Coding for Security and Reliability in Distributed Systems

Thesis by

Wentao Huang

In Partial Fulfillment of the Requirements for the
Degree of
Doctor of Philosophy

CALIFORNIA INSTITUTE OF TECHNOLOGY
Pasadena, California

2017
Defended May 15, 2017

Wentao Huang
ORCID: 0000-0003-0963-3624

To my parents and Lucy

ACKNOWLEDGEMENTS

I am grateful to my advisor, Jehoshua (Shuki) Bruck, for his invaluable guidance and generous encouragement during the course of the research leading to this thesis. Shuki introduced me to the world of coding theory, showed me its depth and broadness, and guided me through this unbelievably colorful, fruitful and delightful journey of exploration. Along this journey, his constant gentle smile always encouraged me. I will try to follow this philosophy of life and keep smiling.

I would like to thank the other members of my candidacy and thesis committee: Michelle Effros, Babak Hassibi, Michael Langberg and Steven Low. Their insightful questions and suggestions greatly helped me to improve this thesis.

During my graduate studies I was fortunate to collaborate with and learn from many wonderful researchers, to whom I am grateful. Special gratitude goes to Tracey Ho, for introducing me to network coding, and for helping me countless times, with her kindness, brightness and patience, to overcome the challenges I encountered in the research that eventually became Part IV of this thesis. I am also grateful to Michael Langberg, for the countless enlightening and inspiring discussions with him, and for helping me to write and communicate better; to Joerg Kliewer, for exposing me to many exciting concepts in coding and security; to Sidharth Jaggi, for his illuminating comments on our work on network error correction; and to Hongyi Yao, for his many helpful advices and for his kind and generous support as a friend.

I am grateful to Ting Wang, who mentored me during my internship at IBM Research. I had a great time during the three months and would like to thank my other collaborators at IBM: Xin Hu, Jiyong Jang and Theodoros Salonidis.

I am grateful to the members of the Paradise Lab. It was a great pleasure to collaborate with Eyal En Gad, whose clarity of thought and commitment to perfection inspired me. I had a wonderful experience collaborating with Yue Li, who provided me with much valuable advice, and taught me a lot about systems. I am also thankful to the other lab members: Farzad Farnoud, Siddharth Jain, and Jin Sima, for creating an ever friendly and cheerful atmosphere in the lab.

Finally, I am grateful to my parents Jianming and Youjun, for their unconditional love and support; and to Lucy, for firmly supporting me as my family away from home.

ABSTRACT

This dissertation studies the use of coding techniques to improve the reliability and security of distributed systems. The first three parts focus on distributed storage systems, and study schemes that encode a message into n shares, assigned to n nodes, such that any $n - r$ nodes can decode the message (reliability) and any colluding z nodes cannot infer any information about the message (security). The objective is to optimize the computational, implementation, communication and access complexity of the schemes during the process of encoding, decoding and repair. These are the key metrics of the schemes so that when they are applied in practical distributed storage systems, the systems are not only reliable and secure, but also fast and cost-effective.

Schemes with highly efficient computation and implementation are studied in Part I. For the practical high rate case of $r \leq 3$ and $z \leq 3$, we construct schemes that require only $r + z$ XORs to encode and z XORs to decode each message bit, based on practical erasure codes including the B, EVENODD and STAR codes. This encoding and decoding complexity is shown to be optimal. For general r and z, we design schemes over a special ring from Cauchy matrices and Vandermonde matrices. Both schemes can be efficiently encoded and decoded due to the structure of the ring. We also discuss methods to shorten the proposed schemes.

Part II studies schemes that are efficient in terms of communication and access complexity. We derive a lower bound on the decoding bandwidth, and design schemes achieving the optimal decoding bandwidth and access. We then design schemes that achieve the optimal bandwidth and access not only for decoding, but also for repair. Furthermore, we present a family of Shamir's schemes with asymptotically optimal decoding bandwidth.

Part III studies the problem of secure repair, i.e., reconstructing the share of a (failed) node without leaking any information about the message. We present generic secure repair protocols that can securely repair any linear schemes. We derive a lower bound on the secure repair bandwidth and show that the proposed protocols are essentially optimal in terms of bandwidth.

In the final part of the dissertation, we study the use of coding techniques to improve the reliability and security of network communication.

Specifically, in Part IV we draw connections between several important problems

in network coding. We present reductions that map an arbitrary multiple-unicast network coding instance to a unicast secure network coding instance in which at most one link is eavesdropped, or a unicast network error correction instance in which at most one link is erroneous, such that a rate tuple is achievable in the multiple-unicast network coding instance if and only if a corresponding rate is achievable in the unicast secure network coding instance, or in the unicast network error correction instance. Conversely, we show that an arbitrary unicast secure network coding instance in which at most one link is eavesdropped can be reduced back to a multiple-unicast network coding instance. Additionally, we show that the capacity of a unicast network error correction instance in general is not (exactly) achievable. We derive upper bounds on the secrecy capacity for the secure network coding problem, based on cut-sets and the connectivity of links. Finally, we study optimal coding schemes for the network error correction problem, in the setting that the network and adversary parameters are not known a priori.

PUBLISHED CONTENT AND CONTRIBUTIONS

[1] W. Huang and J. Bruck, "Generic secure repair for distributed storage," *ArXiv:1706.00500*, 2017,
W. Huang participated in the conception of the project, designed the repair schemes and wrote the manuscript.

[2] ——, *Secret sharing with optimal decoding and repair bandwidth*, accepted to IEEE International Symposium on Information Theory (ISIT), 2017,
W. Huang participated in the conception of the project, designed the coding schemes and wrote the manuscript.

[3] ——, *Secure RAID schemes from EVENODD and STAR codes*, accepted to IEEE International Symposium on Information Theory (ISIT), 2017,
W. Huang participated in the conception of the project, designed the coding schemes and wrote the manuscript.

[4] ——, "Secure RAID schemes for distributed storage," in *IEEE International Symposium on Information Theory (ISIT)*, 2016. DOI: 10.1109/ISIT.2016.7541529,
W. Huang participated in the conception of the project, designed the coding schemes and wrote the manuscript.

[5] W. Huang, T. Ho, M. Langberg, and J. Kliewer, *Single-unicast secure network coding and network error correction are as hard as multiple-unicast network coding*, submitted to IEEE Transactions on Information Theory, 2016,
W. Huang participated in the conception of the project, designed the reduction mappings and wrote the manuscript.

[6] W. Huang, M. Langberg, J. Kliewer, and J. Bruck, "Communication efficient secret sharing," *IEEE Transactions on Information Theory*, vol. 62, no. 12, pp. 7195–7206, 2016. DOI: 10.1109/TIT.2016.2616144,
W. Huang participated in the conception of the project, designed the coding schemes and wrote the manuscript.

[7] E. E. Gad, W. Huang, Y. Li, and J. Bruck, "Rewriting flash memories by message passing," in *IEEE International Symposium on Information Theory (ISIT)*, 2015. DOI: 10.1109/ISIT.2015.7282534,
W. Huang participated in designing and implementing the coding schemes and in writing the manuscript.

[8] W. Huang, M. Langberg, and J. Kliewer, "Connecting multiple-unicast and network error correction: Reduction and unachievability," in *IEEE International Symposium on Information Theory (ISIT)*, 2015. DOI: 10.1109/ISIT.2015.7282477,

W. Huang participated in the conception of the project, designed the reduction mappings and wrote the manuscript.

[9] W. Huang, T. Wang, X. Hu, J. Jang, and T. Salonidis, "Rateless and pollution-attack-resilient network coding," in *IEEE International Symposium on Information Theory (ISIT)*, 2015. DOI: 10.1109/ISIT.2015.7282931,
W. Huang participated in the conception of the project, designed the coding schemes and wrote the manuscript.

[10] W. Huang, T. Ho, M. Langberg, and J. Kliewer, "Reverse edge cut-set bounds for secure network coding," in *IEEE International Symposium on Information Theory (ISIT)*, 2014. DOI: 10.1109/ISIT.2014.6874804,
W. Huang participated in the conception of the project, designed the coding schemes and wrote the manuscript.

[11] ——, "Single-source/sink network error correction is as hard as multiple-unicast," in *Allerton Conference on Communication, Control and Computing*, 2014. DOI: 10.1109/ALLERTON.2014.7028486,
W. Huang participated in the conception of the project, designed the reduction mappings and wrote the manuscript.

[12] ——, "On secure network coding with uniform wiretap sets," in *IEEE International Symposium on Network Coding (NetCod)*, 2013. DOI: 10.1109/NetCod.2013.6570814,
W. Huang participated in the conception of the project, designed the reduction mappings and wrote the manuscript.

[13] W. Huang, T. Ho, H. Yao, and S. Jaggi, "Rateless resilient network coding against byzantine adversaries," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2013. DOI: 10.1109/INFCOM.2013.6566776,
W. Huang participated in the conception of the project, designed the coding schemes and wrote the manuscript.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

Chapter 1

INTRODUCTION

Distributed systems are the foundation of today's information infrastructure. For a system to accommodate more users or loads, one of the most common solutions is to scale out by adding more nodes (e.g., servers) to the system, which then becomes "more distributed". For example, in 2015 Amazon operated more than 30 data centers around the world, and each data center housed between 50,000 and 80,000 nodes [1]. Data centers like these serve most of our daily "online" activities, ranging from browsing web pages and sharing files to booking a hotel room and hailing a ride.

The first three parts of this thesis focus on designing better distributed storage systems. Today's storage systems face many challenges, reliability being a critical one. While a data center should never lose data, node failures occur on a daily basis as confirmed by the industry statistics [2]. Extensive studies on erasure coding have established a rich theory addressing this challenge, with immense impact in practice. Coding techniques, from early Reed-Solomon codes and MDS array codes to recent regenerating and locally repairable codes, lie at the heart of distributed disk arrays such as RAID [3], [4], and many large scale distributed storage systems such as Facebook Analytics Hadoop [2], Microsoft Azure [5] and Google Colossus [6].

Security, namely protecting the privacy of the data[1], is another challenge with increasing importance for distributed storage systems. While the challenge of reliability can be solved by using more redundancy and better erasure codes[2], the challenge of security is more complex and difficult to address. In 2016 alone, more than 2.2 billion data records were reported stolen [7], and this number will likely further increase: not until 2016 did Yahoo discover and disclose that 1.5 billion of their user records were stolen in 2013 and 2014 [8], [9].

In this thesis we investigate the use of coding to strengthen the security (in addition to reliability) of distributed storage systems. Specifically, we study the problem of encoding a message into n shares, assigned to n nodes, such that any $n-r$ nodes can

[1] Security has different meanings in different contexts. In coding theory, security usually refers to the secrecy and privacy of information. We follow this convention in this thesis.

[2] Google cloud storage, for example, guarantees 99.999999999% durability of its data.

Node 1	Node 2	Node 3	Node 4
u	$u \oplus m_1$	$u \oplus m_2$	$u \oplus m_1 \oplus m_2$

Figure 1.1: A simple scheme with $n = 4$, $r = 1$ and $z = 1$. All symbols are bits and all operations are XORs. (m_1, m_2) is the message and u is a random bit. The scheme can tolerate one eavesdropping node as the bit stored by any single node is independent of the message. The scheme can tolerate one node failure as Node 4 stores a parity bit.

decode the message but any colluding z nodes cannot infer any information about the message. Refer to Figure 1.1 for a simple example with $n = 4$, $r = 1$ and $z = 1$. These schemes can find a wide array of applications including, for example, securing disk arrays [10] (where nodes are disks), securing cloud storage [11] (where nodes are different availability zones of a cloud provider or different cloud providers) and securing wireless networks [12] (where nodes are wireless devices).

From a broader perspective, we remark that security is a complex multi-faceted goal that oftentimes requires multiple mechanisms to achieve. Access control and encryption are two parallel approaches to achieve security. Namely, for an adversary to mount a successful attack, it needs both the encryption key (if the message is encrypted) and the access to the nodes storing the message. In this respect, schemes like the one in Figure 1.1 improve security by strengthening the effect of access control. Namely, they enable "fault-tolerant" access control in the sense that no information will be leaked even if access control enforcement fails on any z nodes. Therefore, the schemes allow security to be achieved in a distributed system even if access control enforcement on individual nodes is not perfect, e.g., due to theft, node retirement, hacking, malicious node owner/administrator and so on.

While access control and encryption are most often used together to provide security, there are situations that one of them is more preferable to the other. Encryption is the only choice when access control is not practical, e.g., sending an email over the Internet. However, for most distributed storage systems, access control is enforced and is often advantageous to encryption in terms of performance, cost, flexibility and security [13], [14]. Additionally, most storage systems have already implemented certain kinds of codes for reliability. Therefore, the proposed "secure coding schemes" find natural applications in distributed storage systems, where they fit naturally into the current architectures and improve the effectiveness of access control.

In the first three parts of this thesis, we study how to design such secure coding

schemes with optimal efficiency, summarized in three central questions:

- *How to design schemes with optimal computational complexity and low implementation complexity?* The scheme in Figure 1.1 is extremely simple and efficient in computation and implementation, using only two XORs to encode and one XOR to decode each message bit. But how to design schemes with similar simplicity and efficiency for flexible parameters, e.g., for larger n, r and z? We study this problem in Part I.

- *How to design schemes with optimal communication and access complexity?* We can decode the two message bits in Figure 1.1 by accessing and communicating three bits stored by any three nodes. Is it possible to access and communicate fewer bits? Surprisingly, we show that the answer is yes, if the scheme is designed more carefully. In Part II, we study lower bounds on the amount of bits that need to be accessed and communicated during decoding, and how to design schemes achieving these bounds. Furthermore, we study schemes that achieve optimal access and communication complexity not only for decoding, but also for repair, which is the process of reconstructing the information stored by a node in the event of a node failure.

- *How to securely repair a node failure?* Suppose that a node in Figure 1.1 has failed and lost its bit. To recover from this failure, we can ask a trusted dealer to receive the three bits from the remaining nodes, reconstruct the lost bit, and reassign it to the failed node or a replacement node. The dealer needs to be trusted because it will receive enough information to decode the message. If a trusted dealer is not available, is it possible to repair the failure securely so that no information about the message will be leaked? We study methods to achieve secure repair and their communication complexity in Part III.

So far we have discussed how to achieve security and reliability in distributed storage systems by the means of coding, under the assumption that nodes are connected in a perfect network with secure and reliable channels (e.g., when a user sends/receives a bit to/from a node). In reality, nodes in distributed systems are connected by networks that are subject to errors, eavesdropping and connectivity constraints. In part IV, we study how to achieve secure and reliable communication in general networks by the technique of coding, namely, *network coding*. We remark that the communication problem is a generalization of the storage problem, as storage can

be modeled as communication over time. For network coding, we are interested in the following questions.

- *What is the optimal rate, i.e., throughput, of a network under different connection requirements? What if some of the links are subject to adversarial errors and/or eavesdropping? How to design network codes that achieve the optimal rate?* We remark that, depending on the specific setting, many embodiments of the above questions are central long-standing open problems. For this case, we study the connections between these intriguing open problems using the technique of reduction: *can we reduce an open problem into other open problems?*

Below we briefly summarize the contributions of the four parts.

1.1 Secure RAID Scheme

Codes that can tolerate both node failures and node eavesdropping are initially studied under the context of *secret sharing schemes*. While the literature on secret sharing schemes is very rich and schemes with optimal space efficiency are well known, such as Shamir's scheme [15] and its generalizations [16], [17], application of secret sharing schemes in practical distributed storage systems is rather limited. An important reason is their relatively high computational and implementation complexity. Particularly, no existing secret sharing schemes can achieve an encoding and decoding complexity that is constant with respect to n, even for the first non-trivial case of $r = 2$ or $z = 2$.

In Part I, we present multiple new constructions of schemes with optimal computation and simple implementation. We call these schemes *secure RAID schemes*, because many of them are constructed from efficient erasure codes suitable for practical distributed storage systems, particularly for RAID (Redundant Array of Independent Disks), including the B [18], EVENODD [19] and STAR codes [20]. For $r \leq 3$, $z \leq 3$, the proposed secure RAID schemes require only $r + z$ XORs to encode and z XORs to decode each message bit. This encoding and decoding complexity is shown to be optimal, in the sense that the generator matrices of the schemes are the sparest possible. We remark that the case of $r, z \leq 3$ corresponds to the high rate regime, which is the most relevant configuration for many distributed storage systems. Indeed, all standard RAID levels and most non-standard RAID levels can tolerate no more than three failures.

Furthermore, for general r and z, we construct two secure RAID schemes over a special ring from Vandermonde matrices and Cauchy matrices. The structure of the ring and the structure of the Vandermonde/Cauchy matrix allow the schemes to be efficiently encoded and decoded with low implementation complexity. To the best of our knowledge, the proposed schemes have significantly better encoding and decoding complexity than existing XOR-based secret sharing schemes of general parameters.

Finally, we discuss the shortening of secure RAID schemes. Shortening is a technique of modifying codes in order to obtain flexible code lengths (i.e., n). It plays an important role in the practical deployment of codes, because it allows a specific code to adapt to different configurations when the number of nodes varies. We show that shortening of secret sharing schemes faces new challenges compared to shortening of erasure codes. We show that two of the proposed secure RAID schemes have the desirable property that they can be flexibly shortened to any length.

1.2 Communication Efficient Secret Sharing

Consider the scenario that a user wishes to decode the message by receiving information from the nodes. Referring to the amount of information received by the user as the decoding bandwidth, a natural and important question is to determine and achieve the minimum decoding bandwidth.

In most existing secret sharing schemes, a common practice in decoding is that the user will communicate with a minimum set of nodes, i.e., exactly $n - r$ nodes (even if more nodes are available) and receive all information stored by these nodes. However, this paradigm is not optimal in terms of decoding bandwidth. We show that by receiving information from more than $n - r$ nodes, and by communicating only part of information stored by these nodes, the decoding bandwidth can be reduced. We show a tight lower bound on the decoding bandwidth, which decreases strictly in the number of nodes that participate in decoding. We design two schemes that achieve the optimal decoding bandwidth. One of them is optimal universally, namely, the scheme achieves the optimal decoding bandwidth when d nodes participate in decoding, universally for all $n - r \leq d \leq n$.

While we have designed schemes with optimal decoding bandwidth, it remains an important question whether it is possible to improve the decoding bandwidth of existing schemes, particularly the widely used Shamir's scheme. We answer this question affirmatively by constructing a family of Shamir's schemes that is

asymptotically optimal in the decoding bandwidth.

In addition to the decoding bandwidth, another important aspect of communication efficiency in distributed storage is the repair bandwidth, which is the amount of information communicated during the process of repairing an erasure. We design two schemes that not only achieve the optimal decoding bandwidth, but also achieve the optimal repair bandwidth.

Finally, the decoding and repair bandwidth are naturally related to the access complexity, i.e., the amount of information that needs to be accessed and read from disks, during decoding and repair. The lower bound on the bandwidth is naturally a lower bound on the access complexity. For most of the schemes above, the amount of information accessed is equal to the amount of information communicated. Therefore these schemes are optimal in terms of both bandwidth and access.

1.3 Secure Repair

In the event of a node failure, the system needs to reconstruct the information originally stored by the failed node. With the help of a trusted dealer, this can be achieved easily. The dealer will receive information from the available helper nodes, reconstruct the lost information and then forward it to the failed node. The bandwidth optimal schemes from the previous part assume such a dealer. The challenge in this part is how to repair without such a dealer while maintaining the security guarantee that any colluding z nodes cannot infer the message.

This problem is studied in the literature on secure regenerating codes, e.g., [21]–[24]. The idea there is to design the code carefully and introduce more randomness to scramble the message and protect it from the dealer, so that even the dealer cannot infer the message. This removes the need of a trusted dealer and the failed node can act as the dealer itself. However, achieving secure repair in this way comes at a high cost in rate and other aspects of efficiency.

We address the problem of secure repair from a different perspective, without needing to take the penalty in efficiency as in the case of secure regenerating codes. The key is to utilize ideas from secure multi-party computation and allow a more flexible repair protocol: secure regenerating codes implicitly assume a simple "one-round" repair protocol, in which the helper nodes transmit information to the failed nodes, but they themselves do not receive information from other nodes. We show that, just by slightly relaxing this assumption and allowing a "two-round" protocol, it becomes possible to securely repair *any* secret sharing scheme in a black-box

manner, in the sense that the proposed repair protocol is generic and there is no need to design or modify the secret sharing scheme. We prove a lower bound on the secure repair bandwidth, and propose generic secure repair schemes with essentially optimal secure repair bandwidth. We show that when n dominates z (e.g., the high rate case), the secure repair bandwidth of the proposed repair schemes approaches the non-secure repair bandwidth (or the repair bandwidth with a trusted dealer).

1.4 Network Coding

In network coding, a set of source nodes transmit information to a set of terminal nodes over a network with noiseless links; internal nodes of the network may mix received information before forwarding it. The connection requirements between the sources and terminals range from the simplest unicast, where there is a single source node whose information is demanded by a single terminal node, to the general multiple-unicast, where there are multiple source nodes, each of them demanded by a single and different terminal node. The central question for network coding is to determine whether a rate (e.g., for unicast) or rate tuple (e.g., for multiple-unicast) is achievable. Despite extensive effort, determining the achievability of a rate tuple for multiple-unicast network coding remains an intriguing, central, open problem, e.g., [25]–[27].

We connect multiple-unicast network coding to two other fundamental network coding problems, namely secure network coding and network error correction, using the idea of reduction. A reduction from problem $\mathcal{A}$ to problem $\mathcal{B}$ is a mapping that maps a problem instance $A \in \mathcal{A}$ to a problem instance $B \in \mathcal{B}$, such that the solution to A can be derived easily from the solution to B. Therefore, given a method that solves the instances of $\mathcal{B}$, then by the reduction it also solves the instances of $\mathcal{A}$.

In the secure network coding problem, a subset of links can be eavesdropped, and a valid code design needs to ensure the security of the source message. We construct a reduction that maps an arbitrary multiple-unicast network coding instance to a particular unicast secure network coding instance with an extremely simple setup. Namely, there is a single source, a single terminal, and a single eavesdropped link which can be any link in the network. The only assumption that makes the problem non-trivial is that non-source nodes are allowed to generate independent randomness. It is surprising that this single assumption makes an otherwise trivial problem at least as hard as multiple-unicast network coding. Conversely, we construct a reduction that maps any unicast secure network coding instance in which at most one link can

be eavesdropped to a multiple-unicast network coding instance, hence showing an equivalence between the two problems.

In the network error correction problem, a subset of links are subject to adversarial errors, and a valid code design needs to ensure the reliability of communication. We construct a reduction that maps an arbitrary multiple-unicast network coding instance to a particular unicast network error correction instance with an extremely simple setup. Namely, there is a single source, a single terminal, and a single error link. The only assumption that makes the problem non-trivial is that a subset of links are not subject to error (i.e., the single error link can be any link in the network except this subset). Again, it is surprising that this single assumption makes an otherwise trivial problem at least as hard as multiple-unicast network coding. We show that our reduction is sensitive to the precise definition of achievability, which has an interesting implication that the capacity of a unicast network error correction instance in general is not exactly achievable.

While finding the capacity in the secure network coding problem is hard when non-source nodes can generate randomness, we derive an upper bound on the capacity based on cut-sets and the connectivity of the links. We show that the bound is the tightest possible given the information that is input to the bound.

Finally, we study code construction for the network error correction problem in the setting that any z links are subject to error. While optimal codes are known for this setting [28]–[31], their construction relies on the value of z and the min-cut of the network. In many scenarios, obtaining these values a priori is difficult or impractical. In this regard, we study coding schemes that are rateless, i.e., that do not require prior knowledge of z and the network min-cut. Particularly, the schemes will adapt to the correct z and min-cut during multiple stages of communication and achieve the optimal rate.

Part I

Secure RAID Schemes

Chapter 2

INTRODUCTION TO SECURE RAID

In the RAID (Redundant Array of Independent Disks) architecture [3], information is stored distributively among multiple nodes in a redundant manner that is resilient to individual node failures. Over the past decades, RAID and the fundamental idea of dispersing information to improve reliability, availability and performance have become a ubiquitous principle that lies at the heart of most of today's distributed storage systems [4]–[6].

As distributed storage systems are increasingly being used to store critical and sensitive data, the challenge of protecting data privacy is critical. In the first part of this thesis, we study the design of distributed storage systems that are not only failure-resilient, but also resistant to adversarial eavesdropping of individual nodes. Specifically, we study the problem of storing a message among n nodes such that any $n-r$ nodes can decode the message but any colluding z nodes cannot infer any information about the message. These schemes can find a wide array of applications including, for example, securing disk arrays [10] (where nodes are disks), securing cloud storage [11] (where nodes are different availability zones of a cloud provider or different cloud providers) and securing wireless networks [12] (where nodes are wireless devices).

This problem is initially studied in the literature in the context of *secret sharing schemes*, and rate-optimal (i.e., space-optimal) schemes are known, such as Shamir's scheme [15] and its generalizations [16]. While secret sharing schemes are extensively used as building blocks of numerous secure protocols in cryptography and distributed computing, their application to distributed storage systems has been limited by the relatively high computational complexity (in terms of encoding and decoding) and implementation complexity (in terms of field size) [10], [32]. Secret sharing schemes with improved computational and/or implementation complexity are studied in, e.g., [32], [33]. However, to the best of our knowledge, all existing secret sharing schemes are still rather computationally intensive, limiting their application in distributed storage. We illustrate this by an example.

Node 1	Node 2	Node 3	Node 4	Node 5	Node 6
u	$u \oplus m_1$	$u \oplus m_2$	$u \oplus m_3$	$u \oplus m_4$	$u \oplus \left(\bigoplus_{i=1}^{4} m_i\right)$

Figure 2.1: A simple and optimal secret sharing scheme with $n = 6$, $r = 1$ and $z = 1$ over $\mathbb{F}_2$. All symbols are bits and all operations are XORs. $(m_1, \cdots, m_4)$ is the message and u is a random bit, referred to as a key. The scheme can tolerate one eavesdropping node as the bit stored by any single node is one-time-padded (XORed) by the key. The scheme can tolerate one node failure as Node 6 stores a parity bit.

Motivating Example

Consider the simple scheme in Figure 2.1. The scheme uses 8 XORs to encode 4 message bits and therefore the normalized encoding complexity is 2 XORs per message bit. This encoding complexity is optimal because in order to tolerate z eavesdropping nodes, each message bit has to be padded by z key bits (Node 2 to 5 in the example), resulting in $z = 1$ XOR; and that in order to tolerate r node failures, each message bit has to be checked by r parity bits (Node 6 in the example), resulting in another $r = 1$ XOR. The normalized decoding complexity of the scheme is $z = 1$ XOR per message bit, which is also optimal because at least $z = 1$ key has to be canceled in order to decode a message bit.

Figure 2.1 can be viewed as the secure version of RAID 5 (a standard RAID level that uses the parity code to tolerate one node failure). In many applications, we may want to tolerate more eavesdroppers and failures. Particularly, consider a scheme that can tolerate $r = 2$ failures and $z = 2$ eavesdroppers, e.g., the secure version of the extensively deployed RAID 6. On account of our previous discussion, the lower bound on the encoding complexity is $z + r = 4$ XORs per message bit and the lower bound on the decoding complexity is $z = 2$ XORs per message bit. Unfortunately, all existing secret sharing schemes are not even close to meet these bounds. In fact, no existing schemes can achieve a constant encoding or decoding complexity with respect to n. Therefore, a natural and important question is, does there exist a scheme with encoding and decoding complexity meeting these lower bounds?

In Chapter 4, we settle this problem affirmatively for the high rate case, i.e., the case of $r \leq 3$ and $z \leq 3$, by designing secret sharing schemes with optimal encoding and decoding complexity achieving the lower bounds. We refer to these secret sharing schemes with efficient computation and simple implementation as *secure RAID schemes*, as they are particularly suitable for the RAID architecture and many of the schemes are constructed from practical RAID codes. Refer to Figure 2.2 for

an example of an optimal scheme with $r = 2$ and $z = 2$. We remark that the high rate case is arguably the most relevant regime for distributed storage. Particularly, all standard RAID levels and most non-standard RAID levels can tolerate no more than 3 failures.

Node 1	Node 2	Node 3	Node 4	Node 5	Node 6
u_1	u_2	u_3	u_4	u_5	u_6
$u_3 \oplus u_5 \oplus m_1$	$u_6 \oplus u_3 \oplus m_2$	$u_2 \oplus u_1 \oplus m_3$	$u_5 \oplus u_6 \oplus m_4$	$u_1 \oplus u_4 \oplus m_5$	$u_4 \oplus u_2 \oplus m_6$
$u_2 \oplus u_6 \oplus$ $m_3 \oplus m_5$	$u_4 \oplus u_5 \oplus$ $m_6 \oplus m_3$	$u_6 \oplus u_4 \oplus$ $m_2 \oplus m_1$	$u_1 \oplus u_3 \oplus$ $m_5 \oplus m_6$	$u_3 \oplus u_2 \oplus$ $m_1 \oplus m_4$	$u_5 \oplus u_1 \oplus$ $m_4 \oplus m_2$

Figure 2.2: A secure RAID scheme constructed from the B codes [18]. Symbols are bits and operations are XORs. Each node stores three bits. $m_1, ..., m_6$ are message bits and $u_1, ..., u_6$ are random key bits. The scheme is able to correct $r = 2$ node erasures/failures and is secure against $z = 2$ eavesdropping nodes. The scheme is optimal in several senses. It has optimal rate and optimal field size. It follows a generalized systematic form: all keys are stored uncoded in the first row; all message bits are stored uncoded in the second row, each padded by an optimal number of two keys necessary to resist two eavesdropping nodes; and the third row is redundant. The systematic form implies optimal decoding complexity as the message bits can be decoded by canceling the least amount of keys. The scheme is also optimal in terms of encoding complexity: every key and message bit is checked by an optimal number of two parities in the redundant (third) row necessary to correct two erasures.

Specifically, in Section 4.1 we design a secure RAID scheme with $r \leq 2$, $z \leq 2$, and length $n = p + 2$ for any prime p from the EVENODD codes [19]. The scheme is essentially optimal in computation, i.e., the encoding complexity is approximately 4 XORs per message bit and the decoding complexity is approximately 2 XORs per message bit. In Section 4.4 we design a secure RAID scheme with $r \leq 3$, $z \leq 3$, and $n = p + 3$ for any prime p from the STAR codes [20], with essentially optimal complexity, i.e., approximately 6 XORs to encode and 3 XORs to decode each message bit. In Section 4.5 we present a secure RAID scheme with $r \leq 2$, $z \leq 2$, and length $n = p - 1$ for any prime p from the B codes [18]. Again the scheme is essentially optimal in encoding and decoding. In Section 4.6, we present a secure RAID scheme that is strictly optimal in encoding and decoding, with length $n = p - 1$ for all prime $p \leq 53$, again from the B code. We remark that the B, EVENODD and STAR codes are well-known optimal erasure codes widely used in distributed storage. To the best of our knowledge, the secure B, secure EVENODD and secure STAR are the first schemes that have comparable computational and implementation complexity as these practical erasure codes. They are also the first schemes that are shown to have optimal encoding and decoding complexity.

We highlight two ideas from our constructions: Firstly, we generalize the concept of systematic encoding to the secure setting. Refer to Fig. 2.2 for an example of a systematic scheme. Secondly, we leverage the results on efficient erasure codes, notably on array codes, and construct secure schemes from them and their dual codes. Specifically, the codeword of an array code is a $t \times n$ array; each node stores a column of the array so erasure and distance are defined column-wise. The B, EVENODD and STAR codes are high rate MDS array codes with optimal computation, in the sense that their generator matrices are "low-density" (sparse), and so encoding requires an optimal or almost optimal number of XOR operations. Their dual codes also have "low-density" generator matrices and we show that, in the secure setting, this implies optimal or almost optimal decoding complexity.

So far we have focused on schemes with $r \leq 3$ and $z \leq 3$. For the general case, we design a secure RAID scheme for arbitrary parameters n, r and z based on Reed-Solomon codes. The scheme can be viewed as a systematic version of Shamir's scheme. While the scheme is significantly more efficient than Shamir's scheme in encoding and decoding, it is over a finite field $\mathbb{F}_q$ of size $q > n$, which affects computation and implementation. An interesting problem of practical importance is to design efficient XOR-based secure RAID schemes of general parameters. We construct two such schemes in Chapter 5. Both schemes are over a special ring in which several important ring operations are computationally efficient, in the sense that they can be performed by a smaller number of XORs, and are easy to implement. Several well known families of efficient array codes including the EVENODD and STAR codes are constructed over this ring [19], [20], [34], [35]. Our first scheme over the ring is based on Cauchy matrices. In the high rate regime, the encoding complexity of the scheme is $O(n)$ XORs per message bit and the decoding complexity is $O(z)$ XORs per message bit. Our second scheme, based on Vandermonde matrices, is a generalization of Shamir's scheme to the ring. In the high rate regime, the encoding complexity of the scheme is $O(n)$ XORs per message bit and the decoding complexity is $O(n - r - z)$ XORs per message bit. To our knowledge, these schemes have the best encoding and decoding complexity among XOR-based secret sharing schemes of general parameters (the schemes in [32], [33] have comparable encoding complexity but significantly higher decoding complexity).

Node 1	Node 2	Node 3	Node 4
$c_1 = u$	$c_2 = m_1 + u$	$c_3 = m_2 + u$	$\sum c_i = m_1 + m_2 + u$

(a) A simple scheme with $n = 4$, $r = 1$, $z = 1$. u is a key bit and m_1, m_2 are message bits. Security achieved by one-time-pad and reliability achieved by the parity bit.

Node 1	Node 2	Node 3 (suppressed)	Node 4
$c_1 = u$	$c_2 = m_1 + u$	$c_3 = 0$	$\sum c_i = m_1$

(b) Shortened scheme. The bit c_3 is set to be 0 and does not need to be stored. Node 3 acts as a placeholder only for the purpose of encoding. The scheme is not secure as Node 4 leaks the message bit.

Figure 2.3: An example of naive shortening of a secure RAID scheme compromises security.

Shortening

In coding theory, *shortening* is an important technique of modifying codes in order to obtain flexible code lengths. Specifically, given any $[n, k]$ systematic code and an arbitrary integer $0 < s < k$, one can directly obtain an $[n-s, k-s]$ code of the same distance as the original code, by suppressing s information symbols in the codeword and setting them to be 0 (and therefore without needing to store them) [19]. Shortening plays an important role in the practical deployment and implementation of codes, because it allows a specific code deployed and implemented in a system to adapt to different configurations when the number of nodes varies. Unfortunately, for secure RAID schemes, while the same shortening technique for codes, i.e., suppressing a subset of entries in the codeword, will maintain the reliability parameter r, it may reduce the security parameter z. Refer to Figure 2.3 for an example. However, in Section 4.3 and 5.3 we show that the secure EVENODD and the secure RAID scheme over ring based on Vandermonde matrix both have the desirable property that they can be flexibly shortened to arbitrary lengths without compromising z, if the entries in the codeword are carefully suppressed. As discussed, this property is particularly important in practice.

In summary, our results in this part of the thesis suggest that building "keyless", information-theoretic security into distributed storage and the RAID architecture is practical. Particularly, for many erasure coded distributed storage systems, extending them to employ the proposed secure RAID schemes requires only minor modification to the implementation, with small computational and performance overhead.

Part of the material in Chapter 3 and 4 was presented in [36] and [37].

Chapter 3

MODELS, BOUNDS AND GENERAL CONSTRUCTION FRAMEWORK

3.1 Setup and Definitions

Let $\mathcal{Q}$ be an alphabet, denote $\{1, ..., n\}$ by $[n]$ and denote $\{m, m+1, \cdots, n\}$ by $[m, n]$. For an index set $I \subset [n]$ and a vector $(c_1, ..., c_n)$, denote $c_I = (c_i)_{i \in I}$. An $(n, k, r, z)_{\mathcal{Q}}$ *secret sharing scheme* encodes a message of k symbols over $\mathcal{Q}$ into n shares, each share a symbol over $\mathcal{Q}$, such that 1) the message can be decoded from any $n - r$ shares, and 2) any z shares do not reveal any information about the message. Formally, an $(n, k, r, z)_{\mathcal{Q}}$ secret sharing scheme is a randomized encoding function F that maps a (secret) message $\boldsymbol{m} = (m_1, \cdots, m_k) \in \mathcal{Q}^k$ and a uniformly distributed vector $\boldsymbol{u} = (u_1, \cdots, u_v) \in \mathcal{Q}^v$, also referred to as *keys*, to the codeword $\boldsymbol{c} = (c_1, \cdots, c_n) = F(\boldsymbol{m}, \boldsymbol{u}) \in \mathcal{Q}^n$, such that:

1) (Reliability) $\forall I \subset [n], |I| \geq n - r : \ H(\boldsymbol{m}|c_I) = 0$, implying a decoding function D_I such that $D_I(c_I) = \boldsymbol{m}$.

2) (Secrecy) $\forall I \subset [n], |I| \leq z : \ I(\boldsymbol{m}; c_I) = 0$.

We call the secret sharing schemes constructed in this part of the thesis the *secure RAID schemes*, because they are efficient schemes suitable for distributed storage and particularly for the RAID architecture. We focus on two kinds of linear schemes, namely *scalar* schemes and *array*[1] schemes. For a scalar scheme, $\mathcal{Q}$ is a finite field $\mathbb{F}_q$ and the encoding function F is linear over $\mathbb{F}_q$. For an array scheme, $\mathcal{Q}$ is a vector space $\mathbb{F}_q^t$. In this case, $m_i = (m_{i,1}, \cdots, m_{i,t})$, $u_i = (u_{i,1}, \cdots, u_{i,t})$ and $c_i = (c_{i,1}, \cdots, c_{i,t})$. The encoding function is linear over $\mathbb{F}_q$, taking $m_{i,j}$'s and $u_{i,j}$'s as inputs, and outputs $c_{i,j}$'s. We frequently regard the codeword $\boldsymbol{c}$ as an $t \times n$ array over $\mathbb{F}_q$, in which $c_{i,j}$ is the (j, i)-th entry, namely, the i-th column of the array corresponds to c_i. Note that under the array representation erasure and

[1] Also referred to as *vector linear* in the literature.

eavesdropping are column-wise. Denote

$$\bar{\boldsymbol{m}} = (m_{1,1}, \cdots, m_{1,t}, \cdots, m_{k,1}, \cdots, m_{k,t}) \tag{3.1}$$

$$\bar{\boldsymbol{u}} = (u_{1,1}, \cdots, u_{1,t}, \cdots, u_{v,1}, \cdots, u_{v,t}) \tag{3.2}$$

$$\bar{\boldsymbol{c}} = (c_{1,1}, ..., c_{1,t}, ..., c_{n,1}, ..., c_{n,t}). \tag{3.3}$$

Scalar schemes are special cases of array schemes with $t = 1$. Without loss of generality, in the remaining part of the chapter it is assumed that the secure RAID schemes are array schemes. An $[n, k]_{\mathbb{F}_q^t}$ array code $\mathcal{C}$ of minimum distance $d_{\min}(\mathcal{C}) = r + 1$, where the Hamming distance is defined with respect to $\mathbb{F}_q^t$, is equivalent to an $(n, k, r, 0)_{\mathbb{F}_q^t}$ secure RAID scheme. Denote the dual code of $\mathcal{C}$ by $\mathcal{C}^{\perp}$.

The *rate* of an (n, k, r, z) secure RAID scheme is k/n and characterizes the space efficiency of the scheme. The following proposition gives an upper bound on the rate.

Proposition 3.1.1. *For any (n, k, r, z) secret sharing scheme, it follows that*

$$k \leq n - r - z, \tag{3.4}$$

and the rate of the scheme is at most $\frac{n-r-z}{n}$.

Proof. Let the message $\boldsymbol{m}$ be uniformly distributed, then

$$\begin{aligned} k = H(\boldsymbol{m}) &\overset{(a)}{=} H(\boldsymbol{m}|c_{[z]}) && (3.5)\\ &\leq H(\boldsymbol{m}, c_{[n-r]}|c_{[z]}) \\ &\overset{(b)}{=} H(\boldsymbol{m}|c_{[n-r]}, c_{[z]}) + H(c_{[n-r]}|c_{[z]}) && (3.6)\\ &\overset{(c)}{=} H(c_{[n-r]}|c_{[z]}) && (3.7)\\ &= H(c_{[z+1,n-r]}) \leq n - r - z, \end{aligned}$$

where (a) follows from the security requirement, (b) follows from the chain rule, and (c) follows from the reliability requirement. □

A secure RAID scheme is associated with an encoding algorithm and multiple decoding algorithms. The encoding algorithm is the algorithm of evaluating the encoding function F, and the decoding algorithms are the algorithms of evaluating the decoding functions D_I for $|I| \geq n - r$. We distinguish two cases: *systematic*

decoding when $|I| = n$ and *erasure decoding* when $|I| < n$. We remark that for the application of distributed storage, systematic decoding may be the more common and performance-critical form of decoding.

In reminiscence of linear codes, we define the *generator matrix* of a linear secure RAID scheme to be a $(v+k)t \times nt$ matrix G over $\mathbb{F}_q$ such that $(\bar{\boldsymbol{u}}, \bar{\boldsymbol{m}})G = \bar{\boldsymbol{c}}$. We refer to the first vt rows of G as the *key rows* which correspond to the keys, and refer to the remaining kt rows as the *message rows* which correspond to the messages. Define the *density* of a vector or matrix to be the number of non-zero entries. We are interested in designing secure RAID schemes with low-density generator matrices. Such schemes require a small number of operations in encoding/decoding and therefore are computationally efficient. We remark that the computational efficiency of secure RAID schemes is of practical importance as it is closely related to the read and write performances of the storage systems in terms of throughput and delay.

In this part we also address the efficiency of secure RAID schemes in terms of random access, i.e., the operation of decoding partial messages. Specifically, we study the computational and communication efficiency of decoding a single entry or more generally, a subset of entries of $\boldsymbol{m}$.

3.2 Bounds on Computation

In this section we study lower bounds on the density of the generator matrices of secure RAID schemes. A related important problem is to determine the amount of independent randomness, i.e., the number of keys, required by a scheme. We first address this problem. The following lemma is useful. Note that throughout the thesis, logarithm is base q unless otherwise specified.

Lemma 3.2.1. *For any rate-optimal $(n, k, r, z)_{\mathbb{F}_q^t}$ secure RAID scheme, and any $J \subset [n]$ such that $|J| = z$, it follows that $H(c_J) = zt$.*

Proof. Let the message $\boldsymbol{m}$ be uniformly distributed and suppose for the sake of contradiction that there exists $J \subset [n]$, $|J| = z$, such that $H(c_J) = zt - \epsilon$ for some $\epsilon > 0$. For the ease of notation, we assume without loss of generality (by permuting the indexes if necessary) that $J = [z]$. By the chain rule, $H(c_J) = \sum_{i=1}^{z} H(c_i|c_{[i-1]}) = zt - \epsilon$, and it follows that there exists $i' \in [z]$ such that $H(c_{i'}|c_{[i'-1]}) \leq t - \epsilon'$ for some $\epsilon' > 0$. Hence $H(c_{i'}|c_{[z]\setminus\{i'\}}) \leq t - \epsilon'$. Therefore, without loss of generality (again by permuting the indexes if necessary) let us assume

that $i' = 1$. Recall that $[i, j] = \{i, i+1, ..., j\}$, it follows that

$$\begin{aligned}
I(\, m; c_{[2,z+1]}) &\overset{(a)}{=} I(\boldsymbol{m}; c_{[z+1]}) - I(\boldsymbol{m}; c_1 | c_{[2,z+1]}) \\
&\overset{(b)}{=} I(\boldsymbol{m}; c_{[z+k]}) - I(\boldsymbol{m}; c_{[z+2,z+k]} | c_{[z+1]}) - I(\boldsymbol{m}; c_1 | c_{[2,z+1]}) \\
&\overset{(c)}{=} kt - I(\boldsymbol{m}; c_{[z+2,z+k]} | c_{[z+1]}) - I(\boldsymbol{m}; c_1 | c_{[2,z+1]}) \\
&\geq kt - H(c_{[z+2,z+k]}) - I(\boldsymbol{m}; c_1 | c_{[2,z+1]}) \\
&\geq kt - (k-1)t - I(\boldsymbol{m}; c_1 | c_{[2,z+1]}) \\
&= t - H(c_1 | c_{[2,z+1]}) + H(c_1 | c_{[2,z+1]}, \boldsymbol{m}) \\
&\geq t - H(c_1 | c_{[2,z+1]}) \\
&\geq t - H(c_1 | c_{[2,z]}) \\
&\overset{(d)}{\geq} \epsilon', \qquad (3.8)
\end{aligned}$$

where (a) and (b) follow from the chain rule; (c) follows from the fact that $H(\boldsymbol{m}) = kt$ and that $\boldsymbol{m}$ can be decoded from $c_{[z+k]}$, as $z + k = n - r$; and (d) follows from $H(c_1|c_{[2,z]}) \leq t - \epsilon'$. But (3.8) contradicts the secrecy requirement which implies that $I(\boldsymbol{m}; c_{[2,z+1]}) = 0$. This completes the proof. □

Theorem 3.2.1. *A linear rate-optimal $(n, k, r, z)_{\mathbb{F}_q^t}$ secure RAID scheme uses at least zt keys over $\mathbb{F}_q$ (i.e., $v \geq z$), and the encoding of the scheme is equivalent to a scheme that uses exactly zt keys.*

Proof. Consider any linear $(n, k, r, z)_{\mathbb{F}_q^t}$ scheme such that $k = n - r - z$. Recall that the keys is a length-v vector $\boldsymbol{u}$ over $\mathbb{F}_q^t$. Let the message $\boldsymbol{m}$ be uniformly distributed. We have

$$\begin{aligned}
H(\boldsymbol{u}) &\geq I(c_{[z]}; \boldsymbol{u} | \boldsymbol{m}) \\
&= H(c_{[z]} | \boldsymbol{m}) - H(c_{[z]} | \boldsymbol{u}, \boldsymbol{m}) \\
&\overset{(e)}{=} H(c_{[z]} | \boldsymbol{m}) \\
&\overset{(f)}{=} H(c_{[z]}) \\
&\overset{(g)}{=} zt, \qquad (3.9)
\end{aligned}$$

where (e) follows from the fact that $c_{[z]}$ is a function of $\boldsymbol{u}$ and $\boldsymbol{m}$; (f) follows from the secrecy requirement; and (g) follows from Lemma 3.2.1. Equation (3.9) implies that $v \geq z$ because $H(\boldsymbol{u}) \leq vt$. This proves that the scheme uses at least zt keys over $\mathbb{F}_q$. It remains to show that the scheme is equivalent to a scheme that uses exactly zt keys (e.g., with $v = z$).

Denote the generator matrix of the scheme by G, i.e., G is a $(v+k)t \times nt$ matrix with entries from $\mathbb{F}_q$. Denote by G_1 the submatrix formed by the first vt rows (i.e., the key rows) and the first zt columns of G, denote by G_2 the submatrix formed by the last kt rows (i.e., the message rows) and the first zt columns of G, and denote by $\bar{\boldsymbol{u}}' = \bar{\boldsymbol{u}}G_1$. Then $\bar{c}_{[zt]} = \bar{\boldsymbol{u}}G_1 + \bar{\boldsymbol{m}}G_2 = \bar{\boldsymbol{u}}' + \bar{\boldsymbol{m}}G_2$. Let J be an arbitrary subset of $[nt]$ such that $|J| = (z+k)t$, $[zt] \subset J$ and such that $\boldsymbol{m}$ can be decoded from $\bar{c}_J$. Clearly, the index set of the symbols stored by the first z nodes plus by any k additional nodes is a valid J. We have,

$$\begin{aligned}
H(\bar{c}_J|\bar{\boldsymbol{m}}, \bar{\boldsymbol{u}}') &= H(\bar{c}_J) - I(\bar{c}_J; \bar{\boldsymbol{m}}, \bar{\boldsymbol{u}}') \\
&\overset{(h)}{=} H(\bar{c}_J) - I(\bar{c}_J; \bar{\boldsymbol{m}}, \bar{c}_{[zt]}) \\
&\leq (z+k)t - I(\bar{c}_J; \bar{\boldsymbol{m}}, \bar{c}_{[zt]}) \\
&\overset{(i)}{=} (z+k)t - I(\bar{c}_J; \bar{\boldsymbol{m}}) - I(\bar{c}_J; \bar{c}_{[zt]}|\bar{\boldsymbol{m}}) \\
&\overset{(j)}{=} zt - I((\bar{c}_J; \bar{c}_{[zt]}|\bar{\boldsymbol{m}}) \\
&= zt - H(\bar{c}_{[zt]}|\bar{\boldsymbol{m}}) + H(\bar{c}_{[zt]}|\bar{\boldsymbol{m}}, \bar{c}_J) \\
&\overset{(k)}{=} zt - H(\bar{c}_{[zt]}|\bar{\boldsymbol{m}}) \\
&\overset{(l)}{=} zt - H(\bar{c}_{[zt]}) \\
&\overset{(m)}{=} 0, \qquad (3.10)
\end{aligned}$$

where (h) follows from $\bar{c}_{[zt]} = \bar{\boldsymbol{u}}' + \bar{\boldsymbol{m}}G_2$; (i) follows from the chain rule; (j) follows from $H(\bar{\boldsymbol{m}}|\bar{c}_J) = 0$ and so $I(\bar{c}_J; \bar{\boldsymbol{m}}) = kt$; (k) follows from $[zt] \subset J$; (l) follows from the secrecy requirement; and (m) follows from Lemma 3.2.1. For any $i \in J$, (3.10) implies that there exists a function f such that $\bar{c}_i = f(\bar{\boldsymbol{u}}', \bar{\boldsymbol{m}})$. Since the scheme is linear, f is linear. Note that for any $i \in [nt]$, there exists J such that $i \in J$. Also note that $\bar{\boldsymbol{u}}'$ is a vector of length-zt with entries i.i.d. uniformly distributed over $\mathbb{F}_q$. Hence there exists a matrix G' such that $\bar{\boldsymbol{c}} = (\bar{\boldsymbol{u}}'\ \bar{\boldsymbol{m}})G'$, i.e., G' is the generator matrix of an equivalent scheme that uses exactly zt keys. This completes the proof. □

Theorem 3.2.1 shows that for rate-optimal schemes, zt keys are sufficient and necessary. In the remaining part of the thesis we assume that a rate-optimal $(n, k, r, z)_{\mathbb{F}_q^t}$ secure RAID scheme or secret sharing scheme uses exactly zt keys, and as such the generator matrix G of the scheme has size $(z+k)t \times nt$. The following theorem lower bounds the density of G.

Theorem 3.2.2. *Consider the generator matrix of a rate-optimal $(n, k, r, z)_{\mathbb{F}_q^t}$ secure RAID scheme, then the density of a key row is at least $n - z + 1$, and the density of a message row is at least $r + 1$.*

Proof. Denote by G the generator matrix. Let the message $\boldsymbol{m}$ be uniformly distributed. Let J be an arbitrary subset of $[n]$ such that $|J| = k + z$, and let Z be an arbitrary subset of J such that $|Z| = z$, then we have

$$\begin{aligned} H(\boldsymbol{c}|c_J) &= H(\boldsymbol{c}, c_J) - H(c_J) \\ &= H(\boldsymbol{c}) - H(c_J) \\ &\overset{(a)}{\leq} (z+k)t - H(c_J) \\ &= (z+k)t - H(c_{J\setminus Z}|c_Z) - H(c_Z) \\ &\overset{(b)}{=} (z+k)t - H(c_{J\setminus Z}|c_Z) - zt \\ &\leq kt - I(\boldsymbol{m}; c_{J\setminus Z}|c_Z) \\ &= kt - H(\boldsymbol{m}|c_Z) + H(\boldsymbol{m}|c_J) \\ &\overset{(c)}{=} kt - H(\boldsymbol{m}|c_Z) \\ &\overset{(d)}{=} 0, \end{aligned} \tag{3.11}$$

where (a) follows from Theorem 3.2.1; (b) follows from Lemma 3.2.1; (c) follows from the fact that $\boldsymbol{m}$ can be decoded from $\boldsymbol{c}_J$; and (d) follows from the secrecy requirement. Equation (3.11) implies the erasure of any $n - k - z = r$ entries of $\boldsymbol{c}$ can be corrected, and so that the row space of G is a code of minimum distance $r + 1$. Therefore each row of G must have at least $r + 1$ non-zero entries.

It remains to lower bound the density of the first zt rows of G. Let Z be an arbitrary subset of $[n]$ such that $|Z| = z$, we have

$$\begin{aligned} H(\boldsymbol{u}|c_Z, \boldsymbol{m}) &= H(\boldsymbol{u}|\boldsymbol{m}) - I(c_Z; \boldsymbol{u}|\boldsymbol{m}) \\ &\overset{(e)}{=} zt - I(c_Z; \boldsymbol{u}|\boldsymbol{m}) \\ &\overset{(f)}{=} zt - I(c_Z; \boldsymbol{u}, \boldsymbol{m}) + I(c_Z; \boldsymbol{m}) \\ &\overset{(g)}{=} zt - I(c_Z; \boldsymbol{u}, \boldsymbol{m}) \\ &\overset{(h)}{=} zt - H(c_Z) \\ &\overset{(i)}{=} 0, \end{aligned} \tag{3.12}$$

where (e) follows from $\boldsymbol{u} \perp \boldsymbol{m}$; (f) follows from the chain rule; (g) follows from the secrecy requirement; (h) follows from the fact that c_Z is a function of $\boldsymbol{u}$ and $\boldsymbol{m}$; and

(i) follows from Lemma 3.2.1. Equation (3.12) implies that, if $\boldsymbol{m}$ is given, then the erasure of any $n-z$ entries of $\boldsymbol{c}$ can be corrected as one can first recover $\boldsymbol{u}$ and then compute $\boldsymbol{c}$. Therefore the row space of the submatrix formed by the first zt rows of G is a code of minimum distance $n-z+1$. Therefore the first zt rows of G each have at least $n-z+1$ non-zero entries. This completes the proof. □

From Theorem 3.2.2 we obtain a lower bound on the encoding complexity of an XOR-based (i.e., $q=2$) secure RAID scheme.

Corollary 3.2.1. *Encoding a rate-optimal (n,k,r,z) secure RAID scheme over $\mathbb{F}_2^t$ requires at least $r+z+\frac{rz-z}{n-r-z}$ XORs per message bit.*

Proof. By Theorem 3.2.2, the density of the key rows is at least $n-z+1$ and the density of the message rows is at least $r+1$. By Theorem 3.2.1 there are zt key rows. As the scheme is rate-optimal there are $(n-r-z)t$ message rows. Therefore the density of the generator matrix is at least $zt(n-z+1)+(n-r-z)t(r+1)$ and encoding it requires at least $zt(n-z+1)+(n-r-z)t(r+1)-nt$ XORs. Therefore, the number of XORs amortized over the message bits is

$$\frac{zt(n-z+1)+(n-r-z)t(r+1)-nt}{(n-r-z)t}=n+r+\frac{rz-z}{n-r-z}. \tag{3.13}$$

□

3.3 Systematic Secure RAID Schemes

Codes for distributed storage are typically encoded in a *systematic* way. Namely, a codeword contains two sets of symbols: the uncoded message symbols that appear "in the clear", which are referred to as the *systematic symbols*, and the set of redundant symbols. Systematic codes have important advantages in terms of computational efficiency. Specifically, encoding systematic codes only requires computing redundant symbols. This is especially important when the rate of the code is high, i.e., the number of redundant symbols is small compared to the number of systematic symbols, which is the usual case in storage. Decoding systematic codes is trivial if no systematic symbols are erased. Likewise, random accessing a subset of message symbols is efficient. For secure RAID schemes, conventional systematic encoding is impossible due to the secrecy requirement. This motivates us to generalize the concept of systematic encoding under the context of secrecy.

Definition 3.3.1. *An $(n,k,r,z)_{\mathbb{F}_q^t}$ secure RAID scheme is systematic if*

(1). The keys $\bar{\boldsymbol{u}}$ are stored in the uncoded form in tv entries of the codeword $\bar{\boldsymbol{c}}$.

(2). The message symbols $\bar{m}_1, ..., \bar{m}_{tk}$ are stored in the uncoded form in tk entries of the codeword $\bar{\boldsymbol{c}}$, each padded by a linear function of the keys. Namely, in $\bar{\boldsymbol{c}}$ there is an entry of the form $\bar{m}_i + f_i(\bar{\boldsymbol{u}})$, for $i = 1, ..., tk$.

(3). For $i = 1, ..., tk$, the padding function $f_i(\bar{\boldsymbol{u}})$ can be computed efficiently.

The tv systematic key symbols and the tk systematic message symbols are collectively referred to as the systematic symbols.

Similar to systematic codes, by requiring the systematic symbols to take a simple form, systematic secure RAID schemes have strong advantages in terms of efficiency. Specifically, in Definition 3.3.1, (1) ensures that encoding and decoding (when no erasure has occurred) the key symbols is trivial; (2) ensures that encoding and decoding (when no erasure has occurred) the systematic message symbols only requires computing the padding functions f_i's; and (3) requires that the f_i's take a form amenable to computation.

We remark on the requirement (3) in Definition 3.3.1. From the density perspective, an optimal f_i will be a function of exactly z keys (over $\mathbb{F}_q$). This is because f_i has to be a function of at least z keys in order to meet the secrecy requirement. Otherwise, an adversary can decode $\bar{m}_i$ by looking at no more than z entries of $\bar{\boldsymbol{c}}$, a contradiction. Unfortunately, constructing schemes meeting this requirement strictly is difficult and we are able to do so in Section 4.6, but only for rather restrictive parameters. For most of our secure RAID constructions, some of the f_i's are functions of more than z keys. In this case, we shall show that there exists an efficient algorithm to compute them.

Finally, note that systematic schemes are also efficient in terms of random access, in the sense that decoding a single entry of $\bar{\boldsymbol{m}}$ requires communicating and canceling a small number of keys.

3.3.1 Method of Constructing Secure RAID Schemes

We introduce a method to design systematic secure RAID schemes. The method falls under the general framework of coset coding, which dates back to Wyner's work [38] on the wiretap channel. However here we put special emphasis on designing efficient and systematic schemes in the context of secure RAID.

Consider an an $[n, k_1]$ array code $\mathcal{C}_1$ and an $[n, k_2]$ array code $\mathcal{C}_2$, both over alphabet $\mathbb{F}_q^t$, such that every codeword of $\mathcal{C}_1$ is a codeword of $\mathcal{C}_2$, i.e., $\mathcal{C}_1$ is a *subcode* of $\mathcal{C}_2$. Given such a pair of codes $\mathcal{C}_1$ and $\mathcal{C}_2$, we construct a secure RAID scheme as follows. Encode $\mathcal{C}_2$ systematically and denote the index set of the systematic symbols in the codeword (as a length-tn vector over $\mathbb{F}_q$, see (3.3)) by I_2. Encode $\mathcal{C}_1$ systematically such that the index set I_1 of its systematic symbols satisfies $I_1 \subset I_2$ (which is possible as $\mathcal{C}_1 \subset \mathcal{C}_2$). For the ease of presentation, assume without loss of generality that $I_1 = [tk_1]$ and $I_2 = [tk_2]$. The secure RAID scheme is encoded in 2 steps.

Step 1: Draw tk_1 random keys $\bar{\boldsymbol{u}}$ independently and uniformly from $\mathbb{F}_q$. Encode $\mathcal{C}_1$ by regarding the keys $\bar{\boldsymbol{u}}$ as information symbols to obtain a codeword, and then puncture (delete) all entries in the codeword that are not in I_2. Denote the punctured codeword by $\boldsymbol{d}$, which is the first tk_2 entries of the original codeword.

Step 2: Let $\bar{\boldsymbol{m}}$ be the secret message of length $t(k_2 - k_1)$ over $\mathbb{F}_q$, and denote by $\boldsymbol{e} = \boldsymbol{d} + (\boldsymbol{0}, \bar{\boldsymbol{m}})$, where $\boldsymbol{0}$ is a length-tk_1 zero vector. Encode $\mathcal{C}_2$ by regarding $\boldsymbol{e}$ as information symbols to obtain a codeword $\bar{\boldsymbol{c}}$. $\bar{\boldsymbol{c}}$ is a a length-tn vector over $\mathbb{F}_q$, and is the output codeword of the secure RAID scheme. Note that the codeword $\boldsymbol{c}$ as a length-n vector over the original alphabet $\mathbb{F}_q^t$ can be obtained by collapsing each length-t segment in $\bar{\boldsymbol{c}}$ into one symbol over $\mathbb{F}_q^t$.

An illustration of the construction method is shown in Figure 3.1.

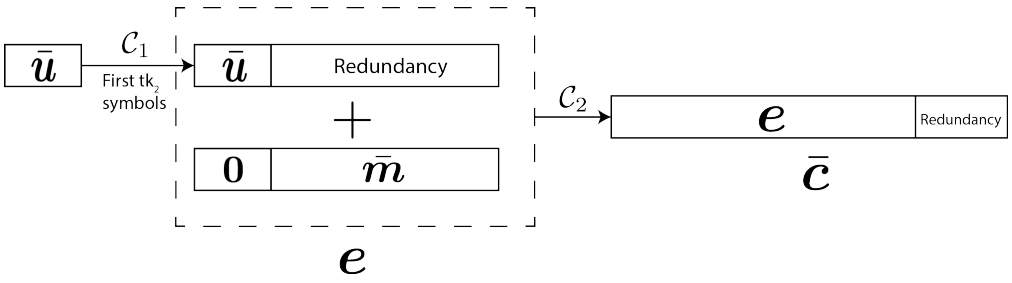


Figure 3.1: Construction of systematic secure RAID scheme from a pair of erasure codes $\mathcal{C}_1$ and $\mathcal{C}_2$ when $I_1 = [tk_1]$ and $I_2 = [tk_2]$.

In general, we can encode $\mathcal{C}_1$ in more flexible ways as long as there exists a I_1 such that $I_1 \subset I_2$ and that $\mathcal{C}_1$ can be decoded from the entries in I_1.

Theorem 3.3.1. *Let $\mathcal{C}_1$ be an $[n, k_1]$ array code and $\mathcal{C}_2$ be an $[n, k_2]$ array code, both over $\mathbb{F}_q^t$, such that $\mathcal{C}_1$ is a subcode of $\mathcal{C}_2$. Then the described encoding scheme*

is an $(n, k_2 - k_1, r, z)$ *secure RAID scheme over* $\mathbb{F}_q^t$*, where* $r = d_{\min}(\mathcal{C}_2) - 1$ *and* $z = d_{\min}(\mathcal{C}_1^{\perp}) - 1$.

Proof. We need to show that the scheme meets the reliability requirement and the secrecy requirement. Because $\boldsymbol{c}$ is a codeword of $\mathcal{C}_2$, and the minimum distance of $\mathcal{C}_2$ is $r+1$, it follows that any r erasures of the entries of $\boldsymbol{c}$ can be corrected. Decoding $\boldsymbol{m}$ from $\boldsymbol{c}$ is simple, as one can read the systematic key entries $\bar{\boldsymbol{u}}$ from $\bar{\boldsymbol{c}}$, and then calculate $\boldsymbol{d}$ from $\bar{\boldsymbol{u}}$, and then cancel $\boldsymbol{d}$ from the systematic message entries in $\boldsymbol{e}$ to obtain $\bar{\boldsymbol{m}}$. This verifies the reliability requirement.

We now prove the security of the scheme. Assume that the adversary observes c_I, for some $I \subset [n]$, $|I| = z$. Recall that $F(\boldsymbol{m}, \boldsymbol{u})$ is the encoding function of the scheme, it suffices to show that $\Pr\{F_I(\boldsymbol{m}, \boldsymbol{u}) = c_I | \boldsymbol{m}\}$ is a constant independent of the choice of $\boldsymbol{m}$, where the probability is taken over the distribution of the keys. Consider the system of linear equations defined by $F_I(\boldsymbol{m}, \boldsymbol{u}) = c_I$ in variables $\boldsymbol{u}$, where $\boldsymbol{m}$ and c_I are given, we are interested in finding the number of solutions to this system.

Let G_2 be the $tk_2 \times tn$ generator matrix of $\mathcal{C}_2$ over $\mathbb{F}_q$, such that $(\bar{\boldsymbol{u}}, \bar{\boldsymbol{m}})G_2 = \bar{\boldsymbol{c}}$. Namely, G_2 is the generator matrix of the proposed secure RAID scheme. Because $\mathcal{C}_1 \subset \mathcal{C}_2$, by the construction method it follows that $(\bar{\boldsymbol{u}}, \boldsymbol{0})G_2$ is a codeword of $\mathcal{C}_1$. Therefore, let G_1 be the submatrix formed by the first tk_1 rows of G_2. Then G_1 is a generator matrix of $\mathcal{C}_1$. Denote by $\bar{I}$ the index set of the entries of $\bar{\boldsymbol{c}}$ corresponding to the set of entries indexed by I in $\boldsymbol{c}$, so $|\bar{I}| = tz$. We claim that the set of columns of G_1 indexed by $\bar{I}$ must be linearly independent. To prove the claim, assume for the sake of contradiction that they are linearly dependent and so there exists a length-tn vector $\bar{\boldsymbol{v}}$ such that $G_1 \bar{\boldsymbol{v}}^T = \boldsymbol{0}$, and such that $\bar{\boldsymbol{v}}$ is non-zero only in the entries indexed by $\bar{I}$. Because G_1 is a parity check matrix of $\mathcal{C}_1^{\perp}$, let $\boldsymbol{v}$ be a length-n vector over $\mathbb{F}_q^t$ obtained by collapsing each length-t segment in $\bar{\boldsymbol{v}}$ into a symbol over $\mathbb{F}_q^t$, then $\boldsymbol{v}$ is a codeword of $\mathcal{C}_1^{\perp}$ that is non-zero only in the entries indexed by I. Since $|I| = z$ but $d_{\min}(\mathcal{C}_1^{\perp}) = z + 1$, this is a contradiction.

Denote the submatrix formed by the last tk_2 rows of G_2 by G_3. For $i = 1, 2, 3$, denote by $G_{i,\bar{I}}$ the submatrix formed by columns of G_i indexed by $\bar{I}$. Then $F_I(\boldsymbol{m}, \boldsymbol{u}) = c_I$ is equivalent to $\bar{\boldsymbol{u}} G_{1,\bar{I}} = \bar{c}_{\bar{I}} - \bar{\boldsymbol{m}} G_{3,\bar{I}}$. Since $G_{1,\bar{I}}$ has full column rank, it follows that the system of equations $\bar{\boldsymbol{u}} G_{1,\bar{I}} = \bar{\boldsymbol{c}}_{\bar{I}} - \bar{\boldsymbol{m}} G_{3,\bar{I}}$ in variables $\bar{\boldsymbol{u}}$ always has a solution, and the number of solutions is exactly $|\text{Null}(G_{1,\bar{I}})|$, where $\text{Null}(A)$ is the left null space of matrix A, i.e., $\{\boldsymbol{x} : \boldsymbol{x}A = \boldsymbol{0}\}$. By the rank-nullity theorem, $|\text{Null}(G_{1,\bar{I}})| =$

$q^{t(k_1-z)}$. Because $\bar{\boldsymbol{u}}$ is uniformly distributed, we have $\Pr\{F_I(\boldsymbol{m},\boldsymbol{u}) = c_I|\boldsymbol{m}\} = |\text{Null}(G_{1,\bar{I}})|/q^{tk_1} = q^{-tz}$, which implies $c_I \perp \boldsymbol{m}$. This completes the proof. □

An $[n,k]$ array code $\mathcal{C}$ is MDS (maximum distance separable) if $d_{\min}(\mathcal{C}) = n-k+1$. An important special case is that $\mathcal{C}_1$ and $\mathcal{C}_2$ are both MDS.

Corollary 3.3.1. *If $\mathcal{C}_1$ and $\mathcal{C}_2$ are MDS codes, then the described encoding scheme is a rate-optimal systematic $(n, k_2-k_1, n-k_2, k_1)$ secure RAID scheme.*

Proof. We first state a known fact.

Lemma 3.3.1. [18], [39] *A code $\mathcal{C}$ is MDS if and only if $\mathcal{C}^\perp$ is MDS.*

Note that the lemma is true for both scalar and array codes. Therefore, $d_{\min}(\mathcal{C}_2) = n-k_2+1$ and $d_{\min}(\mathcal{C}_1^\perp) = k_1+1$. Hence it follows from Theorem 3.3.1 that the scheme is an $(n, k_2-k_1, n-k_2, k_1)$ secure RAID scheme. Clearly the scheme has optimal rate. □

The construction method results in secure RAID schemes that are systematic, where I_1 are the systematic key symbols, and $I_2 \backslash I_1$ are systematic message symbols. The systematic form connects the computational complexity of the scheme to that of the codes. Specifically, the encoding complexity of the scheme is essentially the complexity of encoding $\mathcal{C}_1$ and $\mathcal{C}_2$. A simple systematic decoding algorithm for the scheme is to compute $\boldsymbol{d}$ by encoding $\mathcal{C}_1$ and then cancel it from $\boldsymbol{e}$ to obtain $\bar{\boldsymbol{m}}$, and hence the complexity is dominated by encoding $\mathcal{C}_1$. The erasure decoding algorithm first corrects the erasures by invoking the erasure correction algorithm of $\mathcal{C}_2$, and then invokes the systematic decoding algorithm. So the complexity is essentially the complexity of (erasure) decoding $\mathcal{C}_2$ plus encoding $\mathcal{C}_1$. In summary, to construct efficient secure RAID schemes, it suffices to find a pair of MDS codes $\mathcal{C}_1, \mathcal{C}_2$ of appropriate rates such that $\mathcal{C}_1 \subset \mathcal{C}_2$, and that $\mathcal{C}_1$ can be efficiently encoded, and that $\mathcal{C}_2$ can be efficiently encoded and decoded.

The construction method is also promising in terms of the *simplicity of implementation*. Specifically, the encoder of the secure RAID scheme consists of the encoders of $\mathcal{C}_1$ and $\mathcal{C}_2$. The decoder of the scheme consists of the encoder of $\mathcal{C}_1$ (used in systematic decoding) and the decoder of $\mathcal{C}_2$ (used in correcting erasures). Therefore, if $\mathcal{C}_1$ and $\mathcal{C}_2$ are amenable to implementation then so is the secure RAID scheme.

We remark that the construction method can be interpreted under the framework of coset coding in the following way. Denote by $\boldsymbol{f}$ the codeword of $\mathcal{C}_1$ by encoding $\bar{\boldsymbol{u}}$, and denote by $\boldsymbol{g}$ the codeword of $\mathcal{C}_2$ by encoding $(\mathbf{0}, \bar{\boldsymbol{m}})$. Because $\mathcal{C}_1$ is a subcode of $\mathcal{C}_2$, $\boldsymbol{f}$ is exactly the codeword of $\mathcal{C}_2$ by encoding $\boldsymbol{d}$ (which is the punctured $\boldsymbol{f}$). Therefore it follows from the linearity of $\mathcal{C}_2$ that $\bar{\boldsymbol{c}} = \boldsymbol{f} + \boldsymbol{g}$. Let H_1 be the systematic parity check matrix corresponding to the systematic generator matrix of $\mathcal{C}_1$ that we employ in the scheme, then $H_1 \boldsymbol{f}^T = \mathbf{0}$. And because H_1 is a systematic parity check matrix, we have $H_1 \boldsymbol{g}^T = \bar{\boldsymbol{m}}^T$. Therefore $H_1 \bar{\boldsymbol{c}}^T = H_1(\boldsymbol{f}^T + \boldsymbol{g}^T) = \bar{\boldsymbol{m}}^T$. In this sense, the above encoding scheme can be understood as follows: to encode a secret message $\bar{\boldsymbol{m}}$, the scheme picks a random element from the coset of $\mathcal{C}_1$ whose syndrome is $\bar{\boldsymbol{m}}$.

3.3.2 Secure RAID from Reed-Solomon Codes

A natural choice of $\mathcal{C}_1$ and $\mathcal{C}_2$ in the construction method described in Section 3.3.1 is the Reed-Solomon codes. In fact, Shamir's scheme can be viewed as based on Reed-Solomon codes [40]. However, we show that a systematic scheme based on Reed-Solomon codes has significant advantage over Shamir's scheme in terms of computational efficiency.

Definition 3.3.2. *(*Reed-Solomon Codes [41]*) For $n > k$, let $\mathbb{F}_q$ be a finite field of size $q > n$, and let $\mathcal{S} = \{\alpha_1, ..., \alpha_n\}$ be a set of distinct non-zero elements of $\mathbb{F}_q$, the $[n, k]_{\mathbb{F}_q, \mathcal{S}}$ Reed-Solomon code has a generator matrix*

$$G = \begin{pmatrix} 1 & 1 & ... & 1 \\ \alpha_1 & \alpha_2 & ... & \alpha_n \\ \vdots & & & \vdots \\ \alpha_1^{k-1} & \alpha_2^{k-1} & ... & \alpha_n^{k-1} \end{pmatrix}. \tag{3.14}$$

An equivalent systematic generator matrix G^* can be obtained by performing elementary row operations on G, such that G^* contains an identity submatrix of size k. To construct secure RAID schemes based on Reed-Solomon codes, we let $\mathcal{C}_1$ and $\mathcal{C}_2$ to be Reed-Solomon codes defined on the same $\mathcal{S}$ and such that $\mathcal{C}_1$ has a smaller dimension than $\mathcal{C}_2$.

Theorem 3.3.2. *For any integer n, r and z such that $n - r - z > 0$, a systematic, rate-optimal $(n, n - r - z, r, z)$ secure RAID scheme over $\mathbb{F}_q$ can be constructed by choosing $\mathcal{C}_1$ to be an $[n, z]_{\mathbb{F}_q, \mathcal{S}}$ Reed-Solomon code and $\mathcal{C}_2$ to be an $[n, n - r]_{\mathbb{F}_q, \mathcal{S}}$ Reed-Solomon code in the method described in Section 3.3.1.*

Proof. By Definition 3.3.2, the generator matrix of $\mathcal{C}_1$ is a submatrix of the generator matrix of $\mathcal{C}_2$, and hence $\mathcal{C}_1$ is a subcode of $\mathcal{C}_2$. It is well known that the Reed-Solomon codes are MDS [41], therefore the assertion follows from Corollary 3.3.1. □

Example 3.3.1. *Let $n = 5$, $r = 2$, $z = 2$, $k = 1$ and $q = 5$. Let the generator matrix of $\mathcal{C}_2$ be*

$$G_2 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 \\ 1 & 4 & 2 & 2 & 4 \end{bmatrix},$$

and let the generator matrix G_1 of $\mathcal{C}_1$ be the first two rows of G_2. Then the systematic generator matrices of the codes are

$$G_1^* = \begin{bmatrix} 1 & 0 & 6 & 5 & 4 \\ 0 & 1 & 2 & 3 & 4 \end{bmatrix}, G_2^* = \begin{bmatrix} 1 & 0 & 0 & 1 & 3 \\ 0 & 1 & 0 & 4 & 6 \\ 0 & 0 & 1 & 3 & 6 \end{bmatrix},$$

and the generator matrix of the resulting secure RAID scheme is

$$G = \begin{bmatrix} 1 & 0 & 6 & 5 & 4 \\ 0 & 1 & 2 & 3 & 4 \\ 0 & 0 & 1 & 3 & 6 \end{bmatrix}.$$

Consider an $(n, k = n-r-z, r, z)$ systematic secure RAID scheme based on Reed-Solomon codes. Encoding the scheme is essentially encoding $\mathcal{C}_1$ and $\mathcal{C}_2$, which takes $O((r+z)(n-r))$ operations (multiplications, divisions or additions) over $\mathbb{F}_q$; systematic decoding the scheme is essentially encoding $\mathcal{C}_1$, which takes $O(z(n-z-r))$ operations; erasure/error decoding the scheme can be accomplished by first erasure/error decoding $\mathcal{C}_2$ using the error-erasure version of the Berlekamp-Massey decoding algorithm [39], which takes $O(rn)$ operations, followed by systematic decoding.

In comparison, an $(n, k = n-r-z, r, z)$ Shamir's (ramp) scheme can be viewed as the non-systematic version of the proposed scheme. Encoding Shamir's scheme requires evaluating a polynomial of degree $n-r$ at n points which takes $O(n(n-r))$ operations; decoding Shamir's scheme (with or without erasures) requires interpolating the polynomial which takes $O((n-r)^2)$ operations by Lagrange interpolation. The proposed systematic scheme has significantly better computational efficiency than Shamir's scheme. Particularly, in the high rate regime in which n dominates

r and z, encoding and systematic decoding the systematic scheme both take $O(n)$ operations, whereas encoding and decoding (with or without erasures) Shamir's scheme both take $O(n^2)$ operations. We remark that though (asymptotically) efficient $O(n \log n)$ algorithms are known for encoding and decoding Shamir's scheme, they have large overhead factors and are not commonly used in practice [42]. Finally the systematic scheme is also efficient in random access. Decoding one entry of $\boldsymbol{m}$ in the systematic scheme takes $O(z)$ operations and requires communicating $z + 1$ symbols. Shamir's scheme, however, does not support random access and all entries of $\boldsymbol{m}$ need to be decoded together, requiring $O((n-r)^2) = O((z+k)^2)$ operations and the communication of $z + k$ symbols.

Chapter 4

HIGH RATE SCHEMES WITH OPTIMAL COMPUTATION

In the previous chapter we describe a general framework of constructing secure RAID schemes and present a construction based on Reed-Solomon codes. Reed-Solomon codes are MDS and their parameters are flexible. However, Reed-Solomon codes require computation over finite fields which complicates implementation and affects computational efficiency, especially for the high rate case, i.e., when r is small. For this case, computationally optimal XOR-based array codes, e.g., [19], [18], are proposed and widely used in RAID. The generator matrices of these codes are sparse, and hence encoding requires an optimal or almost optimal number of XOR operations. In this chapter we design XOR-based secure RAID schemes with optimal or almost optimal computational complexity from array codes. Particularly, the schemes have low-density generator matrices that achieve or approach the lower bounds in Section 3.2. Like the array codes, these schemes target the high rate regime, and can tolerate a fixed number (two or three) of erasures and eavesdropping nodes.

A key idea in our constructions is to design $\mathcal{C}_2$ based on high rate array codes and design $\mathcal{C}_1$ based on their dual codes, in the construction method described in Section 3.3.1. There are several benefits of doing this, as the array codes and their duals 1) are both MDS, so that the resulting secure RAID scheme is rate-optimal; 2) have high and low rate, respectively, so that the resulting scheme has high rate; 3) both have low or lowest density generator matrices, implying optimal or almost optimal encoding and decoding complexity for the resulting scheme. However, array codes and their duals are rarely known to contain each other. Surprisingly, we can often modify the codes appropriately to meet the containment condition, while not compromising their complexity and distance. We follow this idea to construct several families of optimal and almost optimal schemes in this chapter.

4.1 Secure EVENODD

In this subsection we construct a family of low-complexity XOR-based secure RAID schemes from the EVENODD codes [19]. We show that the density of the generator matrix of the scheme approaches the lower bound in Theorem 3.2.2, and that the scheme is essentially optimal in terms of encoding complexity and systematic

decoding complexity.

Let p be a prime, the EVENODD code is a $[p+2, p]$ MDS array code over $\mathbb{F}_2^{p-1}$ of minimum distance 3 and with a low density generator matrix [19]. Refer to Fig. 4.1 for an example of $p = 5$. We describe our construction idea using this example. Denote the code in Fig. 4.1 by $\mathcal{C}_2$, which corrects 2 column erasures. To build secrecy into $\mathcal{C}_2$, consider its dual $\mathcal{C}_2^\perp$, obtained by switching the roles of information bit and parity bit. Namely, in Fig. 4.1, for $i \in [4]$, the information bit $c_{6,i}$ is checked by all (parity) entries labeled by i in the top plot. And the information bit $c_{7,i}$ is checked by all entries labeled by i and S in the bottom plot. For example, in the dual code the $(2, 2)$-th entry in the array is $c_{6,2} \oplus c_{7,3}$. Since $\mathcal{C}_2$ is MDS, so is $\mathcal{C}_2^\perp$. $\mathcal{C}_2^\perp$ is a $[p+2, 2]$ code and can be used for secrecy against 2 eavesdropping nodes. Namely, if we encode two columns of keys as information bits according to $\mathcal{C}_2^\perp$ and pad this key array to a message array, then any two columns in the sum array reveal no information about the message. Now we have two efficient codes for reliability and secrecy, respectively. The challenge is to combine them into a single scheme that is both reliable and secure. The straightforward approach for combining codes typically fails. However, as we show in Section 3.3.1, we can construct an efficient secure RAID scheme if $\mathcal{C}_1$ (the code for secrecy) is a subcode of $\mathcal{C}_2$ (the code for reliability). In our example, $\mathcal{C}_2^\perp$ is not a subcode of $\mathcal{C}_2$. However, if we let $\mathcal{C}_1$ be a variant of $\mathcal{C}_2^\perp$ by switching the first and sixth column in its encoding, then one may verify that $\mathcal{C}_1 \subset \mathcal{C}_2$. Based on $\mathcal{C}_1$ and $\mathcal{C}_2$ we construct a secure RAID scheme as follows. Generate two columns of random keys; encode the keys by $\mathcal{C}_1$ but skip the last two columns of the codeword; pad message bits to the 3-rd to 5-th columns of the key array; finally complete the last two columns by encoding $\mathcal{C}_2$. Note that the first 2 columns store only keys, the next 3 columns store uncoded message bits padded by keys, and the last two columns are redundant. The encoding of keys is shown in Fig. 4.2. The scheme corrects 2 erasures, and because $\mathcal{C}_1 \subset \mathcal{C}_2$, the encoding of keys in the last 2 columns is consistent with $\mathcal{C}_1$ (see Fig. 4.2), implying secrecy against 2 eavesdropping nodes. Hence we have the $(7, 3, 2, 2)$ secure EVENODD scheme.

The construction technique can be readily generalized to any prime p. Throughout this section, for an integer a, denote by $\langle a \rangle$ the unique integer m, $0 \leq m < p$, such that $a \equiv m \pmod{p}$. Recall from Section 3.1 that $c_{i,j}$ is the (j, i)-th entry in the codeword array.

Construction 4.1.1. (EVENODD Code [19]) *Let p be a prime, and $m_{i,j}$, $i \in [p]$, $j \in [p-1]$ be the message bits. The codewords of EVENODD form a $(p-1) \times (p+2)$*

1	1	1	1	1	$c_{6,1}$	
2	2	2	2	2	$c_{6,2}$	
3	3	3	3	3	$c_{6,3}$	
4	4	4	4	4	$c_{6,4}$	

1	2	3	4	S		$c_{7,1}$
2	3	4	S	1		$c_{7,2}$
3	4	S	1	2		$c_{7,3}$
4	S	1	2	3		$c_{7,4}$

Figure 4.1: $[7, 5]$ EVENODD code. Codeword is a 4×7 array. The first 5 columns store information bits. Parity bit $c_{6,i}$ is the XOR of all entries labeled by i in the top plot. Parity bit $c_{7,i}$ is the XOR of all entries labeled by i and all entries labeled by S in the bottom plot.

array, described by the following encoding mapping. The first p columns of the array are the systematic symbols, i.e., for $i \in [p]$, $j \in [p-1]$, $c_{i,j} = m_{i,j}$. The last two columns are redundant symbols, i.e., for $j \in [p-1]$, $c_{p+1,j} = \bigoplus_{l=1}^{p} m_{l,j}$ and $c_{p+2,j} = S + \left(\bigoplus_{l=1}^{p} m_{l,\langle j+1-l\rangle}\right)$, where $S = \bigoplus_{l=1}^{p} m_{l,\langle 1-l\rangle}$, and $m_{i,0} \stackrel{\text{def}}{=} 0$.

It is proved in [19] that the EVENODD code is MDS.

Construction 4.1.2. (Secure EVENODD) *Let p be a prime. For $i \in [p-2]$, $l \in [2]$ and $j \in [p-1]$, let $m_{i,j}$ be the message bits, and let $u_{l,j}$ be the uniformly distributed key bits. The codewords of secure EVENODD form a $(p-1) \times (p+2)$ array, described by the following encoding mapping. The first two columns of the array are the systematic key symbols, i.e., $c_{1,j} = u_{1,j}$ for $j \in [p-1]$, and*

$$c_{2,j} = u_{1,j} \oplus u_{2,\langle j+1\rangle} \quad j = 1, \cdots, p-1$$

where $u_{2,0} \stackrel{\text{def}}{=} \bigoplus_{j=1}^{p-1} u_{2,j}$. The 3-rd to p-th columns of the array are the systematic message symbols, i.e., for $i = 3, ..., p$ and $j = 1, \cdots, p-1$

$$c_{i,j} = u_{1,j} \oplus u_{2,\langle i+j-1\rangle} \oplus m_{i-2,j}.$$

The last two columns of the array are redundant symbols, which are computed by encoding the EVENODD code described in Construction 4.1.1, regarding the first p columns of the array as information symbols.

1	1	1	1	1	1	
2	2	2	2	2	2	
3	3	3	3	3	3	
4	4	4	4	4	4	

	2	3	4	Σ	1	1
	3	4	Σ	1	2	2
	4	Σ	1	2	3	3
	Σ	1	2	3	4	4

Figure 4.2: Encoding of keys in the (7,3,2,2) secure EVENODD. The top plot shows how the first column of keys $u_{1,i}$, $i \in [4]$ are propagated in the array and the bottom plot shows how the second column of keys $u_{2,i}$, $i \in [4]$ are propagated. Specifically, in the top plot an entry of i represents the key $u_{1,i}$ being added to this entry in the array. In the bottom plot an entry of i represents the key $u_{2,i}$ being added to this entry in the array, and an entry of Σ represents $\bigoplus_{i=1}^{4} u_{2,i}$ being added. Note that the padding pattern is almost optimal, in the sense that most entries are padded by only two keys and that when more than two keys are padded, Σ only needs to be computed once. We remark that the encoding of keys is identical to the encoding of $\mathcal{C}_1$: this is trivially true for the first five columns by construction, and because $\mathcal{C}_1 \subset \mathcal{C}_2$, it can be shown that the encoding of keys in the last two columns also coincides with $\mathcal{C}_1$.

Note that secure EVENODD is rate-optimal. The following lemma gives an explicit expression of the entries stored in the last two columns of the array.

Lemma 4.1.1. *In Construction 4.1.2,* $c_{p+1,j} = u_{1,j} \oplus u_{2,j} \oplus \left(\bigoplus_{l=1}^{p-2} m_{l,j}\right)$, *and* $c_{p+2,j} = u_{2,j} \oplus S' \oplus \left(\bigoplus_{l=1}^{p-2} m_{l,\langle j-l-1\rangle}\right)$, *for* $j \in [p-1]$, *where* $S' = \bigoplus_{l=1}^{p-2} m_{l,\langle -l-1\rangle}$.

Proof. It follows that

$$
\begin{aligned}
c_{p+1,j} &\overset{(a)}{=} \bigoplus_{l=1}^{p} c_{l,j} \\
&\overset{(b)}{=} \left(\bigoplus_{l=1}^{p} u_{1,j}\right) \oplus \left(\bigoplus_{\substack{l=2 \\ j+l\neq p+1}}^{p} u_{2,\langle j+l-1\rangle}\right) \oplus \left(\bigoplus_{l=1}^{p-1} u_{2,l}\right) \oplus \left(\bigoplus_{l=3}^{p} m_{l-2,j}\right) \\
&= u_{1,j} \oplus \left(\bigoplus_{\substack{l=2 \\ j+l\neq p+1}}^{p} u_{2,\langle j+l-1\rangle}\right) \oplus \left(\bigoplus_{l=1}^{p-1} u_{2,l}\right) \oplus \left(\bigoplus_{l=3}^{p} m_{l-2,j}\right) \\
&= u_{1,j} \oplus \left(\bigoplus_{\substack{l\in[p-1] \\ l\neq j}} u_{2,l}\right) \oplus \left(\bigoplus_{l=1}^{p-1} u_{2,l}\right) \oplus \left(\bigoplus_{l=3}^{p} m_{l-2,j}\right) \\
&= u_{1,j} \oplus u_{2,j} \oplus \left(\bigoplus_{l=3}^{p} m_{l-2,j}\right) \\
&= u_{1,j} \oplus u_{2,j} \oplus \left(\bigoplus_{l=1}^{p-2} m_{l,j}\right),
\end{aligned}
$$

where (a) follows from Construction 4.1.1 and (b) follows from Construction 4.1.2. We also have

$$
\begin{aligned}
S &\overset{(c)}{=} \bigoplus_{l=1}^{p} c_{l,\langle 1-l\rangle} \\
&\overset{(d)}{=} \left(\bigoplus_{l=2}^{p} u_{1,\langle 1-l\rangle}\right) \oplus \left(\bigoplus_{l=2}^{p}\bigoplus_{l'=1}^{p-1} u_{2,l'}\right) \oplus \left(\bigoplus_{l=3}^{p} m_{l-2,\langle 1-l\rangle}\right) \\
&= \left(\bigoplus_{l=2}^{p} u_{1,\langle 1-l\rangle}\right) \oplus \left(\bigoplus_{l=3}^{p} m_{l-2,\langle 1-l\rangle}\right) \\
&= \left(\bigoplus_{l=1}^{p-1} u_{1,l}\right) \oplus \left(\bigoplus_{l=3}^{p} m_{l-2,\langle 1-l\rangle}\right) \\
&= \left(\bigoplus_{l=1}^{p-1} u_{1,l}\right) \oplus \left(\bigoplus_{l=1}^{p-2} m_{l,\langle -l-1\rangle}\right) \\
&= \left(\bigoplus_{l=1}^{p-1} u_{1,l}\right) \oplus S', \qquad (4.1)
\end{aligned}
$$

where (c) follows from Construction 4.1.1 and (d) follows from Construction 4.1.2.

Finally, we have

$$
\begin{aligned}
c_{p+2,j} &\overset{(e)}{=} S \oplus \left(\bigoplus_{l=1}^{p} c_{l,\langle j+1-l\rangle} \right) \\
&\overset{(f)}{=} S \oplus \left(\bigoplus_{\substack{l=1 \\ j+1-l\neq 0}}^{p} u_{1,\langle j+1-l\rangle} \right) \oplus \left(\bigoplus_{\substack{l=2 \\ j+1-l\neq 0}}^{p} u_{2,j} \right) \oplus \left(\bigoplus_{l=3}^{p} m_{l-2,\langle j+1-l\rangle} \right) \\
&= S \oplus \left(\bigoplus_{l=1}^{p-1} u_{1,l} \right) \oplus \left(\bigoplus_{\substack{l=2 \\ j+1-l\neq 0}}^{p} u_{2,j} \right) \oplus \left(\bigoplus_{l=3}^{p} m_{l-2,\langle j+1-l\rangle} \right) \\
&\overset{(g)}{=} S' \oplus \left(\bigoplus_{\substack{l=2 \\ j+1-l\neq 0}}^{p} u_{2,j} \right) \oplus \left(\bigoplus_{l=3}^{p} m_{l-2,\langle j+1-l\rangle} \right) \\
&= S' \oplus u_{2,j} \oplus \left(\bigoplus_{l=3}^{p} m_{l-2,\langle j+1-l\rangle} \right) \\
&= S' \oplus u_{2,j} \oplus \left(\bigoplus_{l=1}^{p-2} m_{l,\langle j-1-l\rangle} \right),
\end{aligned}
$$

where (e) follows from Construction 4.1.1; (f) follows from Construction 4.1.2; and (g) follows from (4.1). □

Theorem 4.1.1. *Secure EVENODD is a* $(p+2, p-2, 2, 2)$ *secure RAID scheme over* $\mathbb{F}_2^{p-1}$*. In particular, the average density of the key rows of the generator matrix is* $\frac{3p-1}{2}$*, and the average density of the message rows is* $\frac{4p-5}{p-1}$*.*

Proof. Since the codewords of secure EVENODD are codewords of the EVENODD code, any two column erasures can be corrected. We focus on proving the security of the scheme. For $i \in [p+2]$ and $j \in [p-1]$, write $c_{i,j} = c'_{i,j} + c''_{i,j}$, where $c'_{i,j}$ is a function of the keys and $c''_{i,j}$ is a function of the message bits. Denote by $\mathcal{C}'$ the $[p+2, 2]$ array code given by the $c'_{i,j}$'s, then to prove the security of secure EVENODD it suffices to show that $\mathcal{C}'$ is MDS. Namely, if $\mathcal{C}'$ is MDS, then all keys can be decoded from any two columns of the array, implying that any two columns of the array are uniformly distributed. Therefore any two columns of secure EVENODD are padded by uniformly distributed random variables and are independent of the

message. By Construction 4.1.2 and Lemma 4.1.1, we have

$$c'_{i,j} = \begin{cases} u_{1,j} \oplus u_{2,\langle i+j-1\rangle} & i = 2, ..., p+1 \\ u_{1,j} & i = 1 \\ u_{2,j} & i = p+2. \end{cases} \tag{4.2}$$

Define $A_k = (a_{ij}^{(k)})$, $1 \le i, j \le p-1$ to be

$$a_{ij}^{(k)} = \begin{cases} 1, & j - i = k \text{ or } i = p - k \\ 0, & \text{otherwise.} \end{cases} \tag{4.3}$$

For example, $A_0 = I$, and for $p = 5$

$$A_1 = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}, \quad A_2 = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}. \tag{4.4}$$

Let G' be the generator matrix of $\mathcal{C}'$ such that

$$(c'_{1,1}, \cdots, c'_{1,p-1}, \cdots, c'_{p+2,1}, \cdots, c'_{p+2,p-1}) = (u_{1,1}, \cdots, u_{1,p-1}, u_{2,1}, \cdots, u_{2,p-1})G',$$

then we have

$$G' = \begin{bmatrix} I & I & I & \cdots & I & 0 \\ 0 & A_1^T & A_2^T & \cdots & A_{\langle p\rangle}^T & I \end{bmatrix}.$$

Switching the first and the $(p+1)$-th column blocks of G', we obtain

$$G'' = \left[\begin{array}{cccc|cc} I & I & \cdots & I & I & 0 \\ A_0^T & A_1^T & \cdots & A_{p-1}^T & 0 & I \end{array}\right].$$

Note that the array code generated by G'' is equivalent the array code generated by G', except that the first and the $(p+1)$-th columns of the array are switched. Therefore G' generates a MDS array code if and only if G'' generates a MDS array code. Notice that G'' is a systematic parity-check matrix of its dual code, and so the systematic generator matrix of the dual code is

$$G''_{\perp} = \left[\begin{array}{cccc|cc} I & 0 & \cdots & 0 & I & A_0 \\ 0 & I & \cdots & 0 & I & A_1 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & I & I & A_{p-1} \end{array}\right]. \tag{4.5}$$

Note that $G''_\perp$ is exactly the generator matrix of the EVENODD code in Construction 4.1.1, which is MDS. By Lemma 3.3.1, the dual EVENODD code, which is equivalent to $\mathcal{C}'$, is also MDS. This completes the proof that secure EVENODD is a $(p+2, p-2, 2, 2)$ secure RAID scheme.

We now analyze the density of the generator matrix of secure EVENODD. Recall that we say a key/message bit is checked by $c_{i,j}$ if the entry in the generator matrix corresponding to the key/message bit and $c_{i,j}$ equals 1. Then by construction, each of the $u_{1,j}$'s is checked for $p+1$ times, and each of the $u_{2,j}$'s is checked for $2(p-1)$ times. Each of the $m_{i,j}$'s, is checked for 3 times if $i+j \neq p-1$, and is checked for $2+p-1 = p+1$ times if $i+j = p-1$. This completes the proof. □

By Theorem 3.2.2, a lower bound on the density of the key rows is $p+1$ and a lower bound on the density of the message rows is 3. Therefore the scheme achieves the lower bound within a factor of 3/2 for the key rows and within a factor of $4/3$ for the message rows.

Systematic decoding the scheme is straightforward by first decoding the keys from the first two columns and then canceling them from the 3-rd to p-th columns. In case of erasures/error, the erasure/error decoding algorithm of EVENODD [19] is invoked, followed by systematic decoding. Encoding secure EVENODD according to Construction 4.1.2 takes a total number of $4p^2 - 7p + 1$ XORs, or on average $4 + \frac{3}{p-2} + \frac{2}{p-1}$ XORs per message bit. Systematic decoding takes a total number of $2p^2 - 4p + 1$ XORs, or on average $2 + \frac{1}{p-2} + \frac{1}{p-1}$ XORs per message bit. By Corollary 3.2.1, encoding each message bit requires at least $4 + \frac{2}{p-2}$ XORs. Moreover, in order to be secure against $z = 2$ eavesdroppers, each message bit has to be padded by at least two keys, and different message bits must not be padded by the same pair of keys, so decoding each message bit requires at least 2 XORs. Therefore secure EVENODD has almost optimal encoding and systematic decoding complexity.

4.2 Algebraic Description of Secure EVENODD

In this subsection we present an algebraic description of the EVENODD code and the secure EVENODD scheme. Let p be a prime, and let $M_p(x) = \sum_{i=0}^{p-1} x^i$ be a polynomial over $GF(2)$. Let $\mathcal{R}_p$ be the ring of polynomials of degree less than $p-1$ over $GF(2)$ with multiplication taken modulo $M_p(x)$. We shall use the indeterminate α to refer to polynomials in $\mathcal{R}_p$, and reserve the indeterminate x for polynomials in $\mathbb{F}_2[x]$. Note that the multiplicative order of α is p, i.e., $\alpha^p = 1$. We interpret the $p-1$ coefficients of an element of $\mathcal{R}_p$ as the $p-1$ bits stored by a

node, i.e., a column in the array. Below we give an alternative description of the EVENODD code.

Construction 4.2.1. (EVENODD Code) *Let p be a prime, the EVENODD code is a $[p+2, p]$ MDS array code over $\mathbb{F}_2^{p-1}$. Specifically, let $m_1(\alpha), ..., m_p(\alpha)$ be p message polynomials each representing $p-1$ message bits. The message polynomials are encoded into $p+2$ codeword polynomials $c_i(\alpha)$, such that $c_i(\alpha)$ represents the $p-1$ bits to be stored on the i-th node. Then $(c_1(\alpha), ..., c_{p+2}(\alpha)) = (m_1(\alpha), ..., m_p(\alpha))\, G_{\text{EO}}$, where G_{EO} is the generator matrix of the EVENODD code over $\mathcal{R}_p$:*

$$G_{\text{EO}} = \left[\begin{array}{cccc|cc} 1 & 0 & \cdots & 0 & 1 & 1 \\ 0 & 1 & \cdots & 0 & 1 & \alpha \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 1 & \alpha^{p-1} \end{array}\right]. \tag{4.6}$$

We now give an algebraic description of the secure EVENODD.

Construction 4.2.2. (Secure EVENODD over $\mathcal{R}_p$) *Let $u_1(\alpha), u_2(\alpha)$ be two key polynomials selected i.i.d. uniformly at random from $\mathcal{R}_p$, and let $m_i(\alpha)$, $i \in [p-2]$ be the message polynomials (each representing $p-1$ bits of information). Then the codeword polynomials are*

$$(c_1(\alpha), ..., c_{p+2}(\alpha)) = (u_1(\alpha), u_2(\alpha), m_1(\alpha), ..., m_{p-2}(\alpha))\, G_{\text{pad}}\, G_{\text{EO}},$$

where G_{pad} is a square matrix that pads the key polynomials to the message polynomials, given in (4.7), and G_{EO} is the generator matrix of the EVENODD code, given in (4.6).

$$G_{\text{pad}} = \left[\begin{array}{cc|ccc} 1 & 1 & 1 & \cdots & 1 \\ 0 & \alpha & \alpha^2 & \cdots & \alpha^{p-1} \\ \hline 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{array}\right]. \tag{4.7}$$

Theorem 4.2.1. *Construction 4.2.2 is a $(p+2, p-2, 2, 2)$ secure RAID scheme.*

Proof. The scheme can correct two erasures because the EVENODD code can correct two erasures. We focus on proving the security of the scheme. Let G_{top} be the first two rows of G_{pad}, then $G_{\text{top}}G_{\text{EO}}$ gives the encoding of keys in the codeword array, namely $(c_1(\alpha), \cdots, c_{p+2}(\alpha)) = (u_1(\alpha), u_2(\alpha))G_{\text{top}}G_{\text{EO}} + M$, where

M is a matrix whose entries are functions of the message polynomials. Define for short $G' = G_{\text{top}}G_{\text{EO}}$ Then to prove security it suffices to prove that the $[p+2, 2]$ code $\mathcal{C}'$ generated by G' is MDS. Namely, if $\mathcal{C}'$ is MDS, then $u_1(\alpha), u_2(\alpha)$ can be decoded from any two entries (over $\mathcal{R}_p$) of $(u_1(\alpha), u_2(\alpha))G'$, implying that any two entries of $(u_1(\alpha), u_2(\alpha))G'$ are uniformly distributed and so any two entries of $(c_1(\alpha), \cdots, c_{p+2}(\alpha))$ are uniformly distributed and independent of the message.

Calculating the matrix product, we have

$$G' = G_{\text{top}}G_{\text{EO}} = \left[\begin{array}{cc|ccc|cc} 1 & 1 & 1 & \cdots & 1 & 1 & 0 \\ 0 & \alpha & \alpha^2 & \cdots & \alpha^{p-1} & 1 & 1 \end{array}\right]. \tag{4.8}$$

Switching the first and the $(p+1)$-th column of G', we obtain

$$G'' = \left[\begin{array}{cccc|cc} 1 & 1 & \cdots & 1 & 1 & 0 \\ 1 & \alpha & \cdots & \alpha^{p-1} & 0 & 1 \end{array}\right]. \tag{4.9}$$

Then the code generated by G'' is equivalent to $\mathcal{C}'$. But G'' is exactly the parity-check matrix of the EVENODD code. Since the EVENODD code is MDS and has minimum distance 3, any two columns of G'' are linearly independent. Therefore $\mathcal{C}'$ is MDS and the proof is complete. □

We interpret the secure EVENODD scheme under the construction framework in Section 3.3.1. It is clear that $\mathcal{C}_2$ is the EVENODD code, and from the proofs of Theorem 4.1.1 and Theorem 4.2.1, $\mathcal{C}_1$ is a variant of the dual EVENODD code, where the first and the $(p+1)$-th entries of the codeword are switched. It is interesting to note that, while Construction 4.1.2 and 4.2.2 both follow this general construction idea, their encodings are slightly different. The reason for the difference is that, in Construction 4.1.2, we regard the generator matrix of the EVENODD code as a $p(p-1) \times (p+2)(p-1)$ matrix over $\mathbb{F}_2$, while in Construction 4.2.2, the generator matrix is a $p \times (p+2)$ matrix over $\mathcal{R}_p$. Therefore, when we "transpose" the generator matrix to obtain the generator matrix for dual code, the granularity of transposition is different.

We now analyze the complexity of Construction 4.2.2. Consider the operation of multiplying a polynomial $f(\alpha) = \sum_{i=0}^{p-2} f_i\alpha^i$ by α^j. Then the resulting polynomial is

$$\alpha^j f(\alpha) = \sum_{\substack{i=0 \\ \langle i+j\rangle \neq p-1}}^{p-2} f_i\alpha^{\langle i+j\rangle} + \sum_{i=0}^{p-2} f_{p-1-j}\alpha^i, \tag{4.10}$$

where we define $f_{p-1} = 0$. Note that the first summation in (4.10) is simply a cyclic shift of $f(\alpha)$ except that the $(p-1-j)$-th entry becomes 0. Therefore the multiplication in (4.10) takes at most $p-1$ XORs to compute. Consider the encoding complexity of Construction 4.2.2. Encoding the first p columns of the array takes $p-1+3(p-1)(p-2)$ XORs and encoding the last two columns by the EVENODD code takes $2(p-1)^2+p-2$ XORs. The complexity of systematic decoding is the same as encoding the first p columns. Therefore, normalized over the $(p-2)(p-1)$ message bits, the encoding complexity is approximately 5 XORs per message bit, and the decoding complexity is approximately 3 XORs per message bit.

Particularly, Construction 4.1.2 has a slightly better encoding and decoding complexity than Construction 4.2.2. On the other hand, the clean algebraic description allows us to prove stronger properties for Construction 4.2.2, which we shall explore in the next section.

4.3 Shortened Secure EVENODD

In this section we study the shortening of secure EVENODD. For a prime p, secure EVENODD is a $(n = p+2, k = p-2, r = 2, z = 2)$ secure RAID scheme over alphabet $\mathbb{F}_2^{p-1}$ with essentially optimal computational and random access complexity. While the length of the secure EVENODD is restricted to $p+2$, in practice it is often desirable to obtain schemes with arbitrary length. For erasure codes, this goal is achieved by the technique of shortening. As shown in Figure 2.3, for secure RAID schemes while the standard shortening technique for erasure codes will maintain the reliability parameter r, it can reduce the security parameter z. However, in this section we show that secure EVENODD has the desirable property that it can be flexibly shortened without compromising z. Namely, from a $(p+2, p-2, 2, 2)$ secure EVENODD scheme one can obtain a $(p+2-s, p-2-s, 2, 2)$ scheme for any $0 < s < p$.

Let p be a prime, recall that $M_p(x) = \sum_{i=0}^{p-1} x^i$ is a polynomial over $GF(2)$, $\mathcal{R}_p$ is the ring of polynomials of degree less than $p-1$ over $GF(2)$ with multiplication taken modulo $M_p(x)$, and that we shall use the indeterminate α instead of x to refer to polynomials in $\mathcal{R}_p$. We remark that $\mathcal{R}_p$ is a field if and only if 2 is a primitive element in $GF(p)$. In this section we focus on the case that $\mathcal{R}_p$ is indeed a field. This is not a significant restriction as it is conjectured that 2 is a primitive element in $GF(p)$ for a constant fraction (≈ 0.374) of primes p [35].

Construction 4.3.1. (Shortened Secure EVENODD) *Let* $0 < s < p-2$ *be an*

integer. The shortened secure EVENODD of length $p+2-s$ *and dimension* $p-2-s$ *is encoded by*

$$(u_1(\alpha), u_2(\alpha), m_1(\alpha), ..., m_{p-2-s}(\alpha))\ G'_{\text{pad}}\ G'_{\text{EO}},$$

where $u_1(\alpha)$, $u_2(\alpha)$ *are randomly selected key polynomials,* $m_1(\alpha), ..., m_{p-2-s}(\alpha)$ *are the message polynomials, and* G'_{pad} *is obtained by deleting the* 3*-rd to* $(s+2)$*-th rows and columns from* G_{pad}*, given in (4.7), and* G'_{EO} *is obtained by deleting the* 3*-rd to* $(s+2)$*-th rows and columns from* G_{EO}*, given in (4.6).*

Note that the length and dimension of the shortened secure EVENODD are decreased by s compared to the secure EVENODD, and so the shortened scheme is rate-optimal. Also note that by deleting the rows and columns from the matrices we are essentially suppressing the 3-rd to $(s+2)$-th entries in the codeword of Construction 4.2.2 to be 0.

We first present a standard lemma useful for proving the security of a scheme.

Lemma 4.3.1. *Consider random variables* $\boldsymbol{c}, \boldsymbol{u}$ *and* $\boldsymbol{m}$*, if* $H(\boldsymbol{c}) \leq H(\boldsymbol{u})$*,* $\boldsymbol{u} \perp \boldsymbol{m}$*,* $H(\boldsymbol{c}|\boldsymbol{u}, \boldsymbol{m}) = 0$ *and* $H(\boldsymbol{u}|\boldsymbol{c}, \boldsymbol{m}) = 0$*, then* $H(\boldsymbol{m}|\boldsymbol{c}) = H(\boldsymbol{m})$*.*

Proof.

$$\begin{aligned}
H(\boldsymbol{m}|\boldsymbol{c}) &\overset{(a)}{=} H(\boldsymbol{m}) - H(\boldsymbol{c}) + H(\boldsymbol{c}|\boldsymbol{m}) \\
&\overset{(b)}{=} H(\boldsymbol{m}) - H(\boldsymbol{c}) + H(\boldsymbol{c}|\boldsymbol{m}) - H(\boldsymbol{c}|\boldsymbol{u}, \boldsymbol{m}) \\
&= H(\boldsymbol{m}) - H(\boldsymbol{c}) + I(\boldsymbol{c}; \boldsymbol{u}|\boldsymbol{m}) \\
&= H(\boldsymbol{m}) - H(\boldsymbol{c}) + H(\boldsymbol{u}|\boldsymbol{m}) - H(\boldsymbol{u}|\boldsymbol{c}, \boldsymbol{m}) \\
&\overset{(c)}{=} H(\boldsymbol{m}) - H(\boldsymbol{c}) + H(\boldsymbol{u}|\boldsymbol{m}) \\
&\overset{(d)}{=} H(\boldsymbol{m}) - H(\boldsymbol{c}) + H(\boldsymbol{u}) \\
&\overset{(e)}{\geq} H(\boldsymbol{m}),
\end{aligned}$$

where (a) follows from $I(\boldsymbol{m}; \boldsymbol{c}) = H(\boldsymbol{m}) - H(\boldsymbol{m}|\boldsymbol{c}) = I(\boldsymbol{c}; \boldsymbol{m}) = H(\boldsymbol{c}) - H(\boldsymbol{c}|\boldsymbol{m})$; (b) and (c) follow from the hypothesis that $H(\boldsymbol{c}|\boldsymbol{u}, \boldsymbol{m}) = H(\boldsymbol{u}|\boldsymbol{c}, \boldsymbol{m}) = 0$; (d) follows from $\boldsymbol{d} \perp \boldsymbol{m}$; and (e) follows from $H(\boldsymbol{c}) \leq H(\boldsymbol{u})$. □

We remark on the use of the lemma. Typically $\boldsymbol{c}$ is the z codeword entries observed by the adversary, $\boldsymbol{u}$ is the keys, and $\boldsymbol{m}$ is the messages. Therefore the statement

$H(\boldsymbol{m}|\boldsymbol{c}) = H(\boldsymbol{m})$ implies that no information about the message is leaked. For all schemes discussed in this thesis, $H(\boldsymbol{c}) \leq H(\boldsymbol{u})$, $\boldsymbol{u} \perp \boldsymbol{m}$ and $H(\boldsymbol{c}|\boldsymbol{u}, \boldsymbol{m})$ follow trivially by construction, and to invoke the lemma, the only non-trivial condition that one needs to establish is $H(\boldsymbol{u}|\boldsymbol{c}, \boldsymbol{m}) = 0$.

We are ready to show the correctness of the shortened secure EVENODD.

Theorem 4.3.1. *If $\mathcal{R}_p$ is a field, then the shortened secure EVENODD is a $(p+2-s, p-2-s, 2, 2)$ secure RAID scheme.*

Proof. It is easy to see that the shortened scheme maintains the same level of reliability as secure EVENODD, and can tolerate any two erasures. Particularly, the same decoding algorithm can be used, except that the shortened (suppressed) entries in the codeword are set to be 0 by default. It remains to be shown that the shortened scheme is also secure in the presence of two eavesdropping nodes.

By Lemma 4.3.1, the scheme is secure if the following claim is true: let $c_{i_1}(\alpha), c_{i_2}(\alpha)$ be any two entries of the shortened codeword, then $u_1(\alpha)$ and $u_2(\alpha)$ are functions of $c_{i_1}(\alpha), c_{i_2}(\alpha)$ and $m_i(\alpha)$, $i = 1, ..., p-2-s$. To prove the claim, we reformulate it in the context of Construction 4.2.2. Note that encoding Construction $4.3.1$ is equivalent to encoding Construction 4.2.2 and suppressing the 3-rd to $(s+2)$-th entries in the codeword to be 0. Therefore, let $\mathcal{S} = \{3, 4, ..., s+2\}$ be the index set of the shortened entries, then an equivalent claim is: in Construction 4.2.2, for any $i_1, i_2 \in [p+2]\backslash\mathcal{S}$, $u_1(\alpha)$ and $u_2(\alpha)$ are functions of $c_{i_1}(\alpha), c_{i_2}(\alpha)$, $\{c_i(\alpha) : i \in \mathcal{S}\}$, and $m_i(\alpha)$, $i \in [p-2]\backslash\mathcal{S}$. In the following we prove this claim by showing that one can recover $u_1(\alpha)$ and $u_2(\alpha)$ from $c_{i_1}(\alpha), c_{i_2}(\alpha)$, $\{c_i(\alpha) : i \in \mathcal{S}\}$, and $m_i(\alpha)$, $i \in [p-2]\backslash\mathcal{S}$. Note that the generator matrix of Construction 4.2.2 is

$$G_{\text{pad}}\, G_{\text{EO}} = \left(\begin{array}{cc|ccc|cc} 1 & 1 & 1 & \cdots & 1 & 1 & 0 \\ 0 & \alpha & \alpha^2 & \cdots & \alpha^{p-1} & 1 & 1 \\ \hline 0 & 0 & 1 & \cdots & 0 & 1 & \alpha^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 1 & \alpha^{p-1} \end{array}\right). \tag{4.11}$$

We remove the rows corresponding to the message polynomials $m_i(\alpha), i \in [p-2]\backslash\mathcal{S}$,

namely the $(3+s)$-th to the p-th rows from (4.11) to obtain

$$G_s = \left(\begin{array}{cc|ccc|ccc|cc} 1 & 1 & 1 & \cdots & 1 & 1 & \cdots & 1 & 1 & 0 \\ 0 & \alpha & \alpha^2 & \cdots & \alpha^{s+1} & \alpha^{s+2} & \cdots & \alpha^{p-1} & 1 & 1 \\ \hline 0 & 0 & 1 & \cdots & 0 & 0 & \cdots & 0 & 1 & \alpha^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 & \cdots & 0 & 1 & \alpha^{s+1} \end{array}\right).$$

It suffices to show that column vectors $e_1 = (1, 0, \cdots, 0)$ and $e_2 = (0, 1, 0, \cdots, 0)$ are in the column span of the space generated by the 3-rd to $(s+2)$-th columns plus the i_1-th and i_2-th columns of G_s. Clearly, if both the i_1-th and i_2-th columns are not the last two columns of G_s, then since $\mathcal{R}_p$ is a field, the i_1-th and i_2-th columns span e_1 and e_2. In the remaining part of the proof we focus on the cases that at least one of i_1 and i_2 is equal to $p+1$ or $p+2$. We also need to distinguish the case that s is odd from the case that it is even. We begin with the case that s is odd.

Case 1 ($i_1 = p+1,\ i_2 < p+1$): sum the 3-rd to $(s+2)$-th columns and the i_1-th column to obtain $u = (0, 1+\sum_{i=2}^{s+1}\alpha^i, 0, \cdots, 0)$. This vector together with the i_2-th column span e_1, e_2.

Case 2 ($i_1 = p+2,\ i_2 < p+1$): for $i = 3, ..., s+2$, scale the i-th column by α^{i-1} and add it to the i_1-th column to obtain the vector $v = (\sum_{j=2}^{s+1}\alpha^j, 1 + \sum_{j=2}^{s+1}\alpha^{2j}, 0, \cdots, 0)$. Now if $i_2 = 1$, then clearly v and the first column span $e_1,\ e_2$. Otherwise, scale the i_2-th column by $\sum_{j=2}^{s+1}\alpha^j$ and add to v to obtain $(0, 1+\sum_{j=2}^{s+1}\alpha^{j+i_2-1} + \sum_{j=2}^{s+1}\alpha^{2j}, 0, \cdots, 0)$. We only need to show that

$$\rho = 1 + \sum_{j=2}^{s+1}\alpha^{j+i_2-1} + \sum_{j=2}^{s+1}\alpha^{2j} \neq 0. \tag{4.12}$$

Note that $\alpha^p = 1$ and (4.12) is trivially true when $s = 1$ or $p = 5$. Now we prove (4.12) assuming $p > 5$ and $s > 1$. First suppose that $s < \frac{p+3}{2}$ so that the summation $\sum_{j=2}^{s+1}\alpha^{2j}$ includes α^4, α^6 but does not include α^5. $\sum_{j=2}^{s+1}\alpha^{j+i_2-1}$, however, sums consecutive powers of α and therefore, if it includes α^5, then it must include either α^4 or α^6 or both. Therefore ρ must either 1) include both α^4 and α^6 but not include α^5, or 2) include α^5 but not include at least one of α^4 and α^6. In both cases ρ is not zero. Now suppose that $s \geq \frac{p+3}{2}$, then $\sum_{j=2}^{s+1}\alpha^{2j}$ includes α^1, α^3 but does not include α^2. By the same argument as above again it follows that $\rho \neq 0$. This proves (4.12) and so v and the i_2-th column span e_1, e_2.

Case 3 ($i_1 = p+1, i_2 = p+2$): obtain u as in Case 1 and obtain v as in Case 2. Then u, v span e_1, e_2.

We now turn to the regime that s is even.

Case $1'$ ($i_1 = p+1,\ i_2 < p+1$): sum the 3-rd to $(s+2)$-th columns and the i_1-th column to obtain $u' = (1, 1+\sum_{i=2}^{s+1}\alpha^i, 0, \cdots, 0)$. This vector together with the i_2-th column span e_1, e_2.

Case $2'$ ($i_1 = p+2,\ i_2 < p+1$): proof is identical to the proof of Case 2.

Case $3'$ ($i_1 = p+1, i_2 = p+2$): obtain u' as in Case $1'$. Add u' to the j-th column to obtain

$$w_j = (0, 1+\sum_{\substack{k=2\\k\neq j-1}}^{s+1}\alpha^k, 0, \cdots, 1, \cdots, 0),\ \ j = 3, ..., s+2$$

where the entry of 1 is the j-th entry. Now scale w_j by α^{j-1} and sum all of them to the $(p+2)$-th column to obtain:

$$v' = \left(0, 1+\sum_{j=2}^{s+1}\left(\alpha^j\left(1+\sum_{l=2,\ l\neq j}^{s+1}\alpha^l\right)\right), 0, \cdots, 0\right) \tag{4.13}$$

$$= \left(0, 1+\sum_{j=2}^{s+1}\alpha^j, 0, \cdots, 0\right). \tag{4.14}$$

Then u', v' span e_1, e_2. The proof is complete. □

4.4 Secure STAR

The secure EVENODD scheme can tolerate $r \leq 2$ erasures and $z \leq 2$ eavesdroppers. A natural and important question is how to construct secure RAID schemes that can tolerate more erasures and eavesdroppers. In this section we construct an efficient secure RAID scheme based on the STAR code [20], which is a generalization of the EVENODD code. The STAR code is a family of MDS array codes capable of tolerating 3 erasures with almost optimal encoding complexity. The resulting secure RAID scheme can tolerate $r \leq 3$ erasures and $z \leq 3$ eavesdroppers, with almost optimal encoding and decoding complexity. We start with describing the STAR code. Define $M_p(x)$, $\mathcal{R}_p$ and α as in Section 4.3. Recall that the multiplicative order of α is p.

Construction 4.4.1. (STAR code [20]) *Let p be a prime, the STAR code is a $[p+3, p]$ MDS array code over $\mathbb{F}_2^{p-1}$. Specifically, let $m_1(\alpha), ..., m_p(\alpha)$ be p message polynomials each representing $p-1$ message bits. Then the codeword*

polynomials $(c_1(\alpha), ..., c_{p+3}(\alpha)) = (m_1(\alpha), ..., m_p(\alpha))\ G_{\text{STAR}}$*, where* G_{STAR} *is the generator matrix of the STAR code:*

$$G_{\text{STAR}} = \left(\begin{array}{cccc|ccc} 1 & 0 & \cdots & 0 & 1 & 1 & 1 \\ 0 & 1 & \cdots & 0 & 1 & \alpha & \alpha^{-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 1 & \alpha^{p-1} & \alpha^{-(p-1)} \end{array}\right). \tag{4.15}$$

We now describe the secure STAR scheme.

Construction 4.4.2. (Secure STAR) *Let* $u_1(\alpha), u_2(\alpha), u_3(\alpha)$ *be three key polynomials selected i.i.d. uniformly at random from* $\mathcal{R}_p$*, and let* $m_i(\alpha)$*,* $i \in [p-3]$ *be the message polynomials (each representing* $p-1$ *bits of information). The key and message polynomials are encoded into* $p+3$ *codeword polynomials as*

$$(c_1(\alpha), ..., c_{p+3}(\alpha)) = (u_1(\alpha), u_2(\alpha), u_3(\alpha), m_1(\alpha), ..., m_{p-3}(\alpha))\ G''_{\text{pad}}\ G_{\text{STAR}} \tag{4.16}$$

where G''_{pad}*, defined in (4.17), is a square matrix that pads the key polynomials to the message and* G_{STAR} *is defined in (4.15).*

$$G''_{\text{pad}} = \left(\begin{array}{cc|ccc|c} 1 & 1 & 1 & \cdots & 1 & 1 \\ 1 & \alpha & \alpha^2 & \cdots & \alpha^{p-2} & \alpha^{p-1} \\ 1 & \alpha^{-1} & \alpha^{-2} & \cdots & \alpha^{-(p-2)} & \alpha^{-(p-1)} \\ \hline 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \end{array}\right). \tag{4.17}$$

Note that the secure STAR is rate-optimal. The following result shows the correctness of the secure STAR.

Theorem 4.4.1. *The secure STAR is a* $(n = p+3, k = p-3, r = 3, z = 3)$ *secure RAID scheme over* $\mathbb{F}_2^{p-1}$.

Proof. Because the STAR code can tolerate three erasures and the codewords of secure STAR are codewords of the STAR code, secure STAR can also tolerate three erasures. It remains to be shown that the scheme can tolerate three eavesdropping nodes.

By Lemma 4.3.1, it suffices to show that from any three entries of the codeword $c_{i_1}(\alpha)$, $c_{i_2}(\alpha), c_{i_3}(\alpha)$ and $m_i(\alpha)$, $i = 1, ..., p-3$, one can recover $u_1(\alpha), u_2(\alpha)$ and $u_3(\alpha)$. To prove this claim, note that the generator matrix of secure STAR is

$$G''_{\text{pad}}\, G_{\text{STAR}} = \left(\begin{array}{cc|ccc|c|ccc} 1 & 1 & 1 & \cdots & 1 & 1 & 1 & 0 & 0 \\ 1 & \alpha & \alpha^2 & \cdots & \alpha^{p-2} & \alpha^{p-1} & 0 & 0 & 1 \\ 1 & \alpha^{-1} & \alpha^{-2} & \cdots & \alpha^{-(p-2)} & \alpha^{-(p-1)} & 0 & 1 & 0 \\ \hline 0 & 0 & 1 & \cdots & 0 & 0 & 1 & \alpha^2 & \alpha^{-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 & 1 & \alpha^{p-2} & \alpha^{-(p-2)} \end{array}\right). \tag{4.18}$$

Let G_{top} be the matrix formed by the first three rows of the matrix in (4.18), then G_{top} is a systematic parity check matrix of the STAR code if the $(p+2)$-th and $(p+3)$-th columns are swapped. Because the STAR code is MDS, any three columns of its parity check matrix are linearly independent. Therefore any three columns of G_{top} are linearly independent. This proves the claim and the theorem. □

On account of the proof of Theorem 4.4.1, we can interpret the secure STAR scheme under the framework of Theorem 3.3.1, where $\mathcal{C}_2$ is the STAR code, and $\mathcal{C}_1$ is the dual STAR code.

4.4.1 Encoding Secure STAR

We analyze the computational complexity of secure STAR. As discussed in Section 4.2, multiplying an element of $\mathcal{R}_p$ by α^i requires at most $p-1$ XORs. Consider the encoding complexity of secure STAR, in the first phase we multiply the key and message polynomials by $G_{\text{pad}''}$. This takes at most $10(p-1) + 5(p-3)(p-1)$ XORs. The second phase, which is to encode the standard STAR code, takes at most $3(p-1)^2 + 2(p-2)$ XORs. Therefore the normalized encoding complexity of secure STAR is

$$\frac{10(p-1) + 5(p-3)(p-1) + 3(p-1)^2 + 2(p-2)}{(p-3)(p-1)} \approx 8$$

XORs to encode each bit of message. By Corollary 3.2.1, a lower bound on the normalized encoding complexity is $6 + \frac{6}{p-3} \approx 6$ XORs to encode each message bit. Therefore the encoding complexity of secure STAR is almost optimal. In the following we show an improved encoding scheme of secure STAR to further reduce the encoding complexity. The normalized encoding complexity of the improved scheme is approximately 6XORs per message bit, meeting the lower bound.

Specifically, consider the (binary) generator matrix of the STAR code by regarding a polynomial $f(\alpha)$ as a binary row vector of length $p-1$. And so G_{STAR} expands into a $p(p-1)$ by $(p+3)(p-1)$ binary matrix, i.e., each entry in the matrix in (4.18) expands into a $(p-1)$ by $(p-1)$ block:

$$G'_{\text{STAR}} = \left(\begin{array}{cccc|ccc} I & 0 & \cdots & 0 & I & A_0 & A_0 \\ 0 & I & \cdots & 0 & I & A_1 & A_{\langle -1\rangle} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & I & I & A_{p-1} & A_{\langle -(p-1)\rangle} \end{array}\right) \tag{4.19}$$

where I is the identity matrix of order $p-1$, 0 is the zero matrix, and $A_k = (a_{ij}^{(k)})$, $1 \le i, j \le p-1$ is defined by:

$$a_{ij}^{(k)} = \begin{cases} 1, & j - i = k \text{ or } i = p - k \\ 0, & \text{otherwise.} \end{cases} \tag{4.20}$$

Note that $A_0 = I$, and refer to (4.4) for more examples. The binary parity check matrix corresponding to the systematic generator matrix in (4.19) is :

$$H'_{\text{STAR}} = \left(\begin{array}{cccc|ccc} I & I & \cdots & I & I & 0 & 0 \\ A_0^t & A_1^t & \cdots & A_{p-1}^t & 0 & I & 0 \\ A_0^t & A_{\langle -1\rangle}^t & \cdots & A_{\langle -p-1\rangle}^t & 0 & 0 & I \end{array}\right).$$

Consider the complexity of encoding the dual code of the STAR code by multiplying a message vector $(\boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{u}_3)$ with the matrix H'_{STAR}, where $\boldsymbol{u}_i$ is a binary row vector of length $p-1$. Then multiplying $\boldsymbol{u}_i$ with A_j^t is simply a cyclic shift of $\boldsymbol{u}_i$ (by j entries to the left) except that the $(p-j)$-th entry in the result becomes $u_i^* = \sum_{k=1}^{p-1} u_{ik}$. Therefore the only computation required in multiplying $\boldsymbol{u}_i$ with A_j^t is to compute u_i^*, which only needs to be performed once for each $\boldsymbol{u}_i$.

Now to encode secure STAR, instead of using the padding matrix G''_{pad} in (4.17), we use the following matrix G'_{pad}:

$$\left(\begin{array}{cc|ccc|c} I & I & I & \cdots & I & I \\ A_0^t & A_1^t & A_2^t & \cdots & A_{p-2}^t & A_{p-1}^t \\ A_0^t & A_{\langle -1\rangle}^t & A_{\langle -2\rangle}^t & \cdots & A_{\langle -(p-2)\rangle}^t & A_{\langle -(p-1)\rangle}^t \\ \hline 0 & 0 & I & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & 0 \\ 0 & 0 & 0 & \cdots & I & 0 \end{array}\right).$$

Replacing G''_{pad} by G'_{pad} does not affect the security of the scheme. This is because the first three rows of G''_{pad} and of G'_{pad} span the same space, i.e., the space of the dual STAR code, with the last three entries in the codeword deleted.

The improved padding matrix reduce the encoding complexity of the padding phase to at most $2(p-2)+6(p-1)+3(p-3)(p-1)$ XORs. Therefore, the overall normalized encoding complexity of the improved scheme is

$$\frac{4(p-2)+6(p-1)+3(p-3)(p-1)+3(p-1)^2}{(p-1)(p-3)} \approx 6$$

XORs per message bit, which is essentially optimal.

4.4.2 Decoding Secure STAR

Next we consider the decoding complexity of secure STAR. In general one can decode by multiplying the codeword vector to the inverse of the generator matrix, but matrix inversion is an expensive operation (requiring $O(n^6)$ XORs). Even if the cost of matrix inversion is amortized (as the inverse can be pre-computed), matrix multiplication is still expensive (requiring $O(n^4)$ XORs). In the following we show that the construction of secure STAR entails a very efficient decoding algorithm, requiring only $O(n^2)$ XORs in total.

The decoding algorithm can be divided into three steps: First, if any of the first p entries in the codeword is erased, recover them by erasure decoding. Secondly, decode the key polynomials $u_1(\alpha), u_2(\alpha), u_3(\alpha)$ and hence all the key bits from $c_1(\alpha), c_2(\alpha), c_p(\alpha)$. Finally, cancel the keys from $c_i(\alpha)$, $i = 3, ..., p-1$ to obtain the message polynomials. For the first step, since the codewords of secure STAR are codewords of the STAR code, recovering the erased symbols is equivalent to recovering from erasures in the STAR code. A major advantage of the STAR code is that it has a very efficient erasure decoding algorithm [20], requiring at most $O(n^2)$ XORs to recover any three erasures. In the following we focus on the latter two steps that deal with the arguably more important issue of "decrypting" the message, as erasure decoding is needed only when erasures occur, but "decryption" is always required whenever one wants to retrieve the information.

We first describe the third step of canceling the keys, which is simply to "re-pad" the keys to the codeword in the same way as they were padded to the messages during the encoding phase. Since the padding scheme G'_{pad} is almost optimal, i.e., most entries in the array are padded by only three key bits, the minimum number of keys to tolerate three eavesdroppers, the complexity of canceling the keys is essentially

optimal. Namely, for most entries in the array, recovering the message bit stored in that entry only requires 3 XORs to cancel the keys.

We now describe the second step of decoding the key polynomials. For ease of notation, denote for short $a_i \triangleq u_{1i}$, $b_i \triangleq u_{2i}$, $c_i \triangleq u_{3i}$, $i = 1, ..., p-1$, and $a_0 \triangleq u_1^*$, $b_0 \triangleq u_2^*$, $c_0 \triangleq u_3^*$ (recall that $u_j^* = \sum_{i=1}^{p-1} u_{ji}$). Then the coefficients of $c_1(\alpha)$ are $a_i + b_i + c_i$, the coefficients of $c_2(\alpha)$ are $a_i + b_{\langle i+1 \rangle} + c_{\langle i-1 \rangle}$ and the coefficients of $c_p(\alpha)$ are $a_i + b_{\langle i-1 \rangle} + c_{\langle i+1 \rangle}$, $i = 1, ..., p-1$. Therefore the coefficients of $c_1(\alpha) + c_2(\alpha)$ are $u_i \triangleq b_i + b_{\langle i+1 \rangle} + c_{\langle i-1 \rangle} + c_i$, and the coefficients of $c_1(\alpha) + c_p(\alpha)$ are $v_i \triangleq b_{\langle i-1 \rangle} + b_i + c_i + c_{\langle i+1 \rangle}$, $i = 1, ..., p-1$.

For $i = 0, ..., p-3$, by XORing $v_{\langle i+1 \rangle}$ and $u_{\langle i+2 \rangle}$ we obtain $w_i = b_i + b_{\langle i+1 \rangle} + b_{\langle i+2 \rangle} + b_{\langle i+3 \rangle}$. Since $b_0 = u_2^* = \sum_{i=1}^{p-1} b_i$, we have $w_0 = \sum_{i=4}^{p-1} b_i$, and $w_{p-3} = \sum_{i=1}^{p-4} b_i$. We consider two cases: Case 1: $p \mod 4 = 1$. Therefore 4 divides $p-5$ and we can combine the w_i's to obtain $\sum_{i=5}^{p-1} b_i$. Canceling it from w_0 we obtain b_4. Similarly, 4 divides $p-9$ and so we can obtain $\sum_{i=6}^{p-4} b_i$. Canceling $\sum_{i=6}^{p-4} b_i$ and w_1 from w_{p-3} we obtain b_5. By symmetry we can also obtain c_4 and c_5. Case 2: $p \mod 4 = 3$. Therefore 4 divides $p-3$ and we can combine the w_i's to obtain $\sum_{i=3}^{p-1} b_i$. Canceling w_0 from it we obtain b_3. Similarly, 4 divides $p-7$ and we can obtain $\sum_{i=4}^{p-4} b_i$. Canceling it from w_{p-3} we obtain $b_1 + b_2 + b_3$. Finally cancel it from w_1 and we obtain b_4. By symmetry we can also obtain c_3 and c_4.

Therefore, there always exists an i so that we can obtain b_i, b_{i+1} and c_i, c_{i+1}. Now cancel b_i, c_i and c_{i+1} from v_i to obtain b_{i-1} and cancel b_{i+1}, c_i and c_{i+1} from u_{i+1} to obtain b_{i+2}. By symmetry we can also obtain c_{i-1} and c_{i+2}. By induction we obtain all b_i, c_i, $i = 1, ..., p-1$. Finally, cancel the b_i's and the c_i's from the coefficients of $c_1(\alpha)$ to obtain a_i, $i = 1, ..., p-1$. This completes the decoding of all key bits.

We summarize the computational complexity of systematic decoding, i.e., the complexity of the second and third steps of the algorithm. The second step requires no more than $18(p-1)$ XORs and the third step requires no more than $3(p-1) + 3(p-3)(p-1)$ XORs. Therefore the normalized decoding complexity is

$$\frac{18(p-1) + 3(p-1) + 3(p-3)(p-1)}{(p-3)(p-1)} \approx 3$$

XORs per message bit. Since every message bit has to be padded by at least three key bits in order to tolerate three eavesdropping nodes, the decoding complexity of the scheme is essentially optimal.

4.5 Secure B

In the previous sections of this chapter we have constructed several families of secure RAID schemes with optimal encoding and decoding complexity; all of them are related to the ring $\mathcal{R}_p$. In this section we construct another family of optimal secure RAID schemes from the B codes [18], which are related to graph factorization.

Specifically, the B codes are equivalent to perfect one-factorization of complete graphs [18]. For any prime p, the perfect one-factorization of K_{p+1}, the complete graph of $p+1$ vertexes, is known [43], and geometrically defines a family of B codes, also equivalent to the codes in [44]. Here we present a simplified algebraic description of this family of B codes.

We start with the dual B codes which are conceptually simpler. For any prime p, let $t = \frac{p-1}{2}$, the dual B code is a $[p-1, 2]$ MDS array code over $\mathbb{F}_2^t$ of minimum distance $p-2$. We refer the reader to Figure 4.3 for an example of the dual B code of $p = 7$ and an informal description of the construction. Let a, b be integers, we denote by $\langle \frac{a}{b} \rangle$ by the unique integer m, $0 \leq m < p$, such that $a \equiv bm \pmod{p}$.

Node 1	Node 2	Node 3	Node 4	Node 5	Node 6
m_1	m_2	m_3	m_4	m_5	m_6
$m_2 \oplus m_6$	$m_4 \oplus m_5$	$m_6 \oplus m_4$	$m_1 \oplus m_3$	$m_3 \oplus m_2$	$m_5 \oplus m_1$
$m_3 \oplus m_5$	$m_6 \oplus m_3$	$m_2 \oplus m_1$	$m_5 \oplus m_6$	$m_1 \oplus m_4$	$m_4 \oplus m_2$

Figure 4.3: Dual B code of length 6. All symbols are binary bits and all operations are XORs. The code is MDS and is able to correct $6 - 2 = 4$ node (column) erasures. Note that each message bit is checked by exactly 4 parities, implying optimal encoding complexity because this is necessary to correct 4 erasures. In general, dual B codes with similar properties can be constructed for any length $p-1$, where p is prime, in the following simple way: node i stores m_i as well as all sums of the form $m_a \oplus m_b$ such that $\langle a + b \rangle = i$.

Construction 4.5.1. (Dual B Code). *Let p be a prime, $t = \frac{p-1}{2}$ and let $m_1, ..., m_{p-1}$ be the message bits. The codewords of the dual B code form a $t \times (p-1)$ array, described by the following encoding mapping. Let $c_{i,j}$ be the (i,j)-th entry of the code array, then $c_{i,j} = m_{\langle i \cdot j \rangle} \oplus m_{\langle (1-i) \cdot j \rangle}$, for $i \in [t]$, $j \in [p-1]$, where $m_0 \stackrel{\text{def}}{=} 0$.*

Note that the first row of the array consists of the systematic symbols, and the 2-nd to t-th rows consist of the parity (i.e., redundant) symbols.

Theorem 4.5.1. *The dual B code is MDS.*

Proof. Note that the dual B codes have dimension $k = 2$ because there are $p - 1$ message bits and $t = \frac{p-1}{2}$. Therefore it suffices to prove that all message bits can be decoded from any two nodes. Suppose the two nodes are node u and v. Then by construction, for $x \in \{u, v\}$, node x stores $\{m_a + m_b | a + b = x, 0 \leq a, b \leq p - 1\}$. Let $i = \langle u/2 \rangle$ and $j = \langle v/2 \rangle$. We describe a path, in which vertexes represent the indexes of the message bits, and edges represent the encoded bits stored in node u and v, i.e., the edge (a, b) represents $m_a + m_b$. The path consists of p vertexes $x_1, ..., x_p$ and $p - 1$ edges, defined as follows. Let the first vertex be $x_1 = i$. Let the odd edges (i.e., the 1-st, 3-rd, ..., $(p - 2)$-th edges of the path) come from node v, i.e., they are elements of $\{(a, b) | a + b = v = 2j\}$, and let the even edges come from node u, i.e., they are elements of $\{(a, b) | a + b = u = 2i\}$. The path is well-defined by construction, namely, $x_2 = 2j - i$, since node v stores $m_i + m_{2j-i}$ and stores no other encoded bits involving m_i; and $x_3 = 3i - 2j$, since node u stores $m_{2j-i} + m_{3i-2j}$ and stores no other encoded bits involving m_{2j-i}. By induction, it is straightforward to see that $\{x_1, ..., x_p\} = \{\langle i + 2a(i - j) \rangle | a = 0, \pm 1, ..., \pm t\}$.

We claim that the path is simple, i.e., $|\{x_1, ..., x_p\}| = p$. Suppose $i + 2a(i - j) \equiv i + 2a'(i - j) \mod p$, then because $i \not\equiv j \mod p$, it follows that $a \equiv a' \mod p$, proving the claim. Because $\mathbb{F}_p$ has exactly p elements, it follows that $\{x_1, ..., x_p\} = \{0, ..., p - 1\}$. Particularly, the path contains a vertex labeled by 0, whose neighbors on the path are vertexes u and v. Since $m_0 = 0$ is known, we cut the path at the vertex 0, obtaining two decoding paths, such that one starts with vertex u, and the other starts with vertex v. Following the decoding paths, all message bits on the path can be decoded one by one by cancellation, starting with canceling m_u and m_v which are stored in the clear. This completes the proof. □

In the $2t \times (p - 1)t$ generator matrix of the dual B code, each row has exactly $p - 2$ 1's. This meets the obvious lower bound on the number of 1's (the dual B code has minimum distance $p - 2$), and therefore the dual B code has a lowest density generator matrix. This matrix is a (systematic) parity check matrix of the B code, from which we can immediately obtain a generator matrix of the B code, by recalling that $[A|I_{(n-k)t}]$ is a parity-check matrix of an $[n, k]$ code $\mathcal{C}$ over $\mathbb{F}_q^t$ if and only if $[I_{kt}| - A^T]$ is a generator matrix of $\mathcal{C}$. We refer the readers to Figure 4.4 for an example of the B code of $p = 7$ and an informal description of the construction.

Construction 4.5.2. (B Code). *Let p be a prime, $t = \frac{p-1}{2}$ and let $m_{i,j}$, $i \in [t - 1]$, $j \in [p - 1]$ be the message bits. The codewords of the B code forms a $t \times (p - 1)$*

Node 1	Node 2	Node 3	Node 4	Node 5	Node 6
$m_{1,1}$	$m_{1,2}$	$m_{1,3}$	$m_{1,4}$	$m_{1,5}$	$m_{1,6}$
$m_{2,1}$	$m_{2,2}$	$m_{2,3}$	$m_{2,4}$	$m_{2,5}$	$m_{2,6}$
$m_{1,4} \oplus m_{1,6} \oplus$ $m_{2,5} \oplus m_{2,3}$	$m_{1,1} \oplus m_{1,5} \oplus$ $m_{2,3} \oplus m_{2,6}$	$m_{1,5} \oplus m_{1,4} \oplus$ $m_{2,1} \oplus m_{2,2}$	$m_{1,2} \oplus m_{1,3} \oplus$ $m_{2,6} \oplus m_{2,5}$	$m_{1,6} \oplus m_{1,2} \oplus$ $m_{2,4} \oplus m_{2,1}$	$m_{1,3} \oplus m_{1,1} \oplus$ $m_{2,2} \oplus m_{2,4}$

Figure 4.4: B code of length 6. All symbols are binary bits and all operations are XORs. The code is MDS and is able to correct 2 node (column) erasures. Note that each message bit is checked by exactly 2 parities, implying optimal encoding complexity because this is necessary to correct 2 erasures. In general, for any prime p, B codes of length $p-1$ can be constructed in the following way: construct the dual B code of length $p-1$ and switch the role of information bits and parity bits. Specifically, in the dual B code (see Figure 4.3), an information bit m_i is checked by $n-2$ parities; in the B code, these $n-2$ parities become information bits, and they are exactly the set of information bits checked by the parity bit of node i.

array, described by the following encoding mapping. The first $t-1$ rows of the array consist of the systematic symbols, i.e., $c_{i,j} = m_{i,j}$, for $i \in [t-1]$, $j \in [p-1]$. The t-th row consists of the redundant symbols, i.e., $c_{t,j} = \bigoplus_{k=1}^{t-1} \left(m_{k,\langle \frac{j}{k+1} \rangle} \oplus m_{k,\langle -\frac{j}{k} \rangle} \right)$, for $j \in [p-1]$.

By Lemma 3.3.1, the B codes are MDS and can correct 2 node erasures. In the $(p-3)t \times (p-1)t$ generator matrix of the B code, each row has exactly three 1's, meeting the obvious lower bound (the B code has minimum distance 3), and therefore the B code has a lowest density generator matrix.

We are ready to describe the $(n = p-1, k = p-5, r = 2, z = 2)$ secure RAID scheme based on the B code.

Construction 4.5.3. (Secure B). *Let p be a prime and $t = \frac{p-1}{2}$. Let $u_1, ..., u_{p-1}$ be the uniformly distributed key bits and let $m_{i,j}$, $i \in [t-2]$, $j \in [p-1]$ be the message bits. The codewords of secure B form a $t \times (p-1)$ array, described by the following encoding mapping. The first row of the array consists of the key symbols, i.e., $c_{1,j} = u_j \oplus u_{\langle 2 \cdot j \rangle} \oplus u_{\langle -j \rangle}$, $j \in [p-1]$. The 2-nd to $(t-1)$-th rows are the systematic message symbols, i.e., $c_{i,j} = u_{\langle (i+1) \cdot j \rangle} \oplus u_{\langle -i \cdot j \rangle} \oplus m_{i-1,j}$, for $i \in [2, t-1]$, $j \in [p-1]$. The t-th row consists of the redundant symbols, which are computed by encoding the B code described in Construction 4.5.2, regarding the first $(t-1)$-rows of the array as information symbols.*

An example of the scheme is shown in Fig. 4.5. On account of the construction method discussed in Section 3.3.1, the construction idea is to let $\mathcal{C}_2$ be the B code and design $\mathcal{C}_1$ to take a form similar to the dual B code, because it is low rate,

Node 1	Node 2	Node 3	Node 4	Node 5	Node 6
$u_1 \oplus u_2 \oplus u_6$	$u_2 \oplus u_4 \oplus u_5$	$u_3 \oplus u_6 \oplus u_4$	$u_4 \oplus u_1 \oplus u_3$	$u_5 \oplus u_3 \oplus u_2$	$u_6 \oplus u_5 \oplus u_1$
$u_3 \oplus u_5 \oplus m_1$	$u_6 \oplus u_3 \oplus m_2$	$u_2 \oplus u_1 \oplus m_3$	$u_5 \oplus u_6 \oplus m_4$	$u_1 \oplus u_4 \oplus m_5$	$u_4 \oplus u_2 \oplus m_6$
$u_\Sigma \oplus u_1 \oplus$ $u_4 \oplus m_3 \oplus m_5$	$u_\Sigma \oplus u_2 \oplus$ $u_1 \oplus m_6 \oplus m_3$	$u_\Sigma \oplus u_3 \oplus$ $u_5 \oplus m_2 \oplus m_1$	$u_\Sigma \oplus u_4 \oplus$ $u_2 \oplus m_5 \oplus m_6$	$u_\Sigma \oplus u_5 \oplus$ $u_6 \oplus m_1 \oplus m_4$	$u_\Sigma \oplus u_6 \oplus$ $u_3 \oplus m_4 \oplus m_2$

Figure 4.5: The (6,2,2,2) secure B scheme. $u_\Sigma = \bigoplus_{i=1}^{p-1} u_i$. The first row stores the (relaxed) systematic key bits, the middle row(s) stores the systematic message bits, and the last row is redundant. Note that the scheme is essentially optimal in key padding, in the sense that only two keys are padded to each entry of the array except the first and the last rows. In the last row three keys are padded (regarding u_Σ as a key), which is only slightly suboptimal. In the first row, while the keys are not stored in the clear, the cost (less than $2p$ XORs) in encoding is marginal especially when amortized over the message bits. Decoding the keys from the first row is efficient, see Algorithm 1.

MDS, and has optimal encoding complexity. However, the dual B code is not contained in the B code, and we need to design $\mathcal{C}_1$ carefully to meet $\mathcal{C}_1 \subset \mathcal{C}_2$ without compromising efficiency. Indeed, the encoding of keys in Construction 4.5.3 is similar to Construction 4.5.1, e.g., compare the third row in Figure 4.3 to the second row in Figure 4.5.

The following lemma gives an explicit expression of the entries in the last row of the array.

Lemma 4.5.1. *In Construction 4.5.3,*

$$c_{t,j} = u_\Sigma \oplus u_j \oplus u_{\langle j/2 \rangle} \oplus \left(\bigoplus_{k=2}^{t-1} \left(m_{k-1,\langle \frac{j}{k+1} \rangle} \oplus m_{k-1,\langle -\frac{j}{k} \rangle} \right) \right),$$

where $u_\Sigma = \bigoplus_{i=1}^{p-1} u_i$.

Proof. We have

$$
\begin{aligned}
c_{t,j} &\overset{(a)}{=} \bigoplus_{k=1}^{t-1}\left(c_{k,\langle\frac{j}{k+1}\rangle}\oplus c_{k,\langle-\frac{j}{k}\rangle}\right)\\
&= c_{1,\langle\frac{j}{2}\rangle}\oplus c_{1,\langle-j\rangle}\oplus\left(\bigoplus_{k=2}^{t-1}\left(c_{k,\langle\frac{j}{k+1}\rangle}\oplus c_{k,\langle-\frac{j}{k}\rangle}\right)\right)\\
&\overset{(b)}{=} u_{\langle j/2\rangle}\oplus u_j\oplus u_{\langle-j/2\rangle}\oplus u_{\langle-j\rangle}\oplus u_{\langle-2j\rangle}\oplus u_j\oplus\left(\bigoplus_{k=2}^{t-1}\left(c_{k,\langle\frac{j}{k+1}\rangle}\oplus c_{k,\langle-\frac{j}{k}\rangle}\right)\right)\\
&= u_{\langle j/2\rangle}\oplus u_{\langle-j/2\rangle}\oplus u_{\langle-j\rangle}\oplus u_{\langle-2j\rangle}\oplus\left(\bigoplus_{k=2}^{t-1}\left(c_{k,\langle\frac{j}{k+1}\rangle}\oplus c_{k,\langle-\frac{j}{k}\rangle}\right)\right)\\
&\overset{(c)}{=} u_{\langle j/2\rangle}\oplus u_{\langle-j/2\rangle}\oplus u_{\langle-j\rangle}\oplus u_{\langle-2j\rangle}\oplus\\
&\qquad\bigoplus_{k=2}^{t-1}\left(u_{\langle\frac{(k+1)j}{k+1}\rangle}\oplus u_{\langle-\frac{kj}{k+1}\rangle}\oplus m_{k-1,\langle\frac{j}{k+1}\rangle}\oplus u_{\langle-\frac{(k+1)j}{k}\rangle}\oplus u_{\langle\frac{kj}{k}\rangle}\oplus m_{k-1,\langle-\frac{j}{k}\rangle}\right)\\
&= u_{\langle j/2\rangle}\oplus u_{\langle-j/2\rangle}+\oplus u_{\langle-j\rangle}\oplus u_{\langle-2j\rangle}\oplus\left(\bigoplus_{k=2}^{t-1}\left(u_{\langle-\frac{kj}{k+1}\rangle}\oplus u_{\langle-\frac{(k+1)j}{k}\rangle}\right)\right)\oplus\\
&\qquad\qquad\bigoplus_{k=2}^{t-1}\left(m_{k-1,\langle\frac{j}{k+1}\rangle}\oplus m_{k-1,\langle-\frac{j}{k}\rangle}\right)\\
&\overset{(d)}{=} u_{\langle j/2\rangle}\oplus u_{\langle-j\rangle}\oplus\left(\bigoplus_{k=1}^{t-1}\left(u_{\langle-\frac{kj}{k+1}\rangle}\oplus u_{\langle-\frac{(k+1)j}{k}\rangle}\right)\right)\oplus\\
&\qquad\qquad\bigoplus_{k=2}^{t-1}\left(m_{k-1,\langle\frac{j}{k+1}\rangle}\oplus m_{k-1,\langle-\frac{j}{k}\rangle}\right)\\
&\overset{(e)}{=} u_{\langle j/2\rangle}\oplus u_{\langle-j\rangle}\oplus u_\Sigma\oplus u_j\oplus u_{\langle-j\rangle}\oplus\left(\bigoplus_{k=2}^{t-1}\left(m_{k-1,\langle\frac{j}{k+1}\rangle}\oplus m_{k-1,\langle-\frac{j}{k}\rangle}\right)\right)\\
&= u_\Sigma\oplus u_j\oplus u_{\langle j/2\rangle}\oplus\left(\bigoplus_{k=2}^{t-1}\left(m_{k-1,\langle\frac{j}{k+1}\rangle}\oplus m_{k-1,\langle-\frac{j}{k}\rangle}\right)\right)
\end{aligned}
$$

where (a) follows from Construction 4.5.2; (b) and (c) follows from Construction 4.5.3; (d) follows from merging $u_{\langle-j/2\rangle}$ and $u_{\langle-2j\rangle}$ into the summation; and (e) follows from the fact that $\bigoplus_{k=1}^{t-1}\left(u_{\langle-\frac{kj}{k+1}\rangle}+u_{\langle-\frac{(k+1)j}{k}\rangle}\right)=u_\Sigma\oplus u_j\oplus u_{\langle-j\rangle}$, which we now prove. Note that $\langle\frac{k}{k+1}\rangle=\langle\frac{k'}{k'+1}\rangle$ implies $\langle k\rangle=\langle k'\rangle$; $\langle\frac{k+1}{k}\rangle=\langle\frac{k'+1}{k'}\rangle$ implies $\langle k\rangle=\langle k'\rangle$; $\langle\frac{k}{k+1}\rangle=\langle\frac{k'+1}{k'}\rangle$ implies that $\langle k+k'\rangle=p-1$, and therefore it follows that in the summation, the $2(t-1)=p-3$ summands are distinct. Denote by J the set of the indexes of the summands, then J contains $1,2,...,p-1$ except two elements. Because $\langle\frac{k}{k+1}\rangle\neq 1$ and $\langle\frac{k+1}{k}\rangle\neq 1$, it follows that $\langle-j\rangle\notin J$. Because $\langle\frac{k}{k+1}\rangle=\langle-1\rangle$ and $\langle\frac{k+1}{k}\rangle=\langle-1\rangle$ both imply that $\langle k\rangle=t$, it follows that $j\notin J$. Hence

$J = [p-1]\backslash\{j, \langle -j\rangle\}$, implying $\bigoplus_{k=1}^{t-1}\left(u_{\langle -\frac{kj}{k+1}\rangle} \oplus u_{\langle -\frac{(k+1)j}{k}\rangle}\right) = u_\Sigma \oplus u_j \oplus u_{\langle -j\rangle}$. This completes the proof. □

Theorem 4.5.2. *Secure B is a* $(p-1, p-5, 2, 2)$ *secure RAID scheme over* $\mathbb{F}_2^t$*, for any prime* p *and* $t = \frac{p-1}{2}$*. In particular, the density of the key rows of the generator matrix is* $2p-5$*, and the density of the message rows is* 3*.*

Proof. Note that $t = \frac{p-1}{2}$ and the number of message bits is $(t-2)(p-1)$. So $k = (t-2)(p-1)/t = p-5$, and the scheme is rate-optimal.

Since the codewords of secure B are codewords of the B code, any two column erasures can be corrected. We focus on proving that the scheme is secure. By Lemma 4.3.1, it suffices to show that given the message bits and any two columns of the array, all keys can be decoded. For $j \in [p-1]$, consider the encoding of keys, denoted by $\{c'_{i,j} : i \in [t]\}$, in the j-th column. By Construction 4.5.3 and Lemma 4.5.1, we have

$$c'_{i,j} = \begin{cases} u_j \oplus u_{\langle 2j\rangle} \oplus u_{\langle -j\rangle} & i = 1 \\ u_{\langle (i+1)\cdot j\rangle} \oplus u_{\langle -i\cdot j\rangle} & i = 2, \cdots, t-1 \\ u_\Sigma \oplus u_j \oplus u_{\langle j/2\rangle} & i = t. \end{cases} \tag{4.21}$$

Encode the keys $u_1, \cdots, u_{p-1}$ as information by the dual B code, then the entries in the j-th columns are

$$c''_{i,j} = u_{\langle i\cdot j\rangle} \oplus u_{\langle (1-i)\cdot j\rangle} \quad i = 1, \cdots, t. \tag{4.22}$$

Clearly, $c''_{i,j} = c'_{i-1,j}$, $i = 3, \cdots, t-1$. Note that

$$\bigoplus_{i=2}^{t-1} c'_{i,j} = u_\Sigma + u_j + u_{2j} + u_{\langle -j\rangle} + u_{\langle j/2\rangle}. \tag{4.23}$$

Therefore, $c''_{1,j} = u_j = \bigoplus_{i=1}^{t} c'_{i,j}$, and $c''_{2,j} = u_{\langle 2j\rangle} + u_{\langle -j\rangle} = \bigoplus_{i=2}^{t} c'_{i,j}$. It follows that $\{c''_{i,j} : i \in [t]\}$ are functions of $\{c'_{i,j} : i \in [t]\}$. Since the dual B code is MDS, the u_i's can be decoded from any two columns of the $c''_{i,j}$'s, and so can be decoded from any two columns of the $c'_{i,j}$'s. This completes the proof that secure B is a $(p-1, p-5, 2, 2)$ secure RAID scheme.

We now analyze the density of the generator matrix G. We say a key u_i (or a message bit $m_{i,j}$) is *checked* by $c_{a,b}$ if in G the entry corresponding to u_i (or $m_{i,j}$) and $c_{a,b}$ is 1. By construction, u_i is checked by $c_{t,b}$ for $b = 1, ..., p-1$, $b \neq i, \langle 2i\rangle$, and is checked by exactly one element of $\{c_{a,1}, ..., c_{a,t-1}\}$ for $a = 1, ..., p-1$, $a \neq \langle 2i\rangle$.

Therefore u_i is checked for exactly $p-2+p-3 = 2p-5$ times. A message bit $m_{i,j}$ is checked by $c_{i+1,j}$, $c_{t,\langle(i+2)j\rangle}$ and $c_{t,\langle-(i+1)j\rangle}$. Therefore $m_{i,j}$ is checked for exactly 3 times. This completes the proof. □

On account of the proof of Theorem 4.5.2, we can interpret the secure B scheme under the framework of Theorem 3.3.1, where $\mathcal{C}_2$ is the B code, and $\mathcal{C}_1$ is a variant of the dual B code.

By Theorem 3.2.2, a lower bound on the density of the key rows is $p-2$ and a lower bound on the density of the message rows is 3. Therefore for the message rows, the scheme achieves the lowest density. For the key rows, the scheme achieves the lower bound within a factor of 2.

Algorithm 1 $\boldsymbol{m}$ = Dec($\boldsymbol{c}$); Systematic Decoding.

1: **for** $i \leftarrow 1$ to t **do** ▷ Decode keys from $c_{1,j}$, $j \in [p-1]$. Recall that $t = \frac{p-1}{2}$.
2: $x \leftarrow c_{1,\langle i/4\rangle} \oplus c_{1,\langle -i/4\rangle}$ ▷ $x = u_{\langle i/2\rangle} + u_{\langle -i/2\rangle}$
3: $u_i \leftarrow c_{1,\langle i/2\rangle} \oplus x$
4: $u_{-i} \leftarrow c_{1,\langle -i/2\rangle} \oplus x$
5: **end for** ▷ All keys have been decoded.
6: **for** $i \leftarrow 2$ to $t-1$ and $j \leftarrow 1$ to $p-1$ **do**
7: $m_{i-1,j} \leftarrow c_{i,j} \oplus u_{\langle(i+1)\cdot j\rangle} \oplus u_{\langle -i\cdot j\rangle}$ ▷ Cancel keys to obtain message bits.
8: **end for**

Algorithm 1 describes a systematic decoding algorithm for the scheme. In the case of erasures/error, the erasure/error decoding algorithm of the B code [18] is invoked to correct the erasures, and then Algorithm 1 is invoked to decode the message. Encoding the scheme according to Construction 4.5.3 requires a total number of $2p^2 - 9p + 7$ XORs, or on average $4 + \frac{6}{p-5}$ XORs per message bit. Systematic decoding the scheme according to Algorithm 1 requires a total number of $p^2 - \frac{9}{2}p + \frac{7}{2}$ XORs, or on average $2 + \frac{3}{p-5}$ XORs per message bit. By Corollary 3.2.1, encoding each message bit requires at least $4 + \frac{2}{p-5}$ XORs, and decoding each message bit requires at least 2 XORs. Therefore the encoding and decoding complexity of secure B is essentially optimal.

4.6 Optimal Secure B

So far the secure RAID schemes constructed in this chapter are almost optimal in terms of encoding, decoding and the density of generator matrix. Nevertheless, it remains an interesting problem whether the gap can be closed, and whether there

exist schemes achieving the bounds in Theorem 3.2.2 and Corollary 3.2.1 strictly. In this section we provide a partial answer to this question.

Specifically, we are able to construct strictly optimal $(p-1, p-5, 2, 2)$ secure RAID schemes for any prime p ranging from 7 to 53. We conjecture that such strictly optimal schemes exist for an infinite sequence of primes p. Like Section 4.5, the construction is based on the B codes. Unlike Section 4.5, which slightly changes the encoding of the dual B code for key padding, in this section we will preserve strictly the encoding of the dual B code and its optimality. Then, in order to meet the containment condition, we resort to permutation.

Definition 4.6.1. *Let p be a prime, $t = \frac{p-1}{2}$, and let $\sigma : [t] \to [t]$ be a permutation. We say σ is proper with respect to p if $\sigma(1) \neq t$ and that for every codeword $C = (c_{i,j})$ of the dual B code, $\left(c_{\sigma(i),j}\right)$ is a codeword of the B code.*

Construction 4.6.1. (Optimal Secure B) *Let p be a prime, $t = \frac{p-1}{2}$, and let $\sigma : [t] \to [t]$ be a proper permutation with respect to p. Let $u_1, ..., u_{p-1}$ be uniformly distributed key bits. The codewords of optimal secure B form a $t \times (p-1)$ array. The first $t-1$ rows of the array are the systematic key and message symbols, computed as follows. Denote by $C' = c'_{i,j}$ the codeword of the dual B code computed by encoding the keys as information symbols and denote $i^* = \sigma(1)$, then $c_{i^*,j} = c'_{1,j} = u_j$, $j \in [p-1]$; for $i \neq i^*, i \in [t-1]$, $j \in [p-1]$, $c_{i,j} = c'_{\sigma(i),j} \oplus m_{i,j}$, where the $m_{i,j}$'s are the message bits. The t-th row consists of the redundant symbols, which are computed by encoding the B code regarding the first $(t-1)$-rows of the array as information symbols.*

An example of the optimal secure B schemes is shown in Figure 2.2. The proper permutation (in cycle representation [45]) is $\sigma = (1)(2,3)$. It would be helpful to compare Figure 2.2 to Figure 4.3 and Figure 4.4 to see the effect of σ.

Theorem 4.6.1. *Construction 4.6.1 is a $(p-1, p-5, 2, 2)$ secure RAID scheme over $\mathbb{F}_2^t$. In particular, the key rows of the generator matrix have optimal density $p-2$, and the message rows have optimal density 3.*

Proof. Interpreting the scheme using the method described in Section 3.3.1, then $\mathcal{C}_1$ is the dual B code in which the rows of the codeword array are permuted according to σ, and $\mathcal{C}_2$ is the B code. By Corollary 3.3.1 the scheme is a $(p-1, p-5, 2, 2)$ secure RAID scheme.

By the optimality of the dual B code, each key bit appears in exactly $p-2$ of the $c_{i,j}$'s, and by the optimality of the B code 4.5.2, each message bit appears in exactly 3 of the $c_{i,j}$'s. Therefore each key row has density $p-2$ and each message row has density 3, meeting the lower bound in Theorem 3.2.2 and proving the theorem. □

Encoding Construction 4.6.1 requires $4+\frac{2}{p-5}$ XORs per message bit and achieves the lower bound of Corollary 3.2.1. Systematic decoding the scheme by first reading the keys and then canceling them from the systematic message symbols requires 2 XORs per message bit, again achieving the trivial lower bound. Therefore Construction 4.6.1 has optimal encoding and systematic decoding complexity.

It remains to address whether a proper permutation σ exists and how to construct it. We are not aware of a method to construct proper permutations for arbitrary prime p. However, for an arbitrary permutation σ, the following result is useful in determining whether σ is proper.

Lemma 4.6.1. *Let p be a prime, $t=\frac{p-1}{2}$, and let $\sigma:[t]\to[t]$ be a permutation such that $\sigma(1)=i^*\neq t$. Consider five multisets $A_1=\{\langle\frac{\sigma^{-1}(i)}{i+1}\rangle : i\in[t-1], i\neq i^*\}$, $A_2=\{\langle\frac{1-\sigma^{-1}(i)}{i+1}\rangle : i\in[t-1], i\neq i^*\}$, $A_3=\{\langle-\frac{\sigma^{-1}(i)}{i}\rangle : i\in[t-1], i\neq i^*\}$, $A_4=\{\langle\frac{\sigma^{-1}(i)-1}{i}\rangle : i\in[t-1], i\neq i^*\}$ and $A_5=\cup_{i=1}^4 A_i\cup\{\langle\frac{1}{i^*+1}\rangle,\langle-\frac{1}{i^*}\rangle\}$. Then σ is proper with respect to p if and only if $\sigma^{-1}(t)$ and $\langle 1-\sigma^{-1}(t)\rangle$ are elements of A_5 with odd multiplicity and all other elements of A_5 have even multiplicity.*

The lemma can be proved by verifying Definition 4.6.1 according to Construction 4.5.1 and 4.5.2. With Lemma 4.6.1 we can easily check whether a given σ is proper or not. Therefore a proper σ with respect to a given p, if exists, can be found by exhaustive search. Proper σ with respect to $7\leq p\leq 53$ are listed in Table 4.1. We conjecture that proper σ exists with respect to an infinite sequence of p.

p	σ
7	(1) (2 3)
11	(1 4 2) (3) (5)
13	(1 5 3) (2) (4) (6)
17	(1) (2 8 3 6 4 7) (5)
19	(1 2) (3 9 8 4) (5 7) (6)
23	(1) (2 11 10 3 4 9 8 7 6 5)
29	(1) (2 14) (3 13 12 11 10 7 5 4) (6) (8 9)
31	(1) (2 15 12 11 6 5) (3 4) (7 10 9 8) (13 14)
37	(1 3 8 5 4 18 17 16 15 14 11 10 9 2) (6 7) (12 13)
41	(1 9 8 7 6 5 4) (2 3) (10 20 17 14 13 12 11) (15 16) (18 19)
43	(1 15 14 13) (2 12 11 10) (3 9 8 7 18 17 16 21 20 19 6 5) (4)
47	(1 17 9 15 5 4 3 2) (6 14 13 12 7) (8 11 10 16) (18 23 22 21 20) (19)
53	(1 5 4 3 18 8 7 15 14 13 12 24 23 10 9 17 16 6 26) (2 25 11 22 21 20 19)

Table 4.1: Table of proper permutations with respect to p. We use the cycle representation of permutations [45]. We note that proper permutations may not be unique and this table lists only one of the proper permutation(s) with respect to a specific p.

Chapter 5

SCHEMES OF ARBITRARY RATE OVER RING

In the previous chapter we construct several secure RAID schemes with optimal or essentially optimal encoding and decoding complexity. These schemes are high rate and can tolerate $r \leq 3$ erasures and $z \leq 3$ eavesdroppers. In this chapter we turn to designing highly efficient schemes with general parameters. We start with diving into a special ring $\mathcal{R}_p$, over which the schemes will be constructed.

5.1 The $\mathcal{R}_p$ Ring

Let p be a prime, and let $M_p(x) = \sum_{i=0}^{p-1} x^i$ be a polynomial over $GF(2)$. Let $\mathcal{R}_p$ be the ring of polynomials of degree less than $p-1$ over $GF(2)$ with multiplication taken modulo $M_p(x)$. We shall use the indeterminate α to refer to polynomials in $\mathcal{R}_p$, and reserve the indeterminate x for polynomials in $\mathbb{F}_2[x]$. Secure EVENODD and secure STAR are both linear schemes over $\mathcal{R}_p$, in which the coefficients of an element of $\mathcal{R}_p$ are the $p-1$ bits stored by a node. It is well known that $\mathcal{R}_p$ has a nice structure that is advantageous to both computation and implementation [34]. In this section we review and summarize the important properties of $\mathcal{R}_p$, which will be useful later in the design and analysis of efficient secure RAID schemes tolerating an arbitrary number of erasures and eavesdropping nodes.

Lemma 5.1.1. *Given $a(x) = \sum_{i=0}^{p-2} a_i x^i$, let*

$$b(x) = \sum_{i=0}^{p-2} b_i x^i \equiv x^j a(x) \mod M_p(x). \tag{5.1}$$

Namely, $b(\alpha) = \alpha^j a(\alpha)$. Then b_i, $i = 0, \cdots, p-2$ can be computed by Algorithm 2 using no more than p XORs.

Proof. Let $c(x) = \sum_{i=0}^{p-1} c_i x^i \in \mathbb{F}_2[x]$ be the polynomial of degree $< p$ such that

$$c(x) \equiv x^j a(x) \mod x^p - 1. \tag{5.2}$$

Denote $\boldsymbol{a} = (a_0, \cdots, a_{p-1})$, where $a_{p-1} \triangleq 0$. Then $(c_0, \cdots, c_{p-1})$ is a cyclic shift of $\boldsymbol{a}$ to the right by j positions, namely $c_i = a_{\langle i-j \rangle_p}$. Since $x^p - 1 = (x+1)M_p(x)$, it follows that

$$b(x) \equiv c(x) \mod M_p(x). \tag{5.3}$$

Algorithm 2 Compute $b(\alpha) = \alpha^j a(\alpha)$

Input: $a_0, \cdots, a_{p-2}, j$
Output: $b_0, \cdots, b_{p-2}$
1: $a_{p-1} \leftarrow 0$
2: Cyclic shift $(a_0, \cdots, a_{p-1})$ to the right by j positions.
3: **for** $i \leftarrow 0$ to $p-2$ **do**
4: $\quad b_i \leftarrow a_i \oplus a_{p-1}$
5: **end for**

Therefore $b(x) = c(x) - c_{p-1}M_p(x)$, implying that for $i = 0, \cdots, p-2$,

$$b_i = c_i - c_{p-1} \tag{5.4}$$

$$= a_{\langle i-j \rangle_p} - a_{\langle -j-1 \rangle_p}. \tag{5.5}$$

This completes the proof[1]. □

Lemma 5.1.1 shows that multiplying an element in $\mathcal{R}_p$ by a power of α can be computed efficiently. Notice that $x^p - 1 = (x-1)M_p(x)$, implying that $\alpha^p = 1$. Therefore α^i has a multiplicative inverse α^{p-i}, also denoted by α^{-i}.

Lemma 5.1.2. *$\alpha^i - \alpha^j$ has a multiplicative inverse in $\mathcal{R}_p$ if $i \not\equiv j \mod p$.*

Proof. Let $l = \langle i - j \rangle_p \neq 0$, notice that

$$\gcd(x^l - 1, x^p - 1) = x^{\gcd(l,p)} - 1 = x - 1. \tag{5.6}$$

Since p is an odd prime, $M_p(1) \neq 0$. Therefore

$$\gcd(x - 1, M_p(x)) = 1. \tag{5.7}$$

(5.6) and (5.7) imply that $\gcd(x^l - 1, M_p(x)) = 1$ and so $\alpha^l - 1$ has a multiplicative inverse in $\mathcal{R}_p$. Therefore $\alpha^j(\alpha^l - 1) = \alpha^i - \alpha^j$ also as a multiplicative inverse. □

Denote the multiplicative inverse of $\alpha^i - \alpha^j$ by $(\alpha^i - \alpha^j)^{-1}$.

Lemma 5.1.3. *Let $a(\alpha) = \sum_{i=0}^{p-2} a_i \alpha^i$, and define $c_i = a_i + \sum_{j=0}^{p-1} a_j$, $i = 0, \cdots, p-1$, where $a_{p-1} \triangleq 0$. Then for $l \not\equiv 0 \mod p$, the coefficients of*

$$b(\alpha) = \sum_{i=0}^{p-2} b_i \alpha^i = \frac{a(\alpha)}{\alpha^l - 1} \tag{5.8}$$

[1] Note that since we are working over bits, $+$ and $-$ both refer to the operation of XOR, which is denoted by $\oplus$ in the algorithm.

are given by the recursion

$$b_{\langle -kl-1\rangle_p} = b_{\langle -(k-1)l-1\rangle_p} + c_{\langle -(k-1)l-1\rangle_p}, \tag{5.9}$$

for $k = 1, \cdots, p-1$*, where* $b_{p-1} \triangleq 0$*.*

Proof. By (5.8) we have $(\alpha^l - 1)b(\alpha) = a(\alpha)$ and so by Lemma 5.1.1 it follows that, for $i = 0, \cdots, p-1$

$$a_i = b_{\langle i-l\rangle_p} - b_{\langle -l-1\rangle_p} - b_i. \tag{5.10}$$

Summing both sides of (5.10) over $0 \leq i \leq p-1$, we obtain

$$\sum_{i=0}^{p-1} a_i = b_{\langle -l-1\rangle_p}. \tag{5.11}$$

This proves the lemma for $k = 1$. Note that (5.10) and (5.11) imply that

$$b_{\langle i-l\rangle_p} = b_{\langle -l-1\rangle_p} + b_i + a_i \tag{5.12}$$

$$= b_i + c_i. \tag{5.13}$$

Therefore the case of $k > 1$ follows by substituting

$$i = \langle -(k-1)l - 1\rangle_p$$

into (5.13). □

Note that because $\mathbb{F}_p$ is a field, for $l \not\equiv 0 \mod p$, $\{\langle -kl-1\rangle_p : k = 0, \cdots, p-1\} = \{0, \cdots, p-1\}$. Therefore all coefficients of $b(\alpha)$ are obtained by (5.9).

Corollary 5.1.1. *Let* $a(\alpha) = \sum_{i=0}^{p-2} a_i\alpha^i$*, and* c_i *be as in Lemma 5.1.3. Then for* $l \not\equiv m \mod p$*, the coefficients of*

$$b(\alpha) = \sum_{i=0}^{p-2} b_i\alpha^i = \frac{a(\alpha)}{\alpha^l - \alpha^m} \tag{5.14}$$

are given by the recursion

$$b_{\langle -k(l-m)-1\rangle_p} = b_{\langle -(k-1)(l-m)-1\rangle_p} + c_{\langle -(k-1)(l-m)+m-1\rangle_p},$$

for $k = 1, \cdots, p-1$*, where* $b_{p-1} \triangleq 0$*.*

Proof. By (5.14) we have

$$\alpha^{-m}(\alpha^l - \alpha^m)b(\alpha) = (\alpha^{l-m} - 1)b(\alpha) = \alpha^{-m}a(\alpha), \tag{5.15}$$

implying that

$$b(\alpha) = \frac{\alpha^{-m}a(\alpha)}{\alpha^{l-m} - 1}.$$

Now apply Lemma 5.1.3 and the corollary is proved. □

Example 5.1.1. *The coefficients of* $b(\alpha) = a(\alpha)/(\alpha^4 - \alpha)$ *in* $\mathcal{R}_7$ *are given by*

$$\begin{aligned} b_6 &= 0 \\ b_3 &= b_6 + c_0 \\ b_0 &= b_3 + c_4 \\ b_4 &= b_0 + c_1 \\ b_1 &= b_4 + c_5 \\ b_5 &= b_1 + c_2 \\ b_2 &= b_5 + c_6. \end{aligned}$$

Corollary 5.1.1 shows that division by $\alpha^l - \alpha^m$ in $\mathcal{R}_p$, formalized in Algorithm 3, can be efficiently computed using no more than $3p$ XORs. We remark that in the algorithm the calculations of indexes modulo p do not depend on the input bits $a_0, \cdots, a_{p-2}$. Therefore for fixed l and m, the indexes can be precomputed. Furthermore, a simple and efficient circuit design for computing the indexes is presented in [34].

Algorithm 3 Compute $b(\alpha) = a(\alpha)/(\alpha^l - \alpha^m)$

Input: $a_0, \cdots, a_{p-2}, l, m$
Output: $b_0, \cdots, b_{p-2}$
1: $a_{p-1} \leftarrow 0$, $b_{p-1} \leftarrow 0$
2: $a^* \leftarrow \bigoplus_{i=0}^{p-1} a_i$
3: **for** $k \leftarrow 1$ to $p-1$ **do**
4: $\quad b_{\langle -k(l-m)-1\rangle_p} \leftarrow b_{\langle -(k-1)(l-m)-1\rangle_p} \oplus a^* \oplus a_{\langle -(k-1)(l-m)+m-1\rangle_p}$
5: **end for**

5.2 Construction from Cauchy Matrices

In this section we propose an efficient secure RAID scheme of general parameters over $\mathcal{R}_p$ from Cauchy matrices. Consider any n, r and z such that $n > r + z$, let $k = n - r - z$ and let $p > n$ be a prime. We describe the encoding of a linear

(n, k, r, z) secure RAID scheme over $\mathcal{R}_p$, i.e., every node stores $p-1$ bits. Let the key polynomials $u_i(\alpha)$, $i \in [z]$ be selected uniformly at random from $\mathcal{R}_p$, and let $m_i(\alpha) \in \mathcal{R}_p$, $i \in [k]$ be the message polynomials. To simplify notation, in this chapter we denote polynomials like $u_i(\alpha)$ and $m_i(\alpha)$ for short as u_i and m_i when no confusion arises. The encoding process includes three steps.

Step 1: Compute the padding polynomials

$$(v_1, \cdots, v_k) = (u_1, \cdots, u_z)PCQ. \tag{5.16}$$

Here $P = \text{diag}(p_0, \cdots, p_{z-1})$ and $Q = \text{diag}(q_0, \cdots, q_{k-1})$ are two diagonal matrices given by

$$p_i = \prod_{j=0, j\neq i}^{z-1} \frac{1}{(\alpha^i - \alpha^j)}, \quad i = 0, \cdots, z-1 \tag{5.17}$$

$$q_i = \prod_{j=0}^{z-1} (\alpha^{z+i} - \alpha^j), \quad i = 0, \cdots, k-1. \tag{5.18}$$

And C is a z by k Cauchy matrix

$$C = \begin{pmatrix} \frac{1}{1-\alpha^z} & \frac{1}{1-\alpha^{z+1}} & \cdots & \frac{1}{1-\alpha^{z+k-1}} \\ \frac{1}{\alpha-\alpha^z} & \frac{1}{\alpha-\alpha^{z+1}} & \cdots & \frac{1}{\alpha-\alpha^{z+k-1}} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{1}{\alpha^{z-1}-\alpha^z} & \frac{1}{\alpha^{z-1}-\alpha^{z+1}} & \cdots & \frac{1}{\alpha^{z-1}-\alpha^{z+k-1}} \end{pmatrix}. \tag{5.19}$$

Step 2: Compute partial codeword (the systematic part)

$$c_i = u_i, \quad i = 1, \cdots, z \tag{5.20}$$

$$c_{z+i} = v_i + m_i, \quad i = 1, \cdots, k. \tag{5.21}$$

Step 3: Compute the remaining part of the codeword

$$(c_{z+k+1}, \cdots, c_n) = (c_1, \cdots, c_{z+k})P'C'Q'. \tag{5.22}$$

Here $P' = \text{diag}(p'_0, \cdots, p'_{z+k-1})$ and $Q = \text{diag}(q'_0, \cdots, q'_{r-1})$ are two diagonal matrices given by

$$p'_i = \prod_{j=0, j\neq i}^{z+k-1} \frac{1}{(\alpha^i - \alpha^j)}, \quad i = 0, \cdots, z+k-1 \tag{5.23}$$

$$q'_i = \prod_{j=0}^{z+k-1} (\alpha^{z+k+i} - \alpha^j), \quad i = 0, \cdots, r-1. \tag{5.24}$$

And C' is a $z+k$ by r Cauchy matrix

$$C' = \begin{pmatrix} \frac{1}{1-\alpha^{z+k}} & \frac{1}{1-\alpha^{z+k+1}} & \cdots & \frac{1}{\alpha^0-\alpha^{n-1}} \\ \frac{1}{\alpha-\alpha^{z+k}} & \frac{1}{\alpha-\alpha^{z+k+1}} & \cdots & \frac{1}{\alpha-\alpha^{n-1}} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{1}{\alpha^{z+k-1}-\alpha^{z+k}} & \frac{1}{\alpha^{z+k-1}\alpha^{z+k+1}} & \cdots & \frac{1}{\alpha^{z+k-1}-\alpha^{n-1}} \end{pmatrix}. \tag{5.25}$$

The resulting codeword is $(c_1, \cdots, c_n)$, i.e., node i stores the $p-1$ bits corresponding to the coefficients of $c_i(\alpha)$. Note that the scheme is rate-optimal as $k = n - r - z$.

We remark that the scheme is systematic: the key polynomials are stored uncoded in the first z nodes (refer to (5.20)), and the $(z+1)$-th to $(z+k)$-th nodes each stores the sum of a single message polynomial and a function of the key polynomials (refer to (5.21)). The systematic form helps reduce encoding and decoding complexity as (5.20) and (5.21) can be computed easily. The systematic form also allows efficient random access: in the case that one wishes to decode a single message m_i instead of all messages, it suffices to read the keys from the first z nodes, then compute v_i according to (5.16), and finally cancel v_i from c_i to obtain m_i. In comparison, complete decoding of all messages is required for non-systematic schemes which induces significant read/communication/computation overheads.

We analyze the encoding complexity of the scheme. Focusing on Step 1, $(u_1, \cdots, u_z)$ is first multiplied with P, then with C, and finally with Q. The first and the last multiplications can be efficiently computed because P and Q are (sparse) diagonal matrices. The second multiplication is also efficient because every entry in C has the form $(\alpha^l - \alpha^m)^{-1}$ and by Algorithm 3 multiplication of this form can be computed efficiently. Specifically, by Algorithm 2 and 3, the first multiplication takes no more than $3z^2p$ XORs, the second multiplication takes no more than $3zkp$ XORs and the last multiplication takes no more than $3zkp$ XORs. Therefore Step 1 takes no more than $3z(z+2k)p$ XORs. Step 2 clearly takes no more than kp XORs. By a similar analysis, Step 3 takes no more than $3(z+k)(z+k+2r)p$ XORs. Overall, the encoding complexity of the scheme is no more than $6(z+k)np$ XORs. We remark that the encoding complexity is amortized over $k(p-1)$ message bits, and so the normalized encoding complexity is at most $\frac{6(z+k)np}{k(p-1)}$ XORs per message bit. Particularly, in the high rate case that k dominates z, the normalized encoding complexity is $O(6n)$ XORs per message bit.

5.2.1 Correctness

We first prove a connection between Vandermonde matrices and Cauchy matrices. This connection is well known when the entries of the matrix are from a field, see, e.g., [46]. Below we generalize the result to matrix over rings.

Theorem 5.2.1. *Let $\alpha_1, \cdots, \alpha_m$ and $\beta_1, \cdots, \beta_n$ be distinct elements of a (commutative) ring $\mathcal{R}$, such that $\alpha_i - \alpha_j$ is a unit (i.e., has a multiplicative inverse in $\mathcal{R}$) for distinct $i, j \in [m]$ and that $\alpha_i - \beta_j$ is a unit for $i \in [m], j \in [n]$. Let $p_i = \prod_{j\in[m],j\neq i}(\alpha_i - \alpha_j)^{-1}$ for $i \in [m]$, and let $q_i = \prod_{j\in[m]}(\beta_i - \alpha_j)$, for $i \in [n]$. Define diagonal matrices $P = diag(p_1, \cdots, p_m)$ and $q = diag(q_1, \cdots, q_n)$. Define an m by n Cauchy matrix $C = (c_{i,j})$, where $c_{i,j} = (\alpha_i - \beta_j)^{-1}$, for $i \in [m]$, $j \in [n]$. Then*

$$PCQ = -V_1^{-1}V_2, \tag{5.26}$$

where V_1 is an m by m Vandermonde matrix in which the (i, j)-th entry equals α_j^{i-1}, $i, j \in [m]$, V_1^{-1} is the inverse of V_1, and V_2 is a m by n Vandermonde matrix in which the (i, j)-th entry equals β_j^{i-1}, $i \in [m], j \in [n]$.

Proof. For $i \in [m]$, define functions

$$f_i(x) = \frac{f(x)}{f'(\alpha_i)(x - \alpha_i)}, \tag{5.27}$$

where $f(x) = \prod_{i=1}^{m}(x - \alpha_i)$. Note that

$$f'(x) = \sum_{j=1}^{m} \prod_{k\in[m],k\neq j} (x - \alpha_k). \tag{5.28}$$

Therefore $f'(\alpha_i) = \prod_{j\in[m],j\neq i}(\alpha_i - \alpha_j)$. Since by hypothesis all factors in the denominator of (5.27) are units, it follows that $f_i(x)$ is well-defined, and that $f_i(\alpha_i) = 1$ and $f_i(\alpha_j) = 0$ for $j \neq i$. Write $f_i(x) = \sum_{j=1}^{m} f_{i,j}x^{j-1}$, where $f_{i,j} \in \mathcal{R}$, and define a $m \times m$ matrix F in which the (i, j)-th entry equals $f_{i,j}$, then it follows that

$$\begin{aligned} FV_1 &= \begin{pmatrix} f_{1,1} & \cdots & f_{1,m} \\ \vdots & \vdots & \vdots \\ f_{m,1} & \cdots & f_{m,m} \end{pmatrix} \begin{pmatrix} \alpha_1^0 & \cdots & \alpha_m^0 \\ \vdots & \vdots & \vdots \\ \alpha_1^{m-1} & \cdots & \alpha_m^{m-1} \end{pmatrix} \\ &= \begin{pmatrix} f_1(\alpha_1) & \cdots & f_1(\alpha_m) \\ \vdots & \vdots & \vdots \\ f_m(\alpha_1) & \cdots & f_m(\alpha_m) \end{pmatrix} = I. \end{aligned} \tag{5.29}$$

Particularly, we have $F = V_1^{-1}$. Therefore, the j-th column of $V_1^{-1}V_2$ equals

$$F(\beta_j^0, \cdots, \beta_j^{m-1})^T = (f_1(\beta_j), \cdots, f_m(\beta_j))^T, \tag{5.30}$$

where by construction, for $i \in [m]$,

$$f_i(\beta_j) = \frac{\prod_{k\in[m],k\neq i}(\beta_j - \alpha_k)}{\prod_{k\in[m],k\neq i}(\alpha_i - \alpha_k)}. \tag{5.31}$$

Now notice that the (i, j)-th entry of PCQ equals

$$PCQ(i,j) = p_i C(i,j) q_j \tag{5.32}$$

$$= \frac{\prod_{k\in[m]}(\beta_j - \alpha_k)}{(\alpha_i - \beta_j)\prod_{k\in[m],k\neq i}(\alpha_i - \alpha_k)} \tag{5.33}$$

$$= -f_i(\beta_j). \tag{5.34}$$

The proof is complete. □

On account of (5.16), define another diagonal matrix $\bar{Q} = \text{diag}(\bar{q}_0, \cdots, \bar{q}_{k+r-1})$ where for $i \in [0, k + r - 1]$,

$$\bar{q}_i = \prod_{j=0}^{z-1}(\alpha^{z+i} - \alpha^j). \tag{5.35}$$

Note that the first k diagonal entries of $\bar{Q}$ are identical to those of Q. Define a z by $k + r$ Cauchy matrix

$$\bar{C} = \begin{pmatrix} \frac{1}{1-\alpha^z} & \frac{1}{1-\alpha^{z+1}} & \cdots & \frac{1}{1-\alpha^{z+k+r-1}} \\ \frac{1}{\alpha-\alpha^z} & \frac{1}{\alpha-\alpha^{z+1}} & \cdots & \frac{1}{\alpha-\alpha^{z+k+r-1}} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{1}{\alpha^{z-1}-\alpha^z} & \frac{1}{\alpha^{z-1}-\alpha^{z+1}} & \cdots & \frac{1}{\alpha^{z-1}-\alpha^{z+k+r-1}} \end{pmatrix}. \tag{5.36}$$

Note that C consists of the first k columns of $\bar{C}$. The following result relates matrix $P\bar{C}\bar{Q}$ to $P'C'Q'$.

Lemma 5.2.1. *The row space of $[I_{z+k}|P'C'Q']$ contains the row space of $[I_z|P\bar{C}\bar{Q}]$, where I_n is the identity matrix of order n.*

Proof. By Lemma 5.1.2, the condition of Theorem 5.2.1 is met and so it follows by the theorem that $P\bar{C}\bar{Q} = V_1^{-1}V_2$, and $P'C'Q' = V_3^{-1}V_4$, where

$$V_1 = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & \alpha & \cdots & \alpha^{z-1} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & \alpha^{z-1} & \cdots & \alpha^{(z-1)(z-1)} \end{bmatrix} \tag{5.37}$$

$$V_2 = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \alpha^{z} & \alpha^{z+1} & \cdots & \alpha^{z+k+r-1} \\ \vdots & \vdots & \vdots & \vdots \\ \alpha^{z\cdot(z-1)} & \alpha^{(z+1)(z-1)} & \cdots & \alpha^{(z+k+r-1)(z-1)} \end{bmatrix} \tag{5.38}$$

$$V_3 = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & \alpha & \cdots & \alpha^{z+k-1} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & \alpha^{z+k-1} & \cdots & \alpha^{(z+k-1)(z+k-1)} \end{bmatrix} \tag{5.39}$$

$$V_4 = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \alpha^{z+k} & \alpha^{z+k+1} & \cdots & \alpha^{z+k+r-1} \\ \vdots & \vdots & \vdots & \vdots \\ \alpha^{(z+k)\cdot(z+k-1)} & \alpha^{(z+k+1)(z+k-1)} & \cdots & \alpha^{(z+k+r-1)(z+k-1)} \end{bmatrix}. \tag{5.40}$$

Therefore, $[I_z|P\bar{C}\bar{Q}] = [I_z|V_1^{-1}V_2]$ is (by multiplying V_1 on the left) row equivalent to $[V_1|V_2]$. Similarly, $[I_{z+k}|P'C'Q'] = [I_{z+k}|V_3^{-1}V_4]$ is row equivalent to $[V_3|V_4]$. But $[V_1|V_2]$ consists of the first z rows of $[V_3|V_4]$. Therefore the row space of $[V_3|V_4]$ contains the row space of $[V_1|V_2]$, and the proof is complete. □

We are ready to prove the correctness of the scheme.

Theorem 5.2.2. *The scheme described in Section 5.2 is an (n, k, r, z) secure RAID scheme.*

Proof. We first prove that the scheme can correct r erasures. In Step 3, i.e., (5.22), we have applied a systematic erasure code $\mathcal{C}$ with generator matrix $G' = [I_{z+k}|P'C'Q']$, where I is the identity matrix of order $z + k$. On account of the proof of Lemma 5.2.1, G' is row equivalent to the Vandermonde matrix $[V_3|V_4]$. Specifically, the determinant of the submatrix formed by any $z + k$ columns of $[V_3|V_4]$ is a unit. This implies that any $z + k$ columns of G' are linearly independent and that $\mathcal{C}$ is an MDS code that can correct r erasures.

We turn to the security of the scheme. Consider any $A \subset [n]$, $|A| = z$, and suppose that the adversary has access to c_i, $i \in [A]$. By the linearity of the scheme, for $i \in [n]$ we can write $c_i = s_i + t_i$, where s_i is a linear function of the keys $(u_1, \cdots, u_z)$ and t_i is a linear function of the messages $(m_1, \cdots, m_k)$. Then to prove security it suffices to show that $(s_i)_{i \in A}$ is uniformly distributed because it implies that $(c_i)_{i \in A}$ is uniformly distributed and independent of the messages. Write

$$(c_1, \cdots, c_n) = (u_1, \cdots, u_z, m_1, \cdots, m_k)G,$$

then by construction

$$G = \left[\frac{G_u}{G_m} \right], \tag{5.41}$$

where $G_u = [I_z | PCQ] \cdot G'$ is a $z \times n$ matrix and G_m is a $k \times n$ matrix. We have

$$(s_1, \cdots, s_n) = (u_1, \cdots, u_z)G_u. \tag{5.42}$$

Let $\bar{G} = [I_z | P\bar{C}\bar{Q}]$. By Lemma 5.2.1, the row space of $\bar{G}$ is contained in the row space of G', namely each row of $\bar{G}$ is a codeword of $\mathcal{C}$. Note that by construction, the first $z + k$ columns of $\bar{G}$ are $[I_z | PCQ]$. Because $\mathcal{C}$ is an MDS array code, it follows that $\bar{G}$ is the unique matrix in the row space of G' such that its first $z + k$ columns are $[I_z | PCQ]$ (since any r entries of a codeword of $\mathcal{C}$ are uniquely determined by the other $z + k$ entries). On the other hand, G_u is in the row space of G' and the first $z + k$ columns of G_u are $[I_z | PCQ]$. Therefore

$$G_u = \bar{G} = [I_z | P\bar{C}\bar{Q}]. \tag{5.43}$$

By the proof of Lemma 5.2.1, $[I_z | P\bar{C}\bar{Q}]$ is row equivalent to the Vandermonde matrix $[V_1 | V_2]$ whose row space is an $[n, z]$ MDS array code. Therefore $(u_1, \cdots, u_z)$ can be decoded from $(s_i)_{i \in A}$. Because $(u_1, \cdots, u_z)$ is uniformly distributed, $(s_i)_{i \in A}$ is also uniformly distributed, which implies that the scheme is secure. Alternatively, the fact that $(u_1, \cdots, u_z)$ can be decoded from $(s_i)_{i \in A}$ implies that the condition $H(\boldsymbol{u} | \boldsymbol{c}, \boldsymbol{m}) = 0$ of Lemma 4.3.1 is met and so the scheme is secure. The proof is complete. □

5.2.2 Decoding

This section discusses the efficient decoding of the secure RAID scheme in Section 5.2. We remark that the efficiency of decoding is critical for most applications. For example, in storage, the decoding complexity affects critical performance including data read throughput and latency.

Decoding without erasures

We distinguish two scenarios of decoding, namely decoding without erasures (i.e., systematic decoding) and decoding with erasures. The former scenario addresses the situation that the complete codeword is available and no entries are erased. The task is to "decrypt" the message bits from the codeword. The latter scenario addresses the situation in which up to r entries of the codeword are erased, and the task is to recover the erased entries. We note that decoding without erasures is especially important because 1) for many applications such as storage, this is the common situation because the nodes are usually available; 2) in the situation that erasures do occur, after recovering the erased entries, we come back to the problem of decoding without erasure.

Specifically, the algorithm of decoding $(m_1, \cdots, m_k)$ from $(c_1, \cdots, c_{z+k})$ is simple. Since $(c_1, \cdots, c_z) = (u_1, \cdots, u_z)$, we can compute $(v_1, \cdots, v_k)$ by (5.16). Now for $i \in [k]$, we have $m_i = c_{z+i} + v_i$.

We make three remarks. First, the decoding process is very similar to Step 1 and Step 2 of encoding, implying that the same circuit/logic can be reused, which is helpful in terms of implementation. Second, we only assume that the first $z + k$ entries of the codeword are not erased and make no assumptions on the last r entries. Therefore if erasures do occur to the last r entries, for the sake of decoding it is not necessary to correct them. Third, for random access, i.e., the task of decoding $(m_i)_{i \in A}$, for some $A \subset [k]$, $|A| < k$, it suffices to only compute $(v_i)_{i \in A}$ and cancel them from $(c_i)_{i \in A}$. Compared to standard (complete) decoding, this achieves savings in both computation and the number of bits that are read and communicated.

The complexity of decoding without erasures is the same as the complexity of Step 1 and Step 2 of encoding, which amounts to no more than $3z(z + 2k)p + kp$ XORs to decode $k(p - 1)$ message bits. For the practical high rate case that k dominates z, the normalized decoding complexity is $O(6z)$ XORs per message bit. From the perspective that any message bit has to be padded by at least z key bits for the sake of security (otherwise the adversary can decode the message bit by eavesdropping z bits), the decoding complexity is asymptotically optimal up to a constant factor of 6.

Decoding with erasures

We now turn to the case that $1 \leq e \leq r$ entries of the codeword are erased. We assume that at least one of the first $z + k$ entries is erased (otherwise, there is no need to correct the erasures). By Theorem 5.2.2, the scheme can tolerate any r erasures and therefore it can be decoded naively by inverting the encoding matrix using Gaussian elimination. However, this leads to high computational[2] and implementation complexity. Below we present a much more efficient erasure decoding algorithm utilizing the structure of $\mathcal{R}_p$ and a result on the decomposition of the inverse of a Cauchy matrix, credited to Boros, Kailath and Olshevsky [47, Theorem3.1] (see also [48]). We note that although [47] studies matrices over fields, the assumption is immaterial and the proof therein generalizes to matrices over rings as long as the Cauchy matrix is well-defined:

Theorem 5.2.3. *Let $\alpha_1, \cdots, \alpha_n$ and $\beta_1, \cdots, \beta_n$ be elements of a (commutative) ring $\mathcal{R}$, such that $\alpha_i - \alpha_j$ and $\beta_i - \beta_j$ are units for distinct $i, j \in [n]$, and that $\alpha_i - \beta_j$ is a unit for $i, j \in [n]$. Then the Cauchy matrix*

$$C = \begin{bmatrix} \frac{1}{\alpha_1-\beta_1} & \frac{1}{\alpha_1-\beta_2} & \cdots & \frac{1}{\alpha_1-\beta_n} \\ \frac{1}{\alpha_2-\beta_1} & \frac{1}{\alpha_2-\beta_2} & \cdots & \frac{1}{\alpha_2-\beta_n} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{1}{\alpha_n-\beta_1} & \frac{1}{\alpha_n-\beta_2} & \cdots & \frac{1}{\alpha_n-\beta_n} \end{bmatrix} \tag{5.44}$$

is well defined and invertible, and its inverse can be decomposed as:

$$C^{-1} = U_1 U_2 \cdots U_{n-1} D L_{n-1} \cdots L_2 L_1, \tag{5.45}$$

[2]Recovering e erasures requires inverting an $e(p-1) \times e(p-1)$ matrix and requires $O(e^3 p^3)$ operations by Gaussian elimination.

where, for $k \in [n-1]$

$$L_k = \left[\begin{array}{c|c} I_k & \\ \hline & \begin{matrix} \frac{1}{\alpha_{k+1}-\alpha_1} & & & \\ & \frac{1}{\alpha_{k+2}-\alpha_2} & & \\ & & \ddots & \\ & & & \frac{1}{\alpha_n-\alpha_{n-k}} \end{matrix} \end{array}\right] \times \tag{5.46}$$

$$\left[\begin{array}{c|c} I_{k-1} & \\ \hline & \begin{matrix} 1 & & & \\ \beta_k-\alpha_1 & \alpha_{k+1}-\beta_k & & \\ & \ddots & \ddots & \\ & & \beta_k-\alpha_{n-k} & \alpha_n-\beta_k \end{matrix} \end{array}\right] \tag{5.47}$$

$$U_k = \left[\begin{array}{c|c} I_{k-1} & \\ \hline & \begin{matrix} 1 & \beta_1-\alpha_k & & \\ & \alpha_k-\beta_{k+1} & \ddots & \\ & & \ddots & \beta_{n-k}-\alpha_k \\ & & & \alpha_k-\beta_n \end{matrix} \end{array}\right] \times \tag{5.48}$$

$$\left[\begin{array}{c|c} I_k & \\ \hline & \begin{matrix} \frac{1}{\beta_1-\beta_{k+1}} & & & \\ & \frac{1}{\beta_2-\beta_{k+2}} & & \\ & & \ddots & \\ & & & \frac{1}{\beta_{n-k}-\beta_n} \end{matrix} \end{array}\right]. \tag{5.49}$$

and

$$D = \text{diag}(\alpha_1-\beta_1, \alpha_2-\beta_2, \cdots, \alpha_n-\beta_n). \tag{5.50}$$

Example 5.2.1. *For* $n = 3$*, it follows that*

$$\begin{bmatrix} \frac{1}{\alpha_1-\beta_1} & \frac{1}{\alpha_1-\beta_2} & \frac{1}{\alpha_1-\beta_3} \\ \frac{1}{\alpha_2-\beta_1} & \frac{1}{\alpha_2-\beta_2} & \frac{1}{\alpha_2-\beta_3} \\ \frac{1}{\alpha_3-\beta_1} & \frac{1}{\alpha_3-\beta_2} & \frac{1}{\alpha_3-\beta_3} \end{bmatrix}^{-1} = U_1U_2DL_2L_1, \tag{5.51}$$

where

$$L_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{\alpha_2-\alpha_1} & 0 \\ 0 & 0 & \frac{1}{\alpha_3-\alpha_2} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ \beta_1-\alpha_1 & \alpha_2-\beta_1 & 0 \\ 0 & \beta_1-\alpha_2 & \alpha_3-\beta_1 \end{bmatrix}, \tag{5.52}$$

$$L_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \frac{1}{\alpha_3-\alpha_1} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & \beta_2-\alpha_1 & \alpha_3-\beta_2 \end{bmatrix}, \tag{5.53}$$

$$U_1 = \begin{bmatrix} 1 & \beta_1-\alpha_1 & 0 \\ 0 & \alpha_1-\beta_2 & \beta_2-\alpha_1 \\ 0 & 0 & \alpha_1-\beta_3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{\beta_1-\beta_2} & 0 \\ 0 & 0 & \frac{1}{\beta_2-\beta_3} \end{bmatrix}, \tag{5.54}$$

$$U_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & \beta_1-\alpha_2 \\ 0 & 0 & \alpha_2-\beta_3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \frac{1}{\beta_1-\beta_3} \end{bmatrix}, \tag{5.55}$$

$$D = \text{diag}(\alpha_1-\beta_1, \alpha_2-\beta_2, \alpha_3-\beta_3). \tag{5.56}$$

Theorem 5.2.3 suggests an efficient way of multiplying a vector to the inverse of a Cauchy matrix over the ring $\mathcal{R}_p$: the matrices L_i, U_i and D are all sparse, and their entries either have the form of $\alpha_i - \beta_j$ or the form of $(\alpha_i - \beta_j)^{-1}$ (see Example 5.2.1). By Algorithm 2 and 3, multiplications involving these forms over $\mathcal{R}_p$ are both computationally efficient and simple to implement.

We turn to the decoding algorithm. Suppose that $(c_i)_{i\in E}$ are erased for some $E \subset [n]$, $|E| = e$. Let $E_1 = E \cap [z+k]$, $E_2 = E\backslash E_1$, $e_1 = |E_1|$, $e_2 = |E_2|$ and $R_1 = [z+k]\backslash E_1$. Since $r - e_2 \geq e_1$, there exists $R_2 \subset [z+k+1, n]$ such that $R_2 \cap E = \emptyset$ and $|R_2| = e_1$. The task of erasure decoding is to recover $c_{E_1} = (c_i)_{i\in E_1}$. By (5.22), we have

$$c_{R_2} = c_{R_1} P'_{R_1} C'_{R_1,R_2} Q'_{R_2} + c_{E_1} P'_{E_1} C'_{E_1,R_2} Q'_{R_2} \tag{5.57}$$

where P'_A and Q'_A are the square matrices formed by the rows and columns of P' and Q' indexed by A, respectively, and $C'_{A,B}$ is the matrix formed by the rows indexed by A and the columns indexed by B of C'. Therefore, the erased entries c_{E_1} can be recovered by

$$c_{E_1} = (c_{R_2} - c_{R_1} P'_{R_1} C'_{R_1,R_2} Q'_{R_2}) Q'^{-1}_{R_2} C'^{-1}_{E_1,R_2} P'^{-1}_{E_1}. \tag{5.58}$$

We remark that (5.58) can be computed efficiently: the term $c_{R_1} P'_{R_1} C'_{R_1,R_2} Q'_{R_2}$ can be obtained as in Step 3 of encoding; inverting $Q'^{-1}_{R_2}$ and $P'^{-1}_{E_1}$ is trivial since

both matrices are diagonal; multiplying a row vector to $C'^{-1}_{E_1,R_2}$ can be efficiently performed according to (5.45); and finally, all multiplications take the form of multiplying a ring element by either $\alpha^i - \alpha^j$ or $(\alpha^i - \alpha^j)^{-1}$ over R_p, which can be efficiently computed and easily implemented according to Algorithm 2 and Algorithm 3.

We analyze the complexity of erasure decoding. The step of computing the vector $(c_{R_2} - c_{R_1} P'_{R_1} C'_{R_1,R_2} Q'_{R_2})$ requires no more than $3(z+k+2r)(z+k)p$ XORs; the steps of multiplying with $Q'^{-1}_{R_2}$ and $P'^{-1}_{E_1}$ each requires no more than $3r(z+k)p$ XORs; the step of multiplying with $C'^{-1}_{E_1,R_2}$ according to (5.45) requires no more than $7r^2p + 2rp$ XORs. The total erasure decoding complexity is $O(6n(z+k)p + 7r^2p)$ XORs to recover rp erased bits. We note that the erasure decoding complexity is comparable to the encoding complexity of the scheme.

Decoding errors

We briefly discuss the problem of correcting errors in the codeword. Theorem 5.2.2 implies that the minimum distance between the codewords is $r+1$. Therefore given any codeword with no more than $\lceil \frac{r}{2} \rceil$ errors, it can be decoded correctly by exhaustive search. The question is how to correct the errors efficiently. Note that the set of codewords of the scheme is the row space of $[I_{z+k}|P'C'Q']$ which is the same as the row space of $[V_3|V_4]$ by the proof of Lemma 5.2.1. Namely the codewords form a generalized Reed-Solomon code over the ring $\mathcal{R}_p$. If 2 is a primitive element in $\mathbb{F}_p$, then $M_p(x)$ is irreducible and R_p is a field [35]. In this case the codewords form a standard Reed-Solomon code and so correcting errors in the codeword becomes the well studied problem of error decoding for Reed-Solomon codes [39]. For the case that $\mathcal{R}_p$ is not a field, decoding algorithms for generalized Reed-Solomon codes over $\mathcal{R}_p$ is studied in [34].

5.3 Construction from Vandermonde Matrices

In this section we present a secure RAID scheme over $\mathcal{R}_p$ constructed from Vandermonde matrices. Compared to the scheme in Section 5.2, this scheme is not systematic, and therefore is not as efficient in terms of decoding (without erasures) and random access. However, the scheme has the important advantage that it can be *shortened* flexibly. In addition to shortening, there are several other reasons that we believe the scheme worth discussion: 1) It is conceptually simple and is shown to be a natural generalization of Shamir's scheme to the ring $\mathcal{R}_p$. 2) It generalizes

the schemes discussed in [32], [33], with a vastly simplified algebraic description and proof of correctness. 3) The new algebraic description allows us to develop an efficient decoding algorithm, which previously was not available.

Construction 5.3.1. (Shamir's scheme over ring) *For n, r, z such that $n > r + z$, let $k = n - r - z$ and $p > n$ be a prime. Let the keys u_i, $i \in [z]$ be selected uniformly at random from $\mathcal{R}_p$, and let $m_i \in \mathcal{R}_p$, $i \in [k]$ be the messages. The codeword is computed by*

$$(c_1, \cdots, c_n) = (m_1, \cdots, m_k, u_1, \cdots, u_z) \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & \alpha & \cdots & \alpha^{n-1} \\ 1 & \alpha^2 & \cdots & \alpha^{2(n-1)} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & \alpha^{k+z-1} & \cdots & \alpha^{(k+z-1)(n-1)} \end{bmatrix}. \tag{5.59}$$

Theorem 5.3.1. *Construction 5.3.1 is an (n, k, r, z) secure RAID scheme.*

Proof. Any $k+z$ columns of the encoding matrix in (5.59) are a square Vandermonde matrix whose determinant, by Lemma 5.1.2, is a unit. Therefore any $k + z$ columns are linearly independent and the scheme can tolerate any $n - (k + z) = r$ erasures. Let G_{low} be the submatrix formed by the last z rows of the encoding matrix, then by a similar argument any z columns of G_{low} are linearly independent. Therefore the condition $H(\boldsymbol{u}|\boldsymbol{c}, \boldsymbol{m}) = 0$ of Lemma 4.3.1 is met and the scheme is secure. □

We remark that the scheme is a natural generalization of Shamir's scheme (and its ramp version) to the ring $\mathcal{R}_p$. Specifically, an equivalent encoding of the scheme is to generate a polynomial over $\mathcal{R}_p$ of degree $k + z - 1$, in which the degree 0 to $k - 1$ coefficients represent messages and the degree k to $z + k - 1$ coefficients are random, and then the n codeword entries (shares) are the evaluations of the polynomial at $\alpha^0, \alpha^1, \cdots, \alpha^{n-1}$. Shamir's original scheme corresponds to the case that $k = 1$, and that the polynomial is over a field, and that the evaluation points are arbitrarily chosen.

The scheme can be encoded efficiently. Specifically, by Algorithm 2, the encoding complexity of the scheme is at most $(z + k)np$ XORs to encode $k(p - 1)$ message bits. In the high rate case that k dominates z, the normalized encoding complexity is $O(n)$ XORs to encode each message bit. The encoding complexity of the scheme

has the same asymptotic order as the encoding complexity of the scheme in Section 5.2, and has a better constant factor.

Decoding

Schemes equivalent to Construction 5.3.1 and its variants are studied in [32], [33], which focus on the efficiency of encoding. However, while the schemes in [32], [33] can be efficiently encoded, an efficient decoding algorithm is not available, other than naively inverting the encoding matrix by Gaussian elimination, which has a high computational complexity of $O((z+k)^3p^3)$. We remark that although it is possible to pre-compute the inverse matrices, it is not practical to do so unless r is very small because there are $\binom{n}{r}$ different erasure patterns and each erasure pattern requires a different inverse. Furthermore, even if the inverse is pre-computed, in general it is a dense matrix and multiplying a vector to it requires up to $O((z+k)^2p^2)$ XORs. Below we present a significantly more efficient decoding algorithm utilizing the structure of $\mathcal{R}_p$ and a well-known result on the decomposition of the inverse of a Vandermonde matrix credited to Bjorck and Pereyra [49]. Though [49] addresses matrices over fields, generalizing its result to rings is straightforward as long as the Vandermonde matrix is indeed invertible over the ring:

Theorem 5.3.2. *Let $\alpha_1, \cdots, \alpha_n$ be elements of a (commutative) ring $\mathcal{R}$, such that $\alpha_i - \alpha_j$ is a unit for distinct $i, j \in [n]$. Then the Vandermonde matrix*

$$V = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \alpha_1 & \alpha_2 & \cdots & \alpha_n \\ \vdots & \vdots & \vdots & \vdots \\ \alpha_1^{n-1} & \alpha_2^{n-1} & \cdots & \alpha_n^{n-1} \end{bmatrix} \tag{5.60}$$

is invertible and its inverse can be decomposed as:

$$V^{-1} = U_1 U_2 \cdots U_{n-1} L_{n-1} \cdots L_2 L_1 \tag{5.61}$$

where, for $k \in [n-1]$

$$L_k = \left[\begin{array}{c|ccccc} I_{k-1} & & & & & \\ \hline & 1 & & & & \\ & -\alpha_k & 1 & & & \\ & & \ddots & \ddots & & \\ & & & -\alpha_k & 1 \end{array}\right] \tag{5.62}$$

$$U_k = \left[\begin{array}{c|ccccc} I_{k-1} & & & & & \\ \hline & 1 & -1 & & & \\ & & 1 & \ddots & & \\ & & & \ddots & -1 & \\ & & & & 1 & \end{array}\right] \left[\begin{array}{c|ccccc} I_k & & & & & \\ \hline & \frac{1}{\alpha_{k+1}-\alpha_1} & & & & \\ & & \frac{1}{\alpha_{k+2}-\alpha_2} & & & \\ & & & \ddots & & \\ & & & & \frac{1}{\alpha_n-\alpha_{n-k}} & \end{array}\right]. \tag{5.63}$$

Similar to Theorem 5.2.3, Theorem 5.3.2 suggests an efficient way of multiplying a vector to the inverse of a Vandermonde matrix over the ring $\mathcal{R}_p$: the matrices L_i and U_i are sparse, and their entries are either 1, -1, α_i or $(\alpha_i - \alpha_j)^{-1}$. By Algorithm 2 and 3, multiplications involving these forms over $\mathcal{R}_p$ are both computationally efficient and simple to implement.

We turn to the decoding algorithm. Suppose that the set of entries $c_R = \{c_i : i \in R\}$ is available for some $R \subset [n]$, $|R| = z + k$. The task of decoding is to recover $m_1, \cdots, m_k$ from c_R. By (5.59), we have

$$c_R = (m_1, \cdots, m_k, u_1, \cdots, u_z)V_R, \tag{5.64}$$

where V_R is the square matrix formed by the columns indexed by R of the Vandermonde matrix in (5.59). Therefore, the messages can be decoded by

$$\begin{aligned}(m_1, \cdots, m_k, u_1, \cdots, u_z) &= c_R V_R^{-1} &(5.65)\\ &= c_R U_1 \cdots U_{k+z-1} L_{k+z-1} \cdots L_1. &(5.66)\end{aligned}$$

By Algorithm 2 and Algorithm 3, decoding the $k(p-1)$ message bits (as well as the $z(p-1)$ key bits) by (5.66) requires at most $3(k+z)^2 p$ XORs. In the high rate case that k dominates z, the normalized decoding complexity is $O(3k)$ XORs per message bit.

Finally, in the situation that there are errors in the codeword, the discussion from Section 5.2.2 applies.

Shortening

Below we show that Construction 5.3.1 and Shamir's scheme have the desirable property that they can be flexibly shortened to any length, if the entries are suppressed carefully.

We first discuss the shortening of Shamir's scheme. We remark that the original Shamir's scheme has $k = 1$ and cannot be shortened since there is only one message

symbol. The scheme is later generalized to the case of $k > 1$ in [16], [17], which are usually referred to as ramp schemes. Below we present another generalization that is more natural and efficient.

Construction 5.3.2. (Ramp version of Shamir's scheme) *For n, r, z such that $n > r + z$, let $k = n - r - z$ and $q > n$. Let $m_1, ..., m_k$ be message symbols over $\mathbb{F}_q$. Construct a polynomial $f(x)$ of degree $z + k - 1$ over $\mathbb{F}_q$, whose degree 0 to $k - 1$ coefficients are m_1 to m_k, and the degree k to $z + k - 1$ coefficients are random (key symbols). The n shares are $f(\alpha_i)$, $i \in [n]$, where α_i, $i \in [n]$ are distinct non-zero elements of $\mathbb{F}_q$.*

Theorem 5.3.3. *Construction 5.3.2 is a (n, k, r, z) secure RAID scheme.*

The proof of the theorem is identical to that of Theorem 5.3.1.

Construction 5.3.3. (Shortened Shamir's scheme) *Let n, r, z, k, q be as in Construction 5.3.2. For $0 \leq s < k$, let $m_1, \cdots, m_{k-s}$ be message symbols over $\mathbb{F}_q$ and let $u_1, \cdots, u_z$ be random key symbols over $\mathbb{F}_q$. Construct a polynomial $f(x)$ of degree $z + k - 1$ over $\mathbb{F}_q$, whose degree 0 to $k - s - 1$ coefficients are the message symbols, and the degree k to $z + k - 1$ coefficients are the key symbols. Let α_i, $i \in [n]$ be distinct non-zero element of $\mathbb{F}_q$, and define a Vandermonde matrix*

$$V = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \alpha_1 & \alpha_2 & \cdots & \alpha_s \\ \vdots & \vdots & \vdots & \vdots \\ \alpha_1^{z+k-1} & \alpha_2^{z+k-1} & \cdots & \alpha_s^{z+k-1} \end{bmatrix}. \tag{5.67}$$

Then the degree $k - s$ to $k - 1$ coefficients of $f(x)$ are $m'_1, \cdots, m'_s$, given by

$$(m'_1, \cdots, m'_s) = -((m_1, \cdots, m_{k-s})V_{[1,k-s]} - (u_1, \cdots, u_z)V_{[k+1,z+k]})V^{-1}_{[k-s+1,k]}, \tag{5.68}$$

where $V_{[a,b]}$ is the submatrix formed by the a-th to b-th rows of V. The $n - s$ shares are $f(\alpha_i)$, $i \in [s + 1, n]$.

The length and dimension of Construction 5.3.3 are both reduced by s compared to Construction 5.3.2, and so the shortened scheme is rate-optimal. Note that Construction 5.3.3 can be viewed as a modified version of Construction 5.3.2. Namely, in Construction 5.3.2, if we compute $m_{k-s+1}, ..., m_k$ according to (5.68) instead of regarding them as arbitrary message symbols, then this results in Construction

5.3.3. Notice that Construction 5.3.2 outputs n shares while Construction 5.3.3 only outputs $n-s$ shares. This is because in Construction 5.3.3 the s "missing shares" $f(\alpha_1) = f(\alpha_2) = \cdots = f(\alpha_s) = 0$, i.e., they are always equal to 0 and so there is no need to compute or store them. To see this, note that by (5.68)

$$(m_1, \cdots, m_{k-s})V_{[1,k-s]} + (m'_1, \cdots, m'_s)V_{[k-s+1,k]} + (u_1, \cdots, u_z)V_{[k+1,z+k]} = 0 \tag{5.69}$$

implying that $(m_1, \cdots, m_{k-s}, m'_1, \cdots, m'_s, u_1, \cdots, u_z)V = 0$ and that $f(\alpha_i) = 0$, $i \in [s]$.

Theorem 5.3.4. *Construction 5.3.3 is an $(n-s, k-s, r, z)$ secure RAID scheme.*

Proof. By the remark above, Construction 5.3.3 can tolerate r erasures because Construction 5.3.2 can tolerate r erasures. To prove security, suppose that the adversary has access to any z shares, denoted by $\boldsymbol{c}$. By Lemma 4.3.1, it suffices to show that $H(\boldsymbol{u}|\boldsymbol{c}, \boldsymbol{m}) = 0$, where $\boldsymbol{u} = (u_1, \cdots, u_z)$ and $\boldsymbol{m} = (m_1, \cdots, m_{k-s})$. Note that given $\boldsymbol{c}$, $z+s$ evaluations of the function $f(x)$ are known because $\boldsymbol{c}$ contains z evaluations and by construction (see the remark above) $f(x)$ evaluates to 0 at s other points (i.e., α_i, $i \in [s]$). Split $f(x)$ into $f(x) = f_1(x) + f_2(x)$, such that $f_1(x)$ collects all the terms of $f(x)$ of degree $< k-s$ and $f_2(x)$ collects all the terms of degree $\geq k-s$. Namely, the coefficients of $f_1(x)$ are the m_i's and the coefficients of $f_2(x)$ are the m'_i's and the u_i's. Then given additional $\boldsymbol{m}$, $z+s$ evaluations of the polynomial $f_2(x)$ are known, by simply subtracting the evaluations of $f_1(x)$ from the evaluations of $f(x)$. Let $f_3(x) = x^{s-k} f_2(x)$, then the $z+s$ evaluations of $f_2(x)$ give $z+s$ evaluations of $f_3(x)$. $f_3(x)$ has degree $z+s-1$ and can be interpolated by the $z+s$ evaluations. Therefore its coefficients, the m'_i's and the u_i's, can be decoded. Hence $H(\boldsymbol{u}|\boldsymbol{c}, \boldsymbol{m}) = 0$ and the proof is complete. □

Applying the same idea of shortening to Construction 5.3.1, we have

Construction 5.3.4. (Shortened Shamir's scheme over ring) *Let n, r, z, k, p be as in Construction 5.3.1. For $0 \leq s < k$, let $m_1, \cdots, m_{k-s} \in \mathcal{R}_p$ be messages and let $u_1, \cdots, u_z \in \mathcal{R}_p$ be randomly selected. Define a Vandermonde matrix*

$$V = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \alpha & \alpha^2 & \cdots & \alpha^{s-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \alpha^{z+k-1} & \alpha^{2(z+k-1)} & \cdots & \alpha^{(s-1)(z+k-1)} \end{bmatrix}, \tag{5.70}$$

and compute $m'_1, \cdots, m'_s$ *by*

$$(m'_1, \cdots, m'_s) = -((m_1, \cdots, m_{k-s})V_{[1,k-s]} - (u_1, \cdots, u_z)V_{[k+1,z+k]})V^{-1}_{[k-s+1,k]}, \tag{5.71}$$

where $V_{[a,b]}$ *is the submatrix formed by the* a*-th to* b*-th rows of* V*. The codeword is given by*

$$(c_1, \cdots, c_{n-s}) = (m_1, \cdots, m_{k-s}, m'_1, \cdots, m'_s, u_1, \cdots, u_z) \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \alpha^s & \alpha^{s+1} & \cdots & \alpha^{n-1} \\ \vdots & \vdots & \vdots & \vdots \\ \alpha^{s(z+k-1)} & \alpha^{(s+1)(k+z-1)} & \cdots & \alpha^{(n-1)(k+z-1)} \end{bmatrix}. \tag{5.72}$$

Theorem 5.3.5. *Construction 5.3.4 is an* $(n-s, k-s, r, z)$ *secure RAID scheme.*

Proof. Theorem 5.3.5 can be proved in the same way as Theorem 5.3.4 as long as we take proper care of the fact that $\mathcal{R}_p$ is a ring and may not be a field. Specifically, we need to ensure that the polynomial $f_3(x)$ has well-defined values at the $z+s$ evaluation points. This is indeed true because the evaluation points take the form of α^i, which is a unit in $\mathcal{R}_p$, and so the value of x^{s-k} is well-defined at these points. Furthermore, we need to ensure that $f_3(x)$ can be interpolated by the $z+s$ evaluations, namely the Vandermonde matrix corresponding to the $z+s$ evaluation points is invertible. This is indeed true due to Lemma 5.1.2. □

We remark on the usefulness of shortening. First, for Construction 5.3.3 or 5.3.4, one can easily change the parameter s with very minor modification to the scheme. Specifically, the encoding algorithm is only slightly different and the decoding algorithm is identical. This allows one to implement a flexible scheme with adaptable length n and dimension k, which is important for practical deployment, e.g., when the number of nodes is not fixed a priori or may vary over time. Second, shortening will cause overhead in encoding, i.e., computing (5.68) or (5.71). However, it is a rather minor one because the matrix $V^{-1}_{[k-s+1,k]}$ can either be pre-computed (for Construction 5.3.3), or be treated by Theorem 5.3.2 (for Construction 5.3.4).

Chapter 6

CONCLUDING REMARKS

In Part I, we introduce secure RAID schemes, which are secret sharing schemes with efficient encoding, decoding and random access complexity and low implementation complexity. We derive lower bounds on the computational complexity, generalize systematic encoding to the secure setting, and describe a general framework for constructing efficient systematic schemes. For $r \leq 3$, $z \leq 3$, we present secure RAID schemes with optimal computation from the B, EVENODD and STAR codes. For general parameters, we construct efficient schemes over the ring $\mathcal{R}_p$, from Vandermonde and Cauchy matrices. We discuss methods to flexibly shorten the secure EVENODD scheme and the secure RAID scheme over $\mathcal{R}_p$ based on Vandermonde matrix.

Many interesting problems remain open, which we list a few. Regarding the high rate regime, is it possible to design schemes with optimal computation for $r \geq 4$ or $z \geq 4$? We remark that for erasure codes, i.e., for $z = 0$, codes with optimal computation are known up to $r = 8$ [35]. Furthermore, we only know how to construct the strictly optimal scheme in Construction 4.6.1 for a finite set of lengths. Is it possible to generalize this construction to an infinite family? Also, while the secure B, secure EVENODD and secure STAR are infinite families of schemes, they are not strictly optimal and are not strictly systematic. Is it possible to further improve them? Regarding the schemes with general parameters, is it possible to design schemes that require $O(r + z)$ XORs to encode and $O(z)$ XORs to decode each message bit (the construction from Cauchy matrices requires $O(z)$ XORs to decode but $O(n)$ XORs to encode each message bit)? Finally, in addition to secure EVENODD and the scheme over $\mathcal{R}_p$ based on Vandermonde matrices, is it possible to shorten other secure RAID schemes and existing secret sharing schemes?

Part II

Communication Efficient Secret Sharing

Chapter 7

INTRODUCTION TO COMMUNICATION EFFICIENT SECRET SHARING

In Part I we study secure RAID schemes which are a class of secret sharing schemes with low encoding, decoding and implementation complexity. In this part we study the communication and access complexity of secret sharing schemes. We start with introducing general secret sharing schemes.

Consider the scenario that n parties wish to store a secret message securely and reliably. To this end, a dealer encodes the message into n shares, i.e., one share for each party, such that 1) (reliability) a collection $\mathcal{A} \subset 2^{\{1,...,n\}}$ of "authorized" subsets of the parties can decode the message, and 2) (secrecy) a collection $\mathcal{B}$ of "blocked" subsets of the parties cannot collude to deduce any information about the message. A scheme to encode the message into shares with respect to *access structure* $(\mathcal{A}, \mathcal{B})$ is called a secret sharing scheme, initially studied in the seminal works by Shamir [15] and Blakley [50]. A secret sharing scheme is *perfect* if a subset of parties is either authorized or blocked, i.e., $\mathcal{A} \cup \mathcal{B} = 2^{\{1,...,n\}}$. A scheme is often referred to as a *ramp* scheme if it is not perfect [16]. A natural application of secret sharing schemes is distributed storage of private data, where each party is a storage node. Besides, secret sharing is a fundamental cryptographic primitive and is used as a building block in numerous secure protocols [51].

As in Part I, we focus on secret sharing schemes for the *threshold* access structure, i.e., $\mathcal{A}$ contains all subsets of $\{1, ..., n\}$ of size at least $n - r$, and $\mathcal{B}$ contains all subsets of $\{1, ..., n\}$ of size at most z. In other words, the message can be decoded in the absence of any r parties, and any z parties cannot collude to deduce any information about the message. Note that a threshold scheme is perfect if and only if $z = n - r - 1$. The scheme is a ramp scheme if $z < n - r - 1$. The threshold access structure is particularly important in practice, because for this case, efficient secret sharing schemes are known. Specifically, Shamir [15] constructs an elegant and efficient perfect threshold scheme using the idea of polynomial interpolation. Shamir's scheme is later shown to be closely related to Reed-Solomon codes [40] and is generalized to ramp schemes in [16], [17], [52], which allow better space efficiency, i.e., rate, than the original perfect scheme. Shamir's scheme and the

generalized ramp schemes achieve optimal usage of storage space, in the sense that fixing the size of the shares, the schemes store a message of maximum size. The schemes are computationally efficient in the sense that encoding is equivalent to polynomial evaluation and decoding is equivalent to polynomial interpolation. An example of Shamir's scheme is shown in Figure 7.1. Other threshold secret sharing schemes and generalizations of Shamir's scheme may be found in, e.g., [32], [53]. Recently, there has been considerable interest in incorporating secrecy into erasure codes, e.g., [21]–[23]. These codes and the secure RAID schemes studied in Part I are also threshold secret sharing schemes.

Party 1	Party 2	$\cdots$	Party 7
$f(1) =$	$f(2) =$	$\cdots$	$f(7) =$
$m_1 + m_2 + u_1$	$m_1 + 2m_2 + 4u_1$	$\cdots$	$m_1 + 7m_2 + 5u_1$

Figure 7.1: Shamir's scheme (generalized ramp version, see Construction 5.3.2) for $n = 7, r = 4, z = 1$, with symbols over $\mathbb{F}_{11}$. The scheme stores a message of two symbols, denoted by m_1, m_2. u_1 is a uniformly distributed random element of $\mathbb{F}_{11}$. $f(x)$ is the polynomial $m_1 + m_2 x + u_1 x^2$. Note that the share stored by any single party is independent of the message because it is padded by u_1, and that the message can be decoded from the shares stored by any three parties by polynomial interpolation.

Decoding Bandwidth

In addition to space and computational efficiency, this chapter studies the *communication efficiency* for secret sharing schemes. Consider the scenario that a user wishes to decode the message by downloading information from the parties that are available. Referring to the amount of information downloaded by the user as the *decoding bandwidth*, a natural question is to determine and achieve the minimum decoding bandwidth. It is of practical interest to design secret sharing schemes that achieve a small decoding bandwidth, or in other words, that require communicating only a small amount of information during decoding. In such a case, decoding will be completed in a timely manner and the communication resource will be more efficiently utilized.

In many existing secret sharing schemes, e.g., [15]–[17], [21]–[23], [32], [40], [53], a common practice in decoding is that the user will communicate with a minimum set of parties, i.e., exactly $n - r$ parties (even if $d > n - r$ parties are available) and download the whole share stored by these parties. Wang and Wong [54] show that this paradigm is not optimal in terms of communication and that the decoding

bandwidth can be reduced if the user downloads only part of the share from each of the $d > n - r$ available parties. Specifically, given d, for any perfect threshold secret sharing scheme, [54] derives a lower bound on the decoding bandwidth when exactly d parties participate in decoding, and designs a perfect scheme that achieves the lower bound. The field size of the scheme is slightly improved in [55]. However, two interesting and important problems remain open: 1) the schemes in [54], [55] achieve the lower bound on the decoding bandwidth when the number of available parties d equals a specific value fixed in code design, and do not achieve the bound if d takes other values. This raises the question of whether the lower bound is uniformly tight, or in other words, if it is possible to design a single scheme that achieves the lower bound universally for all d in the range of $[n - r, n]$. 2) The results in [54], [55] target the case of perfect secret sharing schemes, i.e., the case of $z = n - r - 1$. By Proposition 3.1.1, the rate of the scheme is at most $1/n$. This raises the question of how to generalize the results and ideas to the case of $z \leq n-r-1$, so that the parameters and rates of the schemes are more flexible. Both problems are of practical importance as the first problem addresses the flexibility of a scheme in terms of decoding, and the second problem addresses the flexibility in terms of rate. In this chapter we settle both problems and construct schemes of flexible rate that achieve the optimal decoding bandwidth universally. Additionally, our schemes are computationally efficient and have optimal space efficiency.

Motivating Example

Consider Shamir's scheme (ramp version) in the example of Figure 7.1, that stores 2 symbols securely and reliably for the setting $n = 7, r = 4$ and $z = 1$. In order to decode the message, a user needs to download 3 symbols from any 3 parties, and therefore the decoding bandwidth is 3 symbols. Suppose the same scheme is repeated 3 times in order to store a message of 6 symbols, as shown in Figure 7.2a. Then to decode the message, the decoding bandwidth is 9 symbols.

We propose a new scheme in Figure 7.2b that also stores a message of 6 symbols for the same setting, using the same amount of storage space, and over the same field size. In this scheme, if any 3 parties are available, then similar to Shamir's scheme, the message can be decoded from the 9 symbols stored by the 3 parties. However, if any 4 parties are available, then the message can be decoded by downloading 2 symbols from each available party. Therefore, the decoding bandwidth is improved to 8 symbols. If all 7 parties are available, then the message can be decoded by

Party 1	Party 2	$\cdots$	Party 7
$m_1 + m_2 + u_1$	$m_1 + 2m_2 + 4u_1$	$\cdots$	$m_1 + 7m_2 + 5u_1$
$m_3 + m_4 + u_2$	$m_3 + 2m_4 + 4u_2$	$\cdots$	$m_3 + 7m_4 + 5u_2$
$m_5 + m_6 + u_3$	$m_5 + 2m_6 + 4u_3$	$\cdots$	$m_5 + 7m_6 + 5u_3$

(a) Shamir's Scheme

Party 1	$\cdots$	Party 7
$f(1) = u_1 + m_1 + m_2 + m_3 + m_4 + m_5 + m_6$	$\cdots$	$f(7) = u_1 + 7m_1 + 5m_2 + 2m_3 + 3m_4 + 10m_5 + 6m_6$
$g(1) = u_2 + m_4 + m_5 + m_6$	$\cdots$	$g(7) = u_2 + 7m_4 + 5m_5 + 2m_6$
$h(1) = u_3 + m_3 + m_6$	$\cdots$	$h(7) = u_3 + 7m_3 + 5m_6$

(b) Proposed Scheme

Figure 7.2: Two secret sharing schemes for $n = 7, r = 4$ and $z = 1$ over $\mathbb{F}_{11}$. Both schemes store a message of six symbols (m_1, ..., m_6). In both schemes, u_1, u_2, u_3 are i.i.d. uniformly distributed random variables, i.e., keys. Scheme (a) is Shamir's scheme (see Figure 7.1) repeated three times. In Scheme (b), $f(x) = u_1 + m_1x + m_2x^2 + m_3x^3 + m_4x^4 + m_5x^5 + m_6x^6$, $g(x) = u_2 + m_4x + m_5x^2 + m_6x^3$, $h(x) = u_3 + m_3x + m_6x^2$, and party i stores evaluations $f(i)$, $g(i)$ and $h(i)$. Note that in (b), if all 7 parties are available, then the message can be decoded by downloading only one symbol $f(i)$ from each party i, and then interpolating $f(x)$. If any 4 parties are available, then the message can be decoded in the following way. Download two symbols $f(i), g(i)$ from each available party i and first interpolate $g(x)$, implying that all coefficients of $f(x)$ of degree larger than 3 (e.g., m_4, m_5, m_6) are decoded. The remaining unknown part of $f(x)$ is a degree-3 polynomial and so we have enough evaluations of $f(x)$ to interpolate it, hence completely decoding the message. Similarly, if any 3 parties are available, then the message can be decoded in the following way. Download all three symbols $f(i), g(i), h(i)$ from each available node i and interpolate $h(x)$, which decodes the degree-3 coefficients of $f(x)$ and $g(x)$. Hence the remaining unknown part of $g(x)$ is a degree-2 polynomial and can be interpolated, which decodes the coefficients of $f(x)$ of degrees $4, 5, 6$. Hence the remaining unknown part of $f(x)$ is a degree-2 polynomial and can be interpolated, decoding the complete message. This shows that the scheme meets the reliability requirement. In fact, for $d = 3, 4, 7$, Scheme (b) achieves the optimal decoding bandwidth when d parties participate in decoding. The secrecy of the scheme derives from the secrecy of the generalized Shamir's scheme, as each polynomials $f(x)$, $g(x)$ and $h(x)$ individually is an instance of Construction 5.3.2, and we show that their combination remains secure. The construction is discussed in detail in Section 8.2.

downloading only 1 symbol from each party and so the decoding bandwidth is further reduced to 7 symbols.

We use the examples in Figure 7.2 to highlight several ideas to reduce the decoding bandwidth. Firstly, the necessary amount of communication decreases strictly as the

number of available parties increases. Secondly, it is helpful to distribute multiple subshares (symbols) to a party (essentially the idea of array schemes discussed in Section 3.1). In contrast, Shamir's scheme only distributes one symbol to each party except for trivial repetitions. Thirdly, during decoding it is not always necessary to download the complete share stored by a party. In general, a party can preprocess its share and the user can download a function of the share.

Specifically, in Section 8.1, we prove a tight information-theoretic lower bound on the decoding bandwidth. Particularly, let I be the set of parties participating in decoding, we show that the decoding bandwidth for the case of $|I| = n$ is only a fraction of $\frac{n(n-z-r)}{(n-z)(n-r)}$ of the decoding bandwidth when $|I| = n - r$. In Section 8.2, we construct a secret sharing scheme using the ideas described in Figure 7.2. The construction utilizes a generalized Shamir's scheme and achieves the optimal decoding bandwidth universally for all $I \in \mathcal{A}$. In Section 8.3, we construct another secret sharing scheme from Reed-Solomon codes. The scheme achieves the optimal decoding bandwidth when $|I| = n$ and $|I| = n - r$. The decoder of the scheme has a simpler structure compared to the decoder of the first scheme. The scheme also offers a stronger level of reliability in that it allows decoding even if more than r shares are partially lost.

In the application of storage where each party is regarded as a disk, it is desirable to optimize the efficiency of disk operations. Our lower bound on the decoding bandwidth is a natural lower bound on the number of disk access, i.e., symbol-read, that occur during decoding. In both schemes mentioned above, the number of disk access during decoding equals the amount of communication. Therefore, our schemes are also optimal in terms of *access complexity*. In addition, by involving more disks for decoding, our schemes balance the load at the disks and achieve a higher degree of parallelization.

As previously discussed, the communication efficiency of Shamir's scheme is sub-optimal. In the standard Shamir's scheme, in order to decode the single message symbol (using the standard decoding algorithm), one needs to download $n - r$ encoded symbols. While we have designed schemes with better decoding bandwidth, Shamir's scheme is still extensively used due to its simplicity and it remains an important problem whether it is possible to reduce the decoding bandwidth of Shamir's scheme.

In Section 8.4 we construct a family of Shamir's schemes that is asymptotically optimal in the decoding bandwidth. Specifically, as opposed to the original $n - r$

symbols, the decoding bandwidth reduces to $n/(1+r)$ symbols as the field size increases. The decoding algorithm follows the framework proposed in [56] of interpolating polynomials by querying partial polynomial evaluations. Our scheme is inspired by the family of Reed-Solomon codes constructed in [57] which has asymptotically optimal repair bandwidth. Decoding Shamir's scheme and repairing Reed-Solomon codes are related because both are essentially the problem of determining the evaluation of a polynomial at a point, given the evaluations of the polynomial at other points. Decoding Shamir's scheme in this sense is a simpler problem because it requires finding the evaluation at a single point, while repairing Reed-Solomon codes requires finding the evaluation at different points, depending on which symbol is being repaired. This simplification allows us to greatly reduce the field size. Specifically, while the codes in [57] require the extension degree of the field to be exponential in n, our scheme only requires an extension degree of $O(n(n-z)^3)$, which makes it quite practical.

Repair Bandwidth

In addition to the decoding bandwidth, another important aspect of communication efficiency in distributed storage is the repair bandwidth, which is the amount of information communicated during the process of repairing an erasure/failure. Erasure codes with low repair bandwidth, referred to as the regenerating codes, are well studied in the literature, e.g., [58]–[60]. Secret sharing schemes with low repair bandwidth, referred to as the secure regenerating codes, are studied in, e.g., [21]–[23], where lower bounds on the repair bandwidth are obtained and optimal schemes are proposed. A natural and important question is whether it is possible to construct a secret sharing scheme with both optimal decoding and repair bandwidth. Rawat et al. [61], by observing that decoding the secret sharing scheme can be viewed as repairing the message symbols in a regenerating code, propose schemes that are bandwidth efficient in both repair and decoding. However, their construction is quite restricted in parameters if rate-optimality is required.

In chapter 9, by formalizing the connection between regenerating codes and secret sharing schemes, and then applying the connection to the regenerating codes in [62], we obtain secret sharing schemes with optimal decoding and repair bandwidth for general parameters. However, these schemes are not practical as they require an extremely large level of sub-packetization (i.e., the number of symbols that need to be stored by a node) that is doubly exponential in n. To reduce sub-packetization,

we use the fact that all message symbols are "repaired" together in a centralized manner during decoding. Therefore we essentially need a regenerating code that allows a hybrid mode of repair: centralized repair of the set of message symbols, and individual repair of the remaining symbols. We generalize the codes in [62] to this model and construct secret sharing schemes with a much smaller sub-packetization level due to the centralized repair pattern. Our generalization also leads to two families of regenerating codes that support centralized repair of groups of nodes of flexible sizes with reduced sub-packetization level, which is a result of separate interest. Finally, among the two bandwidth-optimal schemes that we present, the latter one is also optimal in access complexity during both decoding and repair.

Part of the material in Chapter 8 and 9 was presented in [63] and [64].

Chapter 8

SECRET SHARING SCHEMES WITH OPTIMAL DECODING

8.1 Lower Bound on Decoding Bandwidth

Recall the definition of an $(n, k, r, z)_{\mathcal{Q}}$ secret sharing scheme from Section 3.1. By Proposition 3.1.1 a secret sharing scheme is *rate-optimal* if $k = n-r-z$. A scheme is a *perfect scheme* if $z = n - r - 1$ and is a *ramp scheme* if $z \leq n - r - 1$. The rate of any perfect scheme is at most $1/n$ as $k = 1$. Any scheme of a higher rate is necessarily a ramp scheme.

Suppose that the n shares of the secret message are stored by n parties or distributed storage nodes[1], and a user wants to decode the message. By the reliability requirement, the user can decode by connecting to any $n - r$ nodes and downloading one share, i.e., one $|\mathcal{Q}|$-ary symbol, from each node. Therefore, by communicating $n - r$ symbols, the user can decode a message of $k \leq n - r - z$ symbols. It is clear that a communication overhead of z symbols occurs during *decoding*. The question is, whether it is possible to reduce the communication overhead. We answer this question affirmatively in this chapter.

There are two key ideas for improving the communication overhead. First, in many practical scenarios and particularly in distributed storage systems, oftentimes more than $n - r$ nodes are available. In this case, it is not necessary to restrict the user to download from only $n - r$ nodes. Secondly, it is not necessary to download the complete share stored by the node. Instead, it may suffice to communicate only a part of the share or, in general, a function of the share. In other words, a node can preprocess its share before transmitting it to the user.

Motivated by these ideas, for any $I \subset [n]$, $|I| \geq n-r$, define a class of *preprocessing functions* $E_{I,i} : \mathcal{Q} \to \mathcal{S}_{I,i}$, where $|\mathcal{S}_{I,i}| \leq |\mathcal{Q}|$, that maps c_i to $e_{I,i} = E_{I,i}(c_i)$. Let $\boldsymbol{e}_I = (e_{I,i})_{i\in I}$, and define a class of *decoding functions* $D_I : \prod_{i\in I} \mathcal{S}_{I,i} \to \mathcal{Q}^k$, such that $D_I(\boldsymbol{e}_I) = \boldsymbol{m}$. For a naive example, consider any I such that $|I| = n - r$. Then for $i \in I$, we can let $\mathcal{S}_{I,i} = \mathcal{Q}$, let $E_{I,i}$ be the identity function, and let D_I be the naive decoding function implied by the reliability requirement. In the remaining part of this chapter, when I is clear from the context, we will suppress it in the subscripts of $\mathcal{S}_{I,i}$, $E_{I,i}$, $e_{I,i}$ and $\boldsymbol{e}_I$, and denote them by $\mathcal{S}_i$, E_i, e_i and $\boldsymbol{e}$ instead. We

[1] In what follows we do not distinguish between parties and nodes.

now formally define the notion of communication overhead in decoding. Note that all $\log$ functions are base $Q = |\mathcal{Q}|$.

Definition 8.1.1. *For any I such that $|I| \geq n-r$, define the communication overhead function to be* $\mathrm{CO}(I) = \sum_{i \in I} \log |\mathcal{S}_{I,i}| - k$. *Namely,* $\mathrm{CO}(I)$ *is the amount of extra information, measured in Q-ary symbols, that one needs to communicate in order to decode a message of k symbols, provided that the set of available shares is indexed by I.*

The following result provides a lower bound on the communication overhead function. It generalizes the lower bound in [54] for perfect schemes, i.e., schemes with $z = n - r - 1$.

Theorem 8.1.1. *For any $(n, k, r, z)_{\mathcal{Q}}$ secret sharing scheme with preprocessing functions $\{E_{I,i}\}_{i \in I, |I| \geq n-r}$ and decoding functions $\{D_I\}_{|I| \geq n-r}$, it follows that*

$$\mathrm{CO}(I) \geq \frac{kz}{|I| - z}. \tag{8.1}$$

Proof. Consider arbitrary $I = \{i_1, ..., i_{|I|}\}$ such that $|I| \geq n - r$. Assume without loss of generality that $|\mathcal{S}_{i_1}| \leq |\mathcal{S}_{i_2}| \leq ... \leq |\mathcal{S}_{i_{|I|}}|$. Recall that $\boldsymbol{e}_I = (e_{i_1}, ..., e_{i_{|I|}})$ is the output of the preprocessing functions.

$$\begin{aligned}
H(e_{i_1}, ..., e_{i_{|I|-z}}) &\overset{(a)}{\geq} H(e_{i_1}, ..., e_{i_{|I|-z}} | e_{i_{|I|-z+1}}, ..., e_{i_{|I|}}) \\
&\overset{(b)}{=} H(e_{i_1}, ..., e_{i_{|I|-z}} | e_{i_{|I|-z+1}}, ..., e_{i_{|I|}}) \\
&\qquad + H(\boldsymbol{m} | e_{i_1}, ..., e_{i_{|I|}}) \\
&\overset{(c)}{=} H(\boldsymbol{m}, e_{i_1}, ..., e_{i_{|I|-z}} | e_{i_{|I|-z+1}}, ..., e_{i_{|I|}}) \\
&\geq H(\boldsymbol{m} | e_{i_{|I|-z+1}}, ..., e_{i_{|I|}}) \\
&\overset{(d)}{=} H(\boldsymbol{m}) = k,
\end{aligned} \tag{8.2}$$

where (a) follows from the fact that conditioning reduces entropy, (b) follows from the reliability requirement, (c) follows from the chain rule, and (d) follows from the secrecy requirement. Therefore it follows from (8.2) that

$$\prod_{j=1}^{|I|-z} |\mathcal{S}_{i_j}| \geq Q^{H(e_{i_1}, ..., e_{i_{|I|-z}})} \geq Q^k,$$

and so

$$\sum_{j=1}^{|I|-z} \log |\mathcal{S}_{i_j}| \geq k. \tag{8.3}$$

It then follows from $|\mathcal{S}_{i_1}| \leq ... \leq |\mathcal{S}_{i_{|I|}}|$ that,

$$\log |\mathcal{S}_{i_{|I|-z}}| \geq \frac{k}{|I|-z},$$

and that,

$$\log |\mathcal{S}_{i_{|I|-z+j}}| \geq \log |\mathcal{S}_{i_{|I|-z}}| \geq \frac{k}{|I|-z}, \quad j = 1, ..., z. \tag{8.4}$$

Combining (8.3) and (8.4) we have,

$$\text{CO}(I) = \sum_{j=1}^{|I|} \log |\mathcal{S}_{i_j}| - k \geq \frac{kz}{|I|-z}.$$

□

Theorem 8.1.1 suggests that the communication overhead decreases as the number of available nodes increases. Define the *decoding bandwidth* to be the amount of information downloaded by a use during decoding. By Theorem 8.1.1 we have the following result.

Corollary 8.1.1. *For $d \geq n - r$, the bandwidth of decoding a $(n, k = n - r - z, r, z)_{\mathbb{F}_q^t}$ secret sharing scheme from d nodes is at least $\frac{kdt}{d-z}$ symbols over $\mathbb{F}_q$.*

For rate-optimal schemes, Theorem 8.1.1 implies that if $|I| = n - r$, then the communication overhead is at least z, i.e., the user needs to download the complete share from each available node. The naive decoding function implied by the reliability requirement trivially achieves this bound. The more interesting scenario is the regime in which $|I| > n - r$. In this case, if (8.1) is tight, then one can achieve a non-trivial improvement on decoding bandwidth compared to the naive decoding function. When $k = n - r - z = 1$ (i.e., for perfect schemes) and fixing any $d > n-r$, [54] constructs a rate-optimal scheme that achieves the lower bound (8.1) for any I such that $|I| = d$. However, several interesting and important questions remain open. Firstly, is the lower bound uniformly tight, or in other words, is it possible to construct a scheme that achieves (8.1) universally for any I such that $|I| \geq n - r$ (note that the scheme in [54] does not achieve the lower bound when $|I| \neq d$)? Secondly, is the bound tight when $k > 1$ (i.e., for ramp schemes) and how to design such schemes? We answer these questions in the following section.

8.2 Construction from Shamir's Scheme

In this section we construct a rate-optimal scheme that achieves the optimal decoding bandwidth universally for all I such that $|I| \geq n - r$, i.e., all sets of available nodes that allow decoding. This implies that the lower bound in Theorem 8.1.1 is uniformly tight. The scheme is based on a generalization of Shamir's scheme and preserves its simplicity and efficiency. The scheme is flexible in the parameters n, k, r, z and hence is flexible in rate.

We first refer the readers to Figure 7.2b for an example of the scheme, and use it to describe the general idea of the construction. To construct a scheme that achieves the optimal decoding bandwidth when d nodes are available, for all $d \in \mathcal{D}$, we design a set of polynomials of different degrees. Particularly, for all $d \in \mathcal{D}$, we design a number of polynomials of degree exactly $d - 1$, and store one evaluation of each polynomial at each node. For each polynomial, exactly z of its coefficients are independent keys in order to meet the secrecy requirement. The remaining coefficients encode "information": for the highest-degree (e.g., degree $d_{\max} - 1$, where $d_{\max} = \max_{d \in \mathcal{D}} d$) polynomials, their coefficients encode the entire message; for other polynomials, say $g(x)$, the information encoded in the coefficients of $g(x)$ is the high-degree coefficients of the polynomials of degree higher than $g(x)$. Such an arrangement of the coefficients enables decoding in a successive manner. Consider decoding when d nodes are available, implying that d evaluations of each polynomial are known and hence all polynomials of degree $d - 1$ can be interpolated. Then, roughly speaking, the arrangement ensures that the interpolation will decode the coefficients of degree $\geq d$ of some high-degree polynomials, so that the remaining unknown parts of these polynomials can be interpolated. This in turn allows us to decode coefficients for additional high-degree polynomials and thus to interpolate them. The chain continues until all polynomials of degree higher than $d - 1$ are interpolated, implying that the message is decoded. Note that no polynomials of degree smaller than $d - 1$ are interpolated, and therefore the keys associated with them are not decoded. This leads to the saving in decoding bandwidth and in fact this amount is the best one can expect to save, so that the scheme achieves the optimal bandwidth. Below we describe the scheme formally.

8.2.1 Encoding

Consider arbitrary parameters n, r, z, $\mathcal{D}$ and let $k = n - r - z$. We assume that $n - r \in \mathcal{D}$ since it is implied by the reliability requirement. Choose any prime power $q > n$, the scheme is an array scheme over $\mathbb{F}_q^t$. Namely, each node stores t symbols

over $\mathbb{F}_q$ and the encoding function is linear over $\mathbb{F}_q$. The message $\boldsymbol{m}$ is a vector over $\mathbb{F}_q$ of length $|\boldsymbol{m}| = kt$. The choice of t is determined by $\mathcal{D}$ in the following way. Let $|\boldsymbol{m}|$ be the least common multiple of $\{d - z : d \in \mathcal{D}\}$, i.e., the smallest positive integer that is divisible by all elements of the set. Note that $k + z \in \mathcal{D}$ and so k divides $|\boldsymbol{m}|$, and we let $t = \frac{|\boldsymbol{m}|}{k}$. This is the smallest choice of $|\boldsymbol{m}|$ (and thus t) that ensures when $d \in \mathcal{D}$ nodes are available, the optimal bandwidth, measured by the number of $\mathbb{F}_q$ symbols, is an integer.

We now construct t polynomials over $\mathbb{F}_q$, evaluate each of them at n non-zero points, and let every node store an evaluation of each polynomial. Let $\mathcal{D} = \{d_1, d_2, ..., d_{|\mathcal{D}|}\}$, such that $n \geq d_1 > d_2 > ... > d_{|\mathcal{D}|} = n - r$. For $i \in |\mathcal{D}|$, let

$$p_i = \begin{cases} \frac{|\boldsymbol{m}|}{d_1 - z} & i = 1 \\ \frac{|\boldsymbol{m}|}{d_i - z} - \frac{|\boldsymbol{m}|}{d_{i-1} - z} & i > 1 \end{cases} \tag{8.5}$$

We construct p_i polynomials of degree $d_i - 1$. For all polynomials, their z lowest-degree coefficients are independent random keys. We next define the remaining $d_i - z$ non-key coefficients. We first define them for the highest degree polynomials, and then recursively define them for the lower degree polynomials. For $i = 1$, the non-key coefficients of the polynomials of degree $d_i - 1$ are message symbols. Note that there are $|\boldsymbol{m}|$ message symbols and $\frac{|\boldsymbol{m}|}{d_1 - z}$ polynomials of degree $d_1 - 1$. Each such polynomial has $d_1 - z$ non-key coefficients and so there are exactly enough coefficients to encode the message symbols. For $i > 1$, the non-key coefficients encode the degree d_i to $d_{i-1} - 1$ coefficients of all higher (than $d_i - 1$) degree polynomials. Note that there are $\sum_{j=1}^{i-1} p_j = \frac{|\boldsymbol{m}|}{d_{i-1} - z}$ higher degree polynomials and so the total number of coefficients to encode is $(d_{i-1} - d_i)\frac{|\boldsymbol{m}|}{d_{i-1} - z}$. On the other hand, there are p_i polynomials of degree $d_i - 1$, each of them has $d_i - z$ non-key coefficients, and so the total number of non-key coefficients is $(d_i - z)\left(\frac{|\boldsymbol{m}|}{d_i - z} - \frac{|\boldsymbol{m}|}{d_{i-1} - z}\right)$. It is trivial to verify that the two numbers are equal and so there are exactly enough coefficients to encode. Note that the specific way to map the coefficients is not important and any 1-1 mapping suffices. Finally, evaluate each polynomial at n non-zero points and store an evaluation of each polynomial at each node. This completes the scheme. Note that indeed the total number of polynomials is $\sum_{i=1}^{|\mathcal{D}|} p_i = \frac{|\boldsymbol{m}|}{d_{|\mathcal{D}|} - z} = \frac{|\boldsymbol{m}|}{k} = t$, implying that the scheme is rate-optimal.

8.2.2 Decoding

For any $d_i \in \mathcal{D}$, we describe the decoding algorithm of the scheme when d_i nodes are available. It achieves the optimal decoding bandwidth, and since $d_{|\mathcal{D}|} = n - r$

it implies that the scheme meets the reliability requirement. We first interpolate all polynomials of degree $d_i - 1$. After that, for all polynomials of degree $d_{i-1} - 1$, their coefficients of degree larger than $d_i - 1$ are known (as they are encoded in the coefficients of the polynomials of degree $d_i - 1$) and so they can be interpolated. In general, for $j \leq i$, once the polynomials of degree between $d_j - 1$ and $d_i - 1$ are interpolated, then for the polynomials of degree $d_{j-1} - 1$, their coefficients of degree larger than $d_i - 1$ are known by construction and so they can be interpolated. Therefore we can successively interpolate the polynomials of higher degree until the polynomials of degree $d_1 - 1$ are interpolated and so the message symbols are decoded. The total number of $\mathbb{F}_q$ symbols communicated is $d_i \sum_{j=1}^{i} p_j = d_i \frac{|\boldsymbol{m}|}{d_i - z}$. By Theorem 8.1.1, the decoding bandwidth is at least

$$|\boldsymbol{m}| + \frac{ktz}{d_i - z} = kt + \frac{ktz}{d_i - z} = kt\left(1 + \frac{z}{d_i - z}\right) = \frac{d_i|\boldsymbol{m}|}{d_i - z}$$

$\mathbb{F}_q$ symbols. Therefore the optimal decoding bandwidth is achieved.

8.2.3 Secrecy

We show that the scheme is secure against z eavesdropping nodes. At a high level, each polynomial individually can be viewed as a generalized Shamir's scheme, and so each polynomial individually is secure. The main idea is to show that if these polynomials are combined, the resulting scheme is still secure. We first prove that the generalized Shamir's scheme is indeed a valid secret sharing scheme and so is secure.

Theorem 8.2.1. *The following is an $(n, n-r-z, r, z)$ rate-optimal secret sharing scheme: For $q > n$, let $m_1, ..., m_{n-z-r}$ be message symbols over $\mathbb{F}_q$. Construct a polynomial $f(x)$ of degree $n-r-1$ over $\mathbb{F}_q$, whose degree 0 to $z-1$ coefficients are random keys, and the degree z to $n-r-1$ coefficients are m_1 to m_{n-r-z}. Evaluate $f(x)$ at n distinct non-zero points and assign one evaluation to each party.*

Proof. Note that the scheme described is identical to Construction 5.3.2 except that the positions of key and message symbols are swapped. Below we will prove the theorem assuming a slightly more general $f(x)$. Specifically, we allow $f(x)$ to be any degree-$(n-r-1)$ polynomial such that z of its coefficients of *consecutive* degrees are random keys, and the remaining coefficients are m_1 to m_{n-r-z}. Such a scheme generalizes Shamir's scheme as a special case when $n-r-z=1$ and the message m_1 is set to be the constant coefficient. The proof below also follows a similar line to the proof of Shamir's scheme [15].

The scheme is reliable because $f(x)$ can be interpolated from any $n-r$ evaluations so that the message symbols can be decoded. To prove the secrecy of the scheme, suppose that the degree i to $i+z-1$ coefficients of $f(x)$ are random keys $u_1, ..., u_z$, and that an adversary observes $C_1, ..., C_z$, which are the evaluations of $f(x)$ at z points $x_1, ..., x_z$. Then fixing a specific value of the message symbols, say $(m'_1, ..., m'_{n-r-z})$, there is one and only one specific value of the keys $(u'_1, ..., u'_z)$ such that the evaluation of the corresponding polynomial meets the observation of the adversary. This is because given $(m'_1, ..., m'_{n-r-z})$ and $C_1, ..., C_z$, one knows z evaluations of the polynomial $u'_1 x^i + u'_2 x^{i+1} + ... + u'_z x^{i+z-1}$, denoted by $D_1, ..., D_z$. Then by dividing D_j by x_j^i, $j \in [z]$, z evaluations of the polynomial $u'_1 + u'_2 x + ... + u'_z x^{z-1}$ are known. Therefore the unique $(u'_1, ..., u'_z)$ are obtained by interpolating this degree-$(z-1)$ polynomial. By construction, since every value of the keys is equally likely, the adversary cannot deduce any information about the message. □

The following lemma shows that combining two secure schemes is still secure as long as the keys used in the schemes meet certain independence conditions.

Lemma 8.2.1. *Consider random variables M_1, M_2, U_1, U_2 such that U_2 is independent of $\{M_1, U_1\}$. For $i = 1, 2$ Let F_i be a deterministic function of M_i, U_i. If $I(M_1; F_1) = 0$ and $I(M_2; F_2) = 0$, then $I(M_1; F_1, F_2) = 0$. In addition, if U_1 is independent of M_2, then $I(M_1, M_2; F_1, F_2) = 0$.*

Proof. We start with the first statement. Since F_2 is a function of U_2, M_2 but U_2 is independent of $\{M_1, U_1, F_1\}$, it follows that F_2 is independent of $\{M_1, U_1, F_1\}$ conditioning on M_2, implying the Markov chain $\{M_1, U_1, F_1\} \to M_2 \to F_2$. Therefore, $I(M_1, U_1, F_1, M_2; F_2) = I(M_2; F_2) = 0$, i.e., F_2 and $\{M_1, U_1, F_1, M_2\}$ are independent. Hence $I(M_1; F_1, F_2) = I(M_1; F_2) + I(M_1; F_1|F_2) \overset{(a)}{=} I(M_1; F_1|F_2) \overset{(b)}{=} I(M_1; F_1) = 0$, where (a) and (b) follows from the fact that F_2 is independent from $\{M_1, F_1\}$.

To prove the second statement, note that since U_1 is independent of M_2 and that F_1 is a function of M_1, U_1, we have the Markov Chain $M_2 \to M_1 \to F_1$, by which it follows that $I(M_1, M_2; F_1) = I(M_1; F_1) = 0$. Similarly because U_2 is independent of $\{M_1, U_1, F_1\}$ and that F_2 is a function of M_2, U_2, we have the Markov Chain $\{M_1, F_1\} \to M_2 \to F_2$. By this chain it follows that $I(M_1, F_1, M_2; F_2) = I(M_2; F_2) = 0$, i.e., $\{M_1, F_1, M_2\}$ is independent of F_2.

Therefore $I(M_1, M_2; F_2|F_1) = 0$ and so $I(M_1, M_2; F_1, F_2) = I(M_1, M_2; F_1) + I(M_1, M_2; F_2|F_1) = 0$. □

Suppose that the adversary compromises z nodes and obtains z evaluations of each polynomial. Consider the i-th polynomial in the order that we define them, let $\boldsymbol{f}_i$ denote the adversary's observation of this polynomial, let $\boldsymbol{u}_i$ denote the key coefficients of this polynomial and let $\boldsymbol{m}_i$ denote the non-key coefficients. By Theorem 8.2.1 we have

$$I(\boldsymbol{m}_i; \boldsymbol{f}_i) = 0, \quad i = 1, ..., b. \tag{8.6}$$

Consider the first p_1 polynomials which are polynomials of the highest degree $d_1 - 1$. By construction, $\boldsymbol{m}_1, ..., \boldsymbol{m}_{p_1}$ exactly encode the message $\boldsymbol{m}$. We invoke Lemma 8.2.1 by regarding $\boldsymbol{m}_1, \boldsymbol{u}_1, \boldsymbol{f}_1, \boldsymbol{m}_2, \boldsymbol{u}_2$ and $\boldsymbol{f}_2$ as M_1, U_1, F_1, M_2, U_2 and F_2. By the second statement of the lemma it follows that $I(\boldsymbol{m}_1, \boldsymbol{m}_2; \boldsymbol{f}_1, \boldsymbol{f}_2) = 0$. Inductively, for $1 < i < p_1$, suppose that $I(\boldsymbol{m}_1, ..., \boldsymbol{m}_i; \boldsymbol{f}_1, ..., \boldsymbol{f}_i) = 0$. We regard $\{\boldsymbol{m}_1, ..., \boldsymbol{m}_i\}$ as M_1, $\{\boldsymbol{u}_1, ..., \boldsymbol{u}_i\}$ as K_1, $\{\boldsymbol{f}_1, ..., \boldsymbol{f}_i\}$ as F_1, and regard $\boldsymbol{m}_{i+1}, \boldsymbol{u}_{i+1}, \boldsymbol{f}_{i+1}$ as M_2, U_2, F_2. It follows from Lemma 8.2.1 that $I(\boldsymbol{m}_1, ..., \boldsymbol{m}_{i+1}; \boldsymbol{f}_1, ..., \boldsymbol{f}_{i+1}) = 0$. By induction we have $I(\boldsymbol{m}_1, ..., \boldsymbol{m}_{p_1}; \boldsymbol{f}_1, ..., \boldsymbol{f}_{p_1}) = 0$.

We then regard $\{\boldsymbol{m}_1, ..., \boldsymbol{m}_{p_1}\} = \boldsymbol{m}$ as M_1, $\{\boldsymbol{u}_1, ..., \boldsymbol{u}_{p_1}\}$ as U_1, $\{\boldsymbol{f}_1, ..., \boldsymbol{f}_{p_1}\}$ as F_1, and regard $\boldsymbol{m}_{p_1+1}$, $\boldsymbol{u}_{p_1+1}$, $\boldsymbol{f}_{p_1+1}$ as M_2, U_2, F_2. Then it follows from the first statement of Lemma 8.2.1 that $I(\boldsymbol{m}; \boldsymbol{f}_1, ..., \boldsymbol{f}_{p_1+1}) = 0$. Inductively, for $p_1 < i < b$, suppose that $I(\boldsymbol{m}; \boldsymbol{f}_1, ..., \boldsymbol{f}_i) = 0$. We regard $\boldsymbol{m}$ as M_1, $\{\boldsymbol{u}_1, ..., \boldsymbol{u}_i\}$ as U_1, $\{\boldsymbol{f}_1, ..., \boldsymbol{f}_i\}$ as F_1, and regard $\boldsymbol{m}_{i+1}, \boldsymbol{u}_{i+1}, \boldsymbol{f}_{i+1}$ as M_2, U_2, F_2. By Lemma 8.2.1 we have $I(\boldsymbol{m}; \boldsymbol{f}_1, ..., \boldsymbol{f}_{i+1}) = 0$. By induction it follows that $I(\boldsymbol{m}; \boldsymbol{f}_1, ..., \boldsymbol{f}_t) = 0$, implying that the adversary learns no information about the message $\boldsymbol{m}$. This completes the proof and we have the following theorem.

Theorem 8.2.2. *For $\mathcal{D} \subset [n - r, n]$, the encoding scheme constructed in Section 8.2.1 is a rate-optimal (n, k, r, z) secret sharing scheme. The scheme achieves the optimal decoding bandwidth when d nodes participate in decoding, universally for all $d \in \mathcal{D}$.*

8.2.4 Discussion

We remark on some other important advantages and properties of our construction. Firstly, the scheme also achieves the *optimal number of symbol-read from disks* in decoding. To see this, notice that the lower bound (8.1) on communication is also

a lower bound on the number of Q-ary symbols that need to be read from disks during decoding. The number of symbol-read in the proposed scheme is equal to the amount of communication. Therefore our scheme achieves the lower bound and hence has optimal disk access complexity. Secondly, compared to most existing secret sharing schemes which decode from the minimum number of $n - r - z$ nodes, our scheme allows all available nodes (or more flexibly, any $d \in \mathcal{D}$ nodes) to participate in decoding and hence can help balance the load at the disks and achieve a higher degree of parallelization. Thirdly, the encoding and decoding of the scheme are similar to those of Shamir's scheme and therefore are efficient and practical. Particularly, the scheme works over the same field as Shamir's scheme. Fourthly, the preprocessing functions only rely on $d = |I|$ instead of I, further simplifying implementation. Fifthly, in practice when a user connects to more nodes, the increased latency may offset the benefit of the reduced bandwidth. One can overcome this issue by avoiding connections to nodes of large latency. If the latency of the nodes are not known a priori, one can start with connecting to all nodes and downloading the evaluations of the polynomials in decreasing order of degree. If some nodes do not respond in time, consider them as not available and switch adaptively to the mode of decoding from a smaller number of nodes. Finally, the construction is flexible in the parameters, i.e., it works for arbitrary n, r and z and $\mathcal{D}$.

Connection to other schemes: An important idea in our scheme is to construct multiple correlated polynomials of different degrees in order to facilitate decoding when different numbers of nodes are available. Similar ideas also appear in the schemes in [54], [55]. The main technique that enables the improvement of our schemes is a more careful and flexible design of the number and degree of the polynomials, as well as the arrangement of their coefficients.

Our scheme maps the high-degree coefficients of the higher degree polynomials into the coefficients of the lower degree polynomials, whereas the specific coefficient mapping is not important and any 1-1 mapping suffices. Additionally, for all polynomials in our scheme, the z lowest degree coefficients are independent keys. However, in general this is not necessary: in any polynomial, we can choose any consecutive z coefficients to be independent keys, and use the remaining coefficients to encode information (i.e., message symbols and coefficients of higher degree polynomials). The resulting scheme is a still valid (see the proof of Theorem 8.2.1) and achieves the optimal decoding bandwidth universally. Under this observation,

we note that our scheme generalizes the scheme in a recent independent work [65]. Particularly, our scheme is equivalent to the scheme in [65] if we require a specific coefficient mapping and let the z highest (instead of lowest) coefficients of all polynomial be keys[2].

In what follows we discuss the the advantage of our framework, which allows flexibility in choosing the coefficient mapping and the positions of the keys in a polynomial.

Coefficient mapping: As discussed above we can choose the coefficient mapping to be any 1-1 mapping. The flexibility in choosing the specific mapping is helpful in practice. Particularly, it is possible to improve the (computational) encoding complexity of the scheme substantially by choosing a mapping that maintains the order of the coefficients. Refer to Figure 7.2b for an example. We need to compute $m_4x + m_5x^2 + m_6x^3$ in evaluating $g(x)$, and we can reuse this computation in evaluating $f(x)$, because $f(x)$ contains the same run of consecutive coefficients $m_4x^4 + m_5x^5 + m_6x^6$. This for example will save 2 multiplications and 2 additions.

Arrangement of keys: We remark that choosing the lowest degree coefficients to be keys has several practical advantages. Decoding the scheme involves sequentially interpolating the polynomials through multiple iterations, which can lead to undesirable delay especially when $|\mathcal{D}|$ is large. To mitigate this issue, we wish to decode the message symbols "on the fly" in each iteration. Specifically, if d nodes are available, then each time a polynomial is interpolated, exactly d new message and/or key symbols are decoded. Since the number of symbols decoded in each interpolation, the total number of message symbols and the total number of key symbols to be decoded are all fixed, there is a trade-off between the decoding order of the key and message symbols. The location of keys in the polynomials plays a crucial role in this trade-off.

Formally, let $\beta(i)$ be the number of message symbols decoded after i iterations, i.e., after i polynomial interpolations. The optimal trade-off is to maximize $\beta(i)$ for every i. We first derive a simple upper bound of $\beta(i)$. Note that by the time that i polynomials have been interpolated, di symbols are decoded and among them there are at least zi keys since each polynomial introduces z independent keys for secrecy. Therefore $\beta(i) \leq (d - z)i$.

[2]The scheme in [65] also lets a node evaluate all polynomials at the same point, whereas this is not necessary in our framework.

Our scheme achieves this upper bound for all i. For example, in Figure 7.2b, if $d = 3$ nodes are available, then decoding involves 3 iterations and each iteration outputs $d - z = 2$ message symbols. In general, each iteration in decoding our scheme outputs $d-z$ message symbols. This is because by construction, only coefficients of degree higher than $d_{|\mathcal{D}|} = n - r > z$ will be mapped to the coefficients of the lower degree polynomials. Since we place the keys in the z lowest degree coefficients, they are never mapped. Therefore for any polynomial, its coefficients of degree larger than $z - 1$ encode message symbols. When a polynomial is interpolated, exactly z keys are decoded and the remaining $d - z$ symbols decoded are message symbols.

Achieving $\beta(i) = (d-z)i$ implies that at any moment during the decoding process, our scheme always decodes the maximum number of message symbols. In other words the decoding delay, measured in the number of iterations, averaged over all message symbols, is minimized. Moreover, the fact that each iteration decodes a fixed number of $d-z$ new message symbols may be helpful for implementation. On the other hand, note that choosing the z highest degree coefficients to be keys implies that the keys will be mapped to the coefficients of lower degree polynomials. Hence the keys will be decoded earlier than necessary (since lower degree polynomials are interpolated earlier) and it is not possible to achieve the optimal trade-off. Consider the example in Figure 7.2b, if we switch the keys to high degree coefficients, then the polynomials are $f(x) = m_1 + m_2x + m_3x^2 + m_4x^3 + m_5x^4 + m_6x^5 + k_1x^6$, $g(x) = m_5+m_6x+k_1x^2+k_2x^3$ and $h(x) = m_4+k_2x+k_3x^2$. In the case that $d = 4$ nodes are available, only 2 message symbols m_5, m_6 are decoded in the first iteration and the remaining 4 message symbols are decoded in the second (last) iteration. In comparison, the original scheme performs better by decoding 3 message symbols in each iteration. Finally, we remark that decoding the maximum number of message symbols on the fly is also beneficial in terms of partial decoding, i.e., decoding a subset of message symbols. In this case decoding can finish early if all symbols of interest are decoded, and our scheme maximizes the chance of finishing early.

Finally, we remark that using the random coding argument, we can design another secret sharing scheme that achieves the optimal decoding bandwidth universally [66]. However, such a scheme requires a significantly lager field size.

8.3 Construction from Reed-Solomon Codes

In this section we present another rate-optimal secret sharing scheme that achieves the optimal decoding bandwidth when all n nodes are available, namely $\mathcal{D} =$

$\{n-r, n\}$. The scheme is flexible in other parameters and hence is flexible in rate. The scheme is directly related to Reed-Solomon codes. Particularly, the encoding matrix of the scheme is a generator matrix of Reed-Solomon codes, and so the scheme can be decoded as Reed-Solomon codes. This is an advantage over the scheme in the previous section, which requires recursive decoding. The scheme also provides a stronger level of reliability in the sense that it allows decoding even if more than r shares are partially erased. On the other hand, unlike the previous scheme, this scheme does not achieve the optimal decoding bandwidth universally, but rather only for $d = n - r$ and $d = n$. However, we remark that the case of $d = n$ is particularly important because it corresponds to the best case in terms of decoding bandwidth and is arguably the most relevant case for the application of distributed storage, where the storage nodes are usually highly available.

8.3.1 Encoding

Fix $k = n - r - z$, let $q > n(k + r)$ be a prime power, and let $\mathcal{Q} = \mathbb{F}_q^{k+r}$. Note that each node stores a length $k + r$ vector over $\mathbb{F}_q$. For $j = 1, ..., n$, denote the share of node j by $c_j = (c_{1,j}, ..., c_{k+r,j})$, where $c_{i,j} \in \mathbb{F}_q$. The secret message $\boldsymbol{m}$ is k symbols over $\mathcal{Q}$ and therefore can be regarded as a length-$k(k + r)$ vector over $\mathbb{F}_q$, denoted by $(m_1, ..., m_{k(k+r)})$. The encoder generates keys $\boldsymbol{u} = (u_1, ..., u_{kz}) \in \mathbb{F}_q^{kz}$ and $\boldsymbol{u}' = (u'_1, ..., u'_{rz}) \in \mathbb{F}_q^{rz}$ independently and uniformly at random. The encoding scheme is linear over $\mathbb{F}_q$, and is described by an encoding matrix G over $\mathbb{F}_q$:

$$(c_{1,1}, ..., c_{1,n}, ..., c_{k+r,1}, ..., c_{k+r,n}) = (m_1, ..., m_{k(k+r)}, u_1, ..., u_{kz}, u'_1, ..., u'_{rz})G. \quad (8.7)$$

Note that G has $k(k + r) + kz + rz = nk + rz$ rows and has $n(k + r)$ columns. In the following we discuss the construction of G based on Vandermonde matrices. We start with some notation. Let $\alpha_1, ..., \alpha_{n(k+r)}$ be distinct non-zero elements of $\mathbb{F}_q$, and let $v_{ij} = \alpha_j^{i-1}$, $i = 1, ..., nk + rz$, $j = 1, ..., n(k + r)$, then $V = (v_{ij})$ is a Vandermonde matrix of the same size as G. Suppose $\boldsymbol{f} = (f_0, ..., f_i)$ is an arbitrary vector with entries in $\mathbb{F}_q$, we denote by $\boldsymbol{f}[x]$ the polynomial $f_0 + f_1 x + ... + f_i x^i$ over $\mathbb{F}_q$ with indeterminate x. We construct a set of polynomials as follows:

$$\boldsymbol{f}_i[x] = x^{i-1} \qquad i = 1, ..., kn, \quad (8.8)$$

$$\boldsymbol{f}_{kn+i}[x] = x^{i-1} \prod_{j=1}^{kn} (x - \alpha_j) \qquad i = 1, ..., rz. \quad (8.9)$$

Let $\boldsymbol{f}_i, i = 1, ..., kn + rz$ be the length-$(kn + rz)$ vectors over $\mathbb{F}_q$ corresponding to the polynomials. Stack the $\boldsymbol{f}_i$'s to obtain a square matrix of size $(kn + rz)$:

$$T = \begin{pmatrix} \boldsymbol{f}_1 \\ \vdots \\ \boldsymbol{f}_{kn+rz} \end{pmatrix}.$$

Finally, we complete the construction by setting

$$G = TV.$$

Example 8.3.1. *Consider the setting that* $n = 3, r = 1, z = 1$ *and* $k = n-r-z = 1$. *Let* $q = 7$ *and* $\mathcal{Q} = \mathbb{F}_q^2$. *Then* $\boldsymbol{m} = (m_1, m_2)$, $\boldsymbol{u} = (u_1)$ *and* $\boldsymbol{u}' = (u_1')$. *Construct a Vandermonde matrix over* $\mathbb{F}_q$ *as*

$$V = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 4 & 2 & 2 & 4 & 1 \\ 1 & 1 & 6 & 1 & 6 & 6 \end{pmatrix}. \tag{8.10}$$

Construct polynomials $\boldsymbol{f}_1[x] = 1$, $\boldsymbol{f}_2[x] = x$, $\boldsymbol{f}_3[x] = x^2$ *and*

$$\boldsymbol{f}_4[x] = (x-1)(x-2)(x-3) = 1 + 4x + x^2 + x^3.$$

Therefore,

$$T = \begin{pmatrix} \boldsymbol{f}_1 \\ \boldsymbol{f}_2 \\ \boldsymbol{f}_3 \\ \boldsymbol{f}_4 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 4 & 1 & 1 \end{pmatrix},$$

and the encoding matrix is given by

$$G = TV = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 4 & 2 & 2 & 4 & 1 \\ 0 & 0 & 0 & 6 & 3 & 4 \end{pmatrix}.$$

The properties of G are discussed in the following lemma.

Lemma 8.3.1. *Regard* G *as a block matrix*

$$G = \begin{pmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{pmatrix},$$

where G_{11} *has size* $kn \times kn$, G_{12} *has size* $kn \times rn$, G_{21} *has size* $rz \times kn$, *and* G_{22} *has size* $rz \times rn$. *Then,*

(i) Any $(n-r)(k+r)$ columns of G are linearly independent.

(ii) G_{11} is a Vandermonde matrix.

(iii) $G_{21} = 0$.

(iv) Any rz columns of G_{22} are linearly independent.

Proof. By construction, the polynomials $\boldsymbol{f}_i[x], i = 1, ..., kn + rz$ have distinct degrees and therefore are linearly independent. Therefore the rows of T are linearly independent and so T is full rank. This implies that the row space of G is the same as the row space of V. The row space of V is a linear $(nk+nr, nk+rz)$ Reed-Solomon code because that V is a Vandermonde matrix. Note that $nk+rz = (n-r)(k+r)$, and so the row space of G is a $(nk+nr, (n-r)(k+r))$ Reed-Solomon code. This proves (i).

To prove (ii), note that by (8.8), the first kn rows of G are exactly the first kn rows of V. Therefore G_{11} is a Vandermonde matrix.

To prove (iii), note that by construction the (i,j)-th entry of G_{21} equals $\boldsymbol{f}_{kn+i}[\alpha_j]$. By (8.9), α_j is a root of $\boldsymbol{f}_{kn+i}[x]$, for $i = 1, ..., rz$, $j = 1, ..., kn$. Hence $G_{21} = 0$.

Finally we prove (iv). By construction the (i,j)-th entry of G_{22} equals

$$\boldsymbol{f}_{kn+i}[\alpha_{kn+j}] = \alpha_{kn+j}^{i-1} \prod_{l=1}^{kn} (\alpha_{kn+j} - \alpha_l) = \alpha_{kn+j}^{i-1} \boldsymbol{f}^*[\alpha_{kn+j}], \tag{8.11}$$

where $\boldsymbol{f}^*[x] = \prod_{l=1}^{kn}(x - \alpha_l)$. Since $\alpha_1, ..., \alpha_{(k+r)n}$ are distinct elements, it follows that $\boldsymbol{f}^*[\alpha_{kn+j}] \neq 0$, for $j = 1, ..., rn$. Let $1 \leq j_1 < j_2 < ... < j_{rz} \leq rn$ and consider the submatrix formed by the j_1-th,...,j_{rz}-th columns of G_{22}. By (8.11), the l-th column of the submatrix is formed by consecutive powers of α_{kn+j_l}, scaled by $\boldsymbol{f}^*[\alpha_{kn+j_l}]$. Therefore the determinant of the submatrix is $\prod_{l=1}^{rz} \boldsymbol{f}^*[\alpha_{kn+j_l}] \prod_{1 \leq u < v \leq rz} (\alpha_{kn+j_v} - \alpha_{kn+j_u}) \neq 0$. This shows that any rz columns of G_{22} are linearly independent. □

8.3.2 Decoding

We describe the decoding procedure for two cases: 1) $|I| = n$, i.e., all nodes are available, and 2) $|I| < n$. First consider the case that $|I| = n$, i.e., $I = [n]$. In order to decode, for this case it suffices to read and communicate the first k symbols over $\mathbb{F}_q$ from each share. Formally, the user downloads $\boldsymbol{e} = (c_{1,1}, ..., c_{1,n}, ..., c_{k,1}, ..., c_{k,n})$.

By Lemma 8.3.1(ii), G_{11} is invertible. Denote the inverse of G_{11} by G_{11}^{-1}, then the message can be recovered by

$$\boldsymbol{e}G_{11}^{-1} \overset{(e)}{=} (m_1, ..., m_{k(k+r)}, u_1, ..., u_{kz}),$$

where (e) follows from (8.7) and Lemma 8.3.1(iii). The decoding process involves communicating kn symbols from $\mathbb{F}_q$. The communication overhead is kz symbols over $\mathbb{F}_q$ or $\frac{kz}{k+r} = \frac{kz}{n-z}$ $\mathcal{Q}$-ary symbols, which achieves the lower bound (8.1) and therefore is optimal.

Next consider the case that $n - r \leq |I| < n$. Select an arbitrary subset I' of I of size $n - r$, and download the complete share stored by the nodes in I'. Hence, the downloaded information $\boldsymbol{e}$ is a length-$(n - r)(k + r)$ vector over $\mathbb{F}_q$. By Lemma 8.3.1(i), it follows that any $(n - r)(k + r)$ columns in G are linearly independent and therefore the submatrix formed by these columns is invertible. The message $\boldsymbol{m}$ can then be recovered by multiplying $\boldsymbol{e}$ with the inverse. An alternative way to decode the message is to notice that G is an encoding matrix of a $(nk+nr, nk+rz)$ Reed-Solomon code over $\mathbb{F}_q$. Therefore one may employ the standard decoder of Reed-Solomon code to correct any $r(k+r)$ erasures or $\lfloor r(k+r)/2 \rfloor$ errors of symbols over $\mathbb{F}_q$. Note that when at most r nodes are unavailable, we regard their shares as erased and there are at most $r(k + r)$ erasures of symbols over $\mathbb{F}_q$, and therefore can be corrected. In general, any $r(k+r)$ erasures or $\lfloor r(k+r)/2 \rfloor$ errors of symbols over $\mathbb{F}_q$ are correctable even if they occur to more than r nodes. The decoding process involves communicating $nk + rz$ symbols of $\mathbb{F}_q$. The communication overhead is $(n-r)(k+r) - k(k+r) = z(k+r)$ symbols over $\mathbb{F}_q$, or z symbols over $\mathcal{Q}$, which achieves the lower bound (8.1) if and only if $|I| = n - r$.

8.3.3 Analysis

Theorem 8.3.1. *The encoding scheme constructed in Section 8.3.1 is a rate-optimal (n, k, r, z) secret sharing scheme. The scheme achieves the optimal decoding bandwidth when d nodes participate in decoding, for $d = n$ or $d = n - r$.*

Proof. We need to verify that the encoding scheme meets the reliability requirement and the security requirement of a secret sharing scheme. An explicit decoding scheme and its communication overhead are discussed in Section 8.3.2 and therefore the reliability requirement is met. The scheme is rate-optimal because $k = n-r-z$. We only need to show that the scheme is secure. By Lemma 4.3.1, it suffices to show that $H(\boldsymbol{k}, \boldsymbol{k}'|c_I, \boldsymbol{m}) = 0$, for all I such that $|I| = z$. In other words, the

random symbols generated by the encoder are completely determined by c_I and the message. Denote the submatrix formed by the first $k(k+r)$ rows of G by G_{top} and the submatrix formed by the remaining $(k+r)z$ rows of G by G_{low}. Consider any $I = \{i_1, ..., i_z\}$, and let $c_I = (c_{1,i_1}, ..., c_{1,i_z}, ..., c_{k+r,i_1}, ..., c_{k+r,i_z})$. It then follows from (8.7) that

$$c_I = (m_1, ..., m_{k(r+k)})G_{\text{top},I} + (u_1, ..., u_{kz}, u'_1, ..., u'_{rz})G_{\text{low},I},$$

where $G_{\text{top},I}$ is the submatrix formed by the subset of columns in $\{i+j | i \in I, j = 0, n, ..., (k+r-1)n\}$ of G_{top}, and $G_{\text{low},I}$ is the submatrix formed by the same subset of columns of G_{low}. Therefore, written concisely,

$$(\boldsymbol{u}\ \boldsymbol{u}')G_{\text{low},I} = c_I - \boldsymbol{m}G_{\text{top},I}. \tag{8.12}$$

To study the rank of $G_{\text{low},I}$, note that it is a square matrix of size $(k+r)z$, and we regard it as a block matrix

$$G_{\text{low},I} = \begin{pmatrix} G'_{11} & G'_{12} \\ G'_{21} & G'_{22} \end{pmatrix}, \tag{8.13}$$

where G'_{11} has size $kz \times kz$, G'_{12} has size $kz \times rz$, G'_{21} has size $rz \times kz$ and G'_{22} has size $rz \times rz$. By Lemma 8.3.1(ii), G'_{11} is a block of a Vandermonde matrix and therefore is invertible. By Lemma 8.3.1.(iii), $G'_{21} = 0$. Denote $c_I - \boldsymbol{m}G_{\text{top},I}$ by $(v_1, ..., v_{(k+r)z})$, then the above two facts together with (8.12) imply that

$$\boldsymbol{u} = (v_1, ..., v_{kz})G'^{-1}_{11}. \tag{8.14}$$

Therefore $\boldsymbol{u}$ is a function of $\boldsymbol{m}$ and c_I. It follows from (8.12) that

$$\boldsymbol{u}'G'_{22} = (v_{kz+1}, ..., v_{(k+r)z}) - \boldsymbol{u}G'_{12}.$$

By Lemma 8.3.1(iv), G'_{22} is invertible and therefore

$$\boldsymbol{u}' = \left((v_{kz+1}, ..., v_{(k+r)z}) - \boldsymbol{u}G'_{12}\right) G'^{-1}_{22}. \tag{8.15}$$

This shows that $\boldsymbol{u}'$ is a function of $\boldsymbol{u}$, c_I and $\boldsymbol{m}$, and so

$$H(\boldsymbol{u}, \boldsymbol{u}' | \boldsymbol{c}_I, \boldsymbol{m}) = 0. \tag{8.16}$$

The proof is complete. □

Theorem 8.3.1 shows that the proposed secret sharing scheme is optimal in terms of storage and the best-case (i.e., $|I| = n$) decoding bandwidth. Compared to the scheme in the previous section, this scheme has advantages in terms of implementation and error correction because decoding the scheme is equivalent to decoding standard Reed-Solomon codes. The scheme also provides a stronger level of reliability in the sense that it allows decoding even if more than r shares are partially erased. Similar to previous discussion, the scheme achieves the optimal number of symbol-read from disks when $|I| = n$. Finally, in the scheme all operations are performed over the field $\mathbb{F}_q$, where $q > n(k+r)$. This requirement on the field size can be relaxed in the following simple way. Let β be the greatest common divisor of k and r, then instead of choosing $\mathcal{Q}$ to be $\mathbb{F}_q^{k+r}$, we can let $\mathcal{Q} = \mathbb{F}_q^{\frac{k}{\beta}+\frac{r}{\beta}}$, $\boldsymbol{m} = (m_1, ..., m_{\frac{k(k+r)}{\beta}})$, $\boldsymbol{u} = (u_1, ..., u_{\frac{kz}{\beta}})$ and $\boldsymbol{u}' = (u'_1, ..., u'_{\frac{rz}{\beta}})$. The resulting scheme is a rate-optimal $(n, k, r, z)_{\mathcal{Q}}$ secret sharing scheme with the same decoding bandwidth as the original scheme. For this modified construction, it is sufficient to choose any field size $q > n\frac{k+r}{\beta}$.

8.4 Shamir's Scheme with Optimal Decoding Bandwidth

In this section we look at a different perspective on the decoding bandwidth problem. Instead of constructing new secret sharing schemes with optimal decoding bandwidth, we study the decoding bandwidth of the classical Shamir's scheme [15]. Though we have constructed two schemes with optimal decoding bandwidth in this chapter, improving the decoding bandwidth of Shamir's scheme remains an important problem as it is extensively used due to its simplicity. Below we describe a new family of Shamir's scheme with asymptotically optimal decoding bandwidth by extending the ideas recently developed in [56], [57] on repairing Reed-Solomon codes.

Consider Shamir's original scheme: let F be a finite field of size $|F| > n$, and let $\alpha_0, \cdots, \alpha_{t-1}$ be t elements in F, where α_0 is the secret message and $\alpha_1, \cdots, \alpha_{t-1}$ are randomly selected keys. Let $\lambda_1, \cdots, \lambda_n$ be any n distinct non-zero elements in F and let $f(x) = \sum_{i=0}^{t-1} \alpha_i x^i$. Then the n shares are $f(\lambda_1), \cdots, f(\lambda_n)$. Shamir's scheme is an $(n, k = 1, r = n - t, z = t - 1)$ scheme, denoted for short as an (n, t) threshold scheme, i.e., from any t shares α_0 can be decoded by polynomial interpolation, and any $t-1$ shares reveal no information about α_0. Clearly, decoding α_0 by polynomial interpolation requires communication t symbols over F. In what follows we show that by choosing F and the set of evaluation points $\lambda_1, \cdots, \lambda_n$ carefully, it is possible to reduce the decoding bandwidth (when all n shares are

available) to a fraction of approximately $\frac{n}{t(n-t+1)}$ of the original bandwidth. We need to slightly generalize the way that the secret message m is encoded: rather than setting $\alpha_0 = m$, we let $\alpha_0 = m - \sum_{i=1}^{t-1} \lambda_0^i \alpha_i$, for some $\lambda_0 \in F$. In other words $m = \sum_{i=0}^{t-1} \lambda_0^i \alpha_i = f(\lambda_0)$. The corresponding scheme is an (n, t) threshold scheme as long as $\lambda_i \neq \lambda_0$, $i \in [n]$. Note that Shamir's original scheme corresponds to the case that $\lambda_0 = 0$.

To reduce the decoding bandwidth, we follow the framework proposed in [56] of interpolating polynomials by querying partial polynomial evaluations. Specifically, let F be the extension field of degree l of a subfield K. During decoding, each of the n nodes applies K-linear transforms to the share over F that it holds to obtain a set of symbols over K. The decoder collects these sets of symbols and performs K-linear transforms on them to assemble the secret message over F. Formally, viewing F as a vector space of dimension l over K, it is shown in: [56] that

Lemma 8.4.1. *For a finite field K and its degree-l extension field F, let f be a polynomial over F of degree $< t$, and $f(\lambda_1), \cdots, f(\lambda_n)$ be n evaluations. Let λ_0 be an element in F, and let $g_1(x), \cdots, g_l(x)$ be l polynomials over F of degree $< n - t + 1$, such that $\{g_i(\lambda_0) : i \in [l]\}$ is a basis of F over K. Then to determine $f(\lambda_0)$, it suffices to know the set of values $\bigcup_{i\in[n]} \{\text{tr}\,(g_j(\lambda_i) f(\lambda_i)) : j \in [l]\}$, where $\text{tr} : F \to K$ is the trace function.*

The task of decoding the scheme is equivalent to determining $f(\lambda_0)$ and therefore it suffices to download the set of values $\{\text{tr}\,(g_j(\lambda_i) f(\lambda_i)) : j \in [l]\}$ from node i, for $i \in [n]$. However, due to the linearity of the trace function, we can reconstruct this set from values in $\{\text{tr}(\beta f(\lambda_i)) : \beta \in B_i\}$, where B_i is a basis (over K) of $\text{span}[\{g_j(\lambda_i) : j \in [l]\}]$. Therefore, the number of symbols over K that we need to download from node i equals $|B_i| = \dim(\text{span}[\{g_j(\lambda_i)\}_{j\in[l]}])$. We now discuss a way to select λ_i, $i = 0, \cdots, n$ as well as $g_i(x)$, $i \in [l]$, so that the condition in Lemma 8.4.1 is satisfied and that $|B_i|$, $i \in [n]$ is minimized. We remark that our construction idea is inspired by the codes in [57].

Construction 8.4.1. *For any n, t, let $s = n-t+1$ and $l = \tau s$ for some $\tau \geq n+1$. Let K be a finite field and $h(x) \in K[x]$ be a degree l irreducible polynomial. Let β be a root of $h(x)$, and F be the field generated by β over K. Let $\lambda_0 = \beta$, $\lambda_i = \beta^{is}$, $i \in [n]$, and let $\{g_i(x) : i \in [l]\} = \{\beta^a x^b : a = 0, s, \cdots, (\tau-1)s,\ b = 0, \cdots, s-1\}$.*

Theorem 8.4.1. *The (n, t) Shamir's scheme obtained by choosing F and $\{\lambda_i : i = 0, \cdots, n\}$ according to Construction 8.4.1 attains a decoding bandwidth of less than $\frac{nl}{s} + \frac{n^2 s^2}{4}$ symbols over K.*

Proof. Define $\{g_i(x) : i \in [l]\}$ according to Construction 8.4.1. Then it follows that $\{g_i(\lambda_0), i \in [l]\} = \{\beta^0, \beta^1, \cdots, \beta^{l-1}\}$, which is a basis of F over K. Therefore the condition of Lemma 8.4.1 is satisfied and we invoke the lemma to decode $f(\lambda_0)$. We now analyze the size of $\{g_j(\lambda_i) : j \in [l]\}$, for $i \in [n]$:

$$\begin{aligned}
&\{g_j(\lambda_i) : j \in [l]\} \\
&\quad = \{\beta^a \beta^{isb} : a = us, u \in [0, \tau - 1], b \in [0, s - 1]\} && (8.17) \\
&\quad \subset \{\beta^{us} : u \in [0, \tau - 1]\} \cup \\
&\qquad\qquad \{\beta^{a+isb} : a + isb \geq l, a = us, u \in [0, \tau - 1], b \in [0, s - 1]\}. && (8.18)
\end{aligned}$$

Denote the two sets in the R.H.S. of (8.18) by I_1 and I_2, respectively. Then $|I_1| = \tau = l/s$ and the size of I_2 is bounded by

$$\begin{aligned}
|I_2| &\leq \sum_{j=1}^{s-1} |\{\beta^{a+isb} : a = (\tau - (j-1)i - k)s,\ k \in [i],\ b \in [j, s-1]\}| && (8.19) \\
&= \sum_{j=1}^{s-1} i(s - j) = \frac{is(s-1)}{2}. && (8.20)
\end{aligned}$$

Therefore,

$$\begin{aligned}
\sum_{i=1}^{n} \dim\left(\text{span}[\{g_j(\lambda_i) : j \in [l]\}]\right) &\leq \sum_{i=1}^{n} |\{g_j(\lambda_i) : j \in [l]\}| && (8.21) \\
&\leq n\tau + \sum_{i=1}^{n} \frac{is(s-1)}{2} && (8.22) \\
&= \frac{nl}{s} + \frac{ns(n+1)(s-1)}{4} && (8.23) \\
&< \frac{nl}{s} + \frac{n^2 s^2}{4}. && (8.24)
\end{aligned}$$

By the remarks after Lemma 8.4.1, no more than $\frac{nl}{s} + \frac{n^2 s^2}{4}$ symbols over K need to be downloaded, proving the theorem. □

Note that by Theorem 8.1.1 the lower bound on the decoding bandwidth is $\frac{nl}{s}$ symbols over K and by Theorem 8.4.1, the decoding bandwidth of the proposed scheme is less than $(1 + \frac{ns^3}{4l})\frac{nl}{s}$ symbols. Therefore as $l \to \infty$, the decoding

bandwidth is asymptotically optimal in the sense that the ratio between the actual decoding bandwidth and the optimal decoding bandwidth approaches 1. Particularly, to achieve optimality it suffices to choose an l that dominates ns^3.

Chapter 9

SECRET SHARING SCHEMES WITH OPTIMAL DECODING AND REPAIR

9.1 General Construction Framework

Consider a rate-optimal $(n, k = n - r - z, r, z)$ secret sharing scheme over $\mathbb{F}_q^t$, and let $\boldsymbol{m}$ be the message. Recall that t is the number of $\mathbb{F}_q$ symbols stored by a node and is often referred to as the level of sub-packetization. Let d_2 be the number of nodes participating in decoding $\boldsymbol{m}$, the decoding bandwidth is the number of symbols over $\mathbb{F}_q$ to be transmitted from the d_2 nodes to the decoder. The optimal decoding bandwidth equals $\frac{kd_2t}{d_2-z}$ by Corollary 8.1.1, referred to as the d_2-optimal decoding bandwidth. Similarly, in the case that h nodes are failed, let d_1 be the number of available nodes participating in the repair process, then the *repair bandwidth* is the number of symbols over $\mathbb{F}_q$ to be transmitted from the d_1 nodes (here we assume that the repair process is secure, namely a trusted dealer will receive the symbols and reconstruct the loss symbols). The (d_1, h)-optimal repair bandwidth equals $\frac{hd_1t}{h+d_1-k-z}$ [58]. When $h = 1$, we refer to it as the d_1-optimal repair bandwidth. In this chapter we focus on designing secret sharing schemes with both optimal decoding bandwidth and optimal repair bandwidth.

We first formalize a connection between MDS codes and secret sharing schemes. The following theorem is a generalization of the result in [67] to the case of $k > 1$.

Theorem 9.1.1. *For any k, z, let $k' = k + z$ and $n' > 2k + z$, an $[n', k']$ MDS code $\mathcal{C}$ implies a $(n = n' - k, k, r = n - k', z)$ secret sharing scheme $\mathcal{S}$, obtained as follows: Encode $\mathcal{C}$ systematically so that among the k' information nodes, k of them store secret information and the remaining z nodes store uniformly distributed keys. Then discard the k nodes storing the secret information. The remaining n nodes store the n shares of $\mathcal{S}$.*

Proof. Since the minimum distance of $\mathcal{C}$ is $n' - k' + 1$, the minimum distance of the codewords of $\mathcal{S}$ is $n' - k' + 1 - k = n - k' + 1$. This proves the reliability of $\mathcal{S}$. Denote the secret information by a length-k vector $\boldsymbol{m}$ and denote the keys by a length-z vector $\boldsymbol{u}$. Let G be the encoding matrix of $\mathcal{S}$, i.e., the shares are $(\boldsymbol{m}, \boldsymbol{u})G = \boldsymbol{m}G_{\text{up}} + \boldsymbol{u}G_{\text{low}}$. Then to prove the secrecy of $\mathcal{S}$ it suffices to prove

that any subset of z entries of $\boldsymbol{u}G_{\text{low}}$ is uniformly distributed. Let $G' = [I \mid P']$ be the systematic generator matrix of $\mathcal{C}$, where I is the identity matrix of order k'. Then by construction $G_{\text{low}} = [I \mid P]$ where I is the identity matrix of order z, and P is a submatrix of P'. Since $\mathcal{C}$ is MDS, by [39, Ch.11, Theorem 8], every square submatrix of P' is non-singular and therefore every square submatrix of P is also non-singular. Again by [39] this implies that G_{low} is the systematic generator matrix of a MDS code and therefore $\boldsymbol{u}$ can be decoded from any subset of z entries of $\boldsymbol{u}G_{\text{low}}$. Since $\boldsymbol{u}$ is uniformly distributed, it implies that any subset of z entries of $\boldsymbol{u}G_{\text{low}}$ is uniformly distributed, proving the theorem. □

By Theorem 9.1.1 we can construct secret sharing schemes from MDS codes. Particularly, the resulting secret sharing schemes are rate-optimal. The next result formally connects the decoding and repair bandwidth of the secret sharing schemes to those of the MDS codes. A similar observation is recently made in [61].

Theorem 9.1.2. *If $\mathcal{C}$ allows 1) repair of individual non-discarded nodes from any $d_1 \leq n-1$ of the remaining available nodes with optimal bandwidth and 2) simultaneous repair of all k discarded nodes from any $d_2 \leq n$ of the available nodes with optimal bandwidth, then the resulting secret sharing scheme $\mathcal{S}$ achieves d_1-optimal repair bandwidth and d_2-optimal decoding bandwidth.*

Proof. For $\mathcal{S}$, 1) corresponds to the operation of repairing individual nodes and 2) corresponds to the operation of decoding. First consider 1). Note that the bandwidth of repairing a node in $\mathcal{S}$ from d_1 nodes is lower bounded by the optimal bandwidth of repairing a node from d_1 nodes in an $[n, k']$ MDS code (because $\mathcal{S}$ is a $[n, k']$ MDS code) which equals $\frac{d_1 t}{1+d_1-k'}$ symbols and is achieved by $\mathcal{C}$ by hypothesis.

Now consider 2), the optimal bandwidth to repair k nodes in $\mathcal{C}$ from d_2 nodes is $\frac{kd_2t}{k+d_2-k'} = \frac{kd_2t}{d_2-z}$ symbols which matches the lower bound on decoding bandwidth. This shows that $\mathcal{S}$ achieves d_2-optimal decoding bandwidth. □

By Theorem 9.1.1 and Theorem 9.1.2, we can immediately obtain secret sharing schemes with optimal repair and decoding bandwidth from the regenerating codes in [62].

Corollary 9.1.1. *For any n, r, z, there exists a rate-optimal $(n, k = n-r-z, r, z)$ secret sharing scheme with d_1-optimal repair bandwidth and d_2-optimal decoding bandwidth, universally for all $k+z \leq d_1 \leq n-1$, and $k+z \leq d_2 \leq n$.*

Proof. Apply Theorem 9.1.1 and Theorem 9.1.2 to Construction 3 or Construction 6 in [62]. □

Recall that the access complexity is the amount of symbol-read from disks. We remark that Construction 6 in [62] is both bandwidth-optimal and access-optimal. Therefore the resulting secret sharing scheme not only achieves the optimal repair and decoding bandwidth universally, but also achieves the optimal access complexity universally during repair or decoding.

We note that, however, the above schemes obtained by directly applying Theorems 9.1.1 and 9.1.2 to Construction 3 or Construction 6 in [62] are hardly practical. This is because the degree of sub-packetization t is prohibitive. Specifically, t grows double exponentially at the speed of $O\left(2^{(k+r)^n}\right)$. In the next section we will discuss constructions of schemes with a much smaller t. We first introduce some notation and a general construction framework.

Denote the entries of the codewords of $\mathcal{C}$ by C_i, $i = 1, \cdots, n'$. Each C_i is a column vector of length t over a finite field F. Adopting the framework in [62], we define $\mathcal{C}$ by its parity check equations:

$$\mathcal{C} = \{(C_1, \cdots, C_{n'}) : \sum_{i=1}^{n'} A_{l,i} C_i = 0,\ l = 1, \cdots, r'\} \tag{9.1}$$

where $r' = n' - k'$, and $A_{l,i}$, $l \in [r']$, $i \in [n']$ are $t \times t$ matrices over F. In this chapter we consider array codes that resemble the structure of a Vandermonde matrix, i.e., we let

$$A_{l,i} = A_i^{l-1}, \quad l \in [r'],\ i \in [n'] \tag{9.2}$$

where A_i, $i \in [n']$ are $t \times t$ matrices (with the convention that $A^0 = I$). In the next section we will construct $\mathcal{C}$ by designing specific A_i's.

9.2 Scheme with Optimal Decoding and Repair Bandwidth

The d_2-optimal decoding bandwidth of an $(n, k = n - r - z, r, z)$ scheme is $\frac{k d_2 t}{d_2 - z}$ symbols, implying that each of the d_2 nodes participating in decoding will transmit a fraction of $\frac{k}{d_2 - z}$ of the symbols that it stores. We start with the case that $\frac{k}{d_2 - z} = \frac{1}{\rho}$, where $\rho \geq 1$ is an integer, which allows a simplified scheme. The following construction is a generalization of Construction 2 in [62].

Construction 9.2.1. *Consider any $n, r, z, k = n - r - z$ and $k + z \leq d_1 \leq n - 1$, $k + z \leq d_2 \leq n$ such that $\frac{k}{d_2 - z} = \frac{1}{\rho}$. Let $n' = n + k$, $k' = k + z$, $r' = n' - k'$*

and F be a finite field of size $|F| \geq k\rho + ns$, where $s = d_1 + 1 - k'$. Let $\{\lambda_{i,j}\}_{i\in[n],j\in\{0,\cdots,s-1\}} \bigcup \{\lambda_{n+i,j}\}_{i\in[k],j\in\{0,\cdots,\rho-1\}}$ be distinct elements in F. Consider the code family given by (9.1) and (9.2), where $t = \rho s^n$ and

$$A_i = \sum_{a=0}^{t-1} \lambda_{i,a_i} e_a e_a^T, \quad i = 1, \cdots, n$$
$$A_{n+i} = \sum_{a=0}^{t-1} \lambda_{n+i,a_{n+1}} e_a e_a^T \quad i = 1, \cdots, k.$$

Here $\{e_a : a = 0, \cdots, t-1\}$ is the standard basis of F^t and we represent a using the $(n+1)$-digit notation $a = (a_{n+1}, a_n, \cdots, a_1)$, where $a_{n+1} \in \{0, \cdots, \rho - 1\}$ and $a_i \in \{0, \cdots, s-1\}$, for all $i \in [n]$.

Note that A_i, $i \in [n']$ are diagonal matrices and we can expand the parity-check equations (9.1) coordinatewise. Let $c_{i,a}$ denote the a-th entry of C_i, we have,

$$\sum_{i=1}^{n} \lambda_{i,a_i}^l c_{i,a} + \sum_{i=1}^{k} \lambda_{n+i,a_{n+1}}^l c_{n+i,a} = 0, \tag{9.3}$$

for $a \in \{0, \cdots, t-1\}$, $l \in \{0, \cdots, r'-1\}$.

Lemma 9.2.1. *The array code $\mathcal{C}$ given by Construction 9.2.1 is MDS.*

Proof. Writing (9.3) in matrix form, for all $a = 0, \cdots, t-1$, we have

$$\begin{bmatrix} 1 & \cdots & 1 & 1 & \cdots & 1 \\ \lambda_{1,a_1} & \cdots & \lambda_{n,a_n} & \lambda_{n+1,a_{n+1}} & \cdots & \lambda_{n',a_{n+1}} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \lambda_{1,a_1}^{r'-1} & \cdots & \lambda_{n,a_n}^{r'-1} & \lambda_{n+1,a_{n+1}}^{r'-1} & \cdots & \lambda_{n',a_{n+1}}^{r'-1} \end{bmatrix} \begin{bmatrix} c_{1,a} \\ c_{2,a} \\ \vdots \\ c_{n',a} \end{bmatrix} = 0. \tag{9.4}$$

Therefore any r' columns of the parity check matrix in (9.4) are linearly independent, implying that from any $n' - r' = k'$ elements of $\{c_{1,a}, \cdots, c_{n',a}\}$ we can recover the whole set. This shows that we can recover the set $\{C_1, \cdots, C_{n'}\}$ from any of its k' elements. Hence $\mathcal{C}$ is an MDS array code. □

Lemma 9.2.2. *The array code $\mathcal{C}$ given by Construction 9.2.1 attains optimal bandwidth when 1) repairing a single node i, $i \in [n]$, from any d_1 nodes, and 2) repairing the set of nodes $\{n+1, \cdots, n'\}$ from any d_2 nodes.*

Proof. Let $a(i,u) = (a_{n+1}, \cdots, a_{i+1}, u, a_{i-1}, \cdots, a_1)$. To prove the first statement of the theorem we claim that for $i \in [n]$ and $a = 0, \cdots, t-1$, the set of entries $\{c_{i,a(i,0)}, \cdots, c_{i,a(i,s-1)}\}$ of C_i can be recovered from any subset of d_1 symbols of the following set of $n'-1$ symbols over F:

$$\mu_{j,i}^{(a)} \triangleq \sum_{u=0}^{s-1} c_{j,a(i,u)}, \quad j \in [n'] \backslash i.$$

The claim implies that each of the d_1 nodes only needs to send one symbol in order to recover s symbols in C_i, achieving the optimal bandwidth. To prove the claim, by (9.3), for any $i \in [n]$, $a = 0, \cdots, t-1$, $l = 0, \cdots, r'-1$ and $u = 0, \cdots, s-1$, we have:

$$\lambda_{i,u}^l c_{i,a(i,u)} + \sum_{j \in [n] \backslash \{i\}} \lambda_{j,a_j}^l c_{j,a(i,u)} + \sum_{j=1}^{k} \lambda_{n+j,a_{n+1}}^l c_{n+j,a(i,u)} = 0. \tag{9.5}$$

Assume without loss of generality that $i = 1$. By summing (9.5) over $u = 0, \cdots, s-1$ we obtain

$$\begin{bmatrix} 1 & \cdots & 1 \\ \lambda_{1,0} & \cdots & \lambda_{1,s-1} \\ \vdots & \vdots & \vdots \\ \lambda_{1,0}^{r'-1} & \cdots & \lambda_{1,s-1}^{r'-1} \end{bmatrix} \begin{bmatrix} c_{1,a(1,0)} \\ \vdots \\ c_{1,a(1,s-1)} \end{bmatrix} = - \begin{bmatrix} 1 & \cdots & 1 & 1 & \cdots & 1 \\ \lambda_{2,a_2} & \cdots & \lambda_{n,a_n} & \lambda_{n+1,a_{n+1}} & \cdots & \lambda_{n',a_{n+1}} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \lambda_{2,a_2}^{r'-1} & \cdots & \lambda_{n,a_n}^{r'-1} & \lambda_{n+1,a_{n+1}}^{r'-1} & \cdots & \lambda_{n',a_{n+1}}^{r'-1} \end{bmatrix} \begin{bmatrix} \mu_{2,1}^{(a)} \\ \vdots \\ \mu_{n',1}^{(a)} \end{bmatrix}. \tag{9.6}$$

Note that in (9.6), there are r' equations and $s + (n'-1-d_1)$ unknown variables (s from the L.H.S. and $n'-1-d_1$ from the R.H.S.). Moreover, $s+(n'-1-d_1) = (d_1+1-k')+(n'-1-d_1) = n'-k' = r'$. Therefore the number of equations equals the number of variables. Below we show that it is indeed possible to solve all variables from (9.6). We follow an approach similar to that in [62]. Define polynomials $p_i(x) = x^i \prod_{u=0}^{s-1}(x - \lambda_{1,u})$, $i = 0, \cdots, r'-s-1$. Let $p_i(x) = \sum_{j=0}^{r'-1} p_{i,j} x^j$, and define matrix $P \triangleq (p_{i,j})_{i \in [0,r'-s-1], j \in [0,r'-1]}$. Then by construction, multiply P on

the left to (9.6) we have

$$P\begin{bmatrix} 1 & \cdots & 1 & 1 & \cdots & 1 \\ \lambda_{2,a_2} & \cdots & \lambda_{n,a_n} & \lambda_{n+1,a_{n+1}} & \cdots & \lambda_{n',a_{n+1}} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \lambda_{2,a_2}^{r'-1} & \cdots & \lambda_{n,a_n}^{r'-1} & \lambda_{n+1,a_{n+1}}^{r'-1} & \cdots & \lambda_{n',a_{n+1}}^{r'-1} \end{bmatrix} \begin{bmatrix} \mu_{2,1}^{(a)} \\ \vdots \\ \mu_{n',1}^{(a)} \end{bmatrix} = 0. \tag{9.7}$$

The product of the first two terms in (9.7) equals $Q = (q_{i,j})_{i\in[r'-s],j\in[n'-1]}$, where

$$\begin{aligned} q_{i,j} &= p_0(\lambda_{j+1,a_{j+1}})\lambda_{j+1,a_{j+1}}^{i-1}, \quad i \in [r'-s],\ j \in [n-1] \\ q_{i,n-1+j} &= p_0(\lambda_{n+j,a_{n+1}})\lambda_{j+1,a_{n+1}}^{i-1}, \quad i \in [r'-s],\ j \in [k]. \end{aligned}$$

Therefore Q is a Vandermonde matrix in which each column is scaled by a non-zero constant. Therefore any $r'-s$ columns of Q are linearly independent, implying that from any $n'-1-(r'-s) = d_1$ elements of $\{\mu_{2,1}^{(a)}, \cdots, \mu_{n',1}^{(a)}\}$ we can recover the whole set. And then we can recover $\{c_{1,a(1,0)}, \cdots, c_{1,a(1,s-1)}\}$ from (9.6). This proves the claim and the first statement of the theorem.

We now prove the second statement and claim that for any $a = 0, \cdots, t-1$, the set of entries $\{c_{n+i,a(n+1,j)} : i \in [k], j \in [0, \rho-1]\}$ can be recovered from any subset of d_2 elements of the set $\{\mu_{j,n+1}^{(a)} : j \in [n]\}$. In other words, each of the d_2 nodes only needs to send one symbol in order to decode ρ symbols, achieving the optimal decoding bandwidth. To prove the claim, from (9.3) we have:

$$\sum_{j=1}^{n} \lambda_{j,a_j}^{l} c_{j,a(j,u)} + \sum_{j=1}^{k} \lambda_{n+j,u}^{l} c_{n+j,a(n+1,u)} = 0.$$

Summing over u we have

$$\begin{bmatrix} 1 & \cdots & 1 & \cdots & 1 & \cdots & 1 \\ \lambda_{n+1,0} & \cdots & \lambda_{n+1,\rho-1} & \cdots & \lambda_{n+k,0} & \cdots & \lambda_{n+k,\rho-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \lambda_{n+1,0}^{r'-1} & \cdots & \lambda_{n+1,\rho-1}^{r'-1} & \cdots & \lambda_{n+k,0}^{r'-1} & \cdots & \lambda_{n+k,\rho-1}^{r'-1} \end{bmatrix} \begin{bmatrix} c_{n+1,a(n+1,0)} \\ \vdots \\ c_{n+1,a(n+1,\rho-1)} \\ \vdots \\ c_{n+k,a(n+1,0)} \\ \vdots \\ c_{n+k,a(n+1,\rho-1)} \end{bmatrix}$$

$$= \begin{bmatrix} \mu_{1,n+1}^{(a)} \\ \vdots \\ \mu_{n,n+1}^{(a)} \end{bmatrix}. \tag{9.8}$$

Note that in (9.8) we have r' equations and $k\rho+n-d_2$ unknown variables ($k\rho$ from the L.H.S. and $n-d_2$ from the R.H.S.). But $k\rho+n-d_2=r'$ and the number of equations equals the number of variables. Similar to the way that we treat (9.6), we can recover $\{c_{n+i,a(n+1,j)} : i\in[k], j\in[0,\rho-1]\}$ by solving (9.8). This proves the claim and the second statement of the theorem. □

By Lemmas 9.2.1, 9.2.2 and Theorem 9.1.2, we have:

Theorem 9.2.1. *The secret sharing scheme obtained by applying Theorem 9.1.1 to the code given in Construction 9.2.1, where the last k nodes are discarded, is an $(n, k=n-r-z, r, z)$ scheme with d_1-optimal repair bandwidth and d_2-optimal decoding bandwidth.*

Next we generalize Construction 9.2.1 to the case of arbitrary $\frac{k}{d_2-z}$.

Construction 9.2.2. *For any $n, r, z, k=n-r-z$, $k+z\leq d_1\leq n-1$ and $k+z\leq d_2\leq n$, let $n'=n+k$, $k'=k+z$, $r'=n'-k'$ and $s=d_1+1-k'$. Let $\theta=\gcd(k,d_2-z)$, $\tau=\frac{k}{\theta}$, $\rho=\frac{d_2-z}{\theta}$ and $\delta=\rho-\tau$. Let F be a finite field of size $|F|>sn+\sum_{i=1}^{\tau}(i+\delta)$, and let $\{\lambda_{i,j} : i\in[n], j=0,\cdots,s-1\}\bigcup\{\lambda_{n+i,j} : i\in[k], j=0,\cdots,\lfloor\frac{i-1}{\theta}\rfloor+\delta\}$ be distinct elements in F. Let $t=s^n\prod_{i=1}^{\tau}(i+\delta)$. Consider the code family given by (9.1) and (9.2), where*

$$A_i=\sum_{a=0}^{t-1}\lambda_{i,a_i}e_a e_a^T,\quad i=1,\cdots,n$$
$$A_{n+i}=\sum_{a=0}^{t-1}\lambda_{n+i,a_{n+\lceil\frac{i}{\theta}\rceil}}e_a e_a^T\quad i=1,\cdots,k.$$

Here $\{e_a : a=0,\cdots,t-1\}$ is the standard basis of F^t and we represent a using the $(n+\tau)$-digit notation $a=(a_{n+\tau},\cdots,a_1)$, where $a_i\in\{0,\cdots,s-1\}$, for $i\in[n]$ and $a_{n+i}\in\{0,\cdots,i+\delta-1\}$, for $i\in[\tau]$.

Following an argument similar to the proof of Lemma 9.2.1, it is easy to show that the code given by Construction 9.2.2 is an MDS array code.

Lemma 9.2.3. *The array code $\mathcal{C}$ given by Construction 9.2.2 attains optimal bandwidth when 1) repairing a single node i, $i\in[n]$, from any d_1 nodes, and 2) repairing the set of nodes $\{n+1,\cdots,n'\}$ from any d_2 node.*

Proof. We omit the proof of the first statement because it is similar to the proof of Lemma 9.2.2. To prove the second statement, the key idea is to divide the $(n+1)$-th to the $(n+k)$-th nodes into τ groups and repair the groups one by one iteratively. Formally, let $\mathcal{C}_R \subset \{C_1, \cdots, C_n\}$ be the set of nodes accessed, with $|\mathcal{C}_R| = d_2$. For $i = 1, \cdots, \tau$, let $\mathcal{C}_i = \{C_{n+(i-1)\theta+1}, \cdots, C_{n+(i-1)\theta+\theta}\}$, then we first use $\mathcal{C}_R$ to repair $\mathcal{C}_1$, then use $\mathcal{C}_R \cup \mathcal{C}_1$ to repair $\mathcal{C}_2$, $\cdots$, and finally use $\mathcal{C}_R \cup \mathcal{C}_1 \cup \cdots \cup \mathcal{C}_{\tau-1}$ to repair $\mathcal{C}_\tau$. By the proof of Lemma 9.2.2, we can recover $\mathcal{C}_i$ from $\{\sum_{u=0}^{\delta+i-1} c_{v,a(n+i,u)} : a_{n+i} = 0, C_v \in \mathcal{C}_R \cup \mathcal{C}_{[i-1]}\}$. Since the nodes in $\mathcal{C}_{[i-1]}$ are already recovered, it suffices to know the set of values

$$\Omega_i = \left\{ \sum_{u=0}^{\delta+i-1} c_{v,a(n+i,u)} : a_{n+i} = 0, C_v \in \mathcal{C}_R \right\}.$$

Therefore to recover all τ groups of nodes, it suffices to know the values in the set $\Lambda = \bigcup_{i=1}^{\tau} \Omega_i$. We remark that some values in Λ can be derived from other values in the set and so it suffices to download a spanning set of Λ. Let

$$\Lambda^* = \bigcup_{i=1}^{\tau} \left\{ \sum_{u=0}^{\delta+i-1} c_{v,a(n+i,u)} : a_{n+i} = 0, a_{n+j} < \delta + j - 1, j \in [i-1], C_v \in \mathcal{C}_R \right\}.$$

Clearly $\Lambda^* \subset \Lambda$. We claim that every value in Λ can be determined by the values in Λ^*. To prove the claim it suffices to show that for $i \in [\tau]$, $\Omega_i \subset \Lambda^*$. We prove by induction on i. Clearly $\Omega_1 \subset \Lambda^*$. Now suppose that $\Omega_1, \cdots, \Omega_{i-1} \in \Lambda^*$, and consider the set

$$\Omega_i \backslash \Lambda^* = \left\{ \sum_{u=0}^{\delta+i-1} c_{v,a(n+i,u)} : a_{n+i} = 0, a_{n+j} = \delta + j - 1, j \in [i-1], C_v \in \mathcal{C}_R \right\}.$$

Consider an arbitrary element $\sum_{u=0}^{\delta+i-1} c_{v,a(n+i,u)}$ of $\Omega_i \backslash \Lambda^*$, so that a satisfies $a_{n+i} = 0$ and $a_{n+j} = \delta + j - 1$ for some $j \leq i-1$. Denote by

$$a(j, i, x, y) = (\cdots, a_{j-1}, x, a_{j+1}, \cdots, a_{i-1}, y, a_{i+1}, \cdots).$$

Since $\Omega_j \subset \Lambda^*$, it follows that $\sum_{u_1=0}^{\delta+j-1} c_{v,a(n+j,n+i,u_1,u_2)} \in \Lambda^*$, for all $u_2 = 0, \cdots, \delta+i-1$. Therefore

$$\sum_{u_2=0}^{\delta+i-1} \sum_{u_1=0}^{\delta+j-1} c_{v,a(n+j,n+i,u_1,u_2)} \in \text{span}(\Lambda^*). \tag{9.9}$$

Moreover, by construction $\sum_{u_2=0}^{\delta+i-1} c_{v,a(n+j,n+i,u_1,u_2)} \in \Lambda$ for all $u_1 = 0, \cdots, \delta+j-2$. Therefore

$$\sum_{u_1=0}^{\delta+j-2} \sum_{u_2=0}^{\delta+i-1} c_{v,a(n+j,n+i,u_1,u_2)} \in \text{span}(\Lambda^*). \tag{9.10}$$

Subtracting (9.10) from (9.9) we have

$$\sum_{u_2=0}^{\delta+i-1} c_{v,a(n+j,n+i,\delta+j-1,u_2)} = \sum_{u=0}^{\delta+i-1} c_{v,a(n+i,u)} \in \text{span}(\Lambda^*).$$

This proves that $\Omega_i \in \text{span}(\Lambda^*)$ and the claim. We now analyze the size of Λ^*. Let

$$\Omega_i^* = \left\{ \sum_{u=0}^{\delta+i-1} c_{v,a(n+i,u)} : a_{n+i} = 0, a_{n+j} < \delta + j - 1, j \in [i-1], C_v \in \mathcal{C}_R \right\}$$

so that $\Lambda^* = \bigcup_{i=1}^{\tau} \Omega_i^*$. By counting the elements of the set we have

$$|\Omega_i^*| = s^n \cdot \prod_{j=1}^{i-1} (\delta + j - 1) \cdot \prod_{j=i+1}^{\tau} (\delta + j).$$

We claim that

$$\sum_{j=1}^{i} |\Omega_j^*| = s^n \frac{i}{\delta + i} \prod_{j=1}^{\tau} (\delta + j). \tag{9.11}$$

We prove (9.11) by induction on i. Clearly $|\Omega_1^*| = s^n \prod_{j=2}^{\tau} (\delta+j) = s^n \frac{1}{\delta+1} \prod_{j=1}^{\tau} (\delta + j)$. Now suppose that (9.11) is true up to $i-1$, then it follows

$$\begin{aligned}
\sum_{j=1}^{i} |\Omega_j^*| &= \sum_{j=1}^{i-1} |\Omega_j^*| + |\Omega_i^*| \\
&= s^n \frac{i-1}{\delta+i-1} \prod_{j=1}^{\tau} (\delta+j) + s^n \prod_{j=1}^{i-1} (\delta+j-1) \prod_{j=i+1}^{\tau} (\delta+j) \\
&= s^n \left(\frac{i-1}{\delta+i-1} \prod_{j=1}^{i} (\delta+j) + \prod_{j=1}^{i-1} (\delta+j-1) \right) \prod_{j=i+1}^{\tau} (\delta+j) \\
&= s^n \left(\frac{i-1}{\delta+i-1} \prod_{j=1}^{i} (\delta+j) + \frac{\delta}{(\delta+i)(\delta+i-1)} \prod_{j=1}^{i} (\delta+j) \right) \prod_{j=i+1}^{\tau} (\delta+j) \\
&= s^n \left(\frac{i-1}{\delta+i-1} + \frac{\delta}{(\delta+i)(\delta+i-1)} \right) \prod_{j=1}^{\tau} (\delta+j) \\
&= s^n \frac{i}{\delta+i} \prod_{j=1}^{\tau} (\delta+j),
\end{aligned}$$

proving (9.11). Therefore $|\Lambda^*| = \sum_{j=1}^{\tau} |\Omega_j^*| = \frac{\tau}{\rho} s^n \prod_{j=1}^{\tau} (\delta+j)$. Note that $|\Lambda^*|$ is the number of symbols over F that need to be downloaded from $\mathcal{C}_R$ and the total number of symbols stored by $\mathcal{C}_R$ is $t = s^n \prod_{j=1}^{\tau} (\delta+j)$. Therefore a fraction of $\frac{\tau}{\rho}$ of the symbols are downloaded which attains the lower bound. The proof is complete. □

By Lemmas 9.2.3 and Theorem 9.1.2, we have:

Theorem 9.2.2. *The secret sharing scheme obtained by applying Theorem 9.1.1 to the code given in Construction 9.2.2, where the last k nodes are discarded, is a $(n, k = n - r - z, r, z)$ scheme with d_1-optimal repair bandwidth and d_2-optimal decoding bandwidth.*

Finally, we remark that Construction 9.2.1 and Construction 9.2.2 both have a sub-packetization level of $t = O(s^n)$, which is comparable to existing regenerating codes, e.g., [62], of the same parameters (note that our secret sharing schemes can be viewed as regenerating codes). In fact, an exponential t is shown to be necessary in order to achieve the optimal repair bandwidth [68]. This suggests that the additional optimal decoding requirement has a small impact on t. We also remark that Construction 9.2.2 naturally generalizes to a family of regenerating codes that supports centralized repair of groups of nodes of flexible sizes with reduced sub-packetization, which is a result of separate interest. A similar centralized repair problem was studied in [61] whereas the code construction therein is restricted in parameters.

9.3 Scheme with Optimal Decoding and Repair Access

In this section we construct secret sharing schemes that not only achieve the optimal decoding and repair bandwidth, but also the optimal access complexity during decoding and repair. The d_2-optimal decoding bandwidth and access is achieved if a fraction of $\frac{k}{d_2-z}$ of the symbols stored by each of the d_2 nodes are accessed and transmitted during decoding. As before, we first study the simplified case that $\frac{k}{d_2-z} = \frac{1}{\rho}$ for some integer ρ. The following construction is a generalization of Construction 6 in [62].

Construction 9.3.1. *Consider any $n, r, z, k = n - r - z$ and $k + z \leq d_1 \leq n - 1$, $k + z \leq d_2 \leq n$ such that $\frac{k}{d_2-z} = \frac{1}{\rho}$. Let $n' = n + k$, $k' = k + z$, $r' = n' - k'$ and $s = d_1 + 1 - k'$. Let F be a finite field of size $|F| > \rho s(n + k)$, and let γ be a primitive element of F. Consider the code family given by (9.1) and (9.2), where $t = \rho s^n$ and*

$$A_i = \sum_{a=0}^{t-1} \lambda_i e_a e_{a(i,a_i\oplus 1)}^T, \quad i = 1, \cdots, n \tag{9.12}$$

$$A_{n+i} = \sum_{a=0}^{t-1} \lambda_{n+i} e_a e_{a(n+1,a_{n+1}\oplus 1)}^T \quad i = 1, \cdots, k. \tag{9.13}$$

Here $\{e_a : a = 0, \cdots, t-1\}$ *is the standard basis of* F^t *and we represent* a *using the* $(n+1)$*-digit notation* $a = (a_{n+1}, a_n, \cdots, a_1)$*, where* $a_{n+1} \in \{0, \cdots, \rho - 1\}$ *and* $a_i \in \{0, \cdots, s-1\}$*, for* $i \in [n]$*.* $\oplus$ *denotes addition modular* s *in (9.12) and addition modular* ρ *in (9.13), respectively.* $\lambda_i = \gamma^i$ *for* $i \in [n+k]$*.*

Lemma 9.3.1. *The array code* $\mathcal{C}$ *given by Construction 9.3.1 is MDS.*

Proof. By [62, Lemma VII.3], an array code given by (9.1) and (9.2) is MDS if $A_i - A_j$ is invertible and $A_i A_j = A_j A_i$, for all $i \neq j, i, j \in [n+k]$. To establish the commuting part, note that for distinct $i, j \in [n]$

$$A_i A_j = A_j A_i = \sum_{a=0}^{t-1} \lambda_i \lambda_j e_a e^T_{a(i,j,a_i \oplus 1, a_j \oplus 1)}, \tag{9.14}$$

and that for $i \in [n], j \in [n+1, n+k]$

$$A_i A_j = A_j A_i = \sum_{a=0}^{t-1} \lambda_i \lambda_j e_a e^T_{a(i,j,a_i \oplus 1, a_{n+1} \oplus 1)}, \tag{9.15}$$

and that for $i, j \in [n+1, n+k]$

$$A_i A_j = A_j A_i = \sum_{a=0}^{t-1} \lambda_i \lambda_j e_a e^T_{a(n+1, a_{n+1} \oplus 2)}. \tag{9.16}$$

We now turn to the invertible part. For distinct $i, j \in [n+1, n+k]$, $A_i - A_j = \sum_{a=0}^{t-1} (\lambda_i - \lambda_j) e_a e^T_{a(n+1, a_{n+1} \oplus 1)}$ and therefore is invertible. For $i \in [n], j \in [n+1, n+k]$, consider arbitrary x such that $A_i x = A_j x$. Let $x = \sum_{a=0}^{t-1} x_a e_a$, where $x_a \in F$, then

$$A_i x = \sum_{a=0}^{t-1} \lambda_i x_{a(i, a_i \oplus 1)} e_a \tag{9.17}$$

$$A_j x = \sum_{a=0}^{t-1} \lambda_j x_{a(n+1, a_{n+1} \oplus 1)} e_a. \tag{9.18}$$

Therefore $\lambda_i x_{a(i,a_i \oplus 1)} = \lambda_j x_{a(n+1, a_{n+1} \oplus 1)}$ for $a = 0, \cdots, t-1$, implying that

$$x_a = \frac{\lambda_j}{\lambda_i} x_{a(i, n+1, a_i \ominus 1, a_{n+1} \oplus 1)}. \tag{9.19}$$

Repeating (9.19) ρs times, we have

$$x_a = \frac{\lambda_j^{\rho s}}{\lambda_i^{\rho s}} x_{a(i, n+1, a_i \ominus \rho s, a_{n+1} \oplus \rho s)} = \frac{\gamma^{j \rho s}}{\gamma^{i \rho s}} x_a. \tag{9.20}$$

Because $\gamma^{i\rho s} \neq \gamma^{j\rho s}$, it follows from (9.20) that $x_a = 0$, $a = 0, \cdots, t-1$. Therefore $x = 0$ and $A_i - A_j$ is invertible. By a similar argument, for distinct $i, j \in [n]$, it can be shown that $A_i - A_j$ is invertible. This completes the proof. $\square$

Lemma 9.3.2. *The array code $\mathcal{C}$ given by Construction 9.3.1 attains optimal bandwidth and access when 1) repairing a single node i, $i \in [n]$, from any d_1 nodes, and 2) repairing the set of nodes $\{n+1, \cdots, n'\}$ from any d_2 nodes.*

Proof. We first consider the repair of the set of nodes $\{n+1, \cdots, n'\}$. Expanding (9.1) coordinate-wise, then for $a = 0, \cdots, t-1$ and $l = 0, \cdots, r'-1$, we have

$$\sum_{i=1}^{n} \lambda_i^l c_{i,a(i,a_i \oplus l)} + \sum_{i=n+1}^{n+k} \lambda_i^l c_{i,a(n+1,a_{n+1} \oplus l)} = 0. \tag{9.21}$$

Note that $k\rho - 1 = d_2 - z - 1 \leq n - z - 1 = r' - 1$. Therefore, for any $a = 0, \cdots, t-1$ and for $l = a_{n+1}, a_{n+1} + \rho, \cdots, a_{n+1} + (k-1)\rho$, it follows from (9.21) that

$$\sum_{i=n+1}^{n+k} \lambda_i^l c_{i,a} = -\sum_{i=1}^{n} \lambda_i^l c_{i,a(i,n+1,a_i \oplus l, a_{n+1} \ominus l)} \tag{9.22}$$

$$= -\sum_{i=1}^{n} \lambda_i^l c_{i,a(i,n+1,a_i \oplus l, 0)}. \tag{9.23}$$

Fixing a, it is clear that the k equations given by (9.23) forms a system such that the k variables $c_{n+1,a}, \cdots, c_{n+k,a}$ in the L.H.S. can be solved for if the R.H.S. is known. Therefore the set of symbols stored in the erased nodes, i.e., $\{c_{n+i,a} : i \in [k], a = 0, \cdots, t-1\}$, can be recovered from the set of symbols $\{c_{i,a} : i \in [n], a = 0, \cdots, t-1, a_{n+1} = 0\}$. We make the following claim.

Claim: the set of symbols $\{c_{i,a} : i \in [n], a = 0, \cdots, t-1, a_{n+1} = 0\}$ can be recovered from the set of symbols $\{c_{i,a} : i \in \mathcal{D}, a = 0, \cdots, t-1, a_{n+1} = 0\}$, where $\mathcal{D}$ is any subset of $[n]$ of size d_2.

The claim implies that to repair the set of nodes $\{n+1, \cdots, n+k\}$, it suffices to access and download a fraction of $1/\rho$ of the symbols stored in any d_2 nodes, hence achieving the optimal repair bandwidth and access complexity.

To prove the claim, note that by construction the parity check equations are equivalent to

$$\begin{bmatrix} A_1^0 & \cdots & A_n^0 \\ \vdots & \vdots & \vdots \\ A_1^{r'-1} & \cdots & A_n^{r'-1} \end{bmatrix} \begin{bmatrix} C_1 \\ \vdots \\ C_n \end{bmatrix} = - \begin{bmatrix} A_{n+1}^0 & \cdots & A_{n+k}^0 \\ \vdots & \vdots & \vdots \\ A_{n+1}^{r'-1} & \cdots & A_{n+k}^{r'-1} \end{bmatrix} \begin{bmatrix} C_{n+1} \\ \vdots \\ C_{n+k} \end{bmatrix}. \tag{9.24}$$

Let $\mathcal{M}$ be the set of $t \times t$ matrices that commute with $\{A_i : i \in [n+k]\}$. By the proof of Lemma 9.3.1, $A_i \in \mathcal{M}$, for $i \in [n+k]$. Define functions $p_i : \mathcal{M} \to \mathcal{M}$, with

$p_i(x) = x^i \prod_{j=1}^{k}(x^\rho - A_{n+j}^\rho)$, for $i = 0, \cdots, r' - k\rho - 1$. Due to commutativity, we can write $p_i(x) = \sum_{j=0}^{r'-1} p_{i,j}x^i$, where $p_{i,j}$ is a $t \times t$ matrix. Define a $(r' - k\rho)t \times r't$ matrix

$$P = \begin{bmatrix} p_{0,0} & \cdots & p_{0,r'-1} \\ \vdots & \vdots & \vdots \\ p_{r'-k\rho-1,0} & \cdots & p_{r'-k\rho-1,r'-1} \end{bmatrix}. \tag{9.25}$$

By construction $P(A_{n+i}^0, \cdots, A_{n+i}^{r'-1})^T = (p_0(A_{n+i}), \cdots, p_{r'-k\rho-1}(A_{n+i}))^T = 0$, for $i \in [k]$. Therefore multiplying P on the left to (9.24), we have

$$P \begin{bmatrix} A_1^0 & \cdots & A_n^0 \\ \vdots & \vdots & \vdots \\ A_1^{r'-1} & \cdots & A_n^{r'-1} \end{bmatrix} \begin{bmatrix} C_1 \\ \vdots \\ C_n \end{bmatrix} = 0. \tag{9.26}$$

By construction $P(A_i^0, \cdots, A_i^{r'-1})^T = (p_0(A_i), \cdots, p_{r'-k\rho-1}(A_i))^T$, for $i \in [n]$. Therefore it follows from (9.26) that

$$\begin{bmatrix} A_1^0 \prod_{i=1}^{k}(A_1^\rho - A_{n+i}^\rho) & \cdots & A_n^0 \prod_{i=1}^{k}(A_n^\rho - A_{n+i}^\rho) \\ \vdots & \vdots & \vdots \\ A_1^{r'-k\rho-1} \prod_{i=1}^{k}(A_1^\rho - A_{n+i}^\rho) & \cdots & A_n^{r'-k\rho-1} \prod_{i=1}^{k}(A_n^\rho - A_{n+i}^\rho) \end{bmatrix} \begin{bmatrix} C_1 \\ \vdots \\ C_n \end{bmatrix} = 0. \tag{9.27}$$

By an argument similar to that in the proof of Lemma 9.3.1, it can be shown that for $i \in [n]$ and $j \in [n+1, n+k]$, $A_i^\rho - A_j^\rho$ is invertible. Therefore $\prod_{j=1}^{k}(A_i^\rho - A_{n+j}^\rho)$ is invertible for $i \in [n]$. Hence the parity check matrix in (9.27) is a block Vandermonde matrix in which each column is scaled by a full rank matrix. The resulting code is an $[n, n - (r' - k\rho) = d_2]$ MDS array code, i.e., from any d_2 elements of $\{C_1, \cdots, C_n\}$ we can recover the whole set.

Note that by construction $A_{n+i}^\rho = \lambda_{n+i}^\rho I$ for $i \in [k]$. Therefore, the parity check matrix in (9.27) equals:

$$\begin{bmatrix} A_1^0 \prod_{i=1}^{k}(A_1^\rho - I) & \cdots & A_n^0 \prod_{i=1}^{k}(A_n^\rho - I) \\ \vdots & \vdots & \vdots \\ A_1^{r'-k\rho-1} \prod_{i=1}^{k}(A_1^\rho - I) & \cdots & A_n^{r'-k\rho-1} \prod_{i=1}^{k}(A_n^\rho - I) \end{bmatrix}. \tag{9.28}$$

By construction, for $i \in [n]$, A_i is a permutation matrix whose corresponding permutation only affects the digit a_i. Therefore each block of the parity check matrix in (9.28), if expanded, is a sum of products of permutation matrices whose

corresponding permutations do not change a_{n+1}. For $j \in [r' - k\rho]$, consider the j-th blockwise row of (9.28)

$$\left[\begin{array}{ccc} A_1^{j-1} \prod_{i=1}^{k} (A_1^{\rho} - I) & \cdots & A_n^{j-1} \prod_{i=1}^{k} (A_n^{\rho} - I) \end{array} \right], \tag{9.29}$$

then for $a = 1, \cdots, t-1$, the a-th row of (9.29) corresponds to a parity-check equation of the following general form:

$$\sum_{i=1}^{n} \sum_{a'=0}^{t-1} \kappa_{i,a'} c_{i,a'} = 0, \tag{9.30}$$

where by the previous discussion, $\kappa_{i,a'} = 0$ if $a'_{n+1} \neq a_{n+1}$.

Let $C'_i = \{c_{i,a} : a = 0, \cdots, t-1, a_{n+1} = 0\}, i \in [n]$, then the fact that $\{C_1, \cdots, C_n\}$ forms an $[n, d_2]$ MDS array code and the fact that any parity check equation in (9.27) that involves an element of $\cup_{i \in [n]} C'_i$ only involves elements in $\cup_{i \in [n]} C'_i$ implies that from any d_2 elements of $\{C'_1, \cdots, C'_n\}$ the whole set can be recovered. This proves the claim and the d_2-optimal access property of the repair of the set of nodes $\{n+1, \cdots, n+k\}$.

Similarly, consider the case of repairing an individual node m, $m \in [n]$. Since $s \leq n - k' < r'$, by (9.21), for $a = 0, \cdots, t-1$, let $l = a_m$, we have

$$-\lambda_m^l c_{m,a} = \sum_{i \in [n], i \neq m} \lambda_i^l c_{i,a(m,i,a_m \ominus l, a_i \oplus l)} + \sum_{i=n+1}^{n+k} \lambda_i^l c_{i,a(m,n+1,a_m \ominus l, a_{n+1} \oplus l)} \tag{9.31}$$

$$= \sum_{i \in [n], i \neq m} \lambda_i^l c_{i,a(m,i,0,a_i \oplus a_m)} + \sum_{i=n+1}^{n+k} \lambda_i^l c_{i,a(m,n+1,0,a_{n+1} \oplus a_m)}. \tag{9.32}$$

Therefore C_m can be recovered from the set of symbols $\{c_{i,a} : i \in [n+k], i \neq m, a \in [0, t-1], a_m = 0\}$. By setting $p_i(x) = x^i(x^s - A_m^s)$, for $i = 0, \cdots, r' - s - 1$, and following the same line of arguments as before, it can be proved that $\{c_{i,a} : i \in [n+k], i \neq m, a \in [0, t-1], a_m = 0\}$ in turn can be recovered from the set of symbols $\{c_{i,a} : i \in \mathcal{D}, a \in [0, t-1], a_m = 0\}$, where $\mathcal{D}$ is any subset of $[n+k] \backslash \{m\}$ of size d_1. This implies that to repair node m it suffices to access and download a fraction of $1/s$ of the symbols stored in any d_1 nodes, achieving the optimal repair bandwidth and access complexity. The proof is complete. □

By Lemmas 9.3.1, 9.3.2 and Theorem 9.1.2, we have:

Theorem 9.3.1. *The secret sharing scheme obtained by applying Theorem 9.1.1 to the code given in Construction 9.3.1, where the last k nodes are discarded, is a*

$(n, k = n - r - z, r, z)$ scheme with d_1-optimal bandwidth and access in repair and d_2-optimal bandwidth and access in decoding.

Next we generalize Construction 9.3.1 to the case of arbitrary $\frac{k}{d_2 - z}$.

Construction 9.3.2. *For any $n, r, z, k = n - r - z$, $k + z \leq d_1 \leq n - 1$ and $k + z \leq d_2 \leq n$, let $n' = n + k$, $k' = k + z$, $r' = n' - k'$ and $s = d_1 + 1 - k'$. Let $\theta = \gcd(k, d_2 - z)$, $\tau = \frac{k}{\theta}$, $\rho = \frac{d_2 - z}{\theta}$ and $\delta = \rho - \tau$. Let F be a finite field of size $|F| > \rho(n + k)(\max(s, \rho - 1))$, and let γ be a primitive element of F. Let $t = s^n \prod_{i=1}^{\tau}(i + \delta)$. Consider the code family given by (9.1) and (9.2), where*

$$A_i = \sum_{a=0}^{t-1} \lambda_i e_a e_{a(i,a_i \oplus 1)}^T, \quad i = 1, \cdots, n$$

$$A_{n+i} = \sum_{a=0}^{t-1} \lambda_{n+i} e_a e_{a(n+\lceil \frac{i}{\theta} \rceil, a_{n+\lceil \frac{i}{\theta} \rceil} \oplus 1)}^T \quad i = 1, \cdots, k.$$

Here $\{e_a : a = 0, \cdots, t-1\}$ is the standard basis of F^t and we represent a using the $(n + \tau)$-digit notation $a = (a_{n+\tau}, \cdots, a_1)$, where $a_i \in \{0, \cdots, s - 1\}$, for $i \in [n]$ and $a_{n+i} \in \{0, \cdots, i + \delta - 1\}$, for $i \in [\tau]$. $\lambda_i = \gamma^i$ for $i \in [n + k]$.

Lemma 9.3.3. *The array code $\mathcal{C}$ given by Construction 9.3.2 is MDS.*

Proof. Similar to the proof of Lemma 9.3.1, it can be shown that for $i, j \in [n + k]$, $A_i A_j = A_j A_i$. Therefore to prove the lemma it suffices to show that for distinct $i, j \in [n + k]$, $A_i - A_j$ is invertible. Define a function

$$f(i) = \begin{cases} i & i = 1, \cdots, n \\ n + \lceil \frac{i-n}{\theta} \rceil & i = n + 1, \cdots, n + k. \end{cases} \tag{9.33}$$

If $f(i) = f(j)$, then $A_i - A_j = \sum_{a=0}^{t-1} (\lambda_i - \lambda_j) e_a e_{a(f(i), a_{f(i)} \oplus 1)}^T$ which is invertible. Otherwise, consider arbitrary x such that $A_i x = A_j x$. Let $x = \sum_{a=0}^{t-1} x_a e_a$, where $x_a \in F$, then

$$A_i x = \sum_{a=0}^{t-1} \lambda_i x_{a(f(i), a_{f(i)} \oplus 1)} e_a \tag{9.34}$$

$$A_j x = \sum_{a=0}^{t-1} \lambda_j x_{a(f(j), a_{f(j)} \oplus 1)} e_a. \tag{9.35}$$

Therefore for $a = 0, \cdots, t - 1$, $\lambda_i x_{a(f(i), a_{f(i)} \oplus 1)} = \lambda_j x_{a(f(j), a_{f(j)} \oplus 1)}$, implying that,

$$x_a = \frac{\lambda_j}{\lambda_i} x_{a(f(i), f(j), a_{f(i)} \ominus 1, a_{f(j)} \oplus 1)}. \tag{9.36}$$

Define a function

$$g(i) = \begin{cases} s & i = 1, \cdots, n \\ i - n + \delta & i = n+1, \cdots, n+\tau. \end{cases} \tag{9.37}$$

Repeating (9.36) for $G = g(f(i))g(f(j))$ times, we have

$$x_a = \frac{\lambda_j^G}{\lambda_i^G} x_{a(f(i),f(j,a_{f(i)} \ominus G, a_{f(j)} \oplus G)} = \frac{\gamma^{jG}}{\gamma^{iG}} x_a. \tag{9.38}$$

Because $iG < |F|$ and $jG < |F|$, $\gamma^{iG} \neq \gamma^{jG}$, and it follows from (9.38) that $x_a = 0$, $a = 0, \cdots, t-1$. Therefore $x_a = 0$ and $A_i - A_j$ is invertible, proving the lemma. □

Lemma 9.3.4. *The array code $\mathcal{C}$ given by Construction 9.3.2 attains optimal bandwidth and access when 1) repairing a single node i, $i \in [n]$, from any d_1 nodes, and 2) repairing the set of nodes $\{n+1, \cdots, n'\}$ from any d_2 nodes.*

Proof. We omit the proof of the first statement as it is similar to the proof of Lemma 9.3.2. To prove the second statement, the idea is to divide the $(n+1)$-th to the $(n+k)$-th nodes into τ groups and repair the groups one by one iteratively. Formally, let $\mathcal{C}_R \subset \{C_1, \cdots, C_n\}$ be the set of nodes accessed, with $|\mathcal{C}_R| = d_2$. For $i = 1, \cdots, \tau$, let $\mathcal{C}_i = \{C_{n+(i-1)\theta+1}, \cdots, C_{n+(i-1)\theta+\theta}\}$, then we first use $\mathcal{C}_R$ to repair $\mathcal{C}_1$, then use $\mathcal{C}_R \cup \mathcal{C}_1$ to repair $\mathcal{C}_2$, $\cdots$, and finally use $\mathcal{C}_R \cup \mathcal{C}_1 \cup \cdots \cup \mathcal{C}_{\tau-1}$ to repair $\mathcal{C}_\tau$. By the proof of Lemma 9.3.2, we can recover $\mathcal{C}_i$ from $\{c_{v,a} : a_{n+i} = 0, C_v \in \mathcal{C}_R \cup \mathcal{C}_{[i-1]}\}$. Since the nodes in $\mathcal{C}_{[i-1]}$ are already recovered, it suffices to know the set of values $\Omega_i = \{c_{v,a} : a_{n+i} = 0, C_v \in \mathcal{C}_R\}$. Therefore to repair all τ groups of nodes, it suffices to access and communicate the values in the set $\Lambda = \bigcup_{i=1}^{\tau} \Omega_i$.

We analyze the size of Λ. Define $\Lambda_i = \bigcup_{j=1}^{i} \Omega_j$ and note that $\Lambda_\tau = \Lambda$. We prove that $|\Lambda_i| = \frac{i}{\delta+i} d_2 t$ by induction on i. Clearly $|\Lambda_1| = \frac{1}{\delta+1} d_2 t$. Now suppose that $|\Lambda_i| = \frac{i}{\delta+i} d_2 t$ is true, then

$$|\Lambda_{i+1}| = |\Lambda_i| + \frac{1}{\delta+i+1} \left(1 - \frac{|\Lambda_i|}{d_2 t}\right) d_2 t \tag{9.39}$$

$$= \frac{i}{\delta+i} d_2 t + \frac{1}{\delta+i+1} \frac{\delta}{\delta+i} d_2 t \tag{9.40}$$

$$= \frac{(\delta+i)(i+1)}{(\delta+i)(\delta+i+1)} d_2 t = \frac{i+1}{\delta+i+1} d_2 t. \tag{9.41}$$

Therefore by induction $|\Lambda_i| = \frac{i}{\delta+i} d_2 t$ for $i = 1, \cdots, \tau$. So $|\Lambda| = |\Lambda_\tau| = \frac{\tau}{\delta+\tau} d_2 t = \frac{k}{d_2 - z} d_2 t$. Therefore a fraction of $\frac{k}{d_2 - z}$ of the symbols stored in the d_2 nodes are accessed and downloaded which attains the lower bound. The proof is complete. □

By Lemmas 9.3.3, 9.3.4 and Theorem 9.1.2, we have:

Theorem 9.3.2. *The secret sharing scheme obtained by applying Theorem 9.1.1 to the code given in Construction 9.3.2, where the last k nodes are discarded, is an $(n, k = n - r - z, r, z)$ scheme with d_1-optimal bandwidth and access in repair and d_2-optimal bandwidth and access in decoding.*

Chapter 10

CONCLUDING REMARKS

In this part we study the communication complexity and the access complexity of secret sharing schemes during decoding and repair. We prove a tight lower bound on the decoding bandwidth, and design schemes that achieve the optimal decoding bandwidth and access when d nodes participate in decoding, universally for all $n - r \leq d \leq n$. We also design schemes that achieve the optimal bandwidth and access during both decoding and repair. Finally we design a family of Shamir's scheme with asymptotically optimal decoding bandwidth.

An interesting future direction is to combine the objectives of Part I and II, and to design schemes that are optimal in terms of both computation and communication. Particularly, since Construction 5.3.1 in Part I is a natural generalization of Shamir's scheme and the scheme in Section 8.2 is closely related to Shamir's scheme, it seems plausible that the ideas from both sides can be combined, resulting in schemes that are both bandwidth optimal and highly efficient in computation.

More generally, an interesting question is whether the ideas and schemes developed in this part can be applied to a broader scope. For example, is it possible to design schemes with improved decoding bandwidth for non-threshold access structures? Is it possible to extend the ideas to improve the communication efficiency of other secure protocols, particularly those who use secret sharing schemes as building blocks?

Part III

Secure Repair of Secret Sharing Schemes

Chapter 11

INTRODUCTION TO SECURE REPAIR

The problem of repairing secret sharing schemes has attracted significant interests recently. Specifically, a secret sharing scheme encodes a message into n shares, such that the message can be decoded from any $n - r$ shares (reliability), and that any z shares are independent of the message (security). In the setting of distributed storage, a system consists of n nodes and one share is assigned to each node. Therefore a secret sharing scheme can tolerate r node failures (erasures) as well as z colluding adversarial nodes trying to infer information about the message. In the event of node failures, the shares held by the failed nodes are lost and in order to maintain the same level of reliability, the system needs to repair the failures by reconstructing the lost shares and reassigning them to the failed (or replacement) nodes. Two problems arise during the repair process, namely, 1) bandwidth efficiency: it is desirable to minimize the amount of communication induced by the repair process; 2) repair security: the system needs to maintain the security requirement that any colluding z nodes, including the failed (or replacement) nodes, cannot infer any information about the message, during and after the repair process.

Secure regenerating codes, e.g., [21]–[24], are a class of secret sharing schemes that are carefully designed to address the above problems. We classify secure regenerating codes into two categories: codes that only address the bandwidth efficiency problem (i.e., codes with non-secure repair), and codes that address both the bandwidth efficiency and the repair security problems (i.e., codes with secure repair). Specifically, codes with non-secure repair focus on reducing the repair bandwidth without worrying about the security of the repair process. For example, the codes that tolerate Type-I adversary in [24] and the schemes constructed in Chapter 9 belong to this category. For this case one can think of having a trustworthy repair dealer that will receive information from the available helper nodes, reconstruct the lost share and then forward it to the failed node. The repair dealer may receive enough information to gain knowledge of the message, and therefore has to be trustworthy. In comparison, regenerating codes with secure repair guarantee by code design that such a dealer will not learn any information about the message. This in fact removes the need for the dealer to be trustworthy and the failed node can act as the dealer. Unfortunately, the guarantee that the dealer

cannot learn any information about the message is shown to come at a high cost in rate [21], [24], because more independent randomness (keys) is required in order to protect the message from the dealer, resulting in increased overhead. Therefore, codes with non-secure repair in general have a significantly better rate and repair bandwidth (when normalized by rate) than codes with secure repair.

We address the problem of repair security from a different perspective, without needing to take the heavy penalty in rate and other aspects of efficiency as in the case of secure regenerating codes. The key idea is that we allow a more flexible repair protocol: secure regenerating codes implicitly assume a simple "one-round" repair protocol, in which the helper nodes transmit information to the failed nodes but they themselves do not receive information from other nodes. This implicit "one-round" assumption is expensive in terms of efficiency. We show that, just by slightly relaxing this assumption and allowing a "two-round" protocol, it becomes possible to securely repair *any* secret sharing scheme in a black-box manner, in the sense that the proposed repair protocol is generic and there is no need to design or modify the secret sharing scheme. Refer to Figure 11.1 for a simple example of the two-round secure repair protocol.

We remark that a two-round protocol is advantageous in that more nodes are allowed to receive information rather than only the failed node. This is intuitively beneficial because, if $d > z$ nodes can receive information, then we can take advantage of the gap between d and z in the following way. During the repair process, let the information received by any z nodes be independent randomness (so that the security requirement is met), and let the information received by all d nodes reveal useful information on the lost share. We then use an extra round of communication to transmit the information on and only on the lost share from the d nodes to the failed node so that the lost share can be reconstructed. Loosely speaking, we can think of the repair process as letting the failed node "compute" its share securely, so that it only learns the share but nothing else. This is naturally related to the problem of secure multi-party computation and the ideas in [69], [70] play an important role in our repair schemes. We remark that we adopt a formal information-theoretic approach in our analysis and bounds, which differs from many existing works on secure multi-party computation. We also note that relaxing the repair process to involve more than one round is practical. For example, POTSHARDS [14] employs a heuristic multi-round repair scheme to improve the security of the repair process.

Our generic secure repair schemes have two important advantages over secure re-

generating codes with secure repair. First, the generic nature implies that there is no need to compromise the efficiency of the secret sharing scheme for secure repair. Here, aspects of efficiency at stake are not limited to the rate and repair bandwidth discussed earlier, but also include, for example, computational complexity (discussed in Part I) and decoding bandwidth (discussed in Part II), because it is not clear how to construct secure regenerating codes with secure repair that also achieve the optimal computation and/or decoding bandwidth. Second, most secure regenerating codes focus on secure repair by a fixed number of helper nodes. In the case that not enough helper nodes are available due to multiple node failures, it is not clear how secure repair can be achieved.

We briefly summarize the contributions of the paper. In Section 12.1, we present a generic two-round secure repair scheme based on the ideas in [69], [70]. Specifically, in the first round each helper node encodes its share into $z+1$ pieces using a secret sharing scheme, so that any z pieces reveal no information about the share and that the share can be decoded from $z+1$ pieces. The $z+1$ pieces are sent to $z+1$ receiver nodes, and each receiver node receives a piece from each helper node (if the helper node and the receiver node are the same node, then the corresponding piece needs not be transmitted). For example, in Figure 11.1-(b), the helper nodes and receiver nodes are both Nodes 2 and 3. The set of pieces received by all receiver nodes contains enough information to decode the shares of all helper nodes and the lost share. We then need to communicate the information about the lost share, but no extra information about the shares of the helper nodes, to the failed node. To achieve this, each receiver node locally computes a function that takes the pieces received by the node as inputs, and outputs a "distilled" piece such that the set of "distilled" pieces only contains information about the lost share. This set is then transmitted from the receiver nodes to the failed node. Refer to Figure 11.1-(c) for an example.

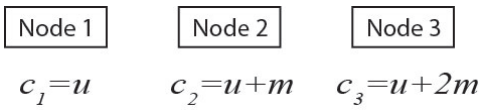
The generic repair scheme in Section 12.1 requires a relatively large repair bandwidth. In Section 12.2, we reduce the repair bandwidth of the scheme significantly by adopting the idea of parallelism in [17]. Instead of repairing one single share at a time, we repair multiple shares together in parallel, therefore amortizing the communication overhead over the multiple shares. This is achieved by letting all n nodes be receiver nodes (instead of $z+1$ nodes) and by using a secret sharing scheme of a higher rate in the first round. The larger gap between the number of receiver nodes and z implies that we can encode more information in the secret

sharing scheme (so that it has a higher rate) and can repair more shares in parallel.

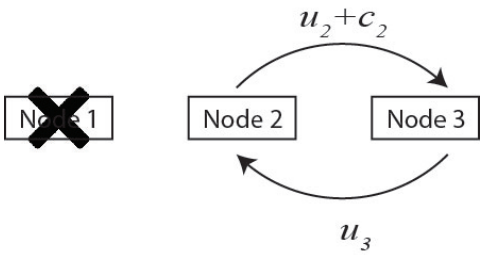
The generic repair schemes in Sections 12.1 and 12.2 can securely repair any *scalar* linear secret sharing schemes. A more general class of schemes are *vector* linear secret sharing schemes. For a vector linear scheme over a field, each node stores multiple elements of the field instead of a single element as in the scalar linear case. Many efficient secret sharing schemes, e.g., schemes with efficient computation (e.g., Chapter 4 and Chapter 5), schemes with efficient decoding bandwidth (e.g., Chapter 8), and schemes with efficient repair bandwidth (e.g., Chapter 9), are intrinsically vector linear. In Section 12.3 we generalize our secure repair schemes to generically repair any vector linear schemes. In particular, this generalization allows us to leverage the property of secret sharing schemes with efficient (non-secure) repair bandwidth, i.e., secure regenerating codes with non-secure repair, to further reduce the (secure) repair bandwidth.

Finally, in Section 12.4 we prove an information-theoretic lower bound on the repair bandwidth of secure repair schemes. The bound implies that the secure repair schemes in Sections 12.2 and 12.3 achieve the optimal repair bandwidth within a small constant factor when n dominates z, or when the secret sharing scheme being repaired has optimal rate.

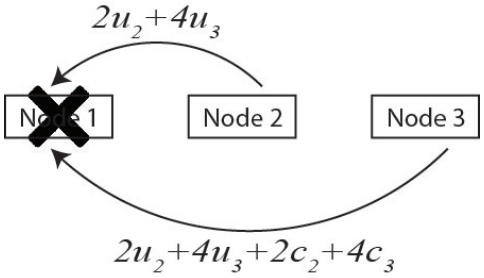
The material in this part of the thesis was presented in part in [71].



(a) A secret sharing scheme over $\mathbb{F}_5$ with $r = 1$ and $z = 1$, where m is a message symbol and u is a random key uniformly distributed over $\mathbb{F}_5$. We denote the three shares by c_1, c_2 and c_3.



(b) Repairing Node 1 (round 1): Node 2 generates a random symbol u_2 and sends $u_2 + c_2$ to Node 3. Node 3 generates a random symbol u_3 and sends it to Node 2.



(c) Repairing Node 1 (round 2): Node 2, having access to u_2 and u_3, computes and sends $2u_2 + 4u_3$ to Node 1. Node 3, having access to $u_2 + c_2$, u_3 and c_3, computes and sends $2u_2 + 4u_3 + 2c_2 + 4c_3$ to Node 1. Node 1 can reconstruct its share since $c_1 = 2c_2 + 4c_3$.

Figure 11.1: Securely repairing a secret sharing scheme. Note that it is impossible to securely repair any node failure under the one-round repair model of regenerating codes, because for the failed node to reconstruct its share it has to collect the shares from the other two nodes, which will violate the security requirement. However, any node failure can be securely repaired by the two-round scheme shown above. To see that the scheme is secure, note that after the repair process Node 1 has access to c_1 and $2u_2 + 4u_3$; Node 2 has access to c_2, u_2 and u_3; Node 3 has access to c_3, u_3 and $u_2 + c_2$. Therefore, any single node has access to only one share as well as some random symbols that are independent of the shares. Therefore no single node can learn any information about the message m.

Chapter 12

SECURE REPAIR SCHEMES

12.1 Generic Secure Repair of Secret Sharing Schemes

An (n, k, r, z) secret sharing scheme over $\mathbb{F}_q$ is a randomized function that maps (encodes) a message $\boldsymbol{m} = (m_1, \cdots, m_k)$ of k symbols over $\mathbb{F}_q$ to n shares $\boldsymbol{c} = (c_1, \cdots, c_n)$ over $\mathbb{F}_q$, such that 1) $\boldsymbol{m}$ can be decoded from any subset of $n-r$ shares; 2) any subset of z shares do not reveal information about $\boldsymbol{m}$. Shamir's scheme is a well known secret sharing scheme with $k = 1$.

Construction 12.1.1. (Shamir's scheme [15]) *For any n, and $z < n$, let $k = 1$, $r = n - z - 1$ and $\mathbb{F}_q$ be a finite field of size $q > n$. Let u_i, $i \in [z]$ be i.i.d. uniformly distributed over $\mathbb{F}_q$ (also referred to as keys) and let α_i, $i \in [n]$ be arbitrary distinct non-zero elements of $\mathbb{F}_q$. The shares corresponding to message m_1 are*

$$(c_1, c_2, \cdots, c_n) = (m_1, u_1, u_2, \cdots, u_z) \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \alpha_1 & \alpha_2 & \cdots & \alpha_n \\ \vdots & \vdots & \vdots & \vdots \\ \alpha_1^z & \alpha_2^z & \cdots & \alpha_n^z \end{bmatrix}. \tag{12.1}$$

Lemma 12.1.1. *Let c_i, $i \in [n]$ be the shares of Shamir's scheme (12.1) on message m_1 and keys u_i, $i \in [z]$, and let c'_i, $i \in [n]$ be the shares of the scheme on message m'_1 and keys u'_i, $i \in [z]$. Then for arbitrary linear function $f : \mathbb{F}_q^2 \to \mathbb{F}_q$, $f(c_i, c'_i)$, $i \in [n]$ are the shares of the scheme on message $f(m_1, m'_1)$ and keys $f(u_i, u'_i)$, $i \in [z]$.*

Proof. Follows from the linearity of (12.1). □

A secret sharing scheme allows secure and reliable storage of information, i.e., it can tolerate the loss of any r shares as well as the exposure of any z shares to an adversary. However, the problem of *repair* is not addressed. Consider the situation that one or more shares are lost or unavailable, and so in order to maintain the same level of reliability one wishes to repair/reconstruct the lost shares. For example, in the application of storage, the n shares are assigned to n storage nodes, and in the situation of node failures, one wishes to repair the failures by reconstructing the shares originally assigned to the failed nodes. The repair problem can be easily

solved if there is a trusted dealer, who can collect the available shares, recompute the lost shares and reassign them to the failed or replacement nodes. However, the assumption of a trusted dealer responsible for centralized repair may not be practical for many applications.

In this chapter we study the situation that a trusted repair dealer is not available and the nodes holding the shares are responsible for carrying out the repair by themselves. A naive repair scheme is to transmit the available shares to the failed node so that it can decode the message and recompute its share. By doing so, however, we have revealed the message to the failed node which conflicts with the underlying threat model of a secret sharing scheme, because now it is possible for one single node to gain knowledge of the message. Indeed, in this case an adversary controlling a single node can compromise security simply by reporting a node failure. Therefore, the main question of interest is *how to repair securely without a trusted dealer*. This is related to the problem of secure multi-party computation, and in particular, the ideas developed in [17], [69], [70] for constructing secure computation protocols would be helpful for us to design mechanisms to securely repair secret sharing schemes. Below we formalize the notion of secure repair.

Definition 12.1.1. (Secure repair scheme) *Consider an (n, k, r, z) secret sharing scheme and n nodes such that node i holds the share c_i. For any $e \in [n]$, and $I \subset [n]$, suppose that node e has failed and nodes in I are available to help repairing node e. A secure repair scheme is a protocol of communication between the nodes, such that 1) the information sent by a node to other nodes is a function of the share it holds, its local coin flips, and the information it received from other nodes; 2) denote by d_i all the information received by node i by the end of the protocol and denote by u_i the result of coin flips at node i, then*

- *(Repairability) $H(c_e|d_e) = 0$.*
- *(Security) $I(\boldsymbol{m}; c_A, u_A, d_A) = 0$, for all $A \subset [n]$, $|A| = z$.*

Note that Definition 12.1.1 naturally extends the threat model of secret sharing, e.g., it maintains the security requirement that any z nodes cannot learn any information about the message, during and after the repair process.

Construction 12.1.2. (Secure repair of linear secret sharing schemes) *Consider any (n, k, r, z) secret sharing scheme, any $e \in [n]$, and any $I = \{i_1, \cdots, i_{|I|}\} \subset [n]$,*

$e \notin I$ such that there exists a linear function f so that $f(c_{i_1}, c_{i_2}, \cdots, c_{i_{|I|}}) = c_e$. Let $J = \{j_1, \cdots, j_{z+1}\}$ be an arbitrary subset of $[n]$ of size $z+1$. The secure repair process involves three steps:

1) *For each node $i \in I$, encode c_i into $c_{i,1}, c_{i,2}, \cdots, c_{i,z+1}$ by a $(z+1, 1, 0, z)$ Shamir's scheme (in Construction 12.1.1 all nodes choose the same α_i's) and send $c_{i,k}$ to node $j_k \in J$.*

2) *For each node $j \in J$, compute $c'_j = f(c_{i_1,j}, c_{i_2,j}, \cdots, c_{i_{|I|},j})$, and send c'_j to node e.*

3) *Node e obtains c_e by decoding the $(z+1, 1, 0, z)$ Shamir's scheme, regarding $c'_{j_1}, c'_{j_2}, \cdots, c'_{j_{z+1}}$ as the $z+1$ shares.*

Theorem 12.1.1. *Construction 12.1.2 is a secure repair scheme.*

Proof. We need to show that Construction 12.1.2 meets the repairability and security requirements in Definition 12.1.1. By Lemma 12.1.1, $c'_{j_1}, c'_{j_2}, \cdots, c'_{j_{z+1}}$ are the shares of a $(z+1, 1, 0, z)$ Shamir's scheme that encodes $f(c_{i_1}, c_{i_2}, \cdots, c_{i_{|I|}}) = c_e$ as message. This proves repairability. We now focus on security. Let A be an arbitrary set of nodes controlled by the adversary, with $|A| = z$. We consider two cases.

Case 1: $e \notin A$. In this case $d_j = (c_{i_1,j}, c_{i_2,j}, \cdots, c_{i_{|I|},j})$ if $j \in J$, and $d_j = 0$ if $j \notin J$. Denote $c_{A,B} = \{c_{i,j} : i \in A, j \in B\}$, we have

$$
\begin{aligned}
I(\boldsymbol{m}; c_A, u_A, d_A) &= I(\boldsymbol{m}; c_A, u_A, c_{I,J\cap A}) && (12.2)\\
&\overset{(a)}{=} I(\boldsymbol{m}; c_A, u_A, c_{I\setminus A,J\cap A})\\
&\overset{(b)}{=} I(\boldsymbol{m}; c_A, u_A | c_{I\setminus A,J\cap A}) + I(\boldsymbol{m}; c_{I\setminus A,J\cap A})\\
&\overset{(c)}{\leq} I(\boldsymbol{m}; c_A, u_A | c_{I\setminus A,J\cap A}) + I(\boldsymbol{c}; c_{I\setminus A,J\cap A})\\
&\overset{(d)}{=} I(\boldsymbol{m}; c_A, u_A | c_{I\setminus A,J\cap A})\\
&\overset{(e)}{=} I(\boldsymbol{m}; c_A, u_A)\\
&\overset{(f)}{=} I(\boldsymbol{m}; c_A)\\
&\overset{(g)}{=} 0. && (12.3)
\end{aligned}
$$

Here (a) is due to the fact that $c_{I\cap A,J}$ is a function of c_A and $\boldsymbol{u}_A$; (b) follows from the chain rule; (c) follows from the data processing inequality and the Markov chain $\boldsymbol{m} \to \boldsymbol{c} \to c_{I\setminus A,J\cap A}$, i.e., $c_{I\setminus A,J\cap A}$ can be dependent on $\boldsymbol{m}$ only via $\boldsymbol{c}$; (d) follows

from the fact that $c_{I\backslash A,J\cap A}$ are the shares of $|I\backslash A|$ independent $(z+1,1,0,z)$ secret sharing schemes and that for each scheme at most $|J\cap A|\leq z$ of its shares are included; (e) follows from the fact that $(\boldsymbol{m},c_A,u_A)\perp c_{I\backslash A,J\cap A}$, implied by (d); (f) follows from $(\boldsymbol{m},c_A)\perp u_A$; and (g) follows from the security of the secret sharing scheme being repaired.

Case 2: $e\in A$. Since $|A|=z$ and $|J|=z+1$, $J\backslash A$ is not empty. Assume with out loss of generality that $j_1\in J\backslash A$. Because $c'_{j_1},c'_{j_2},\cdots,c'_{j_{z+1}}$ are the shares of a $(z+1,1,0,z)$ Shamir's scheme that encodes c_e, it follows that $I(c_e;c'_{j_2},c'_{j_3},\cdots,c'_{j_{z+1}})=0$ and that there exists a linear function g such that $g(c'_{j_1},c'_{j_2},\cdots,c'_{j_{z+1}})=\sum_{k=1}^{z+1}g_kc'_{j_k}=c_e$. This implies that $g_1\neq 0$ and so $c'_{j_1}=(c_e-\sum_{k=2}^{z+1}g_kc'_{j_k})g_1^{-1}$, namely,

$$H(c'_{j_1}|c_e,c'_{J\backslash\{j_1\}})=0. \tag{12.4}$$

We have,

$$\begin{aligned}
I(\boldsymbol{m};c_A,u_A,d_A)&=I(\boldsymbol{m};c_A,u_A,c'_J,c_{I,A\cap J})\\
&\overset{(h)}{\leq} I(\boldsymbol{m};c_A,u_A,c'_J,c_{I,J\backslash\{j_1\}})\\
&\overset{(i)}{=} I(\boldsymbol{m};c_A,u_A,c'_{J\backslash\{j_1\}},c_{I,J\backslash\{j_1\}})\\
&\overset{(j)}{=} I(\boldsymbol{m};c_A,u_A,c_{I,J\backslash\{j_1\}}).
\end{aligned} \tag{12.5}$$

Here (h) follows from $A\cap J\subset J\backslash\{j_1\}$; (i) follows from (12.4); and (j) follows from the fact that $c'_{J\backslash\{j_1\}}$ is a function of $c_{I,J\backslash\{j_1\}}$. We continue the chain of inequality by treating (12.5) in a similar way as Case 1. Namely, applying an argument similar to that of (12.2) - (12.3), we have

$$\begin{aligned}
I(\boldsymbol{m};c_A,u_A,d_A)&\leq I(\boldsymbol{m};c_A,u_A,c_{I,J\backslash\{j_1\}})\\
&=I(\boldsymbol{m};c_A,u_A,c_{I\backslash A,J\backslash\{j_1\}})\\
&=I(\boldsymbol{m};c_A,u_A|c_{I\backslash A,J\backslash\{j_1\}})+I(\boldsymbol{m};c_{I\backslash A,J\backslash\{j_1\}})\\
&\leq I(\boldsymbol{m};c_A,u_A|c_{I\backslash A,J\backslash\{j_1\}})+I(\boldsymbol{c};c_{I\backslash A,J\backslash\{j_1\}})\\
&=I(\boldsymbol{m};c_A,u_A|c_{I\backslash A,J\backslash\{j_1\}})\\
&=I(\boldsymbol{m};c_A,u_A)\\
&=I(\boldsymbol{m};c_A)\\
&=0.
\end{aligned}$$

The proof is complete. □

We remark that Construction 12.1.2 can securely repair any (scalar) linear secret sharing scheme in a black-box manner, in the sense that it does not require modifying the secret sharing scheme. This suggests that secure repair "comes for free" without needing to compromise other aspects of efficiency of the scheme. In comparison, the secure regenerating codes in [21]–[24] allow secure repair at the cost of efficiency. We also remark that multiple failures can be repaired securely by invoking Construction 12.1.2 multiple times.

We analyze the secure repair bandwidth, i.e., the total amount of information that is communicated during the secure repair process. In Step 1, at most $|I|(z+1)$ symbols are transmitted and in Step 2, at most $z+1$ symbols are transmitted. Therefore the total repair bandwidth is at most $(|I|+1)(z+1)$ symbols, which is approximately $z+1$ times of the non-secure repair bandwidth $|I|$.

12.2 Reducing Secure Repair Bandwidth

While Construction 12.1.2 provides a generic approach to repair any linear secret sharing schemes securely, it requires a large overhead in the repair bandwidth. In this section we propose an improved secure repair scheme with a significantly better repair bandwidth. The main idea is that, instead of repairing one single share/symbol at a time, we repair multiple shares together in parallel, and therefore amortizing the communication overhead over the multiple shares. For this to work we need every node to store multiple shares, which is typically the case because the amount of the total information to be stored (e.g., a file) usually exceeds the amount of information that can be stored by a single secret sharing scheme. Therefore the file will be split and stored by multiple independent instances of a secret sharing scheme, resulting in multiple shares to be assigned to a node. In the remainder of this section we assume that there are enough shares in the failed node to be repaired. Then, the main improvement is that in the first round of the repair scheme, rather than using a low rate $(z+1,1,0,z)$ secret sharing scheme, we use a high rate $(n,n-z,0,z)$ scheme. This allows one to repair $n-z$ shares in parallel and reduce the amortized overhead in the repair bandwidth (which are the z keys in the secret sharing schemes of the first round) by $n-z$ times.

Formally, we assume that each node stores $n-z$ shares from $n-z$ independent instances of a secret sharing scheme. We use superscripts to index instances, e.g., $\boldsymbol{m}^{(i)} = (m_1^{(i)}, \cdots, m_k^{(i)})$ is the message encoded by the i-th instance. In the first round of repair we shall use the high rate secret sharing scheme defined in

Construction 5.3.2, which is a generalization of Shamir's scheme to the case of $k > 1$. We present the same construction in matrix form below for concreteness.

Construction 12.2.1. (Ramp version of Shamir's scheme) *For any n, r, z such that $n > r + z$, let $k = n - r - z$ and $\mathbb{F}_q$ be a finite field of size $q > n$. Let u_i, $i \in [z]$ be i.i.d. uniformly distributed over $\mathbb{F}_q$ and let α_i, $i \in [n]$ be arbitrary distinct non-zero elements of $\mathbb{F}_q$. The shares corresponding to message $\boldsymbol{m} = (m_1, m_2, \cdots, m_k)$ are*

$$(c_1, c_2, \cdots, c_n) = (m_1, \cdots, m_k, u_1, \cdots, u_z) \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \alpha_1 & \alpha_2 & \cdots & \alpha_n \\ \vdots & \vdots & \vdots & \vdots \\ \alpha_1^{z+k-1} & \alpha_2^{z+k-1} & \cdots & \alpha_n^{z+k-1} \end{bmatrix}. \tag{12.6}$$

Construction 12.2.1 is an $(n, k = n - r - z, r, z)$ secret sharing scheme.

Construction 12.2.2. (Bandwidth-efficient secure repair) *Consider any (n, k, r, z) secret sharing scheme, any $e \in [n]$, and any $I = \{i_1, \cdots, i_{|I|}\} \subset [n]$, $e \notin I$ such that there exists a linear function f so that $f(c_{i_1}, c_{i_2}, \cdots, c_{i_{|I|}}) = c_e$. The secure repair process involves three steps:*

1) *For each node $i \in I$, encode $c_i^{(1)}, c_i^{(2)}, \cdots, c_i^{(n-z)}$ into $c_{i,1}, c_{i,2}, \cdots, c_{i,n}$ by a $(n, n - z, 0, z)$ scheme according to Construction 12.2.1 (all nodes should choose the same α_i's) and send $c_{i,j}$ to node j.*

2) *For each node $j \in [n]$, compute $c'_j = f(c_{i_1,j}, c_{i_2,j}, \cdots, c_{i_{|I|},j})$, and send c'_j to node e.*

3) *Node e obtains $c_e^{(1)}, c_e^{(2)}, \cdots, c_e^{(n-z)}$ by decoding the $(n, n - z, 0, z)$ scheme, regarding $c'_1, c'_2, \cdots, c'_n$ as the n shares.*

Theorem 12.2.1. *Construction 12.2.2 is a secure repair scheme.*

Proof. Similar to Theorem 12.1.1, repairability follows from the linearity of Construction 12.2.1, which implies that $c'_{[n]}$ are the shares of a $(n, n-z, 0, z)$ secret sharing scheme that encodes $(f(c_{i_1}^{(1)}, c_{i_2}^{(1)}, \cdots, c_{i_{|I|}}^{(1)}), \cdots, f(c_{i_1}^{(n-z)}, c_{i_2}^{(n-z)}, \cdots, c_{i_{|I|}}^{(n-z)})) = (c_e^{(1)}, \cdots, c_e^{(n-z)})$ as message. We now focus on security. Let $A \subset [n]$, $|A| = z$ be an arbitrary set of nodes controlled by the adversary, then $c_e^{[n-z]} \perp c'_A$. Furthermore, since Construction 12.2.1 is rate-optimal, by Lemma 3.2.1 it follows that

$H(c'_A) = z$. We have

$$\begin{aligned} H(c'_{[n]\setminus A}|c_e^{[n-z]}, c'_A) &\overset{(a)}{=} H(c_e^{[n-z]}, c'_{[n]}) - H(c_e^{[n-z]}, c'_A) \qquad (12.7)\\ &\overset{(b)}{\leq} H(c_e^{[n-z]}) + z - H(c_e^{[n-z]}, c'_A)\\ &\overset{(c)}{=} H(c_e^{[n-z]}) + z - H(c_e^{[n-z]}|c'_A) - H(c'_A)\\ &\overset{(d)}{=} H(c_e^{[n-z]}) + z - H(c_e^{[n-z]}) - H(c'_A)\\ &= z - H(c'_A)\\ &\overset{(e)}{=} 0. \qquad (12.8) \end{aligned}$$

Here (a) and (c) follow from the chain rule; (b) follows from the fact that $c'_{[n]}$ is a function of $c_e^{[n-z]}$ and z random keys; (d) follows from $c_e^{[n-z]} \perp c'_A$ and (e) follows from $H(c'_A) = z$.

Consider the case that $e \in A$, we have

$$\begin{aligned} I(\boldsymbol{m}^{[n-z]}; c_A^{[n-z]}, u_A, d_A) &= I(\boldsymbol{m}^{[n-z]}; c_A^{[n-z]}, u_A, c'_{[n]}, c_{I,A}) \qquad (12.9)\\ &\overset{(f)}{=} I(\boldsymbol{m}^{[n-z]}; c_A^{[n-z]}, u_A, c'_A, c_{I,A})\\ &\overset{(g)}{=} I(\boldsymbol{m}^{[n-z]}; c_A^{[n-z]}, u_A, c_{I,A}), \qquad (12.10)\\ &\overset{(h)}{=} I(\boldsymbol{m}^{[n-z]}; c_A^{[n-z]}, u_A, c_{I\setminus A,A})\\ &\overset{(i)}{=} I(\boldsymbol{m}^{[n-z]}; c_A^{[n-z]}, u_A|c_{I\setminus A,A}) + I(\boldsymbol{m}^{[n-z]}; c_{I\setminus A,A})\\ &\overset{(j)}{\leq} I(\boldsymbol{m}^{[n-z]}; c_A^{[n-z]}, u_A|c_{I\setminus A,A}) + I(\boldsymbol{c}^{[n-z]}; c_{I\setminus A,A})\\ &\overset{(k)}{=} I(\boldsymbol{m}^{[n-z]}; c_A^{[n-z]}, u_A|c_{I\setminus A,A})\\ &\overset{(l)}{=} I(\boldsymbol{m}^{[n-z]}; c_A^{[n-z]}, u_A)\\ &\overset{(m)}{=} I(\boldsymbol{m}^{[n-z]}; c_A^{[n-z]})\\ &\overset{(n)}{=} 0, \qquad (12.11) \end{aligned}$$

where (f) follows from (12.8); (g) follows from the fact that c'_A is a function of $c_{I,A}$; (h) follows from the fact that $c_{A,A}$ is a function of $c_A^{[n-z]}$ and u_A; (i) follows from the chain rule; (j) follows from the Markov chain $\boldsymbol{m}^{[n-z]} \to \boldsymbol{c}^{[n-z]} \to c_{I\setminus A,A}$ and the data processing inequality; (k) follows from the fact that $c_{I\setminus A,A}$ are the shares of $|I\setminus A|$ independent $(n, n-z, 0, z)$ secret sharing schemes and that for each scheme only $|A| = z$ of its shares are included; (l) follows from the fact that $(\boldsymbol{m}^{[n-z]}, c_A^{[n-z]}, u_A) \perp c_{I\setminus A,A}$, implied by (k); (m) follows from $(\boldsymbol{m}^{[n-z]}, c_A^{[n-z]}) \perp u_A$; and (n) follows from security of the secret sharing scheme being repaired.

For the case that $e \notin A$, we have $I(\boldsymbol{m}^{[n-z]}; c_A^{[n-z]}, u_A, d_A) = I(\boldsymbol{m}^{[n-z]}; c_A^{[n-z]}, u_A, c_{I,A})$, which can be treated in the same way as (12.10) - (12.11). The proof is complete. □

In Step 1, at most $|I|n$ symbols are communicated and in Step 2, at most n symbols are communicated. Therefore the total repair bandwidth is at most $(|I| + 1)n$ symbols, for repairing $n - z$ symbols. The normalized repair bandwidth to repair each symbol is at most $\frac{(|I|+1)n}{n-z}$ symbols. In the case that n dominates z, the normalized repair bandwidth approaches $|I| + 1$ symbols. Note that $|I|$ is the non-secure repair bandwidth and a trivial lower bound on the secure repair bandwidth. Therefore when n dominates z (e.g., the high rate case), the secure repair bandwidth of Construction 12.2.2 is essentially optimal. Specifically, it is essentially the same as the non-secure repair bandwidth, implying that we can have secure repair essentially for free, even in terms of repair bandwidth.

12.3 Vector Secure Repair

The secure repair schemes in Construction 12.1.2 and 12.2.2 deal with scalar secret sharing schemes, i.e., schemes that are linear over a finite field and such that each share is an element of the field. A more general class of secret sharing schemes are vector linear secret sharing schemes, also referred to as array schemes. A vector linear (n, k, r, z) secret sharing scheme over $\mathbb{F}_q^t$ is a randomized function that maps (encodes) a message $\boldsymbol{m} = (m_1, \cdots, m_k)$ of k symbols over $\mathbb{F}_q^t$ to n shares $\boldsymbol{c} = (c_1, \cdots, c_n)$ over $\mathbb{F}_q^t$, such that the encoding function is linear over $\mathbb{F}_q$ and that the same reliability and security requirements as before are met. We denote $m_i = (m_{i,1}, m_{i,2}, \cdots, m_{i,t})$, where $m_{i,j} \in \mathbb{F}_q$, for $i \in [k], j \in [t]$. Similarly we denote $c_i = (c_{i,1}, c_{i,2}, \cdots, c_{i,t})$, for $i \in [n]$. Note that scalar schemes are special cases of vector linear schemes with $t = 1$.

Many efficient secret sharing schemes, e.g., schemes with efficient computation (e.g., Chapter 4 and Chapter 5), schemes with efficient decoding bandwidth (e.g., Chapter 8), and schemes with efficient repair bandwidth (e.g., Chapter 9), are intrinsically vector linear. In this section we extend our secure repair framework to vector linear schemes. This is especially interesting because it allows us to leverage the property of secret sharing schemes with efficient (non-secure) repair bandwidth, i.e., secure regenerating codes, to further reduce the (secure) repair bandwidth.

We remark that existing secure regenerating codes can be classified into two categories: codes with non-secure repair and codes with secure repair. Secure regenerating codes with non-secure repair focus on reducing the repair bandwidth

without worrying about the security of the repair process. For example, schemes constructed in Chapter 9 belong to this category. For this case one can think of having a trustworthy repair dealer that will reconstruct the lost share and forward it to the failed node. As remarked previously, during the repair process the dealer may gain information on the message and has to be trustworthy. In comparison, regenerating codes with secure repair, by code design, guarantee that such a dealer will not learn any information about the message. This in fact removes the need for a trustworthy dealer as the failed node can act as the dealer. In this sense, secure regenerating codes with secure repair naturally admit a secure repair scheme that meets Definition 12.1.1. Particularly, the secure repair scheme is a simple "one-round" scheme in the sense that the helper nodes will transmit information to the failed node but they themselves do not need to receive information from other nodes. Unfortunately, one-round secure repair comes at a high cost in rate and codes with non-secure repair generally have a much better rate as well as repair bandwidth (when normalized by rate) than codes with secure repair [24]. Our main result in this section implies that this trade-off between rate and secure repair is not necessary: we can apply our generic approach to secure regenerating codes with non-secure repair to achieve secure repair, a good rate, and a good repair bandwidth. The only cost is that the repair process now involves two rounds instead of one round.

Construction 12.3.1. (Vector linear secure repair) *Consider any vector linear* (n, k, r, z) *secret sharing scheme over* $\mathbb{F}_q^t$*, any* $e \in [n]$*, and any* $I = \{i_1, \cdots, i_{|I|}\} \subset [n]$*,* $e \notin I$ *such that there exists* $J \subset [t]$ *and a linear function* f *over* $\mathbb{F}_q$ *that takes* $c_{i,j}$*,* $i \in I$*,* $j \in J$ *as input and outputs* $c_e = (c_{e,1}, c_{e,2}, \cdots, c_{e,t})$*. The secure repair process involves three steps:*

1) *For each node* $i \in I$*, and* $j \in J$*, encode* $c_{i,j}^{(1)}, c_{i,j}^{(2)}, \cdots, c_{i,j}^{(n-z)}$ *into* $c_{i,j,1}$*,* $c_{i,j,2}$*,* $\cdots$*,* $c_{i,j,n}$ *by an* $(n, n-z, 0, z)$ *scheme according to Construction 12.2.1 (choosing the same* α_i*'s) and send* $c_{i,j,k}$ *to node* k*.*

2) *For each node* $k \in [n]$*, compute* $(c'_{k,1}, c'_{k,2}, \cdots, c'_{k,t}) = f(c_{i,j,k})_{i \in I, j \in J}$*, and send* $c'_{k,j}$*,* $j \in [t]$ *to node* e*.*

3) *For* $j \in [t]$*, node* e *obtains* $c_{e,j}^{(1)}, c_{e,j}^{(2)}, \cdots, c_{e,j}^{(n-z)}$ *by decoding the* $(n, n-z, 0, z)$ *scheme, regarding* $c'_{1,j}, c'_{2,j}, \cdots, c'_{n,j}$ *as the* n *shares.*

Theorem 12.3.1. *Construction 12.3.1 is a secure repair scheme.*

Proof. As in Theorem 12.2.1, repairability follows from the linearity of Construction 12.2.1, which implies that $c'_{1,j}, c'_{2,j}, \cdots, c'_{n,j}$ are the shares of a $(n, n-z, 0, z)$ secret sharing scheme that encodes $c^{(1)}_{e,j}, c^{(2)}_{e,j}, \cdots, c^{(n-z)}_{e,j}$ as message, for $j \in [t]$. We now turn to security, and follow a similar flow as the proof of Theorem 12.2.1. Let $A \subset [n]$, $|A| = z$ be an arbitrary set of nodes controlled by the adversary, then $c_e^{[n-z]} \perp c'_{A,j} = 0$. Since Construction 12.2.1 is rate-optimal, by Lemma 3.2.1 it follows that $H(c'_{A,j}) = z$, for $j \in [t]$. We have, for $j \in [t]$,

$$\begin{aligned}
H(c'_{[n]\setminus A,j}|c^{[n-z]}_{e,j}, c'_{A,j}) &= H(c^{[n-z]}_{e,j}, c'_{[n],j}) - H(c^{[n-z]}_{e,j}, c'_{A,j}) \\
&\leq H(c^{[n-z]}_{e,j}) + z - H(c^{[n-z]}_{e,j}, c'_{A,j}) \\
&= H(c^{[n-z]}_{e,j}) + z - H(c^{[n-z]}_{e,j}|c'_{A,j}) - H(c'_{A,j}) \\
&= H(c^{[n-z]}_{e,j}) + z - H(c^{[n-z]}_{e,j}) - H(c'_{A,j}) \\
&= z - H(c'_{A,j}) \\
&= 0, \qquad (12.12)
\end{aligned}$$

where the justification for the steps is similar to that of (12.7) - (12.8). (12.12) implies that

$$H(c'_{[n]\setminus A,[t]}|c^{[n-z]}_{e,[t]}, c'_{A,[t]}) = 0. \qquad (12.13)$$

Now considering the case that $e \in A$, we have

$$\begin{aligned}
I(\boldsymbol{m}^{[n-z]}; c^{[n-z]}_{A,[t]}, u_A, d_A) &= I(\boldsymbol{m}^{[n-z]}; c^{[n-z]}_{A,[t]}, u_A, c'_{[n],[t]}, c_{I,J,A}) \\
&\overset{(a)}{=} I(\boldsymbol{m}^{[n-z]}; c^{[n-z]}_{A,[t]}, u_A, c'_{A,[t]}, c_{I,J,A}) \\
&= I(\boldsymbol{m}^{[n-z]}; c^{[n-z]}_{A,[t]}, u_A, c_{I,J,A}), \qquad (12.14) \\
&= I(\boldsymbol{m}^{[n-z]}; c^{[n-z]}_{A,[t]}, u_A, c_{I\setminus A,J,A}) \\
&= I(\boldsymbol{m}^{[n-z]}; c^{[n-z]}_{A,[t]}, u_A|c_{I\setminus A,J,A}) + I(\boldsymbol{m}^{[n-z]}; c_{I\setminus A,J,A}) \\
&\leq I(\boldsymbol{m}^{[n-z]}; c^{[n-z]}_{A,[t]}, u_A|c_{I\setminus A,J,A}) + I(\boldsymbol{c}^{[n-z]}; c_{I\setminus A,J,A}) \\
&\overset{(b)}{=} I(\boldsymbol{m}^{[n-z]}; c^{[n-z]}_{A,[t]}, u_A|c_{I\setminus A,J,A}) \\
&= I(\boldsymbol{m}^{[n-z]}; c^{[n-z]}_{A,[t]}, u_A) \\
&= I(\boldsymbol{m}^{[n-z]}; c^{[n-z]}_{A,[t]}) \\
&= 0, \qquad (12.15)
\end{aligned}$$

where (a) follows from (12.13); (b) follows from the fact that $c_{I\setminus A,J,A}$ are the shares of $|I\setminus A| \cdot |J|$ independent $(n, n-z, 0, z)$ secret sharing schemes and that

for each scheme only $|A| = z$ of its shares are included; and the remaining equalities/inequalities are similar to (12.9) - (12.11).

For the case that $e \notin A$, we have

$$I(\boldsymbol{m}^{[n-z]}; c_{A,[t]}^{[n-z]}, u_A, d_A) = I(\boldsymbol{m}^{[n-z]}; c_{A,[t]}^{[n-z]}, u_A, c_{I,J,A}),$$

which can be treated in the same way as (12.14) - (12.15). The proof is complete. □

Consider the repair bandwidth of the scheme. In Step 1, at most $n|I||J|$ symbols (over $\mathbb{F}_q$) are transmitted and in Step 2, at most nt symbols are transmitted. Therefore, the total repair bandwidth is at most $(|I||J| + t)n$ symbols, for repairing $(n-z)t$ symbols. The normalized repair bandwidth to repair each symbol is at most $\frac{(|I||J|+t)n}{(n-z)t}$ symbols. In the case that n dominates z, the normalized repair bandwidth approaches $\frac{|I||J|}{t} + 1$. Note that the normalized non-secure repair bandwidth is $\frac{|I||J|}{t}$, and therefore in this case the secure repair bandwidth of Construction 12.3.1 is essentially optimal and is almost the same as the non-secure repair bandwidth.

The MSR secure regenerating codes in Construction 9.2.2, Construction 9.3.2 and in [22] have optimal rate as well as optimal non-secure repair bandwidth (among rate-optimal schemes). Specifically, the rate of the scheme is $\frac{n-k-z}{n}$, and that for $|I|$ helper nodes to non-securely repair a failed node, each helper node will transmit a fraction of $\frac{1}{1+|I|-k-z}$ of the symbols it stores, i.e., $|J| = \frac{t}{1+|I|-k-z}$. By applying Construction 12.3.1 to these codes, we obtain schemes with optimal rate and low secure repair bandwidth. In the next section, we will show that this secure repair bandwidth is in fact optimal up to a small constant factor.

12.4 Lower Bound on Secure Repair Bandwidth

The bandwidths of Construction 12.2.2 and Construction 12.3.1 are significantly better than Construction 12.1.2. A natural question is whether it is possible to do even better, or in other words, what is a lower bound on the secure repair bandwidth. As we previously remarked, when n dominates z, the bandwidths of Constructions 12.2.2 and 12.3.1 approach the non-secure repair bandwidth, which is a naive lower bound. Therefore in this case the bandwidths of Constructions 12.2.2 and 12.3.1 are asymptotically optimal. In this section, we prove a stronger lower bound on the secure repair bandwidth and show that the bandwidths of Constructions 12.2.2 and 12.3.1 are optimal for all parameters up to a constant factor of 2, as long as the secret sharing scheme being repaired is rate-optimal.

Assume that a trustworthy repair dealer is available. The dealer will receive information from the helper nodes, evaluate a *repair function* that outputs the lost share, and reassign the share. In this case, the repair bandwidth is the size of the input to the repair function plus the size of the lost share. Now consider the situation that a trustworthy dealer is not available and a secure repair scheme is used for repair. The secure repair scheme essentially is a method to evaluate the repair function (e.g., f in Constructions 12.1.2, 12.2.2 and 12.3.1) securely at the failed node, and the repair bandwidth again depends on the size of the input to the repair function. The repair function is an intrinsic component of the secret sharing scheme and the size of the input can be minimized by carefully designing the secret sharing scheme, e.g., Chapter 9. Refer to the size of the input to the repair function as the non-secure repair bandwidth of a secret sharing scheme. Below we prove a lower bound on the repair bandwidth of secure repair schemes, given the non-secure repair bandwidth of the secret sharing scheme.

Theorem 12.4.1. *For any rate-optimal $(n, k = n - r - z, r, z)$ secret sharing scheme, let W be the non-secure repair bandwidth of the scheme, then a secure repair scheme requires a bandwidth of at least $\frac{(n-1)W}{2(n-z-1)}$.*

Proof. By Proposition 3.1.1, $k = n - r - z$ implies that the scheme is rate-optimal. Let the message $\boldsymbol{m} = (m_1, m_2, \cdots, m_k)$ be uniformly distributed. Then for any $I \subset [n]$, $|I| = k + z$ and $J \subset I$, $|J| = z$, by the security and the decodability of the scheme we have $I(\boldsymbol{m}; c_J) = 0$ and $I(\boldsymbol{m}; c_I) = H(\boldsymbol{m}) = k$. It follows that

$$\begin{aligned} I(\boldsymbol{m}; c_{I\backslash J}|c_J) &= H(\boldsymbol{m}|c_J) - H(\boldsymbol{m}|c_I) \\ &= H(\boldsymbol{m}) - H(\boldsymbol{m}|c_I) \\ &= I(\boldsymbol{m}; c_I) \\ &= k. \end{aligned} \tag{12.16}$$

Since $|I\backslash J| = k$, $H(c_{I\backslash J}) \leq k$, and hence (12.16) implies that $H(c_{I\backslash J}) = k$ and $c_{I\backslash J} \perp c_J$ and that

$$H(c_{I\backslash J}|\boldsymbol{m}, c_J) = 0. \tag{12.17}$$

Therefore among the n shares of the secret sharing scheme, any $|I\backslash J| = k$ shares are uniformly distributed and that any $|J| = z$ shares are independent of any other k shares. This in turn implies that any $k + z$ shares are uniformly distributed, i.e.,

$$H(c_I) = k + z. \tag{12.18}$$

Assume that c_e is lost, and for $i \in [n]\backslash\{e\}$, let w_i be the information sent by node i to node e for non-secure repair, namely, the input signal to the repair function from node i (with the convention that $w_i = 0$ if node i does not participate in the repair). Then w_i is a function of c_i and $\sum_{i\in[n]\backslash\{e\}} H(w_i) = W$. Now consider any secure repair protocol, and for $i \in [n]\backslash\{e\}$, $j \in [n]$, let $v_{i,j}$ be the set of signals that are sent to node j by node i or sent to node i by node j during the protocol (with the convention that $v_{i,i} = \emptyset$). Then w_i must be a function of the signals incoming to and outgoing from node i, namely, $H(w_i|v_{i,[n]}) = 0$, implying that

$$I(w_i; v_{i,[n]}) = H(w_i). \tag{12.19}$$

Let A be an arbitrary set of nodes controlled by the adversary such that $i \notin A$, $|A| = z$, and let B be an arbitrary set of nodes such that $i \in B$, $|B| = k$, $A \cap B = \emptyset$. We have

$$\begin{aligned}
I(w_i; v_{i,A}) &= H(w_i) - H(w_i|v_{i,A}) \\
&\overset{(a)}{=} H(w_i|c_A) - H(w_i|v_{i,A}) \\
&\leq H(w_i|c_A) - H(w_i|v_{i,A}, c_A) \\
&= I(w_i; v_{i,A}|c_A) \\
&\overset{(b)}{\leq} I(c_i; v_{i,A}|c_A) \\
&\leq I(c_B; v_{i,A}|c_A) \\
&\overset{(c)}{\leq} I(\boldsymbol{m}; v_{i,A}|c_A) \\
&\overset{(d)}{=} I(\boldsymbol{m}; v_{i,A}|c_A) + I(\boldsymbol{m}; c_A) \\
&= I(\boldsymbol{m}; v_{i,A}, c_A) \\
&\overset{(e)}{=} 0.
\end{aligned} \tag{12.20}$$

Here (a) follows from the fact that w_i is a function of c_i and by (12.18), $c_i \perp c_A$; (b) follows from the data processing inequality and the fact that w_i is a function of c_i; (c) follows from the data processing inequality and (12.17), i.e., c_B is a function of $\boldsymbol{m}$ given c_A; (d) follows from the security of the secret sharing scheme; and (e) follows from the security of the repair scheme. Let

$$A^* = \underset{A\subset[n]\backslash\{i\},|A|=z}{\operatorname{argmax}} \sum_{l\in A} H(v_{i,l}),$$

and let $\bar{A}^* = [n]\backslash(\{i\} \cup A^*)$, then by definition, for $j \in \bar{A}^*$ and $j^* \in A^*$, $H(v_{i,j}) \leq H(v_{i,j^*})$. We have

$$\begin{aligned} H(v_{i,\bar{A}^*}) &\geq I(w_i; v_{i,\bar{A}^*}|v_{i,A}) \\ &\overset{(f)}{=} I(w_i; v_{i,\bar{A}^*}|v_{i,A}) + I(w_i; v_{i,A}) \\ &= I(w_i; v_{i,[n]}) \\ &\overset{(g)}{=} H(w_i), \end{aligned}$$

where (f) follows from (12.20) and (g) follows from (12.19). Therefore there exists $j \in \bar{A}^*$ such that $H(v_{i,j}) \geq H(w_i)/|\bar{A}^*|$ and so for $j^* \in A^*$, $H(v_{i,j^*}) \geq H(w_i)/(n-z-1)$. Therefore the amount of information transmitted and received by node i is lower bounded by

$$\begin{aligned} \sum_{j\in[n]} H(v_{i,j}) &\geq H(v_{i,\bar{A}^*}) + |A^*|\frac{H(w_i)}{n-z-1} \\ &= \frac{(n-1)H(w_i)}{n-z-1}. \end{aligned} \tag{12.21}$$

Summing (12.21) over all $i \in [n]\backslash\{e\}$, it follows that the amount of information transmitted and received by nodes in $[n]\backslash\{e\}$ is at least $\frac{(n-1)W}{n-z-1}$. Since the amount of communication is counted exactly twice, i.e., when information is transmitted and when it is received, the repair bandwidth of the scheme is lower bounded by $\frac{(n-1)W}{2(n-z-1)}$. This completes the proof. □

The bandwidths of Constructions 12.2.2 and 12.3.1 are upper bounded by $\frac{(W+1)n}{n-z}$, and therefore are optimal up to a factor of approximately 2 by Theorem 12.4.1.

12.5 Concluding Remarks

We study the problem of repairing a share in a secret sharing scheme securely without leaking any information about the message. Secure regenerating codes, a special class of secret sharing schemes, achieve secure repair at a significant cost of rate and other aspects of efficiency. We show that, by slightly relaxing the repair model and allowing an efficient and simple 2-round repair protocol, any linear secret sharing schemes can be securely repaired in a generic manner. We derive a lower bound on the secure repair bandwidth, and show that the proposed secure repair schemes achieve this bound within a small constant factor either when n dominates z, or when the secret sharing scheme being repaired is rate-optimal. Particularly, when n dominates z, the secure repair bandwidth of the proposed repair schemes approaches the non-secure repair bandwidth.

A natural follow up problem for future study is to consider an active adversary that may deviate from the repair protocol, for example, by sending wrong information. Another problem is to characterize the tradeoff between the secure repair bandwidth and rate. This tradeoff has attracted considerable interest recently but existing works only consider the one-round repair model. It would be interesting to study how does relaxing the one-round restriction affect this tradeoff.

Part IV

Coding for Networks

Chapter 13

INTRODUCTION TO NETWORK CODING

In Parts I - III, we discussed how to achieve security and reliability in distributed storage systems with (various aspects of) optimal efficiency by the means of coding. So far we have assumed that the nodes are connected in a perfect network with secure and reliable channels (e.g., when a user sends/receives a bit to/from a node). In reality, nodes in distributed systems are connected by networks that are subject to errors, eavesdropping and connectivity constraints. In this final part of the thesis, we study how to achieve secure and reliable communication in general networks with optimal efficiency, by the technique of coding, namely, *network coding*. We remark that the communication problem is a generalization of the storage problem as well, as storage can be modeled as communication over time.

In the paradigm of *network coding*, a set of source nodes transmit information to a set of terminal nodes over a network with noiseless links; internal nodes of the network may mix received information before forwarding it. This mixing (or encoding) of information has been extensively studied over the last decade (see e.g., [72]–[76], and the references therein). In particular, the problems of determining the *capacity* of the network and designing optimal codes achieving the capacity are well understood under the *multicast* setting, where there is a single source node whose information is demanded by all terminal nodes. However, much less is known regarding the *multiple-unicast* setting where there are multiple source nodes, each of them demanded by a single and different terminal node (the more general setting of *multiple-multicast*, where each terminal node may demand the information from an arbitrary subset of source nodes, can be converted to an equivalent multiple-unicast setting using the constructions in [77], [78]). Determining the capacity or the achievability of a rate tuple in multiple-unicast network coding remains an intriguing, central, open problem, e.g., [25]–[27].

13.1 Equivalence between Network Coding Problems

We connect multiple-unicast network coding to two other fundamental network coding problems, namely secure network coding and network error correction, using the idea of reduction. We focus on the *decision problem* of determining whether a rate tuple is achievable in the network, as well as the decision problem

of determining whether a rate tuple is in the capacity region of the network. Here a rate tuple is achievable if there exists a code that exactly achieves it, and the capacity region is the closure of the set of all achievable rate tuples. Note that by definition, a rate tuple that is achievable is always in the capacity region, and a rate tuple that is in the capacity region is always asymptotically achievable but may not be (exactly) achievable (e.g., when the rate tuple lies at the boundary of the capacity region and the capacity region is open). We study the connection between different problems using the technique of *reduction*. Loosely speaking, we say a class of decision problems $\mathcal{I}$ can be reduced to a class of decision problems $\mathcal{J}$ if there exists a scheme that maps any problem instance in $\mathcal{I}$ to a corresponding problem instance in $\mathcal{J}$, such that the answers to the two instances are the same. In other words, using such a reduction methodology, one may take an instance of $\mathcal{I}$, map it to a corresponding instance of $\mathcal{J}$, solve the decision problem on the instance of $\mathcal{J}$, and deduce a solution for the original instance of $\mathcal{I}$; implying that if there is a way to solve the problems in $\mathcal{J}$, then there is a way to solve the problems in $\mathcal{I}$.

We construct a reduction, i.e., a mapping, from the problem of multiple-unicast network coding to the problem of unicast secure network coding. Note that the latter problem is asked under the simplest setting of *unicast*, where there is a single source node and a single terminal node in the network. Therefore under the unicast setting the rate tuple degenerates to a scalar rate. Our reduction addresses the problem of determining whether a rate/rate tuple is achievable, as well as the problem of determining whether a rate/rate tuple is in the capacity region. Secondly, we construct another reduction from the problem of multiple-unicast network coding to the problem of unicast network error correction. Note that again the latter problem is asked under the simplest setting of unicast. Surprisingly, we show that our reduction only works for the problem of determining whether a rate/rate tuple is achievable. In contrast, for the problem of determining whether a rate/rate tuple is in the capacity region, we show that the same reduction mapping no longer guarantees that it does not change the answers to the instances. An interesting consequence of this negative result is that the capacity of a unicast network error correction instance in general is not (exactly) achievable.

Compared to the standard network coding problem, secure network coding and network error correction have additional secrecy and reliability requirements, described in the following subsections.

13.1.1 Secure Network Coding

Secure network coding is a natural generalization of network communication to networks with eavesdroppers. Specifically, in the secure network coding problem, a subset $A \in \mathcal{A}$ of links may be eavesdropped, where $\mathcal{A}$ is given and is the collection of all possible eavesdropping patterns. A valid code design for the secure network coding problem needs to ensure the secrecy of the source message. Namely, for any choice of A from $\mathcal{A}$, the mutual information between the set of signals transmitted on the links in A and the source messages must be negligible. The secure network coding problem is well studied in the literature and in particular is well understood in the multicast setting under the assumption that 1) all links have equal capacity, and 2) $\mathcal{A}$ is uniform, i.e., $\mathcal{A}$ includes all subsets of links of size z_w, where z_w is the number of wiretapped links, and 3) only the source node can generate randomness, e.g., [79]–[82]. In the cases that either link capacities are not equal, or $\mathcal{A}$ is arbitrary, or non-source nodes may generate randomness, determining the achievability of a rate or the capacity in the secure network coding problem remains open, e.g., [83]–[86].

We show that an arbitrary multiple-unicast network coding instance $\mathcal{I}$ can be reduced to a particular unicast secure network coding instance $\mathcal{I}_s$ that has a very simple setup. Specifically, the reduction mapping ensures that in $\mathcal{I}_s$ a) there is a single source node and a single terminal node in an acyclic network; b) all links have equal capacity; c) there is a single wiretapped link and this link can be any link in the network, namely, $\mathcal{A}$ is uniform with $z_w = 1$; and d) non-source nodes are allowed to generate randomness. The setup of $\mathcal{I}_s$ is simple in the sense that setup a) is the simplest connection requirement, b) is the simplest assumption on link capacities, and c) gives the simplest structure of a non-trivial $\mathcal{A}$. Indeed, under the setup of a) - c) the secrecy capacity of the network is characterized by the cut-set bounds and is achieved by linear codes [79]. In this sense, our reduction suggests that the addition of setup d) is critical; as the secure network coding problem is simple and well understood under setting a) - c), but under setting a) - d) it is as hard as the long standing open problem of multiple-unicast network coding. We remark that allowing non-source nodes to generate randomness is realistic and preferable because this can significantly increase the secrecy capacity of the network [86], [87].

Our reduction addresses the problem of determining the achievability of a rate/rate tuple, as well as the problem of determining whether a rate/rate tuple is in the capacity region. Furthermore, our reduction holds for different types of security

requirements. Namely, in $\mathcal{I}_s$ we may assume either perfect, strong, or weak security.

Our reduction has an operational aspect. Namely, in our reduction, from a code for $\mathcal{I}_s$ one can construct a code for $\mathcal{I}$. Thus, to construct codes for an instance of the multiple-unicast network coding problem, one may first reduce it to an instance of the unicast secure network coding problem, construct codes for the latter, and finally use them to obtain codes for the original instance. We conclude, speaking loosely, that unicast secure network coding under the simple setting described above is at least as hard as multiple-unicast network coding. Our formal results are given in Section 14.1.

The hardness of the general secure network coding problems are previously studied in [83] and [86]. Specifically, Chan and Grant [83] show that determining the zero-error achievability of a rate in the multicast secure network coding problem with general setup (i.e., arbitrary edge capacities, arbitrary $\mathcal{A}$, and arbitrary nodes may generate randomness) and with perfect security is at least as hard as determining the zero-error achievability of a rate tuple in the multiple-multicast network coding problem. Cui et al. [86] show that determining the capacity of a unicast secure network coding problem is NP-hard if either the edge capacities are arbitrary or $\mathcal{A}$ is arbitrary. Our work significantly strengthens the result in [83] by showing that the secure network coding problem under an extremely simple setup (i.e., unicast, equal link capacities, uniform $\mathcal{A}$ with a single wiretap link) is still hard, under various definitions of achievability and security.

13.1.2 Network Error Correction

We now turn to the *network error correction* problem, which is a natural generalization of network communication to networks with *adversarial* errors. Specifically, in the network error correction problem a subset $B \in \mathcal{B}$ of links may be erroneous, where $\mathcal{B}$ is given and is the collection of all possible link error patterns. A valid code design for the error correction problem needs to ensure reliable communication between the sources and terminals in the *worst case* no matter which set $B \in \mathcal{B}$ of links are chosen to be erroneous and what specific erroneous signals are being transmitted on these links. Namely, in this context, the communication of a message is successful if for any choice of B from $\mathcal{B}$, and for any (error) signals to be transmitted on the links in B, the message is correctly decoded at the terminal nodes. The network error correction problem is extensively studied and in particular is well understood under the multicast setting with the assumption that 1) all links have

equal capacity, and 2) $\mathcal{B}$ is uniform, i.e., $\mathcal{B}$ includes all subsets of links of size z_e, where z_e is the number of erroneous links, e.g., [28]–[30], [88]–[90]. In the cases that either link capacities are not equal or $\mathcal{B}$ is arbitrary, determining the capacity of the network or the achievability of a rate remains an open problem, e.g., [91]–[95].

In a flavor similar to the reduction described in the previous subsection, we show that an arbitrary multiple-unicast network coding instance $\mathcal{I}$ can be reduced to a particular unicast network error correction instance $\mathcal{I}_c$ that has a very simple setup. Specifically, the reduction mapping ensures that in $\mathcal{I}_c$ a) there is a single source node and a single terminal node in an acyclic network; b) all links have equal capacity; c) there is a single error link; and d) the error link can be any link in the network except a given subset of (well protected) links. The setup of $\mathcal{I}_c$ is simple in the sense that setup a) is the simplest connection requirement, b) is the simplest assumption on link capacities and c) gives the smallest number of error links. Indeed, if the error link can be any one link in the network (namely if $\mathcal{B}$ is uniform), then under the setup of a) - c) the capacity of the network is characterized by the cut-set bounds and is achieved by linear codes [28]. In this sense, our reduction suggests that the addition of setup d), which will result in a non-uniform $\mathcal{B}$, is critical; as the network error correction problem is simple and well understood under setting a) - c), but under setting a) - d) it is as hard as the long standing open problem of multiple-unicast network coding.

Our reduction addresses the problem of determining the achievability of a rate/rate tuple. Namely by our reduction one can determine the achievability of a rate tuple in $\mathcal{I}$ by determining the achievability of a corresponding rate in $\mathcal{I}_c$. However, rather interestingly, the same reduction (i.e., the same mapping) does not address the problem of determining whether a rate/rate tuple is in the capacity region. Specifically, we show that there exists a multiple-unicast network coding instance $\mathcal{I}$ and a rate tuple such that the rate tuple is *not* in the capacity region of $\mathcal{I}$, and that after applying our reduction mapping, the corresponding rate is in the capacity region of the corresponding unicast network error correction instance $\mathcal{I}_c$. The fact that our reduction “works" when considering (exact) achievability and does not “work" when considering capacity, implies that in general the capacity of a unicast network error correction instance is not (exactly) achievable, which is a result of separate interest.

Similar to previous discussion, our reduction has an operational aspect that from a code for $\mathcal{I}_c$ one can construct a code for $\mathcal{I}$. Indeed, to construct codes for an instance

of the multiple-unicast network coding problem, one can first reduce it to an instance of the unicast network error correction problem, construct codes for the latter, and finally use them to obtain codes for the original instance. We conclude, speaking loosely, that unicast network error correction under the simple setting described above is at least as hard as multiple-unicast network coding. Our formal results are given in Chapter 15.

13.1.3 Equivalence via Reverse Reduction

The above constructions reduce instances of the multiple-unicast network coding problem to instances of the unicast secure network coding or the unicast network error correction problem. A natural and intriguing question is whether these problems are equivalent, namely, whether it is possible to construct the reverse reductions. Using the technique of A-enhanced networks [96], we show that an arbitrary unicast secure network coding instance in which at most one link is eavesdropped (i.e., $\mathcal{A}$ includes only singleton sets) can be reduced to a multiple-unicast network coding instance, thus implying an equivalence between the two problems. The formal result is presented in Section 14.2. For more complicated $\mathcal{A}$, whether a reverse reduction exists or not remains an open problem. The existence of a reverse reduction from unicast network error correction to multiple-unicast network coding also remains open.

Finally, we remark that reductions between several other network coding problems are studied in, e.g., [77], [78], [97]–[102].

13.2 Bounds for Secure Network Coding

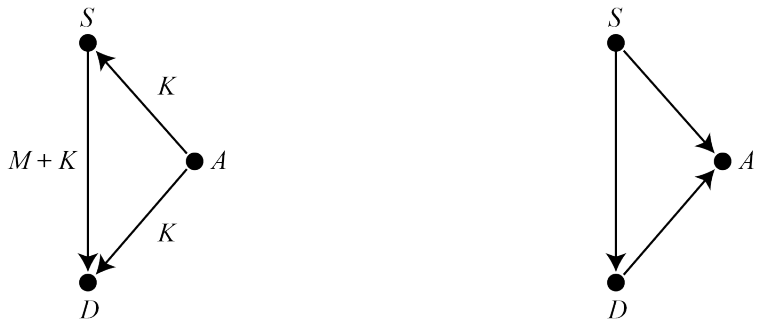
As discussed in Section 13.1.1, the unicast secure network coding problem is well understood under the setting that all links have equal capacity, $\mathcal{A}$ is uniform and only the source node generates randomness. Specifically, assuming that all edges have unit capacity, let μ be the min-cut of the network and let z_w be the number of eavesdropped edges, then the secrecy capacity of the network is $\mu - z_w$ which can be achieved by linear codes [79]. However, as discussed above, the same problem remains open and is at least as hard as the multiple-unicast network coding problem once non-source nodes are allowed to generate randomness. To the best of our knowledge, in this case the only existing bounds on the secrecy capacity are given implicitly in terms of entropy functions/entropic region [27], [85], whereas determining the entropic region is a long standing open problem as well.

We give the first explicit upper bound on the secrecy capacity for the case that

non-source nodes can generate randomness. Our bound is based on cut-sets and has an intuitive graph-theoretic interpretation. The key observation is that unlike standard cut-set bounds which only count forward edges because only forward edges are useful, in a secure network coding problem where non-source nodes can generate randomness, backward edges may also be helpful and should be counted. Refer to Figure 13.1-(a) for an example. Here the backward edge (A, S) can transmit a random key back to the source S to protect the message, enabling the secure communication of the message M. Without the backward edge, secure communication is not possible. Therefore, standard cut-set bounds that only count forward edges are no longer valid upper bounds on the secrecy capacity if non-source nodes can generate randomness. A simple fix of this problem is to ignore the direction of the edges, namely, to regard all edges as forward edges. Unfortunately, this will lead to a very loose bound, because the direction of the edge is indeed important and in many situations, the usefulness of an edge does depend on whether it is pointing forward. Refer to Figure 13.1-(b) for an example, in which the backward edge (D, A) is not useful, but if its direction is reversed, then an edge (A, D) would be useful.

Therefore, it is important to understand when a backward edge is useful and count its contribution carefully in the bound. Notice that in Figure 13.1, the networks in (a) and (b) are identical from the perspective of cuts because each of them contains a cut with two forward edges and a cut with one forward edge and one backward edge. Hence to distinguish them we have to see beyond the cut: in this simple example the backward edge in (a) is helpful because it is connected to a forward edge, while the one in (b) is not. More generally, this motivates us to take into account the connectivity from the backward edges to forward edges, described by a binary connectivity matrix C. We show that the rank structure of the submatrices of C characterizes the utility of the backward edges, and use it to obtain an upper bound on the secrecy capacity.

Finally, we show that given any cut and connectivity pattern, we can construct a network with the given cut and connectivity, such that the proposed bound on the secrecy capacity is tight in this network and can be achieved by linear codes. In this sense, the proposed bound is as tight as possible if the input to the bound is a local cut and the connectivity of the edges beyond the cut, characterized by a binary matrix.



(a) Backward edge is helpful. (b) Backward edge is not helpful.

Figure 13.1: Networks with unit capacity edges and $z = 1$. S is the source and D is the terminal. M is the source message and K is a random key.

13.3 Rateless Network Error Correction

We discussed in Section 13.1.2 that the unicast network error correction problem is well understood under the setting that all links have equal capacity and that $\mathcal{B}$ is uniform. In this case the capacity of the network is characterized by the network singleton bound [28], [29], which is a generalization of the classic singleton bound. Specifically, assuming that all edges have unit capacity, let μ be the min-cut of the network, then the capacity of a network with z_e error edges equals $\mu - 2z_e$, and the capacity of a network with z_e edge erasures equals $\mu - z_e$.

The gap between the network "error-correction" capacity and the network "erasure-correction" capacity is intuitive, as in the case of classic error-correcting codes, we need r redundant symbols to correct r erasures, and need $2r$ redundant symbols to correct r errors. Interestingly, this gap can be closed if we assume a slightly weaker adversary and allow a vanishingly small probability of error. Specifically, assume that there is a secure side channel with vanishingly small capacity between the source and the terminal, so that the bits transmitted by this channel are reliable and private (and the adversary cannot learn them). Then under this model the capacity of a network with z_e error edges is increased to $\mu - z_e$ [30]. It is shown in [31] that the secure side channel can be replaced by a vanishingly small amount of shared private randomness between the source and the terminal.

While the codes in [28]–[31] are rate-optimal, they require the parameters μ and z_e to be known for the purpose of code construction. This may restrict their applications in many practical settings. For example, estimating the min-cut of the network can be costly; the min-cut may change over time; and the number of links controlled by the adversary may not be available. To address this issue, we study *rateless* network

error correction codes, i.e., coding schemes that do not require prior knowledge of the network and adversary parameters.

We design rateless coding schemes for both the secure side channel model and the shared private randomness model. Both coding schemes are optimal and achieve the network capacity $\mu - z_e$. Since the schemes do not require the knowledge of μ and z_e, they have to adapt themselves to the correct parameters. Loosely speaking, the idea is as follows. Regard the source message M as a vector space of dimension k (in reality the message is a basis of the space, but this difference is immaterial [89]), and the communication of M involves multiple stages. In each stage the source makes one transmission of M to the terminal, so that the terminal will receive a space of dimension μ, which contains a subspace of M of dimension $\mu - z_e$ (as the dimension of the space spanned by the error signals injected by the adversary is at most z_e). Therefore after $t = \frac{k}{\mu - z_e}$ stages, the terminal has accumulated a space of dimension $t\mu$ that contains M as a subspace. The problem for the terminal is to pinpoint M from the dimension-$t\mu$ space. The idea is to generate short "signatures" of M, delivered to the terminal by the secure side channel, or by carefully maneuvering the shared private randomness. With the signatures the terminal can reconstruct M efficiently. In contrast, the adversary, without knowing the signatures, is highly unlikely to be able to cause a decoding error by fabricating a different subspace M' whose signatures collide with M.

We also put our schemes into perspective with the rich collection of works, e.g., [103]–[107], on cryptographic error control schemes for network coding systems, which detect and remove error packets injected by a computationally limited adversary. Cryptographic schemes like these operate in a rateless manner independently of the network and adversary parameters. However, in order to remove error packets promptly before they contaminate others, frequent cryptographic verification of packets is necessary at intermediate network nodes. By contrast, our schemes are information-theoretically secure, lightweight, end-to-end and do not require any collaboration from intermediate network nodes.

The material in this part of the thesis was presented in part in [108], [109], [110], [111], [112] and [113].

13.4 Models and Definitions

13.4.1 Multiple-unicast Network Coding

A network is a directed acyclic graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where vertices represent network nodes and edges represent links. Each edge $e \in \mathcal{E}$ has a capacity c_e, which is the number of bits that can be transmitted on e in one transmission. An instance $\mathcal{I} = (\mathcal{G}, \mathcal{S}, \mathcal{T}, B)$ of the *multiple-unicast network coding problem*, includes a network $\mathcal{G}$, a set of source nodes $\mathcal{S} \subset \mathcal{V}$, a set of terminal nodes $\mathcal{T} \subset \mathcal{V}$ and an $|\mathcal{S}|$ by $|\mathcal{T}|$ connection requirement matrix B. The (i, j)-th entry of B equals 1 if terminal j requires the information from source i and equals 0 otherwise. B is assumed to be a permutation matrix so that each source is paired with a single terminal. Denote by $s(t)$ the source required by terminal t. Denote $[n] \triangleq \{1, .., n\}$. Each source $s \in \mathcal{S}$ is associated with an independent message, represented by a random variable M_s uniformly distributed over $[2^{nR_s}]$. A *network code* of length n is a set of encoding functions ϕ_e for every $e \in \mathcal{E}$ and a set of decoding functions ϕ_t for each $t \in \mathcal{T}$. For each $e = (u, v)$, the encoding function ϕ_e is a function taking as input the signals received from the incoming edges of node u, as well as the random variable M_u if $u \in \mathcal{S}$. ϕ_e evaluates to a value in $\{0, 1\}^{nc_e}$, which is the signal transmitted on e. For each $t \in \mathcal{T}$, the decoding function ϕ_t maps the tuple of signals received from the incoming edges of t, to an estimated message $\hat{M}_{s(t)}$ with values in $[2^{nR_{s(t)}}]$.

A network code $\{\phi_e, \phi_t\}_{e \in \mathcal{E}, t \in \mathcal{T}}$ is said to *satisfy* a terminal t under transmission $(m_s, s \in \mathcal{S})$ if $\hat{M}_{s(t)} = m_{s(t)}$ when $(M_s, s \in \mathcal{S}) = (m_s, s \in \mathcal{S})$, namely, terminal t decodes correctly when the message tuple takes the specific value $(m_s, s \in \mathcal{S})$. A network code is said to satisfy the multiple-unicast network coding instance $\mathcal{I}$ with error probability ϵ if the probability that all $t \in \mathcal{T}$ are simultaneously satisfied is at least $1 - \epsilon$. The probability is taken over the joint distribution on $(M_s, s \in \mathcal{S})$. Formally, the network code satisfies $\mathcal{I}$ with error probability ϵ if

$$\Pr_{(M_s, s \in \mathcal{S})} \left\{ \bigcap_{t \in \mathcal{T}} t \text{ is satisfied under } (M_s, s \in \mathcal{S}) \right\} \geq 1 - \epsilon.$$

For an instance $\mathcal{I}$ of the multiple-unicast network coding problem, the rate tuple $(R_s, s \in \mathcal{S})$ is said to be *achievable* if for any $\epsilon > 0$, there exists a network code that satisfies $\mathcal{I}$ with error probability at most ϵ. $(R_s, s \in \mathcal{S})$ is said to be *achievable with zero error* if there exists a network code that satisfies $\mathcal{I}$ with zero error probability. $(R_s, s \in \mathcal{S})$ is said to be *asymptotically achievable* if for any $\delta > 0$, rate tuple $((1 - \delta)R_s, s \in \mathcal{S})$ is achievable. The capacity region of $\mathcal{I}$ is the closure of all rate tuples that are achievable, i.e., the set of all rate tuples that are asymptotically

achievable. Loosely speaking, under our definition, zero-error achievability does not allow any slackness in either the probability of error or rate; achievability allows slackness in the probability of error but not in rate; and asymptotic achievability allows slackness in both the probability of error and rate. We remark that asymptotic achievability is the more commonly used definition in the literature when capacity is concerned. However, in addition to asymptotic achievability, extra efforts will be placed on (exact) achievability in Chapter 15 because interestingly, the reduction therein allows slackness in the probability of error but does not allow slackness in rate.

Without loss of generality, we assume that all entries in the rate tuple are unit, i.e., $R_s = 1, \forall s \in \mathcal{S}$, because a varying rate source s can be modeled by multiple unit rate sources co-located at s. We say that unit rate is achievable, achievable with zero error, or asymptotically achievable if $R_s = 1, \forall s \in \mathcal{S}$ and $(R_s, s \in \mathcal{S})$ is achievable, achievable with zero error, or asymptotically achievable, respectively.

13.4.2 Unicast Secure Network Coding

An instance $\mathcal{I}_s = (\mathcal{G}, s, t, \mathcal{A})$ of the *unicast secure network coding problem* includes a network $\mathcal{G}$, a source node s, a terminal node t and a collection of subsets of links $\mathcal{A} \subset 2^{\mathcal{E}}$ susceptible to eavesdropping. Each node $i \in \mathcal{V}$ generates an independent random variable K_i.[1] The source node holds a rate-R_s secret message M uniformly distributed over $[2^{nR_s}]$. A (secure) network code of length n is a set of encoding functions ϕ_e for every $e \in \mathcal{E}$ and a decoding function ϕ_t. For each $e = (u, v)$, the encoding function ϕ_e is a function taking as input the locally generated randomness K_u, the signals received from the incoming edges of node u, and the message M if $u = s$. ϕ_e evaluates to a value in $\{0, 1\}^{nc_e}$, which is the signal transmitted on e. The decoding function ϕ_t maps the tuple of signals received from the incoming edges of t, to an estimated message $\hat{M}$ with values in $[2^{nR_s}]$.

A secure network code $\{\phi_e, \phi_t\}_{e \in \mathcal{E}}$ is said to *satisfy* instance $\mathcal{I}_s$ with error probability ϵ if the probability that $M = \hat{M}$ is at least $1 - \epsilon$, where the probability is taken over the distribution on M and K_i, $i \in \mathcal{V}$. For any edge $e \in \mathcal{E}$, denote by $X_i(e)$ the signal transmitted on e during the i-th channel use. For a subset of edges A, denote by $X^n(A) = (X_i(e) : 1 \leq i \leq n, e \in A)$. The network code is said to satisfy the

[1] We remark that in the secure network coding problem, allowing non-source nodes to generate randomness can significantly increase the capacity [86], [87] and therefore is preferable. In contrast, for simplicity in the multiple-unicast problem and the network error correction problem we do not assume that non-source nodes can generate randomness.

perfect security requirement if for all $A \in \mathcal{A}$, $I(M; X^n(A)) = 0$; the *strong security* requirement if for all $A \in \mathcal{A}$, $I(M; X^n(A)) \to 0$ as $n \to \infty$; and the *weak security* requirement if $\forall A \in \mathcal{A}$, $\frac{I(M;X^n(A))}{n} \to 0$ as $n \to \infty$.

For a unicast secure network coding instance $\mathcal{I}_s$, rate R_s is said to be *achievable* with perfect, strong, or weak security if for any $\epsilon > 0$, there exist network codes that satisfy $\mathcal{I}_s$ with error probability at most ϵ and the corresponding security requirement. Rate R_s is said to be *achievable with zero error* and with perfect, strong, or weak security if there exist network codes that satisfy $\mathcal{I}_s$ with zero error probability and the corresponding security requirement. Rate R_s is said to be *asymptotically achievable* with perfect, strong, or weak security if for any $\delta > 0$, rate $(1 - \delta)R_s$ is achievable with the corresponding security requirement. The capacity of $\mathcal{I}_s$ under perfect, strong, or weak security is the supremum over all rates that are achievable with the corresponding security requirement.

13.4.3 Unicast Network Error Correction

An instance $\mathcal{I}_c = (\mathcal{G}, s, t, \mathcal{B})$ of the *unicast network error correction problem* includes a network $\mathcal{G}$, a source node s, a terminal node t and a collection of subsets of links $\mathcal{B} \subset 2^{\mathcal{E}}$ susceptible to errors. An error occurs in a link if the output of the link is different from the input. More precisely, the output of a link e is the addition of the input signal and an error signal $r_e \in \{0, 1\}^{nc_e}$, and an error occurs in link e if and only if r_e is not the zero vector[2]. For a subset B of links, a B-error is said to occur if errors occur in every link in B. The source node holds a rate-R_c message M uniformly distributed over $[2^{nR_c}]$, and the decoder of the terminal outputs an estimated message $\hat{M}$.

Let $\boldsymbol{r} = (r_e)_{e\in\mathcal{E}}$ be the tuple of error signals, referred to as an error pattern. Denote by $\mathcal{R}_{\mathcal{B}}$ the set of all possible error patterns, namely, $\mathcal{R}_{\mathcal{B}} = \{\boldsymbol{r} : \text{non-zero entries in } \boldsymbol{r} \text{ correspond to } B\text{-errors}, B \in \mathcal{B}\}$. A network code $\{\phi_e, \phi_t\}_{e\in\mathcal{E}}$, defined similarly as in Section 13.4.1, is said to *satisfy* $\mathcal{I}_c$ under transmission m if $\hat{M} = m$ when $M = m$, regardless of the occurrence of any error pattern $\boldsymbol{r} \in \mathcal{R}_{\mathcal{B}}$. A network code is said to satisfy problem $\mathcal{I}_c$ with error probability ϵ if the probability that $\mathcal{I}_c$ is satisfied is at least $1 - \epsilon$. The probability is taken over the distribution of the source message M. Note that our model targets the worst-case (or adversarial) scenario, namely the probability of error is upper bounded by ϵ even in the occurrence of the

[2]Note that our model and results can be generalized naturally to arbitrary alphabets in which addition may not be defined, as long as we make the convention that $r_e = 0$ if and only if the output of e is identical to the input.

worst case error pattern.

For a unicast network error correction problem $\mathcal{I}_c$, rate R_c is said to be *achievable* if for any $\epsilon > 0$, there exists a network code that satisfies $\mathcal{I}_c$ with error probability at most ϵ. Rate R_c is said to be *achievable with zero error* if there exists a network code that satisfies $\mathcal{I}_c$ with zero error probability. Rate R_c is said to be *asymptotically achievable* if for any $\delta > 0$, rate $(1 - \delta)R_c$ is achievable. The capacity of $\mathcal{I}_c$ is the supremum over all rates that are achievable.

Chapter 14

CONNECTING MULTIPLE-UNICAST AND SECURE NETWORK CODING

14.1 Reducing Multiple-unicast to Unicast Secure Network Coding

Recall from Section 13.4.1 that in a multiple-unicast problem, unit rate is asymptotically achievable if $R_s = 1, \forall s \in \mathcal{S}$ and rate tuple $(R_s, s \in \mathcal{S})$ is asymptotically achievable. The following theorem reduces the problem of determining the asymptotic achievability of unit rate in a general multiple-unicast network coding instance to the problem of determining the asymptotic achievability of a rate in a particular unicast secure network coding instance that has a very simple setup. We remark that although the theorem addresses the achievability of unit rate instead of a general rate tuple in the multiple-unicast network coding problem, this is without loss of generality, because the problem of determining the achievability of an arbitrary rate tuple with rational entries in a multiple-unicast instance can be converted to the problem of determining the achievability of unit rate in a corresponding multiple-unicast instance, by modeling a varying rate source s as multiple unit rate sources co-located at s.

Theorem 14.1.1. *Given any multiple-unicast network coding instance $\mathcal{I}$ with source-terminal pairs $\{(s_i, t_i), i = 1, ..., k\}$, a corresponding unicast secure network coding instance $\mathcal{I}_s = (\mathcal{G}, s, t, \mathcal{A})$, in which $\mathcal{A}$ includes all sets of a single edge (i.e., all singletons), can be constructed according to Construction 14.1.1, such that unit rate is asymptotically achievable in $\mathcal{I}$ if and only if rate k is asymptotically achievable in $\mathcal{I}_s$ under either perfect, strong, or weak security.*

Construction 14.1.1. *Given any multiple-unicast network coding instance $\mathcal{I}$ on a network $\mathcal{N}$ with source-terminal pairs $\{(s_i, t_i), i = 1, ..., k\}$, a unicast secure network coding instance $\mathcal{I}_s$ is constructed as specified in Figure 14.1.*

Proof (of Theorem 14.1.1). "$\Rightarrow$". In this direction, we show that the asymptotic achievability of unit rate in $\mathcal{I}$ implies the asymptotic achievability of rate k in $\mathcal{I}_s$ under perfect secrecy, which in turn implies the asymptotic achievability of rate k in $\mathcal{I}_s$ under strong and weak secrecy.

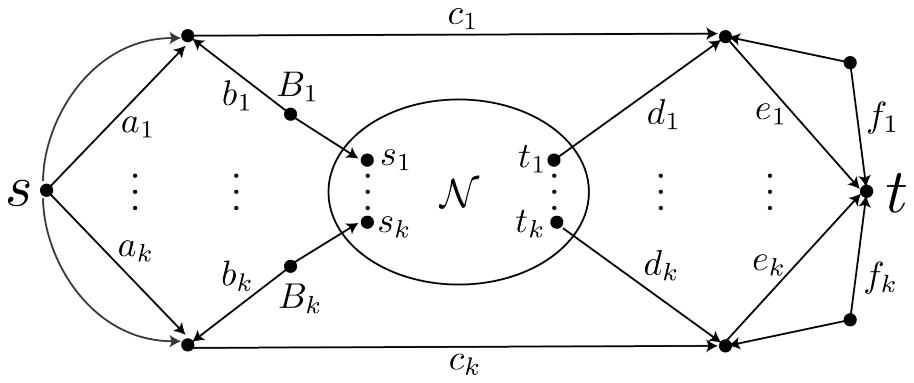


Figure 14.1: In the unicast secure network coding instance $\mathcal{I}_s$, the source s communicates with the terminal t. $\mathcal{N}$ is the network on which $\mathcal{I}$ is defined. All links outside $\mathcal{N}$ (i.e., links for which at least one end-point does not belong to $\mathcal{N}$) have unit capacity. The eavesdropper can wiretap on any single link in the network. Namely, $\mathcal{A}$ includes all sets of a single edge. Note that there are k parallel branches in total going from s to t but only the first and the k-th branches are drawn explicitly.

The scheme asymptotically achieving rate k is described in Figure 14.2. Specifically, the rate of the scheme is $(1-\epsilon)k$ if rate $1-\epsilon$ is achievable in $\mathcal{I}$. Let $\epsilon_i = \Pr\{\hat{V}_{B_i} \neq V_{B_i}\}$, then the probability of error in $\mathcal{I}_s$ is upper bounded by $\sum_{i=1}^{k} \epsilon_i$, which can be made arbitrarily small by choosing the ϵ_i's to be small enough. Note that the scheme achieves perfect security, since links in $\mathcal{N}$ are not downstream of s (and therefore the signals transmitted on them are independent of the message), and all other links are one-time padded by uniformly chosen keys.

"$\Leftarrow$". To prove this direction it suffices to show that asymptotic achievability of rate k in $\mathcal{I}_s$ under weak security implies asymptotic achievability of unit rate in $\mathcal{I}$, because asymptotic achievability of rate k in $\mathcal{I}_s$ under perfect or strong security implies asymptotic achievability of the same rate under weak security.

Suppose in $\mathcal{I}_s$ rate k is asymptotically achieved by a code with length n. Let M be the source input message, then $H(M) = kn$. We use the notation of Figure 14.1. Our objective is to lower bound the mutual information between signals $\boldsymbol{b}_i^n$ and $\boldsymbol{d}_i^n$, for $i = 1, ..., k$. Without loss of generality our analysis will focus on the case of

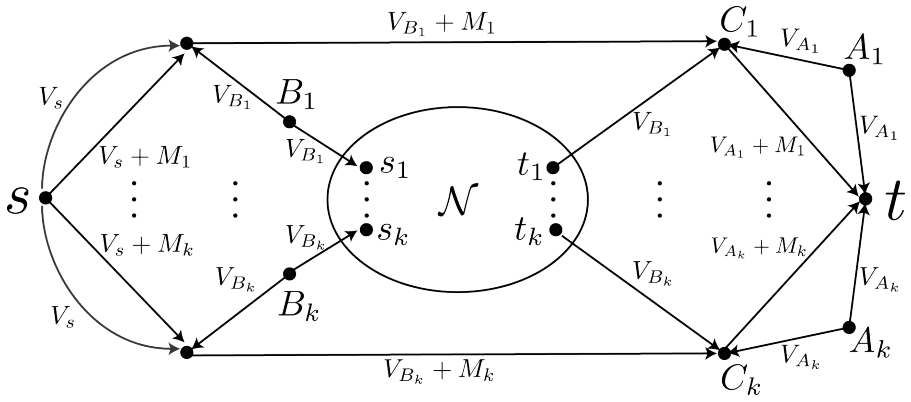


Figure 14.2: A scheme of length n that asymptotically achieves rate k in $\mathcal{I}_s$. Fix any $\epsilon > 0$, for node u, let V_u be a length-$(1-\epsilon)n$ random vector generated by node u. The V_u's are independently generated and are uniformly distributed over $\{0,1\}^{(1-\epsilon)n}$. The source message is a k-tuple of i.i.d. uniformly distributed length-$(1-\epsilon)n$ vectors, i.e., $M = (M_i,\ i = 1, ..., k)$, where the M_i's are i.i.d. uniformly distributed over $\{0,1\}^{(1-\epsilon)n}$. Since unit rate is asymptotically achievable in $\mathcal{I}$, node t_i obtains $\hat{V}_{B_i}$ such that $\hat{V}_{B_i} = V_{B_i}$ with high probability. $\hat{V}_{B_i}$ is then transmitted to node C_i for key cancellation.

$i = 1$. We start with,

$$\begin{aligned} H(M|\boldsymbol{c}_1^n, \boldsymbol{d}_1^n, \boldsymbol{f}_2^n, ..., \boldsymbol{f}_k^n) &\overset{(a)}{=} H(M, \boldsymbol{c}_1^n|\boldsymbol{d}_1^n, \boldsymbol{f}_2^n, ..., \boldsymbol{f}_k^n) - H(\boldsymbol{c}_1^n|\boldsymbol{d}_1^n, \boldsymbol{f}_2^n, ..., \boldsymbol{f}_k^n) \\ &\geq H(M|\boldsymbol{d}_1^n, \boldsymbol{f}_2^n, ..., \boldsymbol{f}_k^n) - H(\boldsymbol{c}_1^n|\boldsymbol{d}_1^n, \boldsymbol{f}_2^n, ..., \boldsymbol{f}_k^n) \\ &\overset{(b)}{\geq} kn - H(\boldsymbol{c}_1^n|\boldsymbol{d}_1^n, \boldsymbol{f}_2^n, ..., \boldsymbol{f}_k^n) \\ &\geq kn - n = (k-1)n, \end{aligned} \tag{14.1}$$

where (a) follows from the chain rule, and (b) follows from our construction which guarantees independence of M and $\{\boldsymbol{d}_1^n,\ \boldsymbol{f}_i^n,\ i = 1, ..., k\}$. On the other hand,

$$\begin{aligned} H(M|\boldsymbol{c}_1^n, \boldsymbol{d}_1^n, \boldsymbol{f}_2^n, ..., \boldsymbol{f}_k^n) &\leq H(M, \boldsymbol{e}_2^n, ..., \boldsymbol{e}_k^n|\boldsymbol{c}_1^n, \boldsymbol{d}_1^n, \boldsymbol{f}_2^n, ..., \boldsymbol{f}_k^n) \\ &\leq H(M|\boldsymbol{c}_1^n, \boldsymbol{d}_1^n, \boldsymbol{f}_2^n, ..., \boldsymbol{f}_k^n, \boldsymbol{e}_2^n, ..., \boldsymbol{e}_k^n) + \\ &\qquad H(\boldsymbol{e}_2^n, ..., \boldsymbol{e}_k^n|\boldsymbol{c}_1^n, \boldsymbol{d}_1^n, \boldsymbol{f}_2^n, ..., \boldsymbol{f}_k^n) \\ &\overset{(c)}{\leq} n\epsilon_n + H(\boldsymbol{e}_2^n, ..., \boldsymbol{e}_k^n|\boldsymbol{c}_1^n, \boldsymbol{d}_1^n, \boldsymbol{f}_2^n, ..., \boldsymbol{f}_k^n) \\ &\leq n\epsilon_n + (k-1)n, \end{aligned} \tag{14.2}$$

where $\epsilon_n \to 0$ as $n \to \infty$ and (c) is due to the cut-set $\{\boldsymbol{c}_1^n, \boldsymbol{d}_1^n, \boldsymbol{f}_2^n, ..., \boldsymbol{f}_k^n, \boldsymbol{e}_2^n, ..., \boldsymbol{e}_k^n\}$

from s to t and Fano's inequality. We lower bound the entropy of $\boldsymbol{c}_1^n$,

$$\begin{aligned} H(\boldsymbol{c}_1^n) &\geq H(\boldsymbol{c}_1^n|\boldsymbol{d}_1^n, \boldsymbol{f}_2^n, ..., \boldsymbol{f}_k^n) \\ &= H(M, \boldsymbol{c}_1^n|\boldsymbol{d}_1^n, \boldsymbol{f}_2^n, ..., \boldsymbol{f}_k^n) - H(M|\boldsymbol{c}_1^n, \boldsymbol{d}_1^n, \boldsymbol{f}_2^n, ..., \boldsymbol{f}_k^n) \\ &\geq H(M|\boldsymbol{d}_1^n, \boldsymbol{f}_2^n, ..., \boldsymbol{f}_k^n) - H(M|\boldsymbol{c}_1^n, \boldsymbol{d}_1^n, \boldsymbol{f}_2^n, ..., \boldsymbol{f}_k^n) \\ &= H(M) - H(M|\boldsymbol{c}_1^n, \boldsymbol{d}_1^n, \boldsymbol{f}_2^n, ..., \boldsymbol{f}_k^n) \\ &\overset{(d)}{\geq} kn - ((k-1)n + n\epsilon_n) = n - n\epsilon_n, \end{aligned} \tag{14.3}$$

where (d) follows from (14.2). We next lower bound the entropy of $\boldsymbol{d}_1^n$,

$$\begin{aligned} H(\boldsymbol{d}_1^n) &\geq H(\boldsymbol{d}_1^n|\boldsymbol{c}_1^n, \boldsymbol{f}_2^n, ..., \boldsymbol{f}_k^n) \\ &= H(M, \boldsymbol{d}_1^n|\boldsymbol{c}_1^n, \boldsymbol{f}_2^n, ..., \boldsymbol{f}_k^n) - H(M|\boldsymbol{c}_1^n, \boldsymbol{d}_1^n, \boldsymbol{f}_2^n, ..., \boldsymbol{f}_k^n) \\ &\geq H(M|\boldsymbol{c}_1^n, \boldsymbol{f}_2^n, ..., \boldsymbol{f}_k^n) - H(M|\boldsymbol{c}_1^n, \boldsymbol{d}_1^n, \boldsymbol{f}_2^n, ..., \boldsymbol{f}_k^n) \\ &\overset{(e)}{=} H(M|\boldsymbol{c}_1^n) - H(M|\boldsymbol{c}_1^n, \boldsymbol{d}_1^n, \boldsymbol{f}_2^n, ..., \boldsymbol{f}_k^n) \\ &\overset{(f)}{\geq} kn - n\delta_n - H(M|\boldsymbol{c}_1^n, \boldsymbol{d}_1^n, \boldsymbol{f}_2^n, ..., \boldsymbol{f}_k^n) \\ &\overset{(g)}{\geq} n - n\epsilon_n - n\delta_n, \end{aligned} \tag{14.4}$$

where $\delta_n \to 0$ as $n \to 0$, (e) follows from the independence between $\{M, \boldsymbol{c}_1^n\}$ and $\{\boldsymbol{f}_i^n,\ i = 1, ..., k\}$, (f) follows from the weak security requirement, and (g) follows from (14.2).

By the independence of $\{M, \boldsymbol{c}_1^n, \boldsymbol{d}_1^n\}$ and $\{\boldsymbol{f}_i^n,\ i = 1, ..., k\}$ we have

$$H(M|\boldsymbol{c}_1^n, \boldsymbol{d}_1^n, \boldsymbol{f}_2^n, ..., \boldsymbol{f}_k^n) = H(M|\boldsymbol{c}_1^n, \boldsymbol{d}_1^n). \tag{14.5}$$

By (14.1) and (14.2), the R.H.S of (14.5) is sandwiched by

$$(k-1)n \leq H(M|\boldsymbol{c}_1^n, \boldsymbol{d}_1^n) \leq n\epsilon_n + (k-1)n. \tag{14.6}$$

Now consider the joint entropy of $M,\ \boldsymbol{c}_1^n,\ \boldsymbol{d}_1^n$ and expand it in two ways

$$\begin{aligned} H(M, \boldsymbol{c}_1^n, \boldsymbol{d}_1^n) &= H(\boldsymbol{c}_1^n|M, \boldsymbol{d}_1^n) + H(M|\boldsymbol{d}_1^n) + H(\boldsymbol{d}_1^n) \\ &= H(M|\boldsymbol{c}_1^n, \boldsymbol{d}_1^n) + H(\boldsymbol{d}_1^n|\boldsymbol{c}_1^n) + H(\boldsymbol{c}_1^n) \\ &\leq (k+1)n + n\epsilon_n, \end{aligned}$$

where the last inequality holds because of (14.6) and $H(\boldsymbol{d}_1^n|\boldsymbol{c}_1^n) \leq n$, $H(\boldsymbol{c}_1^n) \leq n$. Therefore

$$\begin{aligned}
H(\boldsymbol{c}_1^n|M, \boldsymbol{d}_1^n) &= H(M, \boldsymbol{c}_1^n, \boldsymbol{d}_1^n) - H(M|\boldsymbol{d}_1^n) - H(\boldsymbol{d}_1^n) \\
&= H(M|\boldsymbol{c}_1^n, \boldsymbol{d}_1^n) + H(\boldsymbol{d}_1^n|\boldsymbol{c}_1^n) + H(\boldsymbol{c}_1^n) - H(M|\boldsymbol{d}_1^n) - H(\boldsymbol{d}_1^n) \\
&\overset{(h)}{\leq} (k+1)n + n\epsilon_n - H(M|\boldsymbol{d}_1^n) - H(\boldsymbol{d}_1^n) \\
&\overset{(i)}{=} (k+1)n + n\epsilon_n - kn - H(\boldsymbol{d}_1^n) \\
&\overset{(j)}{\leq} 2n\epsilon_n + n\delta_n, \qquad (14.7)
\end{aligned}$$

where (h) follows from (14.6) and $H(\boldsymbol{d}_1^n|\boldsymbol{c}_1^n) \leq n$, $H(\boldsymbol{c}_1^n) \leq n$; (i) follows from the independence between M and $\boldsymbol{d}_1^n$; (j) follows from (14.4). Now we have,

$$\begin{aligned}
H(\boldsymbol{b}_1^n|M, \boldsymbol{c}_1^n) &= H(M, \boldsymbol{b}_1^n, \boldsymbol{c}_1^n) - H(M|\boldsymbol{c}_1^n) - H(\boldsymbol{c}_1^n) \\
&= H(\boldsymbol{c}_1^n|M, \boldsymbol{b}_1^n) + H(M|\boldsymbol{b}_1^n) + H(\boldsymbol{b}_1^n) - H(M|\boldsymbol{c}_1^n) - H(\boldsymbol{c}_1^n) \\
&\leq (k+1)n + H(\boldsymbol{c}_1^n|M, \boldsymbol{b}_1^n) - H(M|\boldsymbol{c}_1^n) - H(\boldsymbol{c}_1^n) \\
&\overset{(k)}{=} (k+1)n + H(\boldsymbol{c}_1^n|M, \boldsymbol{b}_1^n, \boldsymbol{d}_1^n) - H(M|\boldsymbol{c}_1^n) - H(\boldsymbol{c}_1^n) \\
&\leq (k+1)n + H(\boldsymbol{c}_1^n|M, \boldsymbol{d}_1^n) - H(M|\boldsymbol{c}_1^n) - H(\boldsymbol{c}_1^n) \\
&\overset{(l)}{\leq} (k+1)n + 2n\epsilon_n + n\delta_n - H(M|\boldsymbol{c}_1^n) - H(\boldsymbol{c}_1^n) \\
&\overset{(m)}{\leq} n + 2n\epsilon_n + 2n\delta_n - H(\boldsymbol{c}_1^n) \\
&\overset{(n)}{\leq} 3n\epsilon_n + 2n\delta_n, \qquad (14.8)
\end{aligned}$$

where (k) follows from construction, i.e., $H(\boldsymbol{c}_1^n|M, \boldsymbol{b}_1^n) = H(\boldsymbol{c}_1^n|M, \boldsymbol{b}_1^n, \boldsymbol{d}_1^n)$; (l) follows from (14.7); (m) follows from the weak security requirement; (n) follows from (14.3). Therefore,

$$\begin{aligned}
H(\boldsymbol{b}_1^n|\boldsymbol{d}_1^n) &\overset{(o)}{=} H(\boldsymbol{b}_1^n|M, \boldsymbol{d}_1^n) \\
&\leq H(\boldsymbol{b}_1^n, \boldsymbol{c}_1^n|M, \boldsymbol{d}_1^n) \\
&= H(\boldsymbol{b}_1^n|\boldsymbol{c}_1^n, M, \boldsymbol{d}_1^n) + H(\boldsymbol{c}_1^n|M, \boldsymbol{d}_1^n) \\
&\leq H(\boldsymbol{b}_1^n|\boldsymbol{c}_1^n, M) + H(\boldsymbol{c}_1^n|M, \boldsymbol{d}_1^n) \\
&\overset{(p)}{\leq} 3n\epsilon_n + 2n\delta_n + 2n\epsilon_n + n\delta_n = 5n\epsilon_n + 3n\delta_n, \qquad (14.9)
\end{aligned}$$

where (o) follows as M is independent of $\{\boldsymbol{b}_1^n, \boldsymbol{d}_1^n\}$, and (p) follows from (14.8) and (14.7). (14.9) suggests that the signals $\boldsymbol{b}_1^n$ and $\boldsymbol{d}_1^n$ are strongly dependent. Next we

need to lower bound $H(\boldsymbol{b}_1^n)$.

$$\begin{aligned}
H(\boldsymbol{b}_1^n) &= H(M, \boldsymbol{b}_1^n, \boldsymbol{c}_1^n) - H(\boldsymbol{c}_1^n|M, \boldsymbol{b}_1^n) - H(M|\boldsymbol{b}_1^n) \\
&= H(\boldsymbol{b}_1^n|M, \boldsymbol{c}_1^n) + H(M|\boldsymbol{c}_1^n) + H(\boldsymbol{c}_1^n) - H(\boldsymbol{c}_1^n|M, \boldsymbol{b}_1^n) - H(M|\boldsymbol{b}_1^n) \\
&\geq H(M|\boldsymbol{c}_1^n) + H(\boldsymbol{c}_1^n) - H(\boldsymbol{c}_1^n|M, \boldsymbol{b}_1^n) - H(M|\boldsymbol{b}_1^n) \\
&\overset{(q)}{\geq} kn - n\delta_n + H(\boldsymbol{c}_1^n) - H(\boldsymbol{c}_1^n|M, \boldsymbol{b}_1^n) - H(M|\boldsymbol{b}_1^n) \\
&\overset{(r)}{\geq} (k+1)n - n\epsilon_n - n\delta_n - H(\boldsymbol{c}_1^n|M, \boldsymbol{b}_1^n) - H(M|\boldsymbol{b}_1^n) \\
&\overset{(s)}{\geq} n - n\epsilon_n - 2n\delta_n - H(\boldsymbol{c}_1^n|M, \boldsymbol{b}_1^n) \\
&\overset{(t)}{=} n - n\epsilon_n - 2n\delta_n - H(\boldsymbol{c}_1^n|M, \boldsymbol{b}_1^n, \boldsymbol{d}_1^n) \\
&\geq n - n\epsilon_n - 2n\delta_n - H(\boldsymbol{c}_1^n|M, \boldsymbol{d}_1^n) \\
&\overset{(u)}{\geq} n - 3n\epsilon_n - 3n\delta_n, \qquad (14.10)
\end{aligned}$$

where (q) follows from the weak security requirement; (r) follows from (14.3); (s) follows again from the weak security; (t) follows from construction; (u) follows from (14.7).

Finally, by (14.9) and (14.10),

$$I(\boldsymbol{b}_1^n; \boldsymbol{d}_1^n) = H(\boldsymbol{b}_1^n) - H(\boldsymbol{b}_1^n|\boldsymbol{d}_1^n) \geq n - 8n\epsilon_n - 6n\delta_n.$$

The above argument extends to all other paths naturally (by renumbering the notation accordingly), so

$$I(\boldsymbol{b}_i^n; \boldsymbol{d}_i^n) \geq n - 8n\epsilon_n - 6n\delta_n, \ \ \forall i = 1, ..., k. \qquad (14.11)$$

Lemma 14.1.1, stated below, shows that (14.11) implies the asymptotic achievability of unit rate in $\mathcal{I}$, which completes the proof of this direction. Intuitively, since the mutual information between $\boldsymbol{b}_i^n$ and $\boldsymbol{d}_i^n$ is asymptotically n, we can use a random coding argument similar to that used in the proof of the channel coding theorem to show the existence of a network code achieving unit rate asymptotically in $\mathcal{I}$. The reason that we do not use the standard point-to-point channel coding theorem is that there are multiple interacting source-terminal pairs in $\mathcal{I}$ and we need to make sure of the existence of a code that is good for all pairs.

Lemma 14.1.1. *For any $\epsilon > 0$, if there exists a network code of length n for $\mathcal{I}_s$ such that $I(\boldsymbol{b}_i^n; \boldsymbol{d}_i^n) > n(1-\epsilon)$, then unit rate is asymptotically achievable in $\mathcal{I}$.*

Proof (of Lemma 14.1.1). Our proof follows the same lines as the proof of the standard point-to-point channel coding theorem. The differences are that we need to translate the network code originally designed for $\mathcal{I}_s$ to a code for $\mathcal{I}$, as well as taking care of the multiple source-terminal pairs.

First consider problem $\mathcal{I}_s$. By hypothesis, let $\{\phi_e\}_{e\in\mathcal{E}}$ be the network code for $\mathcal{I}_s$. Denote the edge (B_i, s_i) by b'_i, and denote the signal transmitted on it by $\boldsymbol{b}'^n_i$. By construction we have the Markov chain $\boldsymbol{b}^n_i \to \boldsymbol{b}'^n_i \to \boldsymbol{d}^n_i$. Therefore it follows that $I(\boldsymbol{b}'^n_i; \boldsymbol{d}^n_i) \geq I(\boldsymbol{b}^n_i; \boldsymbol{d}^n_i) > n(1-\epsilon)$. Denote the distribution of the signal $\boldsymbol{b}'^n_i$ (generated by node B_i) by $p_i(x)$, $x \in \{0,1\}^n$, $i = 1, ..., k$. Note that the set of signals $\{\boldsymbol{b}'^n_i\}_{i=1}^k$ are independent because by construction they are generated by different nodes.

In problem $\mathcal{I}$ we simulate the same network code $\{\phi_e\}_{e\in\mathcal{E}}$ as in $\mathcal{I}_s$, and regard it as the inner code. We also generate the randomness according to the same distribution as in $\mathcal{I}_s$. Let x be a super symbol of n bits (i.e., same length as $\boldsymbol{b}'^n_i$), we independently generate k outer channel codes $\mathcal{C}_1, ..., \mathcal{C}_k$, each of 2^{mR} codewords and each codeword consists of m super symbols. Precisely, each of the 2^{mR} codewords of $\mathcal{C}_i$ is generated independently according to the distribution $p_i(x^m) = \prod_{j=1}^m p_i(x_j)$. Namely, each super symbol in the codeword is drawn independently according to the distribution of $\boldsymbol{b}'^n_i$. Notice that the overall length of the codeword is mn and the length of the network code for $\mathcal{I}_s$ is n. k messages $M_1, ..., M_k$ are chosen independently according to the uniform distribution: $\Pr\{M_i = w_i\} = 2^{-mR}$, $w_i = 1, ..., 2^{mR}$. Then the M_i-th codeword of $\mathcal{C}_i$, denoted by $X^m_i(M_i)$, is transmitted by invoking the inner network code for m times. Let Y^m_i be the output of terminal t_i using the inner code m times on network $\mathcal{N}$. The distribution of Y^m_i is statistically identical to that of $(\boldsymbol{d}^n_i)^m$ obtained by using the communication scheme for problem $\mathcal{I}_s$ m times. Indeed, in both cases $\mathcal{N}$ performs the same network code, generates randomness according to the same distribution, and receives the same input distribution by the codeword construction. In problem $\mathcal{I}$, t_i is a terminal node and it performs jointly typical decoding. Namely, the decoder at t_i declares that the $\hat{w}_i$-th codeword has been sent if: 1) $(X^m_i(\hat{w}_i), Y^m_i)$ is jointly typical, and 2) There is no other index $w' \neq \hat{w}_i$ such that $(X^m_i(w'), Y^m_i)$ is jointly typical. If no such $\hat{w}_i$ exists, an error is declared.

It remains to be shown that the probability of error $\Pr\{M_i \neq \hat{w}_i\}$ vanishes for all $i = 1, ..., k$ for an appropriate choice of R. As described above, Y^m_i in $\mathcal{I}$ is statistically the same as $(\boldsymbol{d}^n_i)^m$ in $\mathcal{I}_s$. Therefore it follows that $I(X_i; Y_i) = I(\boldsymbol{b}'_i; \boldsymbol{d}_i) > n(1-\epsilon)$.

As a result, we can apply the standard error analysis for jointly typical decoding and standard probabilistic argument (refer to, for example, [25, Chapter 7.7]) to show that for $R = n(1-\epsilon)$ and $\delta > 0$, there exists a large enough length m, such that $\Pr\{M_i \neq \hat{w}_i\} < \delta$, for $i = 1, ..., k$. By the union bound, $\Pr\{\cup_{i=1}^{k} M_i \neq \hat{w}_i\} < k\delta$, which can be made arbitrarily small by choosing a small enough δ.

All in all, combining the inner and outer code, we have shown the existence of a coding scheme of length mn that satisfies the multiple-unicast network coding problem $\mathcal{I}$ with arbitrarily small error probability. The number of codewords for each source-terminal pair is $2^{mR} = 2^{mn(1-\epsilon)}$, and therefore the rate of the scheme (over length mn) is $1-\epsilon$. This implies that unit rate is asymptotically achievable in $\mathcal{I}$. □

This completes the proof of Theorem 14.1.1. □

Theorem 14.1.1 can be easily adapted to the case of zero-error communication.

Corollary 14.1.1. *Given any multiple-unicast network coding instance $\mathcal{I}$ with source-terminal pairs $\{(s_i, t_i), i = 1, ..., k\}$, a corresponding unicast secure network coding instance $\mathcal{I}_s = (\mathcal{G}, s, t, \mathcal{A})$, in which $\mathcal{A}$ includes all sets of a single edge (i.e., all singletons), can be constructed according to Construction 14.1.1, such that unit rate is achievable with zero error in $\mathcal{I}$ if and only if rate k is achievable with zero error in $\mathcal{I}_s$ under perfect security.*

The proof of Corollary 14.1.1 follows the same line as the proof of Theorem 14.1.1, with the difference that all ϵ and δ become strictly 0. For example, (14.9) implies that $\boldsymbol{b}_1^n$ is a function of $\boldsymbol{d}_1^n$, and hence it can be perfectly decoded from $\boldsymbol{d}_1^n$.

We remark that our reduction has an operational aspect that from a code for $\mathcal{I}_s$ one can construct a code for $\mathcal{I}$. Indeed, using our reduction, to solve an instance of the multiple-unicast network coding problem, one may first reduce it to an instance of the unicast secure network coding problem, then solve the latter, and finally use this solution to obtain a solution to the original multiple-unicast problem.

Chan and Grant [83] previously show that determining the zero-error achievability of a rate in the multicast secure network coding problem with general setup (i.e., arbitrary link capacities, arbitrary $\mathcal{A}$, and arbitrary nodes may generate randomness) and with perfect security is at least as hard as determining the zero-error achievability of a rate tuple in multiple-multicast network coding. The result in this section

significantly strengthens the result in [83] by showing that the secure network coding problem under a much simpler setup (i.e., unicast, equal link capacities, uniform $\mathcal{A}$ with a single wiretap link) is still hard, under various definitions of achievability and security.

14.2 Reducing Unicast Secure Network Coding to Multiple-unicast

In the previous section we reduced an arbitrary instance of the multiple-unicast network coding problem into a particular instance of the unicast secure network coding problem with a very simple setup, in which at most one link can be eavesdropped. Conversely, given an arbitrary instance of the unicast secure network coding problem where at most one link can be eavesdropped, we can reduce it into a particular instance of the multiple-multicast network coding problem without security requirements (which can in turn be reduced into an equivalent multiple-unicast network coding problem [77]). We use a general construction that is first proposed in [96] for the purpose of lower bounding the capacity of a secure network coding instance by studying a corresponding multiple-multicast network coding instance. Here we simplify the construction to address the special case that at most one link can be eavesdropped, i.e., $\mathcal{A}$ comprises only singletons. Loosely speaking, we show that for this special case the multiple-multicast network coding instances not only lower bound but also upper bound the capacity of the secure network coding instances, hence giving a reduction.

Construction 14.2.1. *Given a unicast secure network coding instance $\mathcal{I}_s$ on a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with source s, terminal t, and a collection of wiretap sets $\mathcal{A}$ comprising only singletons, we construct a corresponding multiple-multicast network coding instance $\mathcal{I}$ on an augmented graph $\check{\mathcal{G}} = (\check{\mathcal{V}}, \check{\mathcal{E}})$. We define $\check{\mathcal{E}}$ and $\check{\mathcal{V}}$ from $\mathcal{E}$ and $\mathcal{V}$. In $\mathcal{E}$, denote by c_e the capacity of link e, and by $\mathcal{E}_{out}(i)$ the set of outgoing edges of node i.*

1. *For $i \in \mathcal{V}$, add i to $\check{\mathcal{V}}$. For $e \in \mathcal{E}$ such that $\{e\} \notin \mathcal{A}$ (in the remaining part of this subsection we will write $e \notin \mathcal{A}$ or $e \in \mathcal{A}$ instead, because the elements of $\mathcal{A}$ are singletons), add e to $\check{\mathcal{E}}$.*

2. *For $e = (i, j) \in \mathcal{E}$ such that $e \in \mathcal{A}$, create nodes u_e, v_e in $\check{\mathcal{V}}$; create edges (i, u_e), (u_e, j) and (u_e, v_e) in $\check{\mathcal{E}}$, all of capacity c_e.*

3. *Create a key aggregation node v_T in $\check{\mathcal{V}}$. For each node $i \in \mathcal{V}$, create a key*

source node $\bar{v}_i$ in $\check{\mathcal{V}}$. Create two links $(\bar{v}_i, v_T)$ and $(\bar{v}_i, i)$ in $\check{\mathcal{E}}$, both of capacity

$$\check{c}_i = \sum_{e \in \mathcal{E}_{out}(i)} c_e.$$

4. *Create a virtual source node v_s in $\check{\mathcal{V}}$. For all $e \in \mathcal{A}$, create links (v_s, s) and (v_s, v_e) in $\check{\mathcal{E}}$, both of capacity $\sum_{e' \in \mathcal{E}_{out}(s)} c_{e'}$.*

5. *For all $e \in \mathcal{A}$, create a link (v_T, v_e) in $\check{\mathcal{E}}$ of capacity*

$$\sum_{e' \in \mathcal{E}} c_{e'} - c_e.$$

In $\mathcal{I}$, nodes v_s and $\bar{v}_i$, $i \in \mathcal{V}$ are associated with independent messages. Node t demands the message of v_s. For all $e \in \mathcal{A}$, node v_e demands the messages of v_s and all $\bar{v}_i$. Note that in $\mathcal{I}$ there is no security requirement.

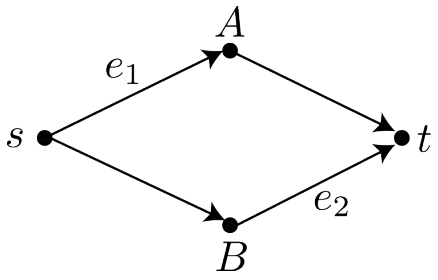
Refer to Figure 14.3 for an example of Construction 14.2.1.

Denote for short $\check{\boldsymbol{c}}_{\mathcal{V}} = (\check{c}_i, i \in \mathcal{V})$. Consider any unicast secure network coding instance $\mathcal{I}_s = (\mathcal{G}, s, t, \mathcal{A})$ such that $\mathcal{A}$ comprises only singletons. Let $\mathcal{I}$ be the multiple-multicast network coding instance obtained from $\mathcal{I}_s$ according to Construction 14.2.1. The next theorem reveals a connection between $\mathcal{I}_s$ and $\mathcal{I}$.

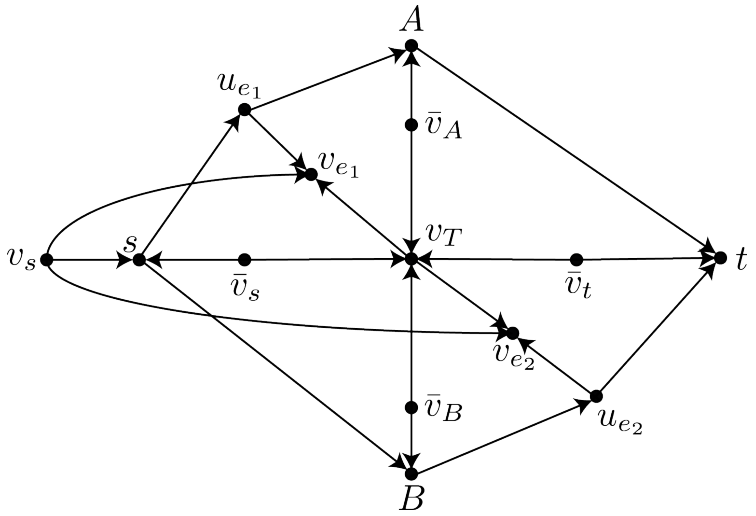
Theorem 14.2.1. *Rate R is asymptotically achievable in $\mathcal{I}_s$ subject to the weak or strong security requirement if and only if rate tuple $(R, \check{\boldsymbol{c}}_{\mathcal{V}})$ is asymptotically achievable in $\mathcal{I}$, where R is the rate of the message of v_s, and $\check{\boldsymbol{c}}_{\mathcal{V}}$ is the rate tuple of messages of $\bar{v}_i$, $i \in \mathcal{V}$.*

Proof. We first note that by [85], the capacity region of $\mathcal{I}_s$ subject to the weak security requirement is the same as the capacity region subject to the strong security requirement. Therefore, a rate is asymptotically achievable in $\mathcal{I}_s$ subject to weak security if and only if it is asymptotically achievable subject to strong security. In the proof we assume that weak security is imposed in $\mathcal{I}_s$, and the case of strong security follows directly from the equivalence.

"$\Leftarrow$". Assuming that rate tuple $(R, \check{\boldsymbol{c}}_{\mathcal{V}})$ is asymptotically achievable in $\mathcal{I}$, it is proved in [96] that rate R is asymptotically achievable in $\mathcal{I}_s$ under the weak security requirement. Here we give a simplified proof of this fact for completeness. By hypothesis, for any $\boldsymbol{\epsilon} > \mathbf{0}$ and $\delta > 0$, there exists a network code ϕ of length n that achieves rate tuple $(R, \check{\boldsymbol{c}}_{\mathcal{V}}) - \boldsymbol{\epsilon}$ in $\mathcal{I}$ with error probability δ. In problem $\mathcal{I}_s$, we



(a) A unicast secure network coding instance $\mathcal{I}_s$ where s is the source, t is the terminal and $\mathcal{A} = \{\{e_1\}, \{e_2\}\}$.



(b) The multiple-multicast network coding instance $\mathcal{I}$ obtained from $\mathcal{I}_s$ according to Construction 14.2.1. In this problem, t demands the message originated from v_s; v_{e_1} and v_{e_2} demand the messages originated from v_s, $\bar{v}_s$, $\bar{v}_A$, $\bar{v}_B$ and $\bar{v}_t$.

Figure 14.3: An example of Construction 14.2.1.

simulate the network code ϕ. Specifically, the message originally associated with v_s is replaced by the secret source message, and the messages originally associated with $\bar{v}_i, i \in \mathcal{V}$ are replaced by the independent random keys generated at node i. Let every edge $e = (i, j)$ in $\mathcal{E}$ simulate the encoding function of edge $e \in \check{\mathcal{E}}$ if $e \notin \mathcal{A}$ or the encoding function of (i, u_e) in $\check{\mathcal{E}}$ if $e \in \mathcal{A}$; then the terminal t, by simulating the decoding function, can decode the source message with error probability δ. Therefore decodability is not a problem.

It remains to show that simulating the code ϕ meets the weak security requirement. For an edge $\check{e} \in \check{\mathcal{E}}$, let $X_{\check{e}}$ be the signal transmitted on $\check{e}$ induced by ϕ. We denote by M the message associated with v_s, and by K_i the message associated with $\bar{v}_i$. For

any $e \in \mathcal{A}$, it follows that

$$\begin{aligned}
H(M|X_{(u_e,v_e)}) &\overset{(a)}{=} H(X_{(v_s,v_e)}|X_{(u_e,v_e)}) + H(M|X_{(v_s,v_e)}, X_{(u_e,v_e)}) - \\
&\qquad\qquad H(X_{(v_s,v_e)}|X_{(u_e,v_e)}, M) \\
&\geq H(X_{(v_s,v_e)}|X_{(u_e,v_e)}) - H(X_{(v_s,v_e)}|X_{(u_e,v_e)}, M) \\
&\overset{(b)}{=} H(X_{(v_s,v_e)}|X_{(u_e,v_e)}) \\
&\overset{(c)}{=} H(X_{(v_T,v_e)}, X_{(u_e,v_e)}, X_{(v_s,v_e)}) - H(X_{(u_e,v_e)}) - \\
&\qquad\qquad H(X_{(v_T,v_e)}|X_{(u_e,v_e)}, X_{(v_s,v_e)}) \\
&\overset{(d)}{\geq} H(X_{(v_T,v_e)}, X_{(u_e,v_e)}, X_{(v_s,v_e)}) - n\sum_{e'\in\mathcal{E}} c_{e'} \\
&\overset{(e)}{=} H(M, K_{\mathcal{V}}, X_{(v_T,v_e)}, X_{(u_e,v_e)}, X_{(v_s,v_e)}) - \\
&\qquad\qquad H(M, K_{\mathcal{V}}|X_{(v_T,v_e)}, X_{(u_e,v_e)}, X_{(v_s,v_e)}) - n\sum_{e'\in\mathcal{E}} c_{e'} \\
&\geq H(M, K_{\mathcal{V}}) - H(M, K_{\mathcal{V}}|X_{(v_T,v_e)}, X_{(u_e,v_e)}, X_{(v_s,v_e)}) - n\sum_{e'\in\mathcal{E}} c_{e'} \\
&\overset{(f)}{\geq} nR - n\epsilon - H(M, K_{\mathcal{V}}|X_{(v_T,v_e)}, X_{(u_e,v_e)}, X_{(v_s,v_e)}) \\
&\overset{(g)}{\geq} nR - n\epsilon - n\delta' \geq H(M) - n(\epsilon + \delta'). \qquad (14.12)
\end{aligned}$$

Here (a) follows from expanding $I(M; X_{(v_s,v_e)}|X_{(u_e,v_e)})$ in two ways; (b) follows from the construction that $X_{(v_s,v_e)}$ is a function of M; (c) follows from the chain rule; (d) follows from the fact that $H(X_{(u_e,v_e)}) \leq nc_{(u_e,v_e)} = nc_e$ and $H(X_{(v_T,v_e)}|X_{(u_e,v_e)}, X_{(v_s,v_e)}) \leq nc_{(v_T,v_e)} = n\sum_{e'\in\mathcal{E}} c_{e'} - nc_e$; (e) follows from the chain rule; (f) follows from $H(M, K_{\mathcal{V}}) \geq nR + n\sum_{e'\in\mathcal{E}} c_{e'} - n\epsilon$, where ϵ is the sum of the entries of $\boldsymbol{\epsilon}$; and (g) follows from Fano's inequality, where $\delta' \to 0$ as $\delta \to 0$. (14.12) implies that $I(M; X_{(u_e,v_e)}) \leq n(\epsilon + \delta')$. And because $X_{(u_e,v_e)}$ can be viewed as the observation of an adversary eavesdropping on edge e, the weak security requirement is met.

"$\Rightarrow$". Assuming that rate R is asymptotically achievable in $\mathcal{I}_s$ under the weak security requirement, we show that rate tuple $(R, \check{\boldsymbol{c}}_{\mathcal{A}})$ is asymptotically achievable in $\mathcal{I}$. For $e \in \mathcal{E}$, denote by X_e the signal transmitted on edge e. By hypothesis, for arbitrary $\epsilon_R > 0$, $\epsilon_S > 0$ and $\delta > 0$, there exists a network code ϕ with length n that achieves rate $R - \epsilon_R$ weakly securely in $\mathcal{I}_s$ with error probability δ, such that $I(M; X_e) \leq n\epsilon_S$, for all $e \in \mathcal{A}$. For arbitrary $\epsilon_E > 0$, without loss of generality we assume that $H(X_e) \geq n(c_e - \epsilon_E)$, for all $e \in \mathcal{E}$. This is because if $H(X_e) < nc_e$, i.e., if X_e is not "almost" uniform, then we can repeat ϕ for m times and perform

a source code of length m (over the supersymbol alphabet $\{0,1\}^n$) on edge e to compress X_e by encoding only the typical sequences [25, Section 3.2], and use the spare capacity of the edge to transmit random bits. Meanwhile, source s and terminal t perform an outer channel code of length m (also over alphabet $\{0,1\}^n$) to keep the error probability small. For sufficiently large m, the overall concatenated code asymptotically achieves the same rate weakly securely as ϕ does, and such that $H(X_e)/mn$, as desired, is arbitrarily close to c_e due to the asymptotic equipartition property. Note that in general a node $i \in \mathcal{V}$ will generate an independent random variable K_i as input to the encoding functions. The rate of K_i is upper bounded by $\check{c}_i$, which is the sum of the capacities of all outgoing edges of i.

We now turn to the problem $\mathcal{I}$. We construct a network code ϕ' of a slightly longer length $n' = (1+\epsilon_L)n$. In the first n channel uses, ϕ' simulates the operation of ϕ. Specifically, node v_s generates a random variable M uniformly distributed over $[2^{n(R-\epsilon_R)}]$, and transmits it to s via edge (v_s, s). For $i \in \mathcal{V}$, node $\bar{v}_i$ generates a random variable K_i uniformly distributed over $[2^{n\check{c}_i}]$, and transmits it to i via edge $(\bar{v}_i, i)$. The rate of M (over code length n') is $\frac{1}{1+\epsilon_L}(R-\epsilon_R)$ and the rate of K_i is $\frac{1}{1+\epsilon_L}\check{c}_i$. The rates of M and $K_{\mathcal{V}}$ can be made arbitrarily close to R and $\check{\mathbf{c}}_{\mathcal{V}}$ by choosing sufficiently small ϵ_R and ϵ_L.

To simulate ϕ, ϕ' performs the same encoding function on edge $e \in \mathcal{E}$ (if e remains in $\check{\mathcal{E}}$) as ϕ did in $\mathcal{I}_s$. Otherwise if e is replaced by (i, u_e) and (u_e, j), then ϕ' performs the same encoding function on $(i, u_e) \in \check{\mathcal{E}}$ as ϕ did on edge $e \in \mathcal{E}$. The induced signal $X_{(i,u_e)}$ is then relayed to edges (u_e, j) and (u_e, v_e). In the extra ϵ_L channel uses, these edges (from $\mathcal{E}$) keep silent (or simply transmit dummy zeros). The terminal node t, by simulating the decoding function, is able to decode M correctly with probability of error δ.

We next show that $(K_i, i \in \mathcal{V})$ and M can be decoded at v_e, for all $e \in \mathcal{A}$. Note that v_e has three incoming edges, i.e., (u_e, v_e), (v_s, v_e) and (v_T, v_e). The signal $X_{(u_e,v_e)}$ transmitted on (u_e, v_e), as described in the previous paragraph, is the same as the signal X_e on edge e in the secure network coding problem $\mathcal{I}_s$. Let edge (v_s, v_e) transmit M. Now consider a distributed source coding problem with three correlated sources $(K_i, i \in \mathcal{V})$, X_e and M. Note the X_e and M are available at v_e by construction, and the question is whether v_e can decode $K_{\mathcal{V}}$ from X_e, M and the

signal received from edge (v_T, v_e). It follows that,

$$\begin{aligned}
H(K_{\mathcal{V}}|X_e, M) &= H(X_e, K_{\mathcal{V}}|M) - H(X_e|M) \\
&= H(X_e|K_{\mathcal{V}}, M) + H(K_{\mathcal{V}}|M) - H(X_e|M) \\
&\overset{(a)}{=} H(K_{\mathcal{V}}|M) - H(X_e|M) \\
&\overset{(b)}{=} H(K_{\mathcal{V}}) - H(X_e|M) \\
&\leq n \sum_{e' \in \mathcal{E}} c_{e'} - H(X_e|M) \\
&= n \sum_{e' \in \mathcal{E}} c_{e'} - (H(X_e) - I(M; X_e)) \\
&\overset{(c)}{\leq} n(\sum_{e' \in \mathcal{E}} c_{e'} - c_e) + n(\epsilon_S + \epsilon_E) \\
&\overset{(d)}{<} n(\sum_{e' \in \mathcal{E}} c_{e'} - c_e) + n\epsilon_L(\sum_{e' \in \mathcal{E}} c_{e'} - c_e) \\
&= n'(\sum_{e' \in \mathcal{E}} c_{e'} - c_e), \qquad (14.13)
\end{aligned}$$

where (a) follows because X_e is a function of $K_{\mathcal{V}}$ and M; (b) follows from the fact that $K_{\mathcal{V}}$ is independent of M; (c) follows from the weak security requirement and that $H(X_e) \geq n(c_e - \epsilon_E)$; and (d) follows by choosing a sufficiently small ϵ_S and ϵ_E such that $\epsilon_S + \epsilon_E < \epsilon_L(\sum_{e' \in \mathcal{E}} c_{e'} - c_e)$, for all $e \in \mathcal{A}$. Note that $\sum_{e' \in \mathcal{E}} c_{e'} - c_e$ is the capacity of edge (v_T, v_e), and so by (14.13) and the Slepian-Wolf Theorem of distributed source coding, the rate tuple of the correlated sources are in the achievable region. Therefore by concatenating an (outer) Slepian-Wolf code with the (inner) network code ϕ', for all $e \in \mathcal{A}$, v_e is able to decode $(M, K_{\mathcal{V}})$ with vanishing probability of error. This completes the proof. □

Chapter 15

REDUCING MULTIPLE-UNICAST TO UNICAST NETWORK ERROR CORRECTION

In this chapter we reduce instances of the multiple-unicast network coding problem to instances of the unicast network error correction problem. We start with the zero-error case.

15.1 Zero-error Achievability

The following theorem reduces the problem of determining the zero-error achievability of a rate in a general multiple-unicast network coding instance to the problem of determining the zero-error achievability of a rate in a particular unicast network error correction instance that has a very simple setup. Recall from the remark before Theorem 14.1.1 that there is no loss of generality in addressing the achievability of a rate instead of a rate tuple in the multiple-unicast network coding problem.

Construction 15.1.1. *Given any multiple-unicast network coding instance $\mathcal{I}$ on a network $\mathcal{N}$ with source-terminal pairs $\{(s_i, t_i), i = 1, ..., k\}$, a unicast network error correction problem $\mathcal{I}_c$ is constructed as specified in Figure 15.1.*

Theorem 15.1.1. *Given any multiple-unicast network coding instance $\mathcal{I}$ with source-terminal pairs $\{(s_i, t_i), i = 1, ..., k\}$, a corresponding unicast network error correction instance $\mathcal{I}_c = (\mathcal{G}, s, t, \mathcal{B})$, in which $\mathcal{B}$ includes sets with at most one edge, can be constructed according to Construction 15.1.1, such that unit rate is achievable with zero-error in $\mathcal{I}$ if and only if rate k is achievable with zero-error in $\mathcal{I}_c$.*

Proof. "$\Rightarrow$". We show that the zero-error achievability of unit rate in $\mathcal{I}$ implies the zero-error achievability of rate k in $\mathcal{I}_c$. A constructive scheme is shown in Figure 15.2. In $\mathcal{I}_c$, the source lets $M = (M_1, ..., M_k)$, where the M_i's are i.i.d. uniformly distributed over $[2^n]$. In $\mathcal{N}$, we simulate the network code for $\mathcal{I}$ that achieves unit rate with zero error. Outside $\mathcal{N}$, let the network code be $a_i(M) = x_i(M) = y_i(M) = z_i(M) = z'_i(M) = M_i$, $i = 1, ..., k$. Note that the signals on edges x_i, y_i and z'_i form a repetition code and node B_i, by performing majority decoding, can correct any single error to obtain M_i. Hence the scheme ensures that $b_i(M) = M_i$ under all possible error patterns, and so rate k is achievable with zero error in $\mathcal{I}_c$.

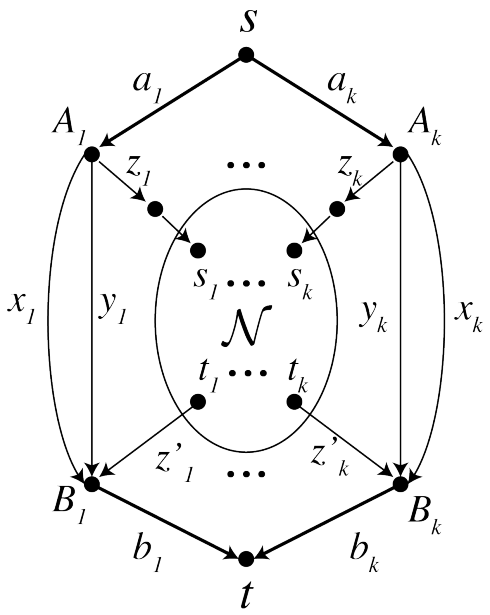


Figure 15.1: In the unicast network error correction instance $\mathcal{I}_c$, s is the source and t is the terminal. $\mathcal{N}$ is the network on which $\mathcal{I}$ is defined. All edges outside $\mathcal{N}$ (i.e., edges for which at least one of its end-points does not belong to $\mathcal{N}$) have unit capacity. There is at most one error in this network, and this error can occur at any edge except $\{a_i, b_i, 1 \le i \le k\}$. Namely, $\mathcal{B}$ includes all singleton sets of a single edge in the network except $\{a_i\}$ and $\{b_i\}$, $i = 1, ..., k$ (the thickened edges). Note that there are k parallel branches in total going from s to t but only the first and the k-th branches are drawn explicitly.

"$\Leftarrow$". We show that the zero-error achievability of rate k in $\mathcal{I}_c$ implies the zero-error achievability of unit zero-error rate in $\mathcal{I}$.

Suppose rate k is achieved with zero-error in $\mathcal{I}_c$ by a network code with length n, and denote the source message by M, which is uniformly distributed over $[2^{nk}]$. Recall from Section 13.4.3 that $\boldsymbol{r} = (r_e)_{e\in\mathcal{E}}$ is the tuple of additive error signals called an error pattern, and $\mathcal{R}_\mathcal{B}$ is the set of all possible error patterns, i.e., $\mathcal{R}_\mathcal{B} = \{\boldsymbol{r} :$ non-zero entries in $\boldsymbol{r}$ correspond to B-errors, $B \in \mathcal{B}\}$. For any edge $e \in \mathcal{E}$, we denote by $e(m, \boldsymbol{r}) : [2^{nk}] \times \mathcal{R}_\mathcal{B} \to [2^n]$ the signal received on edge e when the source message equals m and the error pattern $\boldsymbol{r}$ occurs in the network.

Let $\boldsymbol{b}(m, \boldsymbol{r}) = (b_1(m, \boldsymbol{r}), ..., b_k(m, \boldsymbol{r}))$, then because the edges $b_1, ..., b_k$ form a cut-set from s to t, $\boldsymbol{b}(m, \boldsymbol{r})$ must be injective with respect to m due to the zero error decodability constraint. Formally, for two different messages $m_1 \neq m_2$, it follows from the zero error decodability constraint that $\boldsymbol{b}(m_1, \boldsymbol{r}_1) \neq \boldsymbol{b}(m_2, \boldsymbol{r}_2)$,

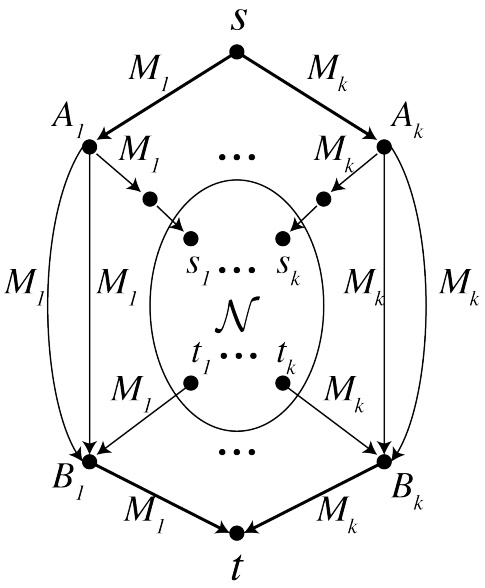


Figure 15.2: A scheme to achieve zero-error rate k in $\mathcal{I}_c$ given that unit rate is achievable with zero error in $\mathcal{I}$. $M = (M_1, ..., M_k)$ and node B_i performs majority decoding.

$\forall \boldsymbol{r}_1, \boldsymbol{r}_2 \in \mathcal{R}_{\mathcal{B}}$. Note that the codomain of $\boldsymbol{b}$ is $[2^n]^k$, which has the same size as the set of messages $[2^{nk}]$. Therefore denote by $\boldsymbol{b}(m) = \boldsymbol{b}(m, \mathbf{0})$, then $\boldsymbol{b}(m)$ is a bijective function. This implies that $\boldsymbol{b}(m, \boldsymbol{r}) = \boldsymbol{b}(m)$, $\forall \boldsymbol{r} \in \mathcal{R}_{\mathcal{B}}$, because otherwise if there exist $m_1, \boldsymbol{r}_1$ such that $\boldsymbol{b}(m_1, \boldsymbol{r}_1) \neq \boldsymbol{b}(m_1)$, then there is a $m_2 \neq m_1$ such that $\boldsymbol{b}(m_1, \boldsymbol{r}_1) = \boldsymbol{b}(m_2)$, violating the decodability requirement. Similarly, let $\boldsymbol{a}(m, \boldsymbol{r}) = (a_1(m, \boldsymbol{r}), ..., a_k(m, \boldsymbol{r}))$ and $\boldsymbol{a}(m) = \boldsymbol{a}(m, \mathbf{0})$, then $\boldsymbol{a}(m)$ is a bijective function and $\boldsymbol{a}(m, \boldsymbol{r}) = \boldsymbol{a}(m)$, $\forall \boldsymbol{r} \in \mathcal{R}_{\mathcal{B}}$.

For any $e \in \mathcal{E}$, denote $e(m) = e(m, \mathbf{0})$. We make the following claim:
Claim: For $i = 1, ..., k$ and any two messages $m_1, m_2 \in [2^{nk}]$ such that $a_i(m_1) \neq a_i(m_2)$, it follows that $x_i(m_1) \neq x_i(m_2)$, $y_i(m_1) \neq y_i(m_2)$ and $z_i(m_1) \neq z_i(m_2)$.
To prove the claim, suppose for contradiction that there exist m_1, m_2 such that $a_i(m_1) \neq a_i(m_2)$ and that the claim is not true, i.e., $x_i(m_1) = x_i(m_2)$ or $y_i(m_1) = y_i(m_2)$ or $z_i(m_1) = z_i(m_2)$. First consider the case that $x_i(m_1) = x_i(m_2)$. Because of the one-to-one correspondence between m and $\boldsymbol{a}$, there exists a message $m_3 \neq m_1$ and such that $\boldsymbol{a}(m_3) = (a_1(m_1), ..., a_{i-1}(m_1), a_i(m_2), a_{i+1}(m_1), ..., a_k(m_1))$. Then $x_i(m_1) = x_i(m_3)$ because by construction $x_i(m_3) = x_i(m_2)$, and by hypothesis $x_i(m_1) = x_i(m_2)$. Consider the following two scenarios. In the first scenario, m_1 is transmitted, and an error turns $y_i(m_1)$ into $y_i(m_3)$; in the second scenario,

m_3 is transmitted, and an error turns $z_i(m_3)$ into $z_i(m_1)$. Then the cut-set signals $a_1, ..., a_{i-1}, x_i, y_i, z_i, a_{i+1}, ..., a_k$ are exactly the same in both scenarios, and so it is impossible for t to distinguish m_1 from m_3, a contradiction to the zero error decodability constraint. Therefore $x_i(m_1) \neq x_i(m_2)$. With a similar argument it follows that $y_i(m_1) \neq y_i(m_2)$ and $z_i(m_1) \neq z_i(m_2)$, and the claim is proved.

The claim above suggests that (the signals on) x_i, y_i and z_i, as functions of (the signals on) a_i, are injective. They are also surjective functions because the domain and codomain are both $[2^n]$. Hence there are one-to-one correspondences between a_i, x_i, y_i and z_i.

Next we show that for any two messages m_1, m_2, if $b_i(m_1) \neq b_i(m_2)$, then $z'_i(m_1) \neq z'_i(m_2)$. Suppose for contradiction that there exists $m_1 \neq m_2$ such that $b_i(m_1) \neq b_i(m_2)$ and $z'_i(m_1) = z'_i(m_2)$. Then if m_1 is transmitted and an error $\boldsymbol{r}_1$ turns $x_i(m_1)$ into $x_i(m_2)$, the node B_i will receive the same signals as in the case that m_2 is transmitted and an error $\boldsymbol{r_2}$ turns $y_i(m_2)$ into $y_i(m_1)$. Therefore $b_i(m_1, \boldsymbol{r}_1) = b_i(m_2, \boldsymbol{r}_2)$. But, as shown above, because $b_i(m_1, \boldsymbol{r}_1) = b_i(m_1)$ and $b_i(m_2, \boldsymbol{r}_2) = b_i(m_2)$, it follows that $b_i(m_1) = b_i(m_2)$, a contradiction. This claim suggests that if $z'_i(m_1) = z'_i(m_2)$ then $b_i(m_1) = b_i(m_2)$ and therefore b_i is a function of z'_i. This function is surjective because b_i takes all 2^n possible values. Then since the domain and the codomain are both $[2^n]$, b_i must be a bijective function of z'_i. With the same argument it follows that b_i is also a bijective function of x_i.

Hence z_i is a bijection of a_i, a_i is a bijection of x_i, x_i is a bijection of b_i, and b_i is a bijection of z'_i. Therefore for all $1 \leq i \leq k$, z_i is a bijection of z'_i, and therefore unit rate is achievable with zero error in $\mathcal{I}$. □

15.2 Achievability Allowing Vanishing Error

In this section we show that Construction 15.1.1 gives a reduction from multiple-unicast network coding to unicast network error correction not only in terms of zero-error achievability, but also in terms of achievability that allows vanishing error. Recall from Section 13.4 that a rate is achievable if there exists a code achieving the rate with a vanishing error probability.

Theorem 15.2.1. *Given any multiple-unicast network coding instance $\mathcal{I}$ with source-terminal pairs $\{(s_i, t_i), i = 1, ..., k\}$, a corresponding unicast network error correction instance $\mathcal{I}_c = (\mathcal{G}, s, t, \mathcal{B})$, in which $\mathcal{B}$ includes sets with at most a single edge, can be constructed according to Construction 15.1.1, such that unit rate is achievable in $\mathcal{I}$ if and only if rate k is achievable in $\mathcal{I}_c$.*

We first prove the forward direction of the theorem, which is simple and is similar to the proof of the zero-error case.

Proof ("⇒" part of Theorem 15.2.1). We show that if unit rate is achievable in $\mathcal{I}$, then rate k is achievable in $\mathcal{I}_c$. Again we use the constructive scheme in Figure 15.2. In $\mathcal{I}_c$, the source lets $M = (M_1, ..., M_k)$, where the M_i's are i.i.d. uniformly distributed over $[2^n]$. In $\mathcal{N}$, we simulate the network code for $\mathcal{I}$ that achieves unit rate. Outside $\mathcal{N}$, let the network code be $a_i(M) = x_i(M) = y_i(M) = z_i(M) = M_i$, $i = 1, ..., k$. Consider the M_i's as the source messages of the simulated $\mathcal{I}$, and denote by $\hat{M}_i$, $i = 1, ..., k$ the outputs of the decoders of $\mathcal{I}$. Let $z'_i(M) = \hat{M}_i$, $i = 1, ..., k$, and let node B_i, $i = 1, ..., k$ performs majority decoding. The terminal t will not decode an error as long as the multiple-unicast instance $\mathcal{I}$ does not commit an error, i.e., $\hat{M}_i = M_i$. The error probability of $\mathcal{I}$ is negligible, which implies the achievability of rate k in $\mathcal{I}_c$. □

In the remainder of this section we prove the other direction of Theorem 15.2.1, i.e., that the achievability of rate k in $\mathcal{I}_c$ implies the achievability of unit rate in $\mathcal{I}$.

The main idea of the proof is as follows. In $\mathcal{I}$ the network will simulate the given network code for $\mathcal{I}_c$, and by doing so the terminal t_i will obtain the signal on z'_i in $\mathcal{I}_c$. The main task therefore is to estimate the signal on z_i, which terminal t_i demands, based on the observed signal on z'_i. This task is accomplished by following two steps. In the first step, we construct an explicit decoding function that, taking the signal on z'_i as input, outputs the "most likely" signal to be transmitted on b_i under the given network code for $\mathcal{I}_c$. We then prove (by combinatorial arguments) that the probability of error of this decoding function is small if the probability of error of the given network code is small. In the second step, we construct another explicit decoding function that takes the signal on b_i as input and outputs the "most likely" signal transmitted on a_i. Again we can prove that if the given network code for $\mathcal{I}_c$ has a small probability of error then so does this decoding function. Finally, terminal t_i, by concatenating the two decoding functions, is able to estimate the signal on a_i, and therefore the signal on z_i correctly with high probability. We now proceed with the formal proof.

Suppose in $\mathcal{I}_c$ a rate of k is achieved by a network code $\mathcal{C} = \{\phi_e, \phi_t\}_{e \in \mathcal{E}}$ with length n, and with a probability of error ϵ. Recall that M is the source message uniformly distributed over $\mathcal{M} = [2^{kn}]$, and $\hat{M}$ is the output of the decoder at the terminal. Let $\mathcal{M}^{\text{good}} = \{m \in [2^{kn}] : \mathcal{C} \text{ satisfies } \mathcal{I}_c \text{ under transmission } m\}$ be the

subset of messages that can be decoded correctly under any error pattern $\boldsymbol{r} \in \mathcal{R}_{\mathcal{B}}$. Denote by $\mathcal{M}^{\text{bad}} = \mathcal{M} \backslash \mathcal{M}^{\text{good}}$, then for any $m \in \mathcal{M}^{\text{bad}}$, there exists an error pattern $\boldsymbol{r} \in \mathcal{R}_{\mathcal{B}}$ such that the decoded value $\hat{M}$ differs from M when $M = m$ and $\boldsymbol{r}$ occurs, i.e., a decoding error occurs. Because $\mathcal{C}$ satisfies $\mathcal{I}_c$ with error probability ϵ (recall from Section 13.4.3 that the probability of error is taken over the distribution on the message M), it follows that $|\mathcal{M}^{\text{bad}}| \leq 2^{kn}\epsilon$ and thus $|\mathcal{M}^{\text{good}}| \geq (1-\epsilon) \cdot 2^{kn}$.

We introduce some notation needed in the proof. In problem $\mathcal{I}_c$, under the network code $\mathcal{C}$, for $i = 1, ..., k$, let $x_i(m, \boldsymbol{r}) : \mathcal{M} \times \mathcal{R}_{\mathcal{B}} \to [2^n]$ be the signal received from channel x_i when $M = m$ and the error pattern $\boldsymbol{r}$ happens. Let $\boldsymbol{r} = \boldsymbol{0}$ denote the case that no error has occurred in the network. Let $x_i(m) = x_i(m, \boldsymbol{0})$, $\boldsymbol{x}(m, \boldsymbol{r}) = (x_1(m, \boldsymbol{r}), ..., x_k(m, \boldsymbol{r}))$ and $\boldsymbol{x}(m) = (x_1(m), ..., x_k(m))$. We define $a_i, b_i, y_i, z_i, z_i', \boldsymbol{a}, \boldsymbol{b}, \boldsymbol{y}, \boldsymbol{z}, \boldsymbol{z}'$ for problem $\mathcal{I}_c$ in a similar way. Note that all of them are functions of m and $\boldsymbol{r}$. We use the hat notation to represent a specific value in the range of a function, e.g., $\hat{a}_i$ represents a specific output of a_i. In other words, $\hat{a}_i$ is a specific n-bit signal.

Notice that the set of edges $a_1, ..., a_k$ forms a cut-set from s to t, and so does the set of edges $b_1, ..., b_k$. Therefore for any $m_1, m_2 \in \mathcal{M}^{\text{good}}$, $m_1 \neq m_2$, it follows from the decodability constraint that $\boldsymbol{a}(m_1) \neq \boldsymbol{a}(m_2)$ and $\boldsymbol{b}(m_1) \neq \boldsymbol{b}(m_2)$. Setting $\mathcal{B}^{\text{good}} = \{\boldsymbol{b}(m) : m \in \mathcal{M}^{\text{good}}\}$, it then follows from $|\mathcal{M}^{\text{good}}| \geq (1-\epsilon) \cdot 2^{kn}$ that $|\mathcal{B}^{\text{good}}| \geq (1-\epsilon) \cdot 2^{kn}$. Setting $\mathcal{B}^{\text{err}} = [2^n]^k \backslash \mathcal{B}^{\text{good}}$, it follows that $|\mathcal{B}^{\text{err}}| \leq 2^{kn}\epsilon$. Similarly, we define $\mathcal{A}^{\text{good}} = \{\boldsymbol{a}(m) : m \in \mathcal{M}^{\text{good}}\}$, and $\mathcal{A}^{\text{err}} = [2^n]^k \backslash \mathcal{A}^{\text{good}}$, then $|\mathcal{A}^{\text{good}}| \geq (1-\epsilon) \cdot 2^{kn}$, $|\mathcal{A}^{\text{err}}| \leq 2^{kn}\epsilon$.

Let $\mathcal{M}(\hat{z}_i', \hat{b}_i) = \{m \in \mathcal{M}^{\text{good}} : z_i'(m) = \hat{z}_i', b_i(m) = \hat{b}_i\}$. Intuitively, $\mathcal{M}(\hat{z}_i', \hat{b}_i)$ represent the set of outcomes (in $\mathcal{I}_c$) that will result in signal $\hat{b}_i$ being transmitted on edge b_i and signal $\hat{z}_i'$ being transmitted on edge z_i'. We define a function $\psi_i : [2^n] \to [2^n]$ as:

$$\psi_i(\hat{z}_i') = \arg\max_{\hat{b}_i} |\mathcal{M}(\hat{z}_i', \hat{b}_i)| \triangleq \hat{b}_{i,\hat{z}_i'}. \tag{15.1}$$

Function ψ_i will be useful later, when we design the network codes in $\mathcal{I}$. Intuitively, in the absence of adversarial errors, ψ_i estimates the signal transmitted on edge b_i given that the signal transmitted on edge z_i' is $\hat{z}_i'$. In the following we analyze how often ψ_i will make a mistake. Define $\mathcal{M}_i^{\psi} = \{m \in \mathcal{M}^{\text{good}} : \psi_i(z_i'(m)) \neq b_i(m)\}$. Notice that $\mathcal{M}_i^{\psi}$ is the set of messages for which, when transmitted by the source, ψ_i will make a mistake in guessing the signal transmitted on b_i. Lemmas 15.2.1 and 15.2.2 analyze the size of $\mathcal{M}_i^{\psi}$.

Lemma 15.2.1. *Let $\mathcal{M}(\hat{z}_i') = \{m \in \mathcal{M}^{good} : z_i'(m) = \hat{z}_i'\}$, then for any $m_1, m_2 \in \mathcal{M}(\hat{z}_i')$ such that $b_i(m_1) \neq b_i(m_2)$, there exists an element of $\mathcal{B}^{err}$ that will be decoded by terminal t to either m_1 or m_2.*

Proof. Consider any $m_1, m_2 \in \mathcal{M}(\hat{z}_i')$ such that $b_i(m_1) \neq b_i(m_2)$. Let $\boldsymbol{r}_1$ be the error pattern that changes the signal on x_i to be $x_i(m_2)$, and let $\boldsymbol{r}_2$ be the error pattern that changes the signal on y_i to be $y_i(m_1)$. Then if m_1 is transmitted by the source and $\boldsymbol{r}_1$ happens, node B_i will receive the same inputs $(x_i(m_2), y_i(m_1), z_i'(m_1) = z_i'(m_2))$ as in the situation that m_2 is transmitted and $\boldsymbol{r}_2$ happens. Therefore $b_i(m_1, \boldsymbol{r}_1) = b_i(m_2, \boldsymbol{r}_2)$, and so either $b_i(m_1, \boldsymbol{r}_1) \neq b_i(m_1)$ or $b_i(m_2, \boldsymbol{r}_2) \neq b_i(m_2)$ because by hypothesis $b_i(m_1) \neq b_i(m_2)$. Consider the first case that $b_i(m_1, \boldsymbol{r}_1) \neq b_i(m_1)$, then the tuple of signals $(b_1(m_1, \boldsymbol{r}_1), ..., b_k(m_1, \boldsymbol{r}_1)) = (b_1(m_1), ..., b_i(m_1, \boldsymbol{r}_1), ..., b_k(m_1))$ will be decoded by the terminal to message m_1 because of the fact that $m_1 \in \mathcal{M}^{\text{good}}$ which is correctly decodable under any error pattern $\boldsymbol{r} \in \mathcal{R}_{\mathcal{B}}$. Therefore this tuple of signals is an element of $\mathcal{B}^{\text{err}}$ since it is not equal to $\boldsymbol{b}(m_1) = (b_1(m_1), ..., b_k(m_1))$ and it is not equal to $\boldsymbol{b}(m)$, for any $m \neq m_1$, $m \in \mathcal{M}^{\text{good}}$, because otherwise it will be decoded by the terminal to m. Similarly in the latter case that $b_i(m_2, \boldsymbol{r}_2) \neq b_i(m_2)$, then $(b_1(m_2, \boldsymbol{r}_2), ..., b_k(m_2, \boldsymbol{r}_2)) = (b_1(m_2), ..., b_i(m_2, \boldsymbol{r}_2), ..., b_k(m_2))$ is an element of $\mathcal{B}^{\text{err}}$ and will be decoded by the terminal to m_2. Therefore in both cases we are able to find an element of $\mathcal{B}^{\text{err}}$ that will be decoded by the terminal to either m_1 or m_2. □

Lemma 15.2.2. $|\mathcal{M}_i^{\psi}| \leq 2\epsilon \cdot 2^{kn}$.

Proof. We can partition $\mathcal{M}_i^{\psi}$ as

$$\mathcal{M}_i^{\psi} = \bigcup_{\hat{z}_i'} \left(\mathcal{M}(\hat{z}_i') \backslash \mathcal{M}(\hat{z}_i', \hat{b}_{i,\hat{z}_i'}) \right),$$

and so

$$|\mathcal{M}_i^{\psi}| = \sum_{\hat{z}_i'} \left(|\mathcal{M}(\hat{z}_i')| - |\mathcal{M}(\hat{z}_i', \hat{b}_{i,\hat{z}_i'})| \right). \tag{15.2}$$

Consider an arbitrary $\hat{z}_i'$ and the set $\mathcal{M}(\hat{z}_i')$. We define an iterative procedure as follows. Initialize $\mathcal{W} := \mathcal{M}(\hat{z}_i')$. If there exist two messages $m_1, m_2 \in \mathcal{W}$ such that $b_i(m_1) \neq b_i(m_2)$, then delete both m_1, m_2 from $\mathcal{W}$. Repeat the operation until there does not exist $m_1, m_2 \in \mathcal{W}$ such that $b_i(m_1) \neq b_i(m_2)$.

After the procedure terminates, it follows that $|\mathcal{W}| \leq |\mathcal{M}(\hat{z}_i', \hat{b}_{i,\hat{z}_i'})|$, because otherwise by definition of $\hat{b}_{i,\hat{z}_i'}$ there must exist $m_1, m_2 \in \mathcal{W}$ such that $b_i(m_1) \neq b_i(m_2)$. Therefore at least $|\mathcal{M}(\hat{z}_i')| - |\mathcal{M}(\hat{z}_i', \hat{b}_{i,\hat{z}_i'})|$ elements are deleted from $\mathcal{M}(\hat{z}_i')$. By Lemma 15.2.1, each pair of elements deleted corresponds to an element of $\mathcal{B}^{\text{err}}$. Also by Lemma 15.2.1 the elements of $\mathcal{B}^{\text{err}}$ corresponding to different deleted pairs are distinct. Summing over all possible values of $\hat{z}_i'$, it follows that the total number of deleted pairs is smaller than the size of $\mathcal{B}^{\text{err}}$:

$$\begin{aligned}\frac{1}{2}\sum_{\hat{z}_i'}\Big(|\mathcal{M}(\hat{z}_i')| - |\mathcal{M}(\hat{z}_i', \hat{b}_{i,\hat{z}_i'})|\Big) & \\ \leq \sum_{\hat{z}_i'} \text{\# of pairs deleted from } \mathcal{M}(\hat{z}_i') & \\ \leq |\mathcal{B}^{\text{err}}| \leq \epsilon \cdot 2^{kn}. & \qquad (15.3)\end{aligned}$$

Combining (15.2) and (15.3) we have $|\mathcal{M}_i^{\psi}| \leq 2\epsilon \cdot 2^{kn}$. □

Next, let $\mathcal{M}(\hat{a}_i, \hat{b}_i) = \{m \in \mathcal{M}^{\text{good}} : a_i(m) = \hat{a}_i, b_i(m) = \hat{b}_i\}$. Intuitively, $\mathcal{M}(\hat{a}_i, \hat{b}_i)$ represent the set of outcomes (in $\mathcal{I}_c$) that will result in signal $\hat{b}_i$ being transmitted on edge b_i and signal $\hat{a}_i$ being transmitted on edge a_i. We define a function $\pi_i : [2^n] \to [2^n]$ as:

$$\pi_i(\hat{b}_i) = \arg\max_{\hat{a}_i} |\mathcal{M}(\hat{a}_i, \hat{b}_i)| \triangleq \hat{a}_{i,\hat{b}_i}. \qquad (15.4)$$

Function π_i will be useful later for designing the network codes in $\mathcal{I}$. Intuitively, in the absence of adversarial errors, π_i estimates the signal transmitted on edge a_i given that the signal transmitted on edge b_i is $\hat{b}_i$. In the following we analyze how often will π_i make a mistake. Define $\mathcal{M}_i^{\pi} = \{m \in \mathcal{M}^{\text{good}} : \pi_i(b_i(m)) \neq a_i(m)\}$. Notice that $\mathcal{M}_i^{\pi}$ is the set of messages for which, when transmitted by the source, π_i will make a mistake in guessing the signal transmitted on a_i. Lemmas 15.2.3 and 15.2.4 analyze the size of $\mathcal{M}_i^{\pi}$.

Lemma 15.2.3. *Define $\mathcal{M}(\hat{a}_i) = \{m \in \mathcal{M}^{good} : a_i(m) = \hat{a}_i\}$. If $|\{b_i(m) : m \in \mathcal{M}(\hat{a}_i)\}| = L$, then there exist $(L-1)|\mathcal{M}(\hat{a}_i)|$ distinct elements of $\mathcal{B}^{err}$ such that each of them will be decoded by terminal t to some message $m \in \mathcal{M}(\hat{a}_i)$.*

Proof. Assume for concreteness that $\{b_i(m) : m \in \mathcal{M}(\hat{a}_i)\} = \{\hat{b}_i^{(1)}, ..., \hat{b}_i^{(L)}\}$, then there exist L messages $m_1, ..., m_L \in \mathcal{M}(\hat{a}_i)$ such that $b_i(m_j) = \hat{b}_i^{(j)}$, $j = 1, ..., L$. For $j = 1, ..., L$, let $\boldsymbol{r}_j$ be the error pattern that changes the signal on z_i' to be

$z_i'(m_j)$. Then if a message $m_0 \in \mathcal{M}(\hat{a}_i)$ is transmitted by the source and $\boldsymbol{r}_j$ happens, the node B_i will receive the same inputs $(x_i(m_0), y_i(m_0), z_i'(m_j))$ as in the situation that m_j is sent and no error happens. Therefore $b_i(m_0, \boldsymbol{r}_j) = \hat{b}_i^{(j)}$, and so $|\{\boldsymbol{b}(m_0, \boldsymbol{r}_j)\}_{j\in[L]}| = |\{b_i(m_0, \boldsymbol{r}_j)\}_{j\in[L]}| = L$. Since $m_0 \in \mathcal{M}^{\text{good}}$, it is correctly decodable under any error pattern $\boldsymbol{r} \in \mathcal{R}_{\mathcal{B}}$, and so all elements of $\{\boldsymbol{b}(m_0, \boldsymbol{r}_j)\}_{j\in[L]}$ will be decoded by the terminal to m_0. Except the element $\boldsymbol{b}(m_0)$, the other $L-1$ elements of $\{\boldsymbol{b}(m_0, \boldsymbol{r}_j)\}_{j\in[L]}$ are elements of $\mathcal{B}^{\text{err}}$. Sum over all $m_0 \in \mathcal{M}(\hat{a}_i)$ and the assertion is proved. □

Lemma 15.2.4. $|\mathcal{M}_i^{\pi}| \leq 3\epsilon \cdot 2^{kn}$.

Proof. Define $\mathcal{A}_{i,1}^{\pi} = \{\hat{a}_i \in [2^n] : |\mathcal{M}(\hat{a}_i)| \leq \frac{1}{2}2^{(k-1)n}\}$, and $\mathcal{A}_{i,2}^{\pi} = \{\hat{a}_i \in [2^n]\backslash\mathcal{A}_{i,1}^{\pi} : |\{b_i(m) : m \in \mathcal{M}(\hat{a}_i)\}| > 1\}$. Then define $\mathcal{M}_{i,1}^{\pi} = \{m \in \mathcal{M}^{\text{good}} : a_i(m) \in \mathcal{A}_{i,1}^{\pi}\}$, and $\mathcal{M}_{i,2}^{\pi} = \{m \in \mathcal{M}^{\text{good}} : a_i(m) \in \mathcal{A}_{i,2}^{\pi}\}$. Notice that by construction $\mathcal{A}_{i,1}^{\pi}$ and $\mathcal{A}_{i,2}^{\pi}$ are disjoint, and $\mathcal{M}_{i,1}^{\pi}$ and $\mathcal{M}_{i,2}^{\pi}$ are disjoint. We claim that,

$$\mathcal{M}_i^{\pi} \subset \mathcal{M}_{i,1}^{\pi} \cup \mathcal{M}_{i,2}^{\pi}. \tag{15.5}$$

To prove the claim, consider any $m \in \mathcal{M}^{\text{good}}$ such that $m \notin \mathcal{M}_{i,1}^{\pi} \cup \mathcal{M}_{i,2}^{\pi}$. We will show that $\pi(b_i(m)) = a_i(m)$. Suppose for the sake of contradiction that $\pi(b_i(m)) = \hat{a}_i \neq a_i(m)$, then it follows that

$$\begin{aligned} |\mathcal{M}(\hat{a}_i, b_i(m))| &\overset{(a)}{>} |\mathcal{M}(a_i(m), b_i(m))| \\ &\overset{(b)}{=} |\mathcal{M}(a_i(m))| \overset{(c)}{>} \frac{1}{2}2^{(k-1)n}, \end{aligned} \tag{15.6}$$

where (a) is due to the definition of π, (b) is due to the fact that $m \notin \mathcal{M}_{i,2}^{\pi}$ and (c) is due to the fact that $m \notin \mathcal{M}_{i,1}^{\pi}$. Let $\mathcal{M}(\hat{b}_i) = \{m' \in \mathcal{M}^{\text{good}} : b_i(m') = \hat{b}_i\}$, then $\mathcal{M}(\hat{a}_i, b_i(m)) \cup \mathcal{M}(a_i(m)) \subset \mathcal{M}(b_i(m))$. Since $\hat{a}_i \neq a_i(m)$, $\mathcal{M}(\hat{a}_i, b_i(m))$ and $\mathcal{M}(a_i(m))$ are disjoint, and it follows that $|\mathcal{M}(b_i(m))| \geq |\mathcal{M}(\hat{a}_i, b_i(m))| + |\mathcal{M}(a_i(m))| > 2^{(k-1)n}$. However, because $|\{(\hat{b}_1, ..., \hat{b}_k) \in [2^n]^k : \hat{b}_i = b_i(m)\}| = 2^{(k-1)n}$, by the pigeonhole principle there must exist two messages $m_1, m_2 \in \mathcal{M}(b_i(m))$ such that $\boldsymbol{b}(m_1) = \boldsymbol{b}(m_2)$. This is a contradiction since the terminal cannot distinguish m_1 from m_2. This proves $\pi(b_i(m)) = a_i(m)$ as well as (15.5).

We next bound the size of $\mathcal{M}_{i,1}^{\pi}$ and $\mathcal{M}_{i,2}^{\pi}$. For any $\hat{a}_i' \in \mathcal{A}_{i,1}^{\pi}$, by definition $\{(\hat{a}_1, ..., \hat{a}_k) \in [2^n]^k : \hat{a}_i = \hat{a}_i'\}\backslash\{(\boldsymbol{a}(m) : m \in \mathcal{M}(\hat{a}_i')\}$ is a subset of $\mathcal{A}^{\text{err}}$ with

size at least $\frac{1}{2}2^{(k-1)n}$. Therefore each element of $\mathcal{A}_{i,1}^{\pi}$ will contribute to at least $\frac{1}{2}2^{(k-1)n}$ distinct elements of $\mathcal{A}^{\text{err}}$. Hence $|\mathcal{A}_{i,1}^{\pi}| \cdot \frac{1}{2}2^{(k-1)n} \leq |\mathcal{A}^{\text{err}}| \leq \epsilon \cdot 2^{kn}$, and so $|\mathcal{A}_{i,1}^{\pi}| \leq 2\epsilon \cdot 2^{n}$. It then follows that $|\mathcal{M}_{i,1}^{\pi}| \leq \frac{1}{2}2^{(k-1)n}|\mathcal{A}_{i,1}^{\pi}| \leq \epsilon \cdot 2^{kn}$.

By Lemma 15.2.3, each elements of $\mathcal{A}_{i,2}^{\pi}$ will contribute to at least $\frac{1}{2}2^{(k-1)n}$ distinct elements in $\mathcal{B}^{\text{err}}$. Therefore $|\mathcal{A}_{i,2}^{\pi}| \cdot \frac{1}{2}2^{(k-1)n} \leq |\mathcal{B}^{\text{err}}| \leq \epsilon \cdot 2^{kn}$, and so $|\mathcal{A}_{i,2}^{\pi}| \leq 2\epsilon \cdot 2^{n}$. It then follows that $|\mathcal{M}_{i,2}^{\pi}| \leq 2^{(k-1)n}|\mathcal{A}_{i,2}^{\pi}| \leq 2\epsilon \cdot 2^{kn}$. Finally, by (15.5) we have $|\mathcal{M}_{i}^{\pi}| \leq |\mathcal{M}_{i,1}^{\pi}| + |\mathcal{M}_{i,2}^{\pi}| \leq 3\epsilon \cdot 2^{kn}$. □

We are now ready to prove Theorem 15.2.1.

Proof ("⇐" part of Theorem 15.2.1). We show the achievability of rate k in $\mathcal{I}_c$ implies the achievability of unit rate in $\mathcal{I}$.

Let $\{\phi_e, \phi_t\}_{e\in\mathcal{E}}$ be the network error correction code of length n that achieves rate k in $\mathcal{I}_c$, with probability of error ϵ. We assume that in this code edge z_i simply relays the signal from edge a_i. This is without loss of generality because for any network code that needs to process the signal on edge a_i to obtain the signal to be transmitted on edge z_i, it is equivalent to relay the signal on edge z_i and perform the processing work at the head node of edge z_i.

Let $\mathcal{E}_{\mathcal{N}} \subset \mathcal{E}$ be the set of edges of the embedded graph $\mathcal{N}$. For the multiple-unicast problem $\mathcal{I}$, we define a length-n network code $\{\tau_e, \tau_{t_i} : e \in \mathcal{E}_{\mathcal{N}}, i \in [k]\}$ as follows.

$$\begin{aligned}
&\tau_e = \phi_e, \quad \forall e \in \mathcal{E}_{\mathcal{N}} \\
&\tau_{t_i} = \phi_{z_i} \circ \pi_i \circ \psi_i \circ \phi_{z_i'}, \quad \forall i = 1, ..., k,
\end{aligned}$$

where $\circ$ denotes function composition; ϕ_{z_i} and $\phi_{z_i'}$ are the encoding functions of edges z_i and z_i' in problem $\mathcal{I}_c$; ψ_i is defined in (15.1); and π_i is defined in (15.4). In the following we show that $\{\tau_e, \tau_{t_i} : e \in \mathcal{E}_{\mathcal{N}}, i \in [k]\}$ achieves unit rate in $\mathcal{I}$ with probability of error upper bounded by $6k\epsilon$.

In problem $\mathcal{I}$, let M_i be the random message associated with source s_i, then M_i, $i = 1, ..., k$ are i.i.d. uniformly distributed over $[2^n]$. Denote for short $\boldsymbol{M} = (M_1, ..., M_k)$, then slightly abusing notation we denote by $\tau_{t_i}(\boldsymbol{M})$ the output of the decoder τ_{t_i} under transmission $\boldsymbol{M}$. The probability of decoding error is given by

$$\Pr\{\bigcup_{i=1}^{k} \tau_{t_i}(\boldsymbol{M}) \neq M_i\},$$

where the probability is taken over the joint distribution of the random messages. Let $\boldsymbol{m} = (m_1, ..., m_k)$ be the realization of $\boldsymbol{M}$. We claim that if there exists a message m of problem $\mathcal{I}_c$ (not to be confused with $\boldsymbol{m}$, a message of $\mathcal{I}$) such that $m \in \mathcal{M}^{\text{good}}$, $m \notin \mathcal{M}_i^{\psi}$, $m \notin \mathcal{M}_i^{\pi}$ and $\boldsymbol{m} = \boldsymbol{z}(m)$, then $\tau_{t_i}(\boldsymbol{m}) = m_i$. To prove the claim, suppose $\boldsymbol{m} = \boldsymbol{z}(m)$ is transmitted in $\mathcal{I}$. Notice that all edges in $\mathcal{N}$ perform the same coding scheme in $\mathcal{I}$ as in $\mathcal{I}_c$, therefore the terminal node t_i, by invoking the function $\phi_{z_i'}$, obtains $z_i'(m)$. Then by the definition of $\mathcal{M}_i^{\psi}$, it follows that $\psi_i(z_i'(m)) = b_i(m)$. And by the definition of $\mathcal{M}_i^{\pi}$, it follows that $\pi(\psi_i(z_i'(m))) = a_i(m)$. Finally since $\boldsymbol{m} = \boldsymbol{z}(m)$, it follows that $\phi_{z_i}(\pi(\psi_i(z_i'(m)))) = \phi_{z_i}(a_i(m)) = z_i(m) = m_i$.

Therefore $\tau_{t_i}(\boldsymbol{m}) = m_i$ if $\boldsymbol{m} \in \{\boldsymbol{z}(m) \in [2^n]^k : m \in \mathcal{M}^{\text{good}}, m \notin \mathcal{M}_i^{\psi}, m \notin \mathcal{M}_i^{\pi}\}$. The probability that τ_{t_i} makes an error, i.e., $\Pr\{\tau_{t_i}(\boldsymbol{M}) \neq M_i\}$, is upper bounded by the probability of the union of the following three events.

$$
\begin{aligned}
E_1 &= \{\boldsymbol{M} = \boldsymbol{m} : \boldsymbol{m} \notin \{\boldsymbol{z}(m) \in [2^n]^k : m \in \mathcal{M}^{\text{good}}\}\} \\
E_2 &= \{\boldsymbol{M} = \boldsymbol{m} : \boldsymbol{m} \in \{\boldsymbol{z}(m) \in [2^n]^k : m \in \mathcal{M}_i^{\psi}\}\} \\
E_3 &= \{\boldsymbol{M} = \boldsymbol{m} : \boldsymbol{m} \in \{\boldsymbol{z}(m) \in [2^n]^k : m \in \mathcal{M}_i^{\pi}\}\}.
\end{aligned}
$$

We upper bound the probability of E_1, E_2, E_3, respectively.

$$
\begin{aligned}
\Pr\{E_1\} &= 1 - \frac{|\{\boldsymbol{z}(m) : m \in \mathcal{M}^{\text{good}}\}|}{2^{kn}} \\
&\overset{(d)}{=} 1 - \frac{|\mathcal{M}^{\text{good}}|}{2^{kn}} \leq 1 - \frac{(1-\epsilon) \cdot 2^{kn}}{2^{kn}} = \epsilon,
\end{aligned} \tag{15.7}
$$

where (d) follows from the fact that $\boldsymbol{z}(m) = \boldsymbol{a}(m) \neq \boldsymbol{a}(m') = \boldsymbol{z}(m')$ for any $m, m' \in \mathcal{M}^{\text{good}}$, $m \neq m'$. By Lemma 15.2.2,

$$
\Pr\{E_2\} = \frac{|\mathcal{M}_i^{\psi}|}{2^{kn}} \leq 2\epsilon. \tag{15.8}
$$

And by Lemma 15.2.4, we have

$$
\Pr\{E_3\} = \frac{|\mathcal{M}_i^{\pi}|}{2^{kn}} \leq 3\epsilon. \tag{15.9}
$$

Combining (15.7), (15.8) and (15.9), it follows that

$$
\Pr\{\tau_{t_i}(\boldsymbol{M}) \neq M_i\} \leq \Pr\{E_1\} + \Pr\{E_2\} + \Pr\{E_3\} \leq 6\epsilon.
$$

Finally, by taking the union bound over the k terminals,

$$
\Pr\{\bigcup_{i=1}^{k} \tau_{t_i}(\boldsymbol{M}) \neq M_i\} \leq 6k\epsilon.
$$

Hence the probability of error is arbitrarily small and this establishes the achievability of unit rate in $\mathcal{I}$. □

The proof above suggests that the achievability of rate k with error probability ϵ in $\mathcal{I}_c$ implies the achievability of unit rate with error probability $6k\epsilon$ in $\mathcal{I}$. By setting $\epsilon = 0$, we generalize the result in Theorem 15.1.1 regarding the zero-error achievability as a special case.

Finally, we remark that our reduction has an operational aspect that from a code for $\mathcal{I}_c$ one can construct a code for $\mathcal{I}$. Indeed, using our reduction, to solve an instance of the multiple-unicast network coding problem, one may first reduce it to an instance of the unicast network error correction problem, then solve the latter, and finally use this solution to obtain a solution to the original multiple-unicast problem.

15.3 Asymptotic Achievability and Counter-example

Theorem 15.1.1 and 15.2.1 show that Construction 15.1.1 gives a reduction from multiple-unicast network coding to unicast network error correction in terms of zero-error achievability and in terms of achievability that allows vanishing error. In this subsection we show that the same construction does not provide a reduction in terms of asymptotic achievability (with vanishing error) by presenting a counter-example.

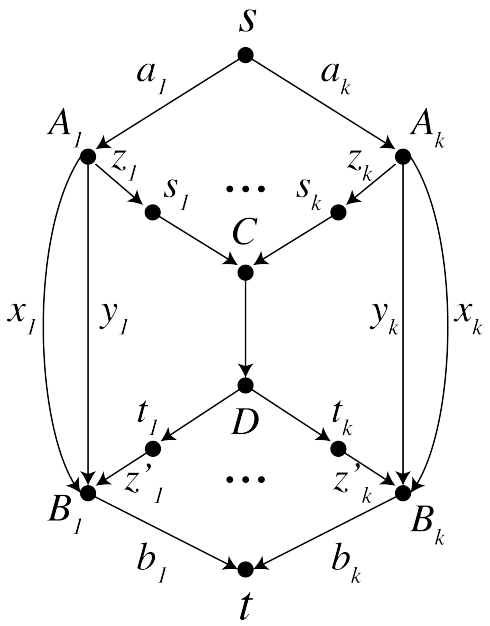


Figure 15.3: Construction of $\mathcal{I}_c$ and $\mathcal{I}$. In $\mathcal{I}_c$, the source is s and the terminal is t. $\mathcal{B}$ includes all singleton sets of a single edge except $\{a_i\}$ and $\{b_i\}$, $i = 1, ..., k$. In $\mathcal{I}$, the source-terminal pairs are $(s_i, t_i), i = 1, ..., k$. All edges have unit capacity.

Theorem 15.3.1. *There exists a multiple-unicast network coding instance $\mathcal{I}$ such*

that unit rate is not asymptotically achievable in $\mathcal{I}$, but in $\mathcal{I}_c$, which is the unicast network error correction instance constructed from $\mathcal{I}$ according to Construction 15.1.1, rate k is asymptotically achievable.

Proof. The construction of $\mathcal{I}$ and the corresponding $\mathcal{I}_c$ are shown in Figure 15.3. In $\mathcal{I}$, $\{(C, D)\}$ is a cut-set separating all sources from the terminals. Therefore by the cut-set bound, any rate $R > 1/k$ is not achievable in $\mathcal{I}$. This shows that unit rate is not asymptotically achievable in $\mathcal{I}$ if $k > 1$.

We prove the remaining part of the theorem by describing a network code with length n that achieves rate $k - k/n$ in $\mathcal{I}_c$. First divide the source message of rate $k - k/n$ into k pieces $M = (M_1, ..., M_k)$, such that M_i, $i = 1, ..., k$ are i.i.d. uniformly distributed over $[2^{n-1}]$. We denote $\phi_e : [2^{k(n-1)}] \to [2^n]$ as the encoding function[1] of edge e, which takes the source message M as input, and outputs the signal to be transmitted on e when there is no error in the network. For all $i = 1, ..., k$, we let

$$\phi_{a_i}(M) = \phi_{x_i}(M) = \phi_{y_i}(M) = \phi_{z_i}(M) = \phi_{(s_i,C)}(M) = M_i.$$

Furthermore, we let

$$\phi_{(C,D)}(M) = \phi_{(D,t_i)}(M) = \phi_{z'_i}(M) = \sum_{j=1}^{k} M_j, \ \ \forall i = 1, ..., k$$

where the summation is bitwise XOR. Note that the edges a_i, x_i, y_i, z_i, (s_i, C), (C, D), (D, t_i), z'_i each have the capacity to transmit n bits. But we only require each of them to transmit $n - 1$ bits. Hence each edge reserves one unused bit.

Node B_i, by observing the (possibly corrupted) signals received from edges x_i, y_i, z'_i, performs error detection/correction in the following way. If the signal (of $n-1$ bits) received from x_i equals the signal received from y_i, forward the signal to edge b_i, and then transmit one bit of 0 using the reserved bit. Otherwise, forward the signal received from z'_i to b_i, and then transmit one bit of 1 using the reserved bit.

Finally, terminal t recovers the source message in the following way. Note that since there is only one corrupted edge in the network, for $i = 1, ..., k$, the reserved bit on b_i equals 0 only if both x_i and y_i are not corrupted and it equals 1 only if either x_i or y_i is corrupted. Therefore, if the reserved bit on b_i is 0 then the remaining $n - 1$ bits received from b_i are exactly M_i. If the reserved bit on b_i is 0 for $i = 1, ..., k$, then $M = (M_1, ..., M_k)$ is decoded correctly at the terminal.

[1] This is called the *global encoding function* in the context of network coding.

Otherwise, among $b_1, ..., b_k$, there is at most one b_l such that the reserved bit on b_l is 1, because there is at most one corrupted edge. The remaining $n-1$ bits received from b_l are exactly $\sum_{j=1}^{k} M_j$. This is because either x_l or y_l is corrupted and so z_l is not corrupted. Now note that the terminal t can decode M_i, $i \neq l$ correctly from the signals received from b_i, $i \neq l$. And so t can decode M_l correctly by evaluating $\sum_{j=1}^{k} M_j - \sum_{j=1, j\neq l}^{k} M_j = M_l$. Hence $M = (M_1, ..., M_k)$ is decoded correctly at the terminal. This shows that rate $k - k/n$ is achievable in $\mathcal{I}_c$ and so rate k is asymptotically achievable in $\mathcal{I}_c$, completing the proof. □

Combining Theorem 15.2.1 with Theorem 15.3.1, it follows that in the unicast network error correction problem, the capacity in general is not achievable. Note that we mean the capacity is not *exactly* achievable, and by definition capacity is always *asymptotically* achievable.

Corollary 15.3.1. *There exists a unicast network error correction problem for which the capacity is not achievable.*

Proof. The construction of the network error correction problem $\mathcal{I}_c$ is shown in Figure 15.3. By the cut-set bounds, the capacity of $\mathcal{I}_c$ is upper bounded by k. By Theorem 15.3.1, rate k is asymptotically achievable in $\mathcal{I}_c$, and so the capacity of $\mathcal{I}_c$ is k. Also by Theorem 15.3.1, unit rate is not achievable in $\mathcal{I}$, and so by Theorem 15.2.1, rate k is not achievable in $\mathcal{I}_c$. This shows that the capacity of $\mathcal{I}_c$ is not achievable. □

Corollary 15.3.1 suggests that although the (unicast) network error correction capacity is (by definition) asymptotically achievable, in general it is not achievable. This is in contrast to the scenario of network error correction with uniform $\mathcal{B}$, i.e., $\mathcal{B}$ is the collection of all subsets containing z_e links. In this case the network capacity can be achieved by linear codes. Unachievability of capacity is also studied for multiple-unicast networks [114] and sum networks [115]. For both cases, networks in which the capacity is not achievable are constructed using matroid theory.

Chapter 16

BOUNDS FOR SECURE NETWORK CODING

16.1 Models

Consider an instance $(\mathcal{G}, s, d, \mathcal{A})$ of the unicast secure network coding problem, where $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a directed graph, s is the source, d is the terminal and $\mathcal{A}$ is the collection of sets of eavesdropped edges. Since $\mathcal{A}$ is arbitrary (i.e., non-uniform), without loss of generality we assume that all edges have unit capacity, because any edge of larger capacity can be replaced by a number of parallel unit capacity edges in both $\mathcal{G}$ and $\mathcal{A}$. Non-source nodes are allowed to generate randomness and for all $i \in \mathcal{V}$, denote by K_i the independent randomness generated by node i. In this chapter we focus on perfect secrecy, i.e., let M be the source message and denote by $X(A)$ the signals transmitted on $A \subset \mathcal{E}$, then $\forall A \in \mathcal{A}$, $I(M; X(A)) = 0$.

Consider an arbitrary cut $V \subset \mathcal{V}$ such that $s \in V$ and $d \in V^c$. Denote $E_V^{\text{fwd}} = \{(i,j) \in \mathcal{E} : i \in V, j \in V^c\}$ as the set of forward edges with respect to V, and $E_V^{\text{bwd}} = \{(i,j) \in \mathcal{E} : i \in V^c, j \in V\}$ as the set of backward edges. Let $x = |E_V^{\text{fwd}}|$ and $y = |E_V^{\text{bwd}}|$, and denote the x forward edges by $e_1^{\text{fwd}}, e_2^{\text{fwd}}, ..., e_x^{\text{fwd}}$, and the y backward edges by $e_1^{\text{bwd}}, e_2^{\text{bwd}}, ..., e_y^{\text{bwd}}$. Define $C_{b\to f} = (c'_{ij})$ to be an $x \times y$ binary matrix characterizing the connectivity from the backward edges to the forward edges. More precisely,

$$c'_{ij} = \begin{cases} 1 & \text{if } \exists \text{ a directed path from head}(e_j^{\text{bwd}}) \text{ to tail}(e_i^{\text{fwd}}) \text{ that does not pass through any nodes in } V^c \\ 0 & \text{otherwise.} \end{cases}$$

16.2 Cut-set Bound

This section presents a cut-set bound on the secrecy capacity with respect to the cut V and its connectivity matrix $C_{b\to f}$. We first prove a lemma before formally introducing the bound.

Lemma 16.2.1. *Given an arbitrary binary matrix $C = (c_{ij})$ of size $a \times b$ and a set of submatrices of C, denoted by $\mathcal{U}$, there is a large enough q such that there exists a matrix $\bar{C} \in \mathbb{F}_q^{a\times b} = (\bar{c}_{ij})$ with the following properties: 1) $\bar{c}_{ij} = 0$ if $c_{ij} = 0$; 2) $\forall$ $U \in \mathcal{U}$, let its size be $m \times n$ and let the corresponding submatrix of $\bar{C}$ be $\bar{U}$, then*

rank$(\bar{U}) = \max_{V \in \mathbb{F}_q^{m\times n}, v_{ij}=0 \text{ if } u_{ij}=0}$ *rank*(V)*, i.e.,* $\bar{U}$ *is rank maximized subject to the zero constraints given in* C*. In particular,* $q > |\mathcal{U}|ab$ *is sufficient.*

Proof. Consider a finite field $\mathbb{F}_q$ of size q, and any $U \in \mathcal{U}$, let

$$\bar{V} = \arg \max_{V \in \mathbb{F}_q^{m\times n}, v_{ij}=0 \text{ if } u_{ij}=0} \text{rank}(V),$$

and let $r_U = \text{rank}(\bar{V})$. So $\bar{V}$ contains an $r_U \times r_U$ full rank submatrix, denoted by $\underline{V}$. Let $\underline{C} = (\underline{c}_{ij})$ be the submatrix of C corresponding to the position of $\underline{V}$. Now consider a polynomial matrix $\underline{V}[\boldsymbol{x}] = (\underline{v}_{ij})$ defined by

$$\underline{v}_{ij} = \begin{cases} 0 & \text{if } \underline{c}_{ij} = 0 \\ x_{ij} & \text{if } \underline{c}_{ij} = 1 \end{cases}$$

where the x_{ij}'s are indeterminates. Then it follows that $\det(\underline{V}[\boldsymbol{x}])$ is not the zero polynomial because otherwise $\det(\underline{V}) = 0$ and $\underline{V}$ cannot be full rank. Now let the non-zero entries of $\underline{V}[\boldsymbol{x}]$, i.e., all the x_{ij}'s, be i.i.d. uniformly distributed on $\mathbb{F}_q$. By the Schwartz-Zippel lemma,

$$\Pr\{\det(\underline{V}[\boldsymbol{x}]) = 0\} \le \frac{r_U^2}{q} \le \frac{ab}{q}.$$

Notice that the polynomial matrix $\underline{V}[\boldsymbol{x}]$ is in fact a submatrix of a $a \times b$ polynomial matrix $B[\boldsymbol{x}] = (b_{ij})$ defined by

$$b_{ij} = \begin{cases} 0 & \text{if } c_{ij} = 0 \\ x_{ij} & \text{if } c_{ij} = 1 \end{cases}$$

where again the x_{ij}'s are indeterminates. Let all the non-zero entries of $B[\boldsymbol{x}]$ follow i.i.d. uniform distribution on $\mathbb{F}_q$, and by the union bound, we have

$$\begin{aligned} \Pr\left\{\bigcup_{U\in\mathcal{U}} \det(\underline{V}[\boldsymbol{x}]) \neq 0\right\} &\ge 1 - \sum_{U\in\mathcal{U}} \Pr\{\det(\underline{V}[\boldsymbol{x}]) = 0\} \\ &\ge 1 - |\mathcal{U}|\frac{ab}{q}. \end{aligned}$$

Therefore if $q > |\mathcal{U}|ab$, there exists an evaluation of $B[\boldsymbol{x}]$ such that $\det(\underline{V}[\boldsymbol{x}]) \neq 0$ for any $U \in \mathcal{U}$. This evaluation gives a desired $\bar{C}$, because for any $U \in \mathcal{U}$, the corresponding submatrix $\bar{U}$ of $\bar{C}$ contains a full rank square submatrix of size r_U, and by definition r_U is the maximum rank $\bar{U}$ can achieve subject to the zero constraints in C. □

Define

$$C = \begin{pmatrix} C_{b\to f} \\ I_y \end{pmatrix},$$

where I_y is the identity matrix of order y. Notice that rows in C correspond to edges crossing the cut in $\mathcal{G}$. Denote $\mathcal{A}_V = \{A \cap (E_V^{\text{fwd}} \cup E_V^{\text{bwd}}) : A \in \mathcal{A}\}$. For $A \in \mathcal{A}_V$, denote by U_A the submatrix of C formed by the rows corresponding to edges in A. Let $\mathcal{U} = \{U_A, A \in \mathcal{A}_V\}$, and let $\bar{C}$ be the rank maximized matrix specified in Lemma 16.2.1 with respect to C and $\mathcal{U}$. For $U_A \in \mathcal{U}$, let $\bar{U}_A$ be the corresponding submatrix of $\bar{C}$. We are now ready to state our main result.

Theorem 16.2.1. *The secrecy capacity is bounded by*

$$\mathfrak{C} \leq x + \min_{A\in\mathcal{A}_V} \left(\text{rank}(\bar{U}_A) - |A|\right).$$

In the special case of uniform wiretap sets, i.e., $\mathcal{A} = \{A \subset \mathcal{E} : |A| \leq z\}$, Theorem 16.2.1 reduces to the following form.

Corollary 16.2.1. *Define* $k_b = \min_{\{\bar{U}:\, z\times y \text{ submatrix of } \bar{C}\}} \text{rank}(\bar{U})$, *then the secrecy capacity is bounded by*

$$\mathfrak{C} \leq x + k_b - z.$$

We prove Theorem 16.2.1 in the remaining part of this section. Given a cut of x forward edges, y backward edges, and the connectivity matrix $C_{b\to f}$, we construct an upper bounding network $\bar{\mathcal{G}}$ as follows: 1) Absorb all nodes downstream of the cut, i.e., all $v \in V^c$, into the terminal d. So for $i \in [x]$, $j \in [y]$, head(e_i^{fwd})=d, tail(e_j^{bwd})=d. 2) Connect the source to each forward edge with an infinite number of unit capacity edges $(s, \text{tail}(e_i^{\text{fwd}}))$. 3) Connect the backward edges to the forward edges according to $C_{b\to f}$. More precisely, create an infinite number of unit capacity edges $(\text{head}(e_j^{\text{bwd}}), \text{tail}(e_i^{\text{fwd}}))$ if and only if $c'_{ij} = 1$. In $\bar{\mathcal{G}}$ we only allow the source and the terminal to generate independent randomness, and the collection of sets of eavesdropped edges is $\mathcal{A}_v$, i.e., only the edges in the cut-set can be eavesdropped.

Lemma 16.2.2. *The secrecy capacity of $\bar{\mathcal{G}}$ upperbounds the secrecy capacity of $\mathcal{G}$.*

Proof. Note that all infinite parallel unit capacity edges are perfectly secure because they can be protected by an infinite number of local keys. This implies that the

assumption that only the source and the terminal can generate randomness is without loss of optimality, because if any other node wishes to generate independent randomness, such randomness can be generated at the source instead and sent by infinite parallel edges. Hence for any coding scheme in $\mathcal{G}$ on the edges crossing the cut, the same coding scheme can be simulated in $\bar{\mathcal{G}}$ securely. □

Since $\bar{\mathcal{G}}$ has a simplified structure, in the proof of Theorem 16.2.1 we shall always consider $\bar{\mathcal{G}}$ instead of $\mathcal{G}$ unless otherwise specified. As mentioned, in $\bar{\mathcal{G}}$ only the edges crossing the cut V are vulnerable, and for notational convenience in what follows we let $\mathcal{A} = \mathcal{A}_V$. Let $F_1, ..., F_x$ be the signals transmitted on edges $e_1^{\text{fwd}}, ..., e_x^{\text{fwd}}$; $B_1, ..., B_y$ be the signals transmitted on edges $e_1^{\text{bwd}}, ..., e_y^{\text{bwd}}$. Consider any $A \in \mathcal{A}$ and the set of signals $f_A = \bigcup_{e \in A} f_e$, defined as follows

$$f_e = \begin{cases} \{B_j\} & \text{if } e = e_j^{\text{bwd}} \\ \{B_j : c'_{ij} = 1\} & \text{if } e = e_i^{\text{fwd}}. \end{cases}$$

The following lemma shows that the rank structure of the submatrices of $\bar{C}$ has interesting properties.

Lemma 16.2.3. *For any $A \in \mathcal{A}$, there exists a partition $A = A_1 \cup A_2$, such that $|f_{A_1}| + |A_2| =$ rank($\bar{U}_A$).*

Proof. The idea of the proof is to infer the structure of U_A given the rank of $\bar{U}_A$ and the fact that $\bar{U}_A$ is rank maximized. Then since U_A characterizes the connectivity to the edges in A it becomes convenient to bound the size of f_A.

Denote for short $r = \text{rank}(\bar{U}_A)$, so $\bar{U}_A$ contains an $r \times r$ submatrix whose determinant is non-zero, and therefore in $\bar{U}_A$ there exist r non-zero entries at different columns and at different rows. Recall that an entry in $\bar{U}_A$ can be non-zero only if this entry is 1 in U_A, hence U_A contains r entries of value 1 at different columns and different rows. Perform column and row permutations to move these 1's such that $U_A(r + 1 - i, i) = 1, \forall 1 \leq i \leq r$, i.e., they become the counter-diagonal entries of the upper-left block formed by the first $r \times r$ entries. See Fig. 16.1 for an example. Note that permutations in U_A are merely reordering of edges, and for notational convenience we denote the matrix after permutations as U_A still.

It then follows that $U_A(i, j) = 0, \forall r < i \leq |A|, r < j \leq y$. Otherwise if any entry in this lower right block is non-zero, setting this and the aforementioned r counter-diagonal entries as 1, and all other entries as 0 yields a matrix that satisfies

the zero constraint in U_A and it has rank $r+1$. But this is a contradiction because $r = \text{rank}(\bar{U}_A)$ is the maximum rank. Hence we label this block as *zero*.

Below we introduce an algorithm that further permutes U_A and labels it blockwise. The algorithm takes a matrix G of arbitrary size $m \times n$ and a positive integer parameter k as input, such that the upper-left block G_{UL} formed by the first $k \times k$ entries of G has all 1's in its counter-diagonal. Now consider $G_{LL} = (g_{ij}), k < i \leq m, 1 \leq j \leq k$ which is the lower-left block of G. If every column of G_{LL} is non-zero, label this block as *non-zero*, label G_{UL} as *counter-diagonal*, label the block $G_{UR} = (g_{ij}), 1 \leq i \leq k, k < j \leq n$ as *zero**, return $t := 0$ and terminate. If $G_{LL} = \mathbf{0}$ or G_{LL} is empty, label this block (if not empty) as *zero*, label G_{UL} as *counter-diagonal*, label G_{UR} as *arbitrary*, return $t := k$ and terminate.

Otherwise G_{LL} contains both zero and non-zero columns. In this case, first perform column permutations in G to move all non-zero columns of G_{LL} to the left and zero columns to the right. Assume that after permutation the first u columns of G_{LL} are non-zero, and the last v columns are all zero. Label the block $(g_{ij}), k < i \leq m, 1 \leq j \leq u$ as *non-zero* and label the block $(g_{ij}), k < i \leq m, u < j \leq k$ as *zero*. At this point some of the 1's originally in the counter-diagonal of G_{UL} are misplaced due to column permutations; perform row permutations to move them back to the counter-diagonal. Note that only the first k rows need to be permuted and the lower labeled block(s) is not affected. Label the block $(g_{ij}), k-u+1 \leq i \leq k, 1 \leq j \leq u$ as *counter-diagonal*, label the block $(g_{ij}), 1 \leq i \leq k-u, 1 \leq j \leq u$ as *arbitrary*, and label the block $(g_{ij}), k-u+1 \leq i \leq k, k < j \leq n$ as *zero**. Then truncate the first u columns and the last $m-k$ rows from G. Notice that the block formed by the first $v \times v$ entries in the truncated G has all 1's in its counter-diagonal. Now invoke the algorithm recursively to the truncated G with parameter $v < k$. The algorithm must terminate because the input parameter is a positive finite integer and cannot decrease indefinitely.

Applying the algorithm to the matrix U_A with parameter $k := r$ will permute the rows and columns of U_A and label it completely. Refer to Figure 16.1 for an example. Notice that the algorithm always labels *counter-diagonal*, *non-zero* and *zero* literally, i.e., by hypothesis all counter-diagonal-label blocks are square and have 1's in their counter-diagonals (but the off-counter-diagonal entries may be arbitrary); all non-zero-label blocks do not contain zero columns; and all zero-label blocks are all zero. The only non-trivial label is *zero**, and we claim that the algorithm also labels *zero** correctly in the sense that a zero*-labeled block is indeed zero.

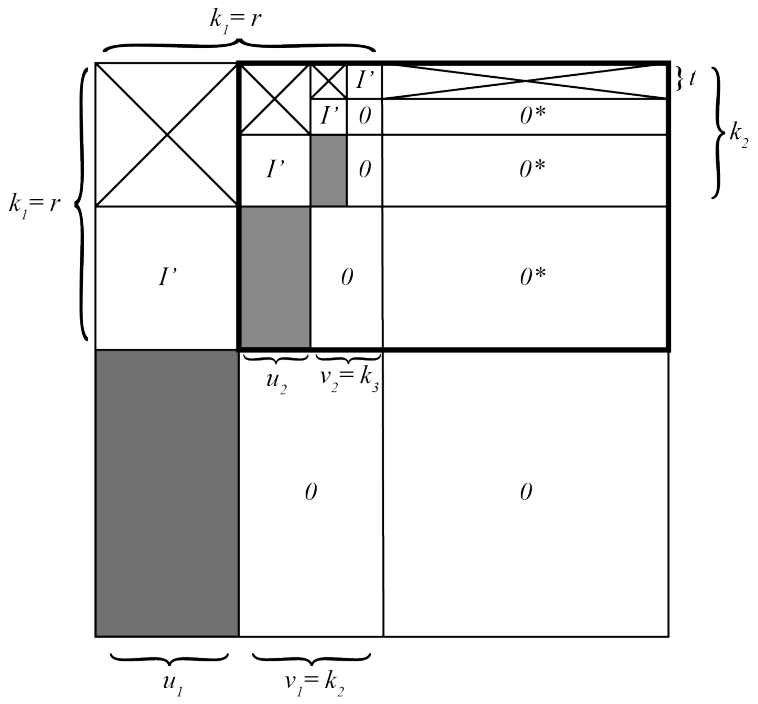


Figure 16.1: An example of a labeled U_A. *zero* blocks are indicated by 0; *zero** blocks are indicated by 0^*; *counter-diagonal* blocks are indicated by I'; *non-zero* blocks are colored in gray and *arbitrary* blocks are crossed. The algorithm terminates in four iterations and returns t. Key parameters of the first two iterations are illustrated and the subscripts denote iteration numbers. The truncated G after the first iteration is highlighted in bold line. Note that the first t rows correspond to A_2, and the remaining rows correspond to A_1.

To prove the claim, notice that all *zero** blocks pile up at the last $y - r$ columns of U_A, and consider any entry α_1 of a *zero** block. By the algorithm the row of α_1 must intersect a unique *counter-diagonal* block, and denote the intersecting counter-diagonal entry of the *counter-diagonal* block as β_1. By the algorithm this intersecting *counter-diagonal* block must lie immediately on top of a *non-zero* block. Therefore the lower *non-zero* block contains a non-zero entry α_2 in the same column as β_1. And again the row of α_2 will intersect a counter-diagonal entry β_2 of a *counter-diagonal* block. In exactly the same way we are able to find a sequence of entries $\alpha_3, \beta_3, \alpha_4, \beta_4...$ until we reach the lowest *non-zero* block. Note that all these entries belong to distinct blocks, and because there is a finite number of blocks, the series is finite. In particular, let w be the number of *counter-diagonal* blocks that lie below or intersect the row of α_1, then we can find $\beta_1, ..., \beta_w$ and $\alpha_1, ..., \alpha_{w+1}$, where α_{w+1} lies in the lowest *non-zero* block. Now suppose for the sake of contradiction that α_1 is non-zero, set $\alpha_1, ..., \alpha_{w+1}$ to 1, set all counter-diagonal entries of all

counter-diagonal blocks except $\beta_1, ..., \beta_w$ to 1, and set all other entries to 0. This produces a matrix of rank $r+1$ because all $r+1$ 1's appears in distinct columns and rows, which contradicts the fact that $\bar{U}_A$ is rank maximized.

Hence all zero*-label blocks are indeed zero. In particular, after the permutations, the block $U_A(i,j), t < i \leq |A|, r-t+1 \leq j \leq y$ is all zero. Now partition A into $A_1 \cup A_2$, where A_2 is the subset of edges corresponding to the first t rows of the permuted U_A. So $|A_2| = t$ and $|A_1| = |A| - t$. But the zero constraints in U_A imply that f_{A_1} contains $r-t$ of the B_j's corresponding to the first $r-t$ columns, hence $|f_{A_1}| = r-t$. Finally $|f_{A_1}| + |A_2| = r$. □

Corollary 16.2.2. *Partition A into $A_1 \cup A_2$ as in Lemma 16.2.3. Further partition A_1 as $A_F \cup A_B$, where $A_F \subset \{e_1^{fwd}, ..., e_x^{fwd}\}$, $A_B \subset \{e_1^{bwd}, ..., e_y^{bwd}\}$, then $H(f_{A_F}|f_{A_B}) \leq$ rank$(\bar{U}_A) - |A_B| - |A_2|$.*

Proof. Suppose for contradiction that $H(f_{A_F}|f_{A_B}) >$ rank$(\bar{U}_A) - |A_B| - |A_2|$, then $|f_{A_F} \backslash f_{A_B}| \geq H(f_{A_F} \backslash f_{A_B}) \geq H(f_{A_F}|f_{A_B}) >$ rank$(\bar{U}_A) - |A_B| - |A_2|$. This implies $|f_{A_1}| >$ rank$(\bar{U}_A) - |A_2|$, a contradiction to Lemma 16.2.3. □

Due to the cyclic nature of $\mathcal{G}$ (i.e., if $\mathcal{G}$ is acyclic then there are no backward edges and traditional cut-set bounds apply), imposing delay constraints on the edges is necessary to avoid stability and causality issues. It suffices to assume that there is a unit delay on edges $e_1^{\text{fwd}}, ..., e_x^{\text{fwd}}, e_1^{\text{bwd}}, ..., e_y^{\text{bwd}}$. Note that any realistic systems should comply with these minimal delay constraints, e.g., it is not possible that a forward signal F_i is a causal output depending on a backward signal B_j, while B_j is also a causal output depending on F_i. Let t be a time index, denote $F_i[t]$ and $B_j[t]$ as the signals transmitted on edges e_i^{fwd} and e_j^{bwd} during the t-th time step. Consider an arbitrary secure coding scheme that finishes within T time steps. Below we show that the rate of this code is upper bounded by $x +$ rank$(\bar{U}_A) - |A|$, $\forall A \in \mathcal{A}$, as claimed in Theorem 16.2.1. We first prove a lemma.

Lemma 16.2.4. *Consider arbitrary random variables X, Y, Z, W, if*

$$(Z, W) \to (Y, W) \to X,$$

then

$$H(X|Z,W) \geq H(X|W) - I(Y;X|W).$$

Proof. Note that $H(X,Y|W) = H(X|Y,W)+H(Y|W) = H(Y|X,W)+H(X|W)$. So $H(X|Y,W) = H(X|W)+H(Y|X,W)-H(Y|W) = H(X|W)-I(Y;X|W)$. Finally because $H(X|Z,W) \geq H(X|Y,W)$, we prove the claim. □

Proof (of Theorem 16.2.1). Define $\mathcal{F}[t] = \{F_1[t], ..., F_x[t]\}$ as all the forward signals at time t, and $\mathcal{B}[t] = \{B_1[t], ..., B_y[t]\}$ as all the backward signals. Let $\mathcal{F} = \{\mathcal{F}[1], ..., \mathcal{F}[T]\}$, $\mathcal{B} = \{\mathcal{B}[1], ..., \mathcal{B}[T]\}$. Consider any $A \in \mathcal{A}$, partition it into $A_1 + A_2$ as in Lemma 16.2.3 and partition A_1 into $A_F + A_B$ as in Corollary 16.2.2. Let $F_A[t] = \{F_i[t] : e_i^{\text{fwd}} \in A_F\}$ denote the signals transmitted on A_F at time t, and likewise let $B_A[t] = \{B_j[t] : e_j^{\text{bwd}} \in A_B\}$. Let $a = |A_F|, b = |A_B|, c = |A_2|$. Recall that $f_{A_F}[t]$ are the signals sent by all backward edges to the edges in A_F at time t, M is the source message, and K_d is all randomness generated by the terminal. Now we upper bound the message rate R_s. It follows,

$$\begin{aligned} TR_s = H(M) &\overset{(a)}{=} H(M|K_d) - H(M|\mathcal{F},\mathcal{B},K_d) \\ &= I(M;\mathcal{F},\mathcal{B}|K_d) \\ &= H(\mathcal{F},\mathcal{B}|K_d) - H(\mathcal{F},\mathcal{B}|M,K_d), \end{aligned} \tag{16.1}$$

where (a) is due to the decoding constraint and the fact that K_d is independent of M. We first study the first term in (16.1). Expand it according to the chain rule, we have

$$\begin{aligned} H(\mathcal{F},\mathcal{B}|K_d) &= H(\mathcal{F}[1], ..., \mathcal{F}[T], \mathcal{B}[1], ..., \mathcal{B}[T]|K_d) \\ &\overset{(b)}{=} \sum_{i=1}^{T} H(\mathcal{F}[i],\mathcal{B}[i]|\mathcal{F}[0...i-1],\mathcal{B}[0...i-1],K_d) \\ &\overset{(c)}{=} \sum_{i=1}^{T} H(\mathcal{F}[i]|\mathcal{F}[0...i-1],\mathcal{B}[0...i-1],K_d) \\ &\overset{(d)}{\leq} \sum_{i=1}^{T} H(\mathcal{F}[i]\backslash F_A[i]|\mathcal{F}[0...i-1],\mathcal{B}[0...i-1],K_d)+ \\ &\qquad\qquad H(F_A[i]|\mathcal{F}[0...i-1],\mathcal{B}[0...i-1],K_d) \\ &\overset{(e)}{\leq} T(x-a) + \sum_{i=1}^{T} H(F_A[i]|F_A[0...i-1],B_A[0...i-1]) \\ &\overset{(f)}{=} T(x-a) + \sum_{i=1}^{T} H(F_A[i]|F_A[0...i-1],B_A[0...i-1],M). \end{aligned} \tag{16.2}$$

Here (b) follows from the chain rule; (c) follows from the fact that $\mathcal{B}[i]$ is a function of the conditions; (d) follows from the chain rule and conditioning reduces entropy; (e) follows from the fact that conditioning reduces entropy; and (f) follows from the secrecy constraint, i.e., M is independent from $F_A[0...T], B_A[0...T]$. Next we deal with the second term in (16.1).

$$\begin{aligned} H(\mathcal{F}, \mathcal{B}|M, K_d) &\geq H(F_A[1...T], B_A[1...T]|M, K_d) \\ &= \sum_{i=1}^{T} H(F_A[i], B_A[i]|F_A[0...i-1], B_A[0...i-1], M, K_d) \\ &\geq \sum_{i=1}^{T} H(F_A[i]|F_A[0...i-1], B_A[0...i-1], M, K_d) \\ &\overset{(g)}{\geq} \sum_{i=1}^{T} H(F_A[i]|F_A[0...i-1], B_A[0...i-1], M) - \\ &\qquad I(f_{A_F}[0...i-1]; F_A[i]|F_A[0...i-1], B_A[0...i-1], M). \end{aligned} \tag{16.3}$$

Where (g) is due to Lemma 16.2.4 by regarding $F_A[i]$ as X; $f_{A_F}[0, ..., i-1]$ as Y; K_d as Z; and $M, F_A[0, ..., i-1], B_A[0, ..., i-1]$ as W. Note that indeed $F_A[i]$ learns everything it can about K_d from $f_{A_F}[0, ..., i-1]$. Substituting (16.2) and (16.3) into (16.1) yields,

$$TR_s \leq T(x-a) + \sum_{i=1}^{T-1} I(f_{A_F}[1...i]; F_A[i+1]|F_A[1...i], B_A[1...i], M). \tag{16.4}$$

Finally we bound the summation in the R.H.S. of (16.4). These terms characterize how the keys generated by the terminal at times $1, ..., i$ contribute to randomizing

(and therefore protecting) the forward signals transmitted at time $i+1$.

$$\begin{aligned}
&\sum_{j=1}^{T-1} I(f_{A_F}[1...j]; F_A[j+1]|F_A[1...j], B_A[1...j], M) \\
&\overset{(h)}{=} \sum_{j=1}^{T-1}\sum_{i=1}^{j} I(f_{A_F}[i]; F_A[j+1]|F_A[1...j], B_A[1...j], f_{A_F}[0...i-1], M) \\
&\overset{(i)}{=} \sum_{i=1}^{T-1}\sum_{j=i}^{T-1} I(f_{A_F}[i]; F_A[j+1]|F_A[1...j], B_A[1...j], f_{A_F}[0...i-1], M) \\
&\overset{(j)}{\leq} \sum_{i=1}^{T-1} I(f_{A_F}[i]; F_A[i+1]|F_A[1...i], B_A[1...i], f_{A_F}[0...i-1], M) + \\
&\qquad \sum_{i=1}^{T-2}\sum_{j=i+1}^{T-1} I(f_{A_F}[i]; F_A[j+1], B_A[j]|F_A[1...j], B_A[1...j-1], f_{A_F}[0...i-1], M) \\
&\overset{(k)}{=} \sum_{i=1}^{T-1} I(f_{A_F}[i]; F_A[i+1...T], B_A[i+1...T-1]|F_A[1...i], B_A[1...i], f_{A_F}[0...i-1], M) \\
&\overset{(l)}{\leq} \sum_{i=1}^{T-1} H(f_{A_F}[i]|B_A[i]) \\
&\overset{(m)}{\leq} (T-1)(\text{rank}(\bar{U}_A) - b - c) \qquad (16.5)
\end{aligned}$$

Here (h) follows from the chain rule for mutual information; (i) follows from changing the order of summation; (j) follows from the fact that $I(X;Y|Z) \leq I(X;Y,Z)$; (k) follows from the chain rule for mutual information; (l) follows from the definition of mutual information and conditioning reduces entropy; and (m) follows from Corollary 16.2.2. Finally substituting (16.5) into (16.4) we have

$$\begin{aligned}
R_S &\leq \frac{T(x-a+\text{rank}(\bar{U}_A)-b-c) - \text{rank}(\bar{U}_A) + b + c}{T} \\
&= \frac{T(x+\text{rank}(\bar{U}_A)-|A|) - \text{rank}(\bar{U}_A) + b + c}{T} \\
&< x + \text{rank}(\bar{U}_A) - |A|.
\end{aligned}$$

□

16.3 Achievability

In this section we construct a scalar linear code that achieves the upper bound of Theorem 16.2.1 in $\bar{\mathcal{G}}$, thereby finding the secrecy capacity of $\bar{\mathcal{G}}$. The achievability result also implies that the upper bound in Theorem 16.2.1 is the tightest possible

if the bound only takes cuts and their connectivity matrices as input. We will build the code on top of $\bar{C}$, with the idea that $\bar{C}$ is rank maximized and therefore suggests an "optimal" way of using the backward keys (i.e., terminal generated randomness) to provide maximum randomization and protection. Hence what remains to be designed is the forward keys (source generated randomness that is independent of the message), and it turns out that $k_f = \max_{A\in\mathcal{A}} |A| - \text{rank}(\bar{U}_A)$ units of forward keys are sufficient in $\bar{\mathcal{G}}$, implying that a rate of $R_s = x - k_f = x + \min_{A\in\mathcal{A}} \text{rank}(\bar{U}_A) - |A|$ can be achieved.

For the ease of presentation we assume that there is no delay in $\bar{\mathcal{G}}$, and construct a code that achieves the capacity exactly. We will show later that extending this code to networks with delay is straightforward, and in this case it achieves the capacity asymptotically.

Let $M_1, \cdots, M_{R_s}$ be the messages, $K_s^1, \cdots, K_s^{k_f}$ be the source generated keys, K_d^1, $\cdots$, K_d^y be the terminal generated keys, all of them are i.i.d. uniformly distributed in $\mathbb{F}_q$. Let $E = (e_{ij}) \in \mathbb{F}_q^{(x+y)\times(x+y)}$ be the encoding matrix, defined by

$$E = \left(\begin{array}{c|c} \dfrac{G}{\mathbf{0}} & \bar{C} \end{array} \right), \tag{16.6}$$

where G is a random matrix of size $x \times x$ with entries i.i.d. uniformly chosen from $\mathbb{F}_q$, and $\mathbf{0}$ is a zero matrix of size $y \times x$. Then the signals transmitted on the cut are

$$\begin{pmatrix} F_1 \\ \vdots \\ F_x \\ B_1 \\ \vdots \\ B_y \end{pmatrix} = E \begin{pmatrix} M_1 \\ \vdots \\ M_{R_s} \\ K_s^1 \\ \vdots \\ K_s^{k_f} \\ K_d^1 \\ \vdots \\ K_d^y \end{pmatrix}. \tag{16.7}$$

Notice that E is a full rank square matrix with high probability since G is generic and the bottom y rows of $\bar{C}$ are linearly independent. Therefore the terminal d can decode the messages and keys with high probability. We only need to show that the code is secure, i.e., any subset of $\{F_1, ..., F_x, B_1, ..., B_y\}$ eavesdropped by an adversary is independent of $\{M_1, ..., M_{R_s}\}$. Since linear independence implies

independence [79], it suffices to show that the row space of

$$E_{\text{Message}} = (I_{R_s} \mid \mathbf{0}_{R_s \times (k_f + y)})$$

and the row space of E_A intersect trivially for all $A \in \mathcal{A}$, where E_A is the submatrix of E formed by the rows that correspond to the edges in A. Let E^r be the submatrix of E by deleting the first R_s columns from E, then it suffices to show that any submatrix E^r_A, $A \in \mathcal{A}$ has full row rank. The following theorem shows this is true when q is sufficiently large.

Theorem 16.3.1. *The code in (16.7) is secure with probability at least* $1 - |\mathcal{A}| \frac{k_f(x+y)}{q}$.

Proof. As mentioned above it suffices to show that any E^r_A, $A \in \mathcal{A}$ has full row rank. Consider an arbitrary E^r_A, and notice that the last y columns of it is exactly $\bar{U}_A$. Assume that the A contains a forward edges and b backward edges. Then due to the structure of $\bar{C}$, the last b rows of $\bar{U}_A$ must be linearly independent. There exist $\text{rank}(\bar{U}_A) - b$ rows among the first a rows such that these rows and the last b rows together form a basis of the row space of $\bar{U}_A$. The remaining $|A| - \text{rank}(\bar{U}_A)$ rows of $\bar{U}_A$, all of them corresponding to forward edges, are in the linear span of this basis. Assuming without loss of generality that they are the first $|A| - \text{rank}(\bar{U}_A)$ rows (otherwise reorder the forward edges), we construct a matrix E^+_A as

$$E^+_A = \left(\begin{array}{c|c} I_{|A| - \text{rank}(\bar{U}_A)} & \\ \mathbf{0} & \bar{U}_A \end{array} \right).$$

Notice that $k_f = \max_{A' \in \mathcal{A}} |A'| - \text{rank}(\bar{U}'_A) \geq |A| - \text{rank}(\bar{U}_A)$, so the number of columns in E^+_A is not larger than the number of columns in E^r_A. We append $k_f + \text{rank}(\bar{U}_A) - |A|$ more zero columns to the left of E^+_A to obtain a matrix E^{++}_A which is of the same size as E^r_A. Note that E^{++}_A has full row rank and satisfies the zero block constraint as defined in (16.6). Hence E^{++}_A contains an $|A| \times |A|$ full rank submatrix, denoted by $\underline{E}_A$. Consider the polynomial matrix $\underline{E}_A[\boldsymbol{x}]$ as a submartix of E^r_A by regarding all random entries in E^r_A as indeterminates, then it follows that $\det(\underline{E}_A[\boldsymbol{x}])$ is not the zero polynomial because $\det(\underline{E}_A) \neq 0$. By the Schwartz-Zippel lemma, under the random selection of entries in E^r_A,

$$\Pr\{\det(\underline{E}_A[\boldsymbol{x}]) = 0\} \leq \frac{k_f |A|}{q}.$$

Finally by the union bound,

$$\begin{aligned}\Pr\left\{\bigcup_{A\in\mathcal{A}}\det(\underline{E}_A[\boldsymbol{x}])\neq 0\right\} &\geq 1-\sum_{A\in\mathcal{A}}\Pr\{\det(\underline{E}_A[\boldsymbol{x}])=0\}\\ &\geq 1-|\mathcal{A}|\frac{k_f(x+y)}{q}.\end{aligned}$$

The proof is complete. □

Extending the above coding scheme to networks with delay is straightforward. It suffices for the source to wait one time slot for the arrival of the first batch of keys, and then start transmitting normally. So the overhead is vanishing as we increase the time duration of the code.

Chapter 17

RATELESS NETWORK ERROR CORRECTION

17.1 Models

17.1.1 Network Model

Consider an instance $(\mathcal{G}, s, t, \mathcal{B})$ of the unicast network error correction problem. Recall from Section 13.4.3 that $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a directed graph, s is the source, t is the terminal and $\mathcal{B}$ is the collection of sets of error edges. In this section we focus on the case that $\mathcal{B}$ is uniform, i.e., $\mathcal{B}$ consists of all subsets of $\mathcal{E}$ of size z_e. Denote the min-cut of $\mathcal{G}$ by μ. Denote by $\bar{\mu}$ the number of out-going edges of s, so $\bar{\mu} \geq \mu$.

We assume that an arbitrary linear network code $\mathcal{C}$ is implemented in the network, such that $\mathcal{C}$ achieves the network capacity μ in the absence of errors[1]. $\mathcal{C}$ and μ are not known a priori to the source and the terminal. To transmit information over the network, the source generates a batch of $\bar{\mu}$ encoded packets of length n as input to the network, represented by a matrix $X \in \mathbb{F}_q^{\bar{\mu} \times n}$, where packets are rows. As the packets travel through the network, they undergo linear transforms defined by the network code $\mathcal{C}$. In the absence of adversarial errors, the terminal t will observe a matrix AX, where $A \in \mathbb{F}_q^{\mu \times \bar{\mu}}$ is the network transform matrix of rank μ. Note that A is not known to the source and the terminal. For notational convenience, we assume that μ and $\mathcal{C}$ do not change over time, whereas our results naturally generalize to the case that they do change.

17.1.2 Adversary Model

The adversary controls $z_e < \mu$ edges[2] in the network, modeled in the following way. For each compromised edge (u, v), the adversary injects an error packet so that the packet received by v from this edge is the addition (over $\mathbb{F}_q$) of the error packet and the packet originally transmitted on the edge. As the injected error packets travel through the network, they undergo linear transforms defined by the network code $\mathcal{C}$. The terminal receives the sum of the linearly transformed error packets and the linearly transformed X. More precisely, the terminal observes a matrix

[1] Otherwise, if $\mathcal{C}$ is not capacity-achieving, then some edges are redundant. Remove these edges and $\mathcal{C}$ is capacity achieving in the modified network.

[2] We consider a model where edges rather than nodes are corrupted. Corrupting a node is equivalent to corrupting the links outgoing from it.

$Y = AX + BZ$, where $B \in \mathbb{F}_q^{\mu \times z_e}$ is the network transform matrix (determined by the network code) from the compromised edges to t, and $Z \in \mathbb{F}_q^{z_e \times n}$ are the z_e injected error packets. The adversary may choose Z carefully in order to corrupt the communication between s and t. Note that z_e, Z and B are not known to the source and the terminal.

17.1.3 Throughput and Capacity

We call one *stage* as the transmission of one batch of encoded packets, i.e., a matrix X. For a scheme involving N stages, denote by X^N and Y^N the sequences of matrices transmitted by s and received by t. Let M be a source message chosen from an alphabet $\mathcal{M}$ and let $\hat{M}$ be the output of the terminal decoder. If there exists a scheme that maps M to X^N, and maps Y^N to $\hat{M}$, such that for all $M \in \mathcal{M}$, $\Pr\{M \neq \hat{M}\} \to 0$ as $q \to \infty$, where the probability is taken over the coin flips of the encoder, decoder and adversary, then we say a throughput or rate of $\frac{\log_q |\mathcal{M}|}{Nn}$ is achievable. The capacity of the network is the supremum over all achievable rates.

17.2 Secret Channel Model

In this section we describe the rateless network error correction code for the secret channel model. We assume that there is a secure side channel between the source and the terminal and that the information transmitted by this channel is reliable and secure from the adversary. Non-rateless network error correction under this model is initially studied in [30]. The secret channel is helpful for two reasons. Firstly, it increases the capacity of the network. Specifically, the network capacity is $\mu - 2z_e$ without the secret channel and is $\mu - z_e$ with the channel [30]. Secondly, as we will discuss in more detail later, the secret channel facilitates verification of the received packets independently of the value of z_e and therefore allows rateless error correction. We require a scheme to use the secret channel to communicate at most a vanishingly small amount of information, as otherwise the problem is trivial.

At a high level, in the scheme the source incrementally sends more linearly dependent redundancy of the message through multiple stages. The message will be contained in the row space of the received packets after a sufficient number of stages. Additionally, the source sends a sequence of short hashes to help the decoder pinpoint the message from the row space. Below we describe the encoder for the source and the decoder for the terminal.

17.2.1 Encoding

Suppose the source wishes to transmit a message of b packets, each consisting of n symbols from $\mathbb{F}_q$, represented by a $b \times n$ matrix M over $\mathbb{F}_q$. The communication of M may last for several stages and during stage i, the source draws a random matrix K_i of size $\bar{\mu} \times b$ with entries i.i.d. uniformly distributed on $\mathbb{F}_q$. The source encodes $X_i = K_i M$, and inputs X_i to the network. Thereafter X_i undergoes the network transform as it travels through the network, as described in Section 17.1.1 and 17.1.2.

Next we discuss the construction of the hash. Let $\alpha_1 = b\bar{\mu} + t$, $t \geq 1$ and $\alpha_i = b\bar{\mu}$, $i > 1$. During the i-th stage, the source draws α_i random symbols $r_1, \cdots, r_{\alpha_i}$ from $\mathbb{F}_q$ privately, and constructs a Vandermonde matrix $D = (d_{kj}) \in \mathbb{F}_q^{n \times \alpha_i}$, where $d_{kj} = (r_j)^{k-1}$. The source computes $H_i = MD_i$, and sends H_i as well as $r_1, \cdots, r_{\alpha_i}$ to the terminal using the secret channel. The size of the secret is $\alpha_i(b+1)$, which is asymptotically negligible in n.

17.2.2 Decoding

During the i-th stage the terminal receives a batch of packets from the network $Y_i = AX_i + B_i Z_i$ corresponding to X_i. Let

$$Y^{(i)} = \begin{bmatrix} Y_1 \\ \vdots \\ Y_i \end{bmatrix}. \tag{17.1}$$

And let

$$H^{(i)} = [H_1 \cdots H_i] \tag{17.2}$$

$$D^{(i)} = [D_1 \cdots D_i]. \tag{17.3}$$

The decoder tries to solve the following system of linear equations:

$$X^s Y^{(i)} D^{(i)} = H^{(i)}, \tag{17.4}$$

where X^s is a $b \times i\mu$ matrix. If (17.4) has a unique solution for X^s, the terminal decodes $M = X^s Y^{(i)}$. If there exists no solution for (17.4), the terminal waits another stage for more redundancy. Otherwise, if there are multiple solutions for (17.4), the terminal declares a decoding failure.

17.2.3 Analysis

In this subsection we analyze the probability of decoding error of the proposed scheme. If decoding is not successful then either of the following two error events

must happen: 1) there does not exist X^s such that $X^s Y^{(i)} = M$; or 2) there exists a X^{s*}, such that X^{s*} is a solution to (17.4) and $X^{s*}Y^{(i)} \neq M$.

We study the probability of the first error event. We start with a useful lemma.

Lemma 17.2.1. *Let A be an arbitrary $u \times v$ matrix with rank u and $\boldsymbol{k}$ be a length-v column vector with entries uniformly i.i.d. distributed over $\mathbb{F}_q$, then $A\boldsymbol{k}$ is a random vector with entries uniformly i.i.d. distributed over $\mathbb{F}_q$.*

Proof. Consider an arbitrary length-u column vector $\boldsymbol{y} \in \mathbb{F}_q^u$, and the system of equations $A\boldsymbol{x} = \boldsymbol{y}$. Because A has full row-rank, its columns span F_q^u. So $A\boldsymbol{x} = \boldsymbol{y}$ has a solution $\boldsymbol{x}^*$ and the set of solutions is $\{\boldsymbol{x}^* + \boldsymbol{x}_0 : \boldsymbol{x}_0 \in \mathfrak{N}(A)\}$, where $\mathfrak{N}(A)$ is the null space or kernel of A. Therefore the size of the set of solutions is $|\mathfrak{N}(A)| = q^{v-u}$, which does not depend on $\boldsymbol{y}$. So $\Pr\{A\boldsymbol{k} = \boldsymbol{y}\} = q^{v-u}/q^v = 1/q^u$, and $A\boldsymbol{k}$ is uniformly distributed over $\mathbb{F}_q^u$. This proves the lemma. □

Recall that $Y_i = AX_i + B_i Z_i$. Define for notational convenience that

$$Z^{(i)} = \begin{bmatrix} Z_1 \\ \vdots \\ Z_i \end{bmatrix}, \tag{17.5}$$

and

$$T^{(i)} = \left[\begin{array}{c|cccc} AK_1 & B_1 & 0 & ... & 0 \\ AK_2 & 0 & B_2 & ... & 0 \\ \vdots & \vdots & & & \\ AK_i & 0 & 0 & ... & B_i \end{array}\right] = \left[A^{(i)} \mid B^{(i)}\right]. \tag{17.6}$$

Then it follows from the network transform that

$$Y^{(i)} = T^{(i)} \begin{bmatrix} M \\ Z^{(i)} \end{bmatrix}. \tag{17.7}$$

Lemma 17.2.2. *If $i(\mu - z_e) \geq b$, then $T^{(i)}$ has full column rank with probability (over the distribution of the K_j's) at least $1 - \frac{1}{q-1}$.*

Proof. Without loss of generality we assume all B_k, $k = 1, ..., i$ have full column ranks z_e, otherwise we can select a basis of the column space of B_k and reformulate the problem with a smaller z_e. Therefore $B^{(i)}$ has rank iz_e. By Lemma 17.2.1, all the entries in $A^{(i)}$ are uniformly i.i.d. distributed. Now consider the first b columns

of $T^{(i)}$. For $k = 1, ..., b$, the probability that the k-th column of T^i is in the linear span of the $iz_e + b - k$ columns after it is equal to $q^{iz_e+b-k}/q^{i\mu}$. Therefore, by the union bound,

$$\begin{aligned}
&\Pr\{T^{(i)} \text{ is not full column rank}\} \\
&\qquad = \Pr\{\bigcup_{k=1}^{b} k\text{-th column in the span of the columns behind}\} \\
&\qquad \leq \sum_{k=1}^{b} \Pr\{k\text{-th column in the span of the columns behind}\} \\
&\qquad = \sum_{k=1}^{b} \frac{q^{iz_e+b-k}}{q^{i\mu}} \\
&\qquad = \frac{q^{iz_e+b-i\mu} - q^{iz_e-i\mu}}{q-1} \leq \frac{1}{q-1}.
\end{aligned}$$

This completes the proof. □

The following result is well-known.

Lemma 17.2.3. *A matrix is left-invertible if and only if it has full column rank.*

Corollary 17.2.1 bounds the probability of the first kind of error.

Corollary 17.2.1. *If $i(\mu - z_e) \geq b$, then with probability at least $1 - 1/(q-1)$, there exists matrix X^s such that $M = X^s Y^{(i)}$.*

Proof. By Lemma 17.2.2, $T^{(i)}$ has full column rank with probability at least $1 - 1/(q-1)$. By Lemma 17.2.3, if $T^{(i)}$ has full column rank, then there exists a matrix V^s such that $V^s T^{(i)} = I_{b+iz_e}$. Therefore by (17.7),

$$V^s Y^{(i)} = \begin{bmatrix} M \\ Z^{(i)} \end{bmatrix}.$$

Let X^s be the first b rows of V^s and we have $X^s Y^{(i)} = M$. □

Next we study the second kind of error events.

Lemma 17.2.4. *For any $M^* \neq M$, the probability (over the distribution of the r_j's) that $M^* D^{(i)} = H^{(i)}$ is upper bounded by $(n/q)^{\sum_{k=1}^{i} \alpha_k}$.*

Proof. It is equivalent to consider the probability that $(M^* - M)D^{(i)} = 0$. Since $M^* - M \neq 0$, there is at least one row in which M^* differs from M. Denote this row of $M^* - M$ as $(x_1, ..., x_n)$, then the j-th entry of the corresponding row of $(M^* - M)D^{(i)}$ is $F(r_j) = \sum_{k=1}^{n} x_k r_j^{k-1}$. Because $F(r_j)$ is not the zero polynomial and has at most $n - 1$ roots, the probability (over r_j) that $F(r_j) = 0$ is at most n/q. Because $D^{(i)}$ has $\sum_{k=1}^{i} \alpha_k$ columns, and all r_j, $1 \leq j \leq \sum_{k=1}^{i} \alpha_k$, are independently chosen, the probability that the entire row is a zero vector is at most $(n/q)^{\sum_{k=1}^{i} \alpha_k}$. This is an upper bound of the probability that the entire matrix of $(M^* - M)D^{(i)}$ is zero. □

Lemma 17.2.5. *The probability that there exists X^{s*} such that $X^{s*}Y^{(i)} \neq M$ but $X^{s*}Y^{(i)}D^{(i)} = H^{(i)}$ is upper bounded by $n^{\sum_{k=1}^{i} \alpha_k}/q^{-ib\mu + \sum_{k=1}^{i} \alpha_k}$.*

Proof. Note that X^{s*} is a matrix of size $b \times i\mu$ over $\mathbb{F}_q$. So the claim follows from Lemma 17.2.4 and the union bound over all possible choices of X^{s*}. □

We are now ready to prove the error performance of the proposed scheme.

Theorem 17.2.1. *Let $\mathfrak{i}$ be the smallest integer such that $\mathfrak{i}(\mu - z_e) \geq b$, i.e., $\mathfrak{i} = \lceil \frac{b}{\mu - z_e} \rceil$, then the terminal is able to decode M correctly after collecting packets for $\mathfrak{i}$ stages, with probability at least $1 - 1/(q-1) - \mathfrak{i} \cdot n^{t+\mathfrak{i}b\bar{\mu}}/q^t$.*

Proof. By Lemma 17.2.5, for any stage $k \leq \mathfrak{i}$, the probability of the second kind of error is upper bounded by $n^{t+kb\bar{\mu}}/q^{t+kb(\bar{\mu}-\mu)} \leq n^{t+\mathfrak{i}b\bar{\mu}}/q^t$. By Corollary 17.2.1, at stage $\mathfrak{i}$ the probability of the first kind of error is upper bounded by $1/(q-1)$. The theorem then follows from the union bound. □

We remark that $\mathfrak{i}(\mu - z_e) \geq b$ is in fact the cut-set bound, i.e., the number of packets received is larger than or equal to the number of message packets plus the number of injected error packets. Therefore Theorem 17.2.1 suggests that the scheme is rate-optimal in the sense that the terminal will decode correctly with high probability after collecting packets for the least possible number of stages.

17.3 Shared Secret Model

In this section we describe rateless network error correction under the shared secret model. Formally, we assume that the source and the terminal share a sequence of symbols i.i.d. uniformly distributed over $\mathbb{F}_q$. The symbols are drawn secretly from the adversary, and are independent of the source message M. Non-rateless network

error correction under this model is initially studied in [31]. We remark that the shared secret is a strictly weaker resource than the secret channel of the previous section. However, the shared secret model is arguably much more realistic than the secret channel model, as the secret can be shared in advance.

Compared to the secret channel model, a challenge for the shared secret model is that the shared secret is independent of the message. Another challenge is that there is no naive way to communicate the hash reliably and securely. A main idea in our scheme is to transmit the hash in the insecure network, but with strong redundancy. Since the hashes are short, the induced overhead is small and is negligible in the packet length. Below we describe the encoder for the source and the decoder for the terminal.

17.3.1 Encoding

Again, suppose that the source wishes to transmit a message of b packets represented by a $b \times n$ matrix M over $\mathbb{F}_q$. As before, during stage i, the source draws a random matrix K_i of size $\bar{\mu} \times b$, encodes $X_i = K_i M$, and inputs X_i to the network. Thereafter X_i undergoes the network transform as it travels through the network, as described in Section 17.1.1 and 17.1.2.

Next we discuss the construction of the hash. Recall that the vectorization of a matrix is a linear transformation which converts the matrix into a column vector by stacking the columns of the matrix on top of one another. Let the column vector $\boldsymbol{m} \in \mathbb{F}_q^{bn}$ be the vectorized M. Let $\alpha_1 = \bar{\mu}(b+1) + t$, $t \geq 1$ and $\alpha_i = \bar{\mu}(b+1)$, $i > 1$ be the length of the hash constructed at the i-th stage. The source draws α_i symbols $\boldsymbol{r}_i = (r_1, ..., r_{\alpha_i})$, and another α_i symbols $\boldsymbol{h}_i = (h_1, ..., h_{\alpha_i})$ uniformly i.i.d. distributed over $\mathbb{F}_q$, from the shared secret randomness. Let $D_i \in \mathbb{F}_q^{\alpha_i \times nb}$ be the matrix whose (u, v)-th entry equals r_u^v, then compute the length-α_i column vector

$$\boldsymbol{l}_i = \boldsymbol{h}_i - D_i \boldsymbol{m}, \tag{17.8}$$

which is the hash of message $\boldsymbol{m}$. To communicate $\boldsymbol{l}_i$, during the i-th stage, the source draws a random vector $\bar{K}_i \in \mathbb{F}_q^{\bar{\mu} \times 1}$ with entries i.i.d uniformly distributed over $\mathbb{F}_q$. It then encodes $\bar{X}_i = \bar{K}_i \boldsymbol{l}_i^T$, and inputs $\bar{X}_i$ into the network. Alternatively, the source may include $\bar{X}_i$ as a small header when it sends X_i. This concludes the operations of the encoder. Thereafter $\bar{X}_i$ travels through the network and undergoes the network transform described in Sections 17.1.1 and 17.1.2.

For stage i, the number of the shared random secret symbols required is $2\alpha_i$, which is asymptotically negligible in n. The communication overhead, i.e., the number of hash-related transmissions, is the length of $\boldsymbol{l}_i$, which equals α_i. Again this is asymptotically negligible in n. The computational cost of encoding is dominated by the operation of computing $\boldsymbol{l}_i$ from (17.8), which is bounded by $O(\bar{\mu}b^2n)$.

We remark that the hashing scheme may be understood in the following way. In (17.8) if we regard $D_i\boldsymbol{m}$ as the hash of $\boldsymbol{m}$ with respect to a check matrix D_i, then $\boldsymbol{l}_i$ is the one-time padded version of $D_i\boldsymbol{m}$.

17.3.2 Decoding

During the i-th stage the terminal receives a batch of packets from the network $Y_i = AX_i + B_iZ_i$ corresponding to X_i. The terminal also receives a batch of packets $\bar{Y}_i = \bar{A}\bar{X}_i + \bar{B}_i\bar{Z}_i$ corresponding to $\bar{X}_i$ (alternatively if $\bar{X}_i$ is the header of X_i, then $\bar{Y}_i$ is the header of the received packets), where $\bar{A}$, $\bar{B}_i$ and $\bar{Z}_i$ are defined similarly as A, B_i and Z_i, cf. Section 17.1.1 and 17.1.2.

The decoder constructs a matrix $P_{i,1}$ from the shared randomness as:

$$P_{i,1} = \begin{bmatrix} D_1 & I_{\alpha_1} & 0 & ... & 0 \\ D_2 & 0 & I_{\alpha_2} & ... & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ D_i & 0 & 0 & ... & I_{\alpha_i} \end{bmatrix},$$

where I_{α_j} is the identity matrix of order α_j. Denote by $\otimes$ the Kronecker product, the decoder constructs another matrix $P_{i,2}$ from the received packets:

$$P_{i,2} = \begin{bmatrix} \left(Y^{(i)}\right)^T \otimes I_b & 0 & \cdots & 0 \\ 0 & \bar{Y}_1^T & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \bar{Y}_i^T \end{bmatrix},$$

where I_b is the identity matrix of order b, and

$$Y^{(i)} = \begin{bmatrix} Y_1 \\ \vdots \\ Y_i \end{bmatrix}. \tag{17.9}$$

Let $P_i = P_{i,1}P_{i,2}$ be a matrix of size $t + i\bar{\mu}(b+1) \times i\mu(b+1)$. Then the decoder solves the system of equations (17.10) in variables $\boldsymbol{x}^s$ and $\bar{\boldsymbol{x}}_k^s$, $k = 1, ..., i$, where

$\boldsymbol{x}^s$ is a column vector of length $ib\mu$ and the $\bar{\boldsymbol{x}}_j^s$'s are column vectors of length μ.

$$P_i \begin{bmatrix} \boldsymbol{x}^s \\ \bar{\boldsymbol{x}}_1^s \\ \vdots \\ \bar{\boldsymbol{x}}_i^s \end{bmatrix} = \begin{bmatrix} \boldsymbol{h}_1 \\ \vdots \\ \boldsymbol{h}_i \end{bmatrix}. \tag{17.10}$$

If (17.10) is not uniquely solvable, i.e., if it has no solution or if there are multiple solutions, the terminal postpones decoding to the next stage so that it will receive more redundancy. If (17.10) is uniquely solvable, unvectorize $\boldsymbol{x}^s$ into a $b \times i\mu$ matrix X^s by rearranging every length-b segment of $\boldsymbol{x}^s$ as a column of X^s. Then the source message is recovered as:

$$\hat{M} = X^s Y^{(i)}. \tag{17.11}$$

Note that the size of X^s is much smaller than the size of $\hat{M}$. To improve computational efficiency, we have used X^s as a proxy for solving $\hat{M}$, i.e., first solve X^s from (17.10) and then obtain $\hat{M}$ from (17.11). Particularly, the computational complexity of solving X^s from (17.10) is bounded by $O(i^3 b^3 \bar{C}^3)$, and is asymptotically negligible in n. The computational cost of decoding is dominated by the matrix multiplication to obtain P_i, which is bounded by $O(i^2 \bar{C}^2 b^2 n)$.

17.3.3 Analysis

In this subsection we analyze the probability of error of the proposed scheme. If decoding is not successful then either of the following two error events must happen: 1) there does not exist X^s such that $X^s Y^{(i)} = M$ and $\boldsymbol{x}^s$ is a solution to (17.10); or 2) there exists X^{s*}, such that $X^{s*} Y^{(i)} \neq M$ and $\boldsymbol{x}^{s*}$ is a solution to (17.10), where $\boldsymbol{x}^{s*}$ is the vectorized X^{s*}.

Lemma 17.3.1 bounds the probability of the first kind of error.

Lemma 17.3.1. *If $i(\mu - z_e) \geq b$, then with probability at least $1 - (i+1)/(q-1)$, there exists a matrix X^s and vectors $\bar{\boldsymbol{x}}_k^s$, $k = 1, ..., i$ such that $M = X^s Y^{(i)}$ and $\boldsymbol{x}^s$, $\bar{\boldsymbol{x}}_k^s$, $k = 1, ..., i$ is a solution to (17.10).*

Proof. By Corollary 17.2.1, the probability that there does not exist X^s such that $X^s Y^{(i)} = M$ is upper bounded by $1/(q-1)$. Because $C > z_e$, by the same argument, for $k = 1, ..., i$ the probability that there does not exist $\bar{\boldsymbol{x}}_k^s$ such that

$\boldsymbol{l}_k = \bar{Y}_k^T \bar{\boldsymbol{x}}_k^s$ is upper bounded by $1/(q-1)$. We next verify that the above $\boldsymbol{x}^s$ and $\bar{\boldsymbol{x}}_k^s$, $k = 1, ..., i$, if exist, are a solution to (17.10). Vectorizing $X^s Y^{(i)} = M$ we obtain

$$\boldsymbol{m} = \left(\left(Y^{(i)} \right)^T \otimes I_b \right) \boldsymbol{x}^s. \tag{17.12}$$

Substitute (17.12) and $\boldsymbol{l}_k = \bar{Y}_k^T \bar{\boldsymbol{x}}_k^s$ into (17.10) we obtain

$$P_i \begin{bmatrix} \boldsymbol{x}^s \\ \bar{\boldsymbol{x}}_1^s \\ \vdots \\ \bar{\boldsymbol{x}}_i^s \end{bmatrix} = \begin{bmatrix} D_1 & I_{\alpha_1} & 0 & ... & 0 \\ D_2 & 0 & I_{\alpha_2} & ... & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ D_i & 0 & 0 & ... & I_{\alpha_i} \end{bmatrix} \begin{bmatrix} \boldsymbol{m} \\ \boldsymbol{l}_1 \\ \vdots \\ \boldsymbol{l}_i \end{bmatrix} = \begin{bmatrix} \boldsymbol{h}_1 \\ \vdots \\ \boldsymbol{h}_i \end{bmatrix}, \tag{17.13}$$

where the second equality follows from (17.8). Finally, apply the union bound and the lemma is proved. □

Next we study the second kind of error events.

Lemma 17.3.2. *For any $(\boldsymbol{m}^*, \boldsymbol{l}_1^*, ..., \boldsymbol{l}_i^*)$ such that $\boldsymbol{m}^* \neq \boldsymbol{m}$, the probability (over the distribution of the $\boldsymbol{r}_j$'s) that*

$$\begin{bmatrix} D_1 & I_{\alpha_1} & 0 & ... & 0 \\ D_2 & 0 & I_{\alpha_2} & ... & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ D_i & 0 & 0 & ... & I_{\alpha_i} \end{bmatrix} \begin{bmatrix} \boldsymbol{m}^* \\ \boldsymbol{l}_1^* \\ \vdots \\ \boldsymbol{l}_i^* \end{bmatrix} = \begin{bmatrix} \boldsymbol{h}_1 \\ \vdots \\ \boldsymbol{h}_i \end{bmatrix}, \tag{17.14}$$

is upper bounded by $(nb/q)^{\sum_{k=1}^i \alpha_k}$.

Proof. By (17.8), the event that $(\boldsymbol{m}^*, \boldsymbol{l}_1^*, ..., \boldsymbol{l}_i^*)$ is a solution of (17.14) is equivalent to the event that

$$\begin{bmatrix} D_1 & I_{\alpha_1} & 0 & ... & 0 \\ D_2 & 0 & I_{\alpha_2} & ... & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ D_i & 0 & 0 & ... & I_{\alpha_i} \end{bmatrix} \begin{bmatrix} \boldsymbol{m}^* - \boldsymbol{m} \\ \boldsymbol{l}_1^* - \boldsymbol{l}_1 \\ \vdots \\ \boldsymbol{l}_i^* - \boldsymbol{l}_i \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{17.15}$$

Denote

$$\begin{bmatrix} \boldsymbol{m}^* - \boldsymbol{m} \\ \boldsymbol{l}_1^* - \boldsymbol{l}_1 \\ \vdots \\ \boldsymbol{l}_i^* - \boldsymbol{l}_i \end{bmatrix} = (\Delta m_1, ..., \Delta m_{nb}, \Delta l_1, ..., \Delta l_{\sum_{k=1}^i \alpha_k})^T,$$

and denote by d_{j1} the $(j,1)$-th entry of the matrix $[D_1^T, ..., D_i^T]^T$. Consider the j-th row of (17.15), by the construction of the D_i's, this row is equivalent to

$$\sum_{k=1}^{nb} \Delta m_k d_{j1}^k + \Delta l_j = 0. \tag{17.16}$$

Note that (17.16) is a non-zero polynomial in variable d_{j1} of degree at most nb. By the fundamental theorem of algebra, this polynomial has at most nb roots. By construction, d_{j1} is uniformly distributed over $\mathbb{F}_q$ and so the probability that (17.16) follows is upper bounded by nb/q. For $j = 1, ..., \sum_{k=1}^{i} \alpha_k$, because the d_{j1}'s are i.i.d. distributed, the probability that all $\sum_{k=1}^{i} \alpha_k$ equations in (17.15) hold is at most $(nb/q)^{\sum_{k=1}^{i} \alpha_k}$. □

Lemma 17.3.3. *The probability (over the distribution of the $\boldsymbol{r}_j$'s) that there exists a solution $(\boldsymbol{x}^{s*}, \bar{\boldsymbol{x}}_1^{s*}, ..., \bar{\boldsymbol{x}}_i^{s*})$ of (17.10) such that $X^{s*}Y^i \neq M$ is upper bounded by*

$$\frac{(nb)^{\sum_{k=1}^{i} \alpha_k}}{q^{-i\mu(b+1)+\sum_{k=1}^{i} \alpha_k}}. \tag{17.17}$$

Proof. Consider a column vector $(\boldsymbol{x}^{s*}, \bar{\boldsymbol{x}}_1^{s*}, ..., \bar{\boldsymbol{x}}_i^{s*})$ such that $X^{s*}Y^i \neq M$. Let

$$(\boldsymbol{m}^*, \boldsymbol{l}_1^*, ..., \boldsymbol{l}_i^*) = P_{i,2}(\boldsymbol{x}^{s*}, \bar{\boldsymbol{x}}_1^{s*}, ..., \bar{\boldsymbol{x}}_i^{s*}),$$

then $\boldsymbol{m}^* \neq \boldsymbol{m}$. It follows that,

$$\begin{aligned}
&\Pr\{\bigcup_{\boldsymbol{x}^{s*}: X^{s*}Y^i \neq M} (\boldsymbol{x}^{s*}, \bar{\boldsymbol{x}}_1^{s*}, ..., \bar{\boldsymbol{x}}_k^{s*}) \text{ is a solution of } (17.10)\} \\
&\qquad \overset{(a)}{\leq} \sum_{\boldsymbol{x}^{s*}: X^{s*}Y^i \neq M} \Pr\{(\boldsymbol{x}^{s*}, \bar{\boldsymbol{x}}_1^{s*}, ..., \bar{\boldsymbol{x}}_k^{s*}) \text{ is a solution of } (17.10)\} \\
&\qquad \overset{(b)}{\leq} q^{ib\mu} q^{i\mu} \left(\frac{nb}{q}\right)^{\sum_{k=1}^{i} \alpha_k} \\
&\qquad = \frac{(nb)^{\sum_{k=1}^{i} \alpha_k}}{q^{-i\mu(b+1)+\sum_{k=1}^{i} \alpha_k}},
\end{aligned}$$

where (a) follows from the union bound. To prove the inequality of (b), notice that if $(\boldsymbol{x}^{s*}, \bar{\boldsymbol{x}}_1^{s*}, ..., \bar{\boldsymbol{x}}_i^{s*})$ is a solution to (17.10), then $(\boldsymbol{m}^*, \boldsymbol{l}_1^*, ..., \boldsymbol{l}_i^*)$ is a solution to (17.14). Therefore (b) follows from Lemma 17.3.2. This completes the proof. □

We are ready to prove the error performance of the proposed scheme.

Theorem 17.3.1. *Let* $\mathfrak{i}$ *be the smallest integer such that* $\mathfrak{i}(\mu - z_e) \geq b$*, i.e.,* $\mathfrak{i} = \lceil \frac{b}{\mu - z_e} \rceil$*, then the terminal is able to decode* M *correctly after collecting packets for* $\mathfrak{i}$ *stages, with probability at least*

$$1 - \frac{\mathfrak{i}+1}{q-1} - \frac{\mathfrak{i} \cdot (nb)^{t+\mathfrak{i}(b+1)\bar{\mu}}}{q^t}. \tag{17.18}$$

Proof. By Lemma 17.3.3, for stage $k \leq \mathfrak{i}$, the probability of the second kind of error is upper bounded by $(nb)^{t+k(1+b)\bar{\mu}}/q^{t+k(b+1)(\bar{\mu}-\mu)} \leq (nb)^{t+\mathfrak{i}(1+b)\bar{\mu}}/q^t$. By Lemma 17.3.1, at stage $\mathfrak{i}$ the probability of the first kind of error is upper bounded by $1 - (\mathfrak{i}+1)/(q-1)$. The theorem then follows from the union bound. □

As in the secret channel model, we remark that $\mathfrak{i}(\mu - z_e) \geq b$ is in fact the cut-set bound. Therefore Theorem 17.3.1 implies that the scheme is rate-optimal in the sense that the terminal will decode correctly with high probability after collecting packets for the least possible number of stages.

Chapter 18

CONCLUDING REMARKS

In this part we present reductions that map an arbitrary multiple-unicast network coding instance to a unicast secure network coding instance in which at most one link is eavesdropped, or a unicast network error correction instance in which at most one link is erroneous, such that a rate tuple is achievable in the multiple-unicast network coding instance if and only if a corresponding rate is achievable in the unicast secure network coding instance, or in the unicast network error correction instance. Our reductions show that solving seemingly very simple instances of the secure network coding problem or of the network error correction problem are in fact as hard as solving the multiple-unicast network coding problem which is a central open problem. Conversely, we show that an arbitrary unicast secure network coding instance in which at most one link is eavesdropped can be reduced back to a multiple-unicast network coding instance, implying an equivalence between the two problems. In addition, we show that the capacity of a unicast network error correction instance in general is not exactly achievable.

While determining the secrecy capacity in the secure network coding problem is hard, we derive an upper bound on the secrecy capacity based on cut-sets and the connectivity of links. We show that the bound is as tight as possible given the input to the bound. Finally, we study code construction for the network error correction problem in the setting that any z_e links are subject to error. We present coding schemes that are rateless, i.e., that do not require prior knowledge of z_e and the network min-cut. The schemes will adapt to the correct parameters over multiple stages of communication and achieve the optimal rate.

Several problems are left open. It would be interesting to study whether the unicast secure network coding problem with more than one eavesdropped link can be reduced to the multiple-unicast network coding problem. Such a reduction, if exists, will imply that the unicast secure network coding problem with only one eavesdropped link is as hard as the general unicast secure network coding problem with an arbitrary number of eavesdropped links. We also leave open the possibility that the unicast network error correction problem can be reduced to the multiple-unicast network coding problem. Such a reduction would imply an equivalence

between the two problems. Finally, it is an interesting fact that in reducing multiple-unicast network coding to unicast network error correction, our construction works for both zero-error achievability and achievability with vanishing error, but not for asymptotic achievability. A natural question is addressing the existence of a reduction for asymptotic achievability.

BIBLIOGRAPHY

[1] R. Miller, *Inside Amazon's cloud computing infrastructure*, Data Center Frontier, 2015. [Online]. Available: http://datacenterfrontier.com/inside-amazon-cloud-computing-infrastructure/.

[2] M. Sathiamoorthy, M. Asteris, D. Papailiopoulos, A. G. Dimakis, R. Vadali, S. Chen, and D. Borthakur, "XORing Elephants: novel erasure codes for big data," *International Conference on Very Large Data Bases (VLDB)*, vol. 6, no. 5, pp. 325–336, 2013.

[3] D. A. Patterson, G. Gibson, and R. H. Katz, "A case for redundant arrays of inexpensive disks (RAID)," in *ACM SIGMOD*, vol. 17, 1988, pp. 109–116.

[4] P. M. Chen, E. K. Lee, G. a. Gibson, R. H. Katz, and D. a. Patterson, "RAID: high-performance, reliable secondary storage," *ACM Computing Surveys*, vol. 26, no. 2, pp. 145–185, 1994.

[5] C. Huang, H. Simitci, Y. Xu, A. Ogus, B. Calder, P. Gopalan, J. Li, and S. Yekhanin, "Erasure coding in Windows Azure storage," *USENIX Annual Technical Conference (ATC)*, 2012.

[6] A. Fikes, "Storage architecture and challenges," in *Google Faculty Summit*, 2010. [Online]. Available: https://cloud.google.com/files/storage_architecture_and_challenges.pdf.

[7] Z. Whittaker, *These were the biggest hacks, leaks and data breaches of 2016*, ZDNet, 2016. [Online]. Available: http://www.zdnet.com/pictures/biggest-hacks-security-data-breaches-2016/.

[8] N. Perlroth, *Yahoo says hackers stole data on 500 million users in 2014*, The New York Times, 2016. [Online]. Available: https://www.nytimes.com/2016/09/23/technology/yahoo-hackers.html.

[9] V. Goel and N. Perlroth, *Yahoo says 1 billion user accounts were hacked*, The New York Times, 2016. [Online]. Available: https://www.nytimes.com/2016/12/14/technology/yahoo-hack.html.

[10] J. K. Resch and J. S. Plank, "AONT-RS: blending security and performance in dispersed storage systems," in *USENIX Conference on File and Storage Technologies (FAST)*, 2011.

[11] A. Bessani, M. Correia, B. Quaresma, F. André, and P. Sousa, "Depsky: Dependable and secure storage in a cloud-of-clouds," *ACM Transactions on Storage*, vol. 9, no. 4, 12:1–12:33, 2013.

[12] H. Luo, P. Zerfos, J. Kong, S. Lu, and L. Zhang, "Self-securing ad hoc wireless networks," in *International Symposium on Computers and Communications*, 2002.

[13] R. J. Anderson, *Security engineering*. Wiley Publishing, 2008.

[14] M. W. Storer, K. M. Greenan, E. L. Miller, and K. Voruganti, "POTSHARDS - a secure, recoverable, long-term archival storage system," *ACM Transactions on Storage*, vol. 5, no. 2, pp. 1–35, 2009.

[15] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, 1979.

[16] G. R. Blakley and C. Meadows, "Security of ramp schemes," *Advances in Cryptology - CRYPTO*, vol. 196, pp. 242–268, 1985.

[17] M. Franklin and M. Yung, "Communication complexity of secure computation," in *ACM symposium on Theory of computing*, 1992, pp. 699–710.

[18] L. Xu, V. Bohossian, J. Bruck, and D. G. Wagner, "Low-density MDS codes and factors of complete graphs," *IEEE Transactions on Information Theory*, vol. 45, no. 6, pp. 1817–1826, 1999.

[19] M. Blaum, J. Brady, J. Bruck, and J. Menon, "EVENODD: an efficient scheme for tolerating double disk failures in RAID architectures," *IEEE Transactions on Computers*, vol. 44, no. 2, pp. 192–202, 1995.

[20] C. Huang and L. Xu, "STAR : an efficient coding scheme for correcting triple storage node failures," in *USENIX Conference on File and Storage Technologies (FAST)*, 2005, pp. 197–210.

[21] S. Pawar, S. E. Rouayheb, and K. Ramchandran, "Securing dynamic distributed storage systems against eavesdropping and adversarial attacks," *IEEE Transactions on Information Theory*, vol. 57, no. 10, pp. 6734–6753, 2011.

[22] N. B. Shah, K. V. Rashmi, and P. V. Kumar, "Information-theoretically secure regenerating codes for distributed storage," in *IEEE GLOBECOM*, 2011.

[23] A. S. Rawat, N. S. Onur Ozan Koyluoglu and, and S. Vishwanath, "Optimal locally repairable and secure codes for distributed storage systems," *IEEE Transactions on Information Theory*, vol. 60, no. 1, pp. 212–236, 2014.

[24] R. Tandon, S. Amuru, T. C. Clancy, and R. M. Buehrer, "Toward optimal secure distributed storage systems with exact repair," *IEEE Transactions on Information Theory*, vol. 62, no. 6, pp. 3477–3492, 2016.

[25] T. Cover and J. Thomas, *Elements of Information Theory*. Wiley-Interscience, 2005.

[26] R. W. Yeung, S. R. Li, N. Cai, and Z. Zhang., *Network Coding Theory*. Now Publishers, 2006.

[27] T. Chan and A. Grant, "Dualities between entropy functions and network codes," *IEEE Transactions on Information Theory*, vol. 54, no. 10, pp. 4470–4487, 2008.

[28] R. W. Yeung and N. Cai, "Network Error Correction, I: Basic Concepts and Upper Bounds," *Communications in Information & Systems*, vol. 6, no. 1, pp. 19–35, 2006.

[29] N. Cai and R. W. Yeung, "Network Error Correction, II: Lower Bounds," *Communications in Information & Systems*, vol. 6, no. 1, pp. 37–54, 2006.

[30] S. Jaggi, M. Langberg, S. Katti, T. Ho, D. Katabi, M. Medard, and M. Effros, "Resilient Network Coding in the Presence of Byzantine Adversaries," *IEEE Transactions on Information Theory*, vol. 54, no. 6, pp. 2596–2603, 2008.

[31] L. Nutman and M. Langberg, "Adversarial Models and Resilient Schemes for Network Coding," in *IEEE International Symposium on Information Theory (ISIT)*, 2008, pp. 171–175.

[32] J. Kurihara, S. Kiyomoto, K. Fukushima, and T. Tanaka, "A new (k, n)-threshold secret sharing scheme and its extension," *International Conference on Information Security*, 2008.

[33] C. Lv, X. Jia, L. Tiany, J. Jing, and M. Sun, "Efficient ideal threshold secret sharing schemes based on Exclusive-OR operations," *International Conference on Network and System Security*, pp. 136–143, 2010.

[34] M. Blaum and R. M. Roth, "New array codes for multiple phased burst correction," *IEEE Transactions on Information Theory*, vol. 39, no. 1, pp. 66–77, 1993.

[35] M. Blaum, J. Bruck, and A. Vardy, "MDS array codes with independent parity symbols," *IEEE Transactions on Information Theory*, vol. 42, no. 2, pp. 529–542, 1996.

[36] W. Huang and J. Bruck, "Secure RAID schemes for distributed storage," in *IEEE International Symposium on Information Theory (ISIT)*, 2016.

[37] ——, *Secure RAID schemes from EVENODD and STAR codes*, accepted to IEEE International Symposium on Information Theory (ISIT), 2017.

[38] A. Wyner, "The wire-tap channel," *Bell System Technical Journal*, 1975.

[39] F. J. MacWilliams and N. J. A. Sloane, *The theory of error-correcting codes*. North Holland Publishing, 1977.

[40] R. J. McEliece and D. V. Sarwate, "On sharing secrets and Reed-Solomon codes," *Communications of the ACM*, vol. 24, no. 9, pp. 583–584, 1981.

[41] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *Journal of the Society for Industrial and Applied Mathematics*, vol. 8, no. 2, pp. 300–304, 1960.

[42] D. Knuth, *The art of computer programming*. Addison-Wesley, 1998.

[43] W. D. Wallis, *One-factorizations*. Norwell, 1997.

[44] G. Zaitsev, V. Zinov‘ev, and N. Semakov, “Minimum-check-density codes for correcting bytes of errors, erasures, or defects,” *Problems of Information Transmission*, vol. 19, no. 3, pp. 197–204, 1983.

[45] D. S. Dummit and R. M. Foote, *Abstract algebra*. Wiley, 2003.

[46] R. Gow, “Cauchy’s matrix, the vandermonde matrix and polynomial interpolation,” *Irish Mathematical Society Bulletin*, vol. 28, pp. 45–52, 1992.

[47] T. Boros, T. Kailath, and V. Olshevsky, “A fast parallel Bjorck-Pereyra-type algorithm for solving Cauchy linear equations,” *Linear Algebra and its Applications*, vol. 302-303, pp. 265–293, 1999.

[48] H. Hou and Y. S. Han, “Cauchy MDS array codes with efficient decoding method,” *ArXiv: 1611.09968*, 2016.

[49] A. Bjorck and V. Pereyra, “Solution of Vandermonde systems of equations,” *Mathematics of Computation*, vol. 24, pp. 893–903, 1970.

[50] G. R. Blakley, “Safeguarding cryptographic keys,” in *International Workshop on Managing Requirements Knowledge*, 1979, pp. 313–317.

[51] A. Beimel, “Secret-Sharing Schemes: A Survey,” in *Coding and Cryptology*, Chapter 2, vol. 6639, Springer Berlin Heidelberg, 2011, pp. 11–46.

[52] H. Yamamoto, “Secret sharing system using (k, l, n) threshold scheme,” *Electronics and Communications in Japan (Part I: Communications)*, vol. 69, no. 9, pp. 46–54, 1986.

[53] E. D. Karnin, J. W. Greene, and M. E. Hellman, “On secret sharing systems,” *IEEE Transactions on Information Theory*, vol. 29, no. 1, pp. 35–41, Jan. 1983.

[54] H. Wang and D. Wong, “On secret reconstruction in secret sharing schemes,” *IEEE Transactions on Information Theory*, vol. 54, no. 1, pp. 473–480, 2008.

[55] Z. Zhang, Y. M. Chee, S. Ling, M. Liu, and H. Wang, “Threshold changeable secret sharing schemes revisited,” *Theoretical Computer Science*, vol. 418, pp. 106–115, 2012.

[56] V. Guruswami and M. Wootters, “Repairing Reed-Solomon codes,” in *ACM symposium on Theory of Computing (STOC)*, 2016.

[57] M. Ye and A. Barg, “Explicit constructions of MDS array codes and RS codes with optimal repair bandwidth,” in *IEEE International Symposium on Information Theory (ISIT)*, 2016.

[58] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran, “Network coding for distributed storage systems,” *IEEE Transactions on Information Theory*, vol. 56, no. 9, pp. 4539–4551, 2010.

[59] K. V. Rashmi, N. B. Shah, and P. V. Kumar, "Optimal exact-regenerating codes for distributed storage at the MSR and MBR points via a product-matrix construction," *IEEE Transactions on Information Theory*, vol. 57, no. 8, pp. 5227–5239, 2011.

[60] I. Tamo, Z. Wang, and J. Bruck, "Zigzag codes: MDS array codes with optimal rebuilding," *IEEE Transactions on Information Theory*, vol. 59, no. 3, pp. 1597–1616, 2013.

[61] A. S. Rawat, O. O. Koyluoglu, and S. Vishwanath, "Centralized repair of multiple node failures with applications to communication efficient secret sharing," *ArXiv:1603.04822*, 2016.

[62] M. Ye and A. Barg, "Explicit constructions of high-rate mds array codes with optimal repair bandwidth," *ArXiv:1604.00454*, 2016.

[63] W. Huang, M. Langberg, J. Kliewer, and J. Bruck, "Communication efficient secret sharing," *IEEE Transactions on Information Theory*, vol. 62, no. 12, pp. 7195–7206, 2016.

[64] W. Huang and J. Bruck, *Secret sharing with optimal decoding and repair bandwidth*, accepted to IEEE International Symposium on Information Theory (ISIT), 2017.

[65] R. Bitar and S. El Rouayheb, "Staircase codes for secret sharing with optimal communication and read overheads," in *IEEE International Symposium on Information Theory (ISIT)*, 2016.

[66] W. Huang, M. Langberg, J. Kliewer, and J. Bruck, "Communication efficient secret sharing," *ArXiv:1505.07515*, 2016.

[67] G. Blakley and G. Kabatianski, "Ideal perfect threshold schemes and MDS codes," in *IEEE International Symposium on Information Theory (ISIT)*, vol. 4, 1995.

[68] S. Goparaju, I. Tamo, and R. Calderbank, "An improved sub-packetization bound for minimum storage regenerating codes," *IEEE Transactions on Information Theory*, vol. 60, no. 5, pp. 2770–2779, 2014.

[69] M. Ben-Or, S. Goldwasser, and A. Wigderson, "Completeness theorems for non-cryptographic fault-tolerant distributed computation," in *ACM symposium on Theory of computing (STOC)*, 1988, pp. 1–10.

[70] D. Chaum, C. Crépeau, and I. Damgård, "Multiparty unconditionally secure protocols," in *ACM symposium on Theory of computing (STOC)*, 1988, pp. 11–19.

[71] W. Huang and J. Bruck, "Generic secure repair for distributed storage," *ArXiv:1706.00500*, 2017.

[72] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network Information Flow," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1204–1216, 2000.

[73] S. Y. R. Li, R. Yeung, and N. Cai, "Linear network coding," *IEEE Transactions on Information Theory*, vol. 49, no. 2, pp. 371–381, 2003.

[74] R. Koetter and M. Medard, "An algebraic approach to network coding," *IEEE/ACM Transactions on Networking*, vol. 11, no. 5, pp. 782–795, 2003.

[75] S. Jaggi, P. Sanders, P. a. Chou, M. Effros, S. Egner, K. Jain, and L. M. G. M. Tolhuizen, "Polynomial time algorithms for multicast network code construction," *IEEE Transactions on Information Theory*, vol. 51, no. 6, pp. 1973–1982, 2005.

[76] T. Ho, M. Medard, R. Koetter, D. R. Karger, M. Effros, J. Shi, and B. Leong, "A Random Linear Network Coding Approach to Multicast," *IEEE Transactions on Information Theory*, vol. 52, no. 10, pp. 4413–4430, 2006.

[77] R. Dougherty and K. Zeger, "Nonreversibility and Equivalent Constructions of Multiple-Unicast Networks," *IEEE Transactions on Information Theory*, vol. 52, no. 11, pp. 5067–5077, 2006.

[78] M. F. Wong, M. Langberg, and M. Effros, "On a capacity equivalence between multiple multicast and multiple unicast," in *Allerton Conference on Communication, Control, and Computing*, 2013, pp. 1537–1544.

[79] N. Cai and R. W. Yeung, "Secure network coding," in *IEEE International Symposium on Information Theory (ISIT)*, 2002.

[80] J. Feldman, T. Malkin, R. A. Servedio, and C. Stein, "On the capacity of secure network coding," in *Allerton Conference on Communication, Control, and Computing*, 2004, pp. 1–10.

[81] S. Y. El Rouayheb and E. Soljanin, "On Wiretap Networks II," in *IEEE International Symposium on Information Theory (ISIT)*, 2007, pp. 551–555.

[82] D. Silva and F. R. Kschischang, "Universal Secure Network Coding via Rank-Metric Codes," *IEEE Transactions on Information Theory*, vol. 57, no. 2, pp. 1124–1135, 2011.

[83] T. Chan and A. Grant, "Network coding capacity regions via entropy functions," in *IEEE Transactions on Information Theory*, vol. 60, 2014, pp. 5347–5374.

[84] ——, "Capacity bounds for secure network coding," in *Australian Communications Theory Workshop*, 2008, pp. 95–100.

[85] S. Jalali and T. Ho, "On capacity region of wiretap networks," *ArXiv:1212.3859*, 2012.

[86] T. Cui, T. Ho, and J. Kliewer, "On Secure Network Coding With Nonuniform or Restricted Wiretap Sets," *IEEE Transactions on Information Theory*, vol. 59, no. 1, pp. 166–176, 2013.

[87] N. Cai and R. W. Yeung, "A Security Condition for Multi-Source Linear Network Coding," *IEEE International Symposium on Information Theory (ISIT)*, pp. 561–565, 2007.

[88] R. Koetter and F. R. Kschischang, "Coding for Errors and Erasures in Random Network Coding," *IEEE Transactions on Information Theory*, vol. 54, no. 8, pp. 3579–3591, 2008.

[89] D. Silva, F. R. Kschischang, and R. Kotter, "A Rank-Metric Approach to Error Control in Random Network Coding," *IEEE Transactions on Information Theory*, vol. 54, no. 9, pp. 3951–3967, 2008.

[90] Z. Zhang, "Linear network error correction codes in packet networks," *IEEE Transactions on Information Theory*, vol. 54, no. 1, pp. 209–218, 2008.

[91] O. Kosut, L. Tong, and D. Tse, "Nonlinear Network Coding is Necessary to Combat General Byzantine Attacks," in *Allerton Conference on Communication, Control, and Computing*, 2009, pp. 593–599.

[92] D. Wang, D. Silva, and F. R. Kschischang, "Robust Network Coding in the Presence of Untrusted Nodes," *IEEE Transactions on Information Theory*, vol. 56, no. 9, pp. 4532–4538, 2010.

[93] O. Kosut, L. Tong, and D. N. C. Tse, "Polytope codes against adversaries in networks," in *IEEE International Symposium on Information Theory (ISIT)*, 2010, pp. 2423–2427.

[94] S. Kim, T. Ho, M. Effros, and A. S. Avestimehr, "Network Error Correction With Unequal Link Capacities," *IEEE Transactions on Information Theory*, vol. 57, no. 2, pp. 1144–1164, 2011.

[95] P. H. Che, M. Chen, T. Ho, S. Jaggi, and M. Langberg, "Routing for Security in Networks with Adversarial Nodes," in *IEEE International Symposium on Network Coding (NetCod)*, 2013, pp. 1–6.

[96] T. K. Dikaliotis, H. Yao, T. Ho, M. Effros, and J. Kliewer, "Network Equivalence in the Presence of an Eavesdropper," *ArXiv:1211.4081*, 2012.

[97] S. El Rouayheb, A. Sprintson, and C. Georghiades, "On the index coding problem and its relation to network coding and matroid theory," *IEEE Transactions on Information Theory*, vol. 56, no. 7, pp. 3187–3195, 2010.

[98] M. F. Wong, M. Langberg, and M. Effros, "On a capacity equivalence between network and index coding and the edge removal problem," in *IEEE International Symposium on Information Theory (ISIT)*, 2013.

[99] S. Kamath, D. Tse, and C.-C. Wang, "Two-unicast is hard," in *IEEE International Symposium on Information Theory (ISIT)*, 2014.

[100] M. F. Wong, M. Langberg, and M. Effros, "Linear capacity equivalence between multiple multicast and multiple unicast," in *IEEE International Symposium on Information Theory (ISIT)*, 2014.

[101] M. F. Wong, M. Effros, and M. Langberg, "On an equivalence of the reduction of k-unicast to 2-unicast capacity and the edge removal property," in *IEEE International Symposium on Information Theory (ISIT)*, 2015.

[102] ——, "A code equivalence between streaming network coding and streaming index coding," in *IEEE International Symposium on Information Theory (ISIT)*, 2017.

[103] C. Gkantsidis and P. Rodriguez, "Cooperative Security for Network Coding File Distribution.," *IEEE International Conference on Computer Communications (INFOCOM)*, pp. 1–13, 2006.

[104] D. Boneh, D. Freeman, J. Katz, and B. Waters, "Signing a Linear Subspace: Signature Schemes for Network Coding," in *Public Key Cryptography*, Springer Berlin Heidelberg, 2009, pp. 68–87.

[105] F. Zhao, T. Kalker, M. Medard, and K. J. Han, "Signatures for Content Distribution with Network Coding," in *IEEE International Symposium on Information Theory (ISIT)*, 2007, pp. 556–560.

[106] Y. Li, H. Yao, M. Chen, S. Jaggi, and A. Rosen, "RIPPLE Authentication for Network Coding," *IEEE International Conference on Computer Communications (INFOCOM)*, pp. 1–9, 2010.

[107] F. Oggier and H. Fathi, "An Authentication Code Against Pollution Attacks in Network Coding," *IEEE/ACM Transactions on Networking*, vol. 19, no. 6, pp. 1587–1596, 2011.

[108] W. Huang, T. Ho, H. Yao, and S. Jaggi, "Rateless resilient network coding against byzantine adversaries," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2013.

[109] W. Huang, T. Ho, M. Langberg, and J. Kliewer, "On secure network coding with uniform wiretap sets," in *IEEE International Symposium on Network Coding (NetCod)*, 2013.

[110] W. Huang, T. Wang, X. Hu, J. Jang, and T. Salonidis, "Rateless and pollution-attack-resilient network coding," in *IEEE International Symposium on Information Theory (ISIT)*, 2015.

[111] W. Huang, T. Ho, M. Langberg, and J. Kliewer, "Reverse edge cut-set bounds for secure network coding," in *IEEE International Symposium on Information Theory (ISIT)*, 2014.

[112] W. Huang, M. Langberg, and J. Kliewer, "Connecting multiple-unicast and network error correction: Reduction and unachievability," in *IEEE International Symposium on Information Theory (ISIT)*, 2015.

[113] W. Huang, T. Ho, M. Langberg, and J. Kliewer, "Single-source/sink network error correction is as hard as multiple-unicast," in *Allerton Conference on Communication, Control and Computing*, 2014.

[114] R. Dougherty, C. Freiling, and K. Zeger, "Unachievability of network coding capacity," *IEEE Transactions on Information Theory*, vol. 52, no. 6, pp. 2365–2372, 2006.

[115] B. K. Rai and B. K. Dey, "On network coding for sum-networks," *IEEE Transactions on Information Theory*, vol. 58, no. 1, pp. 50–63, 2012.