On Block Security of Regenerating Codes at the MBR Point for Distributed Storage Systems

Son Hoang Dau*, Wentu Song†, Chau Yuen‡
Singapore University of Technology and Design, Singapore
Emails: {*sonhoang_dau, †wentu_song, ‡yuenchau}@sutd.edu.sg

***Abstract*—A *passive* adversary can eavesdrop stored content or downloaded content of some storage nodes, in order to learn illegally about the file stored across a distributed storage system (DSS). Previous work in the literature focuses on code constructions that trade storage capacity for *perfect security*. In other words, by decreasing the amount of original data that it can store, the system can guarantee that the adversary, which eavesdrops up to a certain number of storage nodes, obtains *no information* (in Shannon's sense) about the original data. In this work we introduce the concept of *block security* for DSS and investigate minimum bandwidth regenerating (MBR) codes that are *block secure* against adversaries of varied eavesdropping strengths. Such MBR codes guarantee that no information about any *group* of original data units up to a certain size is revealed, *without* sacrificing the storage capacity of the system. The size of such secure groups varies according to the number of nodes that the adversary can eavesdrop. We show that code constructions based on Cauchy matrices provide block security. The opposite conclusion is drawn for codes based on Vandermonde matrices.**

I. Introduction

A. Background

In recent years, the demand for large-scale data storage has grown explosively, due to numerous applications including large files and video sharing, social networks, and back-up systems. Distributed Storage Systems (DSS) store a tremendous amount of data using a massive collection of distributed storage nodes. Applications of DSS include large data centers and P2P storage systems such as OceanStore [1], Total Recall [2], and Dhash++ [3] that deploy a huge number of storage nodes spread widely over the Internet. Since any storage device is individually unreliable and subject to failure, redundancy must be introduced to provide the much-needed system-level protection against data loss due to storage node failure.

The simplest form of redundancy is replication. By storing n identical copies of a file at n distributed nodes, one copy per node, an n-duplication system can guarantee the data availability as long as no more than $(n-1)$ nodes fail. Such systems are very easy to implement and maintain, but extremely inefficient in storage space utilization, incurring tremendous waste in equipment, building space, and cost for powering and cooling. More sophisticated systems employing erasure coding based on maximum distance separable (MDS) codes [4] can expect to considerably improve the storage efficiency. However, a DSS based on MDS codes often incurs considerable communication bandwidth during node repair: a replacement node has to download the whole file to recover the content of just one node. To overcome the disadvantages of both replication and erasure codes, regenerating codes were proposed for DSS in the groundbreaking work of Dimakis *et al.* [5]. In this work, we only consider regenerating codes for DSS and limit ourselves to minimum bandwidth regenerating (MBR) codes with exact repair [5].

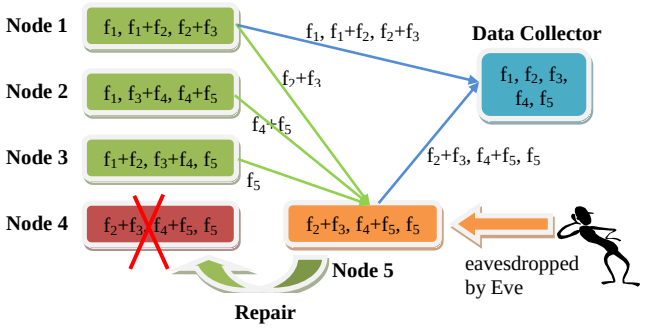

Fig. 1: Example of a DSS with an eavesdropper.

In practice, one important aspect in the design of a DSS code is its security. As the storage nodes can well be located at different places, and new nodes join the network all the time to replace failing nodes, it is possible that at a certain point, some nodes might be compromised by some unauthorized party, referred to as an adversary. The adversary can be either

- *active*, i.e. controlling the node, modifying the data, and sending erroneous data to other nodes [6], [7], [8], [9], or
- *passive*, i.e. knowing the data stored in the node and observing all communication between this node and the other nodes, in order to learn illegally the content of the file stored by the DSS [7], [10], [11], [12], [13].

In this work, we focus on the security of DSS codes against the second type of adversary, the passive adversary. We also refer to this type of adversary as eavesdropper. We illustrate in Fig. 1 a scenario where the adversary can observe all data stored in one storage node. The data file is split into five data units $\boldsymbol{f} = (f_1, f_2, f_3, f_4, f_5)$. Originally there are four storage nodes. At a certain time, Node 4 fails. Node 5 comes in to replace Node 4 (repair). If the content of this node is eavesdropped by Eve, then Eve can obtain the values of two units f_4 and f_5 explicitly. The same thing happens when Eve eavesdrops any node among the three nodes 1, 2, and 3. Therefore the system is not secure against an adversary who can eavesdrop one storage node.

B. Related Work

Hereafter we follow the standard notation in the regenerating code literature (see Section II). In their pioneering work, Dimakis *et al.* [5] established that the maximum file size to be stored in a DSS $\mathcal{D}(n,k,d)$ must satisfy the following inequality

$$\mathcal{M} \leq \sum_{i=1}^{k} \min\{(d-i+1)\beta, \alpha\}. \tag{1}$$

A construction of optimal regenerating codes with exact repair at the MBR point for all n,k,d was proposed by Rashmi *et al.* [14]. When the stored contents of some ℓ nodes are observed by an adversary Eve, Pawar *et al.* [6], [7] showed that the maximum file size to be stored satisfies

$$\mathcal{M}^{\mathrm{s}} \leq \sum_{i=\ell+1}^{k} \min\{(d-i+1)\beta, \alpha\}, \tag{2}$$

provided that Eve gains *no information* (in Shannon's sense) about the file. This type of security is often referred to as *perfect security* or *strong security* in the literature. The parameter ℓ is called the *adversarial strength*. The authors [6], [7] also provided an optimal code construction, based on complete graphs, for the case $d = n-1$ that attains the bound (2). We later argue that an extension of their construction based on regular graphs [15] also produces optimal perfectly secure codes for all d with nd even (see Remark 6). Under the same adversary model, Shah *et al.* [11] constructed optimal codes that attain the bound (2) for all d. The authors used product-matrix codes in their construction.

Comparing (1) and (2), it is clear that in order to achieve the perfect security against an adversary which eavesdrops ℓ storage nodes, the storage capacity of the system has to be decreased by an amount of

$$\Delta\mathcal{M} = \mathcal{M} - \mathcal{M}^{\mathrm{s}} = \sum_{i=1}^{\ell} \min\{(d-i+1)\beta, \alpha\}.$$

Therefore, perfect security is obtained at the cost of lowering the storage capacity. Additionally, according to the security scheme studied thereof, one has to specify beforehand the strength of the adversary, i.e. the number of storage nodes it can eavesdrop, and then modify the input accordingly to obtain the perfect security. When the strength of the adversary exceeds the specified threshold, nothing is guaranteed on the security of the system. In many practical storage systems, perfect security is either too strict and might not be even necessary, or too costly (too much wasted space in the system) and might not be affordable. Hence weaker security levels at lower costs would be preferred in many practical scenarios.

C. Our Contribution

In order to address the aforementioned issues with perfect security, we propose to use a broader concept of security from the network coding literature, namely, *block security* [16], [17], [18] (also known as *security against guessing*). A code is b-block secure against an adversary of strength ℓ, if the adversary, which accesses at most ℓ storage nodes, gains no information about any group of b data units. In the language of information theory, the mutual information between the eavesdropped content and any group of b original data units is zero. The concept of block security describes nicely a hierarchy of different levels of security against eavesdropping. Depending on the adversarial strength, a block secure system can provide a range of different levels of security, from the weakest level to the strongest level:

- 1-block security (also referred to as *weak security)* implies that no individual data unit is revealed,
- 2-block security implies that no information on any group of two data units is revealed,
- $\cdots$,
- $\mathcal{M}$-block security, where $\mathcal{M}$ is the total number of data units (the file size), implies the perfect security, i.e. no information about the original data is revealed.

If the system is weakly secure, although the adversary gains some information about the file, it cannot determine each particular data unit. For instance, if the file is a movie and the data units are movie chunks, then the adversary obtains no information about each individual chunk, and hence cannot play the movie. Furthermore, if the system is b-block secure against an adversary of strength ℓ, even when the adversary can access some ℓ storage nodes and on top of that, gain knowledge on some other $b-1$ data units via some side channel, it still cannot determine each particular data unit.

We investigate the security of the two main existing MBR code constructions [19], [15], [14]. Recall that either Vandermonde matrices or Cauchy matrices are often used in these constructions. We prove the following

- Vandermonde-based codes are *not* block secure,
- Cauchy-based regular-graph codes [19], [15] are inherently block secure,
- Cauchy-based product-matrix codes [14] are inherently block secure.

Based on these results, we are able to conclude that using Cauchy matrices rather than Vandermonde in these constructions of MBR codes automatically guarantees weaker levels of security compared with perfect security, but most importantly, *without* any loss on the storage capacity of the system. We also show that with Cauchy-based constructions of perfectly secure codes [10], [7], [11], certain security level is still guaranteed even when the adversarial strength *surpasses* the security threshold. More specifically, in addition to being *perfectly secure* against adversaries of strength not exceeding a specified threshold λ, the codes remain to be *block secure* against an adversary of strength $\ell > \lambda$ as long as $\ell < k$ (Fig. 2). The level b of block security gradually decreases as the adversarial strength ℓ increases. When $\ell \geq k$, the whole file is revealed.

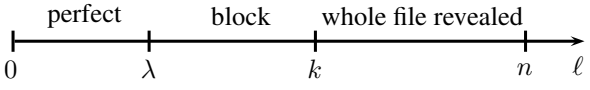

Fig. 2: Security of a Cauchy-based system as the adversarial strength ℓ varies.

The paper is organized as follows. In Section II, we discuss the concept of block security for distributed storage systems. We analyze the security of regular-graph codes and product-matrix codes in Section III and Section IV, respectively. We conclude the paper in Section V.

II. Block Security for Distributed Storage Systems

We denote by $\mathbb{F}_q$ the finite field with q elements. Let $[n]$ denote the set $\{1, 2, \ldots, n\}$. Let $\boldsymbol{u}^{\mathrm{t}}$ denotes the transpose of a vector $\boldsymbol{u}$. For an $m \times n$ matrix $\boldsymbol{M}$, for $i \in [m]$ and $j \in [n]$, let $\boldsymbol{M}_i$ and $\boldsymbol{M}[j]$ denote the row i and the column j of $\boldsymbol{M}$, respectively. We define below standard notions from coding theory (for instance, see [20]).

The (Hamming) *weight* of a vector $\boldsymbol{u} \in \mathbb{F}_q^n$ is the number of the nonzero coordinates of $\boldsymbol{u}$. For the vectors $\boldsymbol{u} = (u_1, u_2, \ldots, u_n) \in \mathbb{F}_q^n$ and $\boldsymbol{v} = (v_1, v_2, \ldots, v_n) \in \mathbb{F}_q^n$, the (Hamming) distance between $\boldsymbol{u}$ and $\boldsymbol{v}$ is defined to be the number of coordinates where $\boldsymbol{u}$ and $\boldsymbol{v}$ differ, namely,

$$\mathsf{d}(\boldsymbol{u}, \boldsymbol{v}) = |\{i \in [n] \ : \ u_i \neq v_i\}|.$$

A k-dimensional subspace $\mathscr{C}$ of $\mathbb{F}_q^n$ is called a linear $[n, k, d]_q$ code over $\mathbb{F}_q$ if the minimum distance of $\mathscr{C}$,

$$\mathsf{d}(\mathscr{C}) \triangleq \min_{\boldsymbol{u} \in \mathscr{C},\, \boldsymbol{v} \in \mathscr{C},\, \boldsymbol{u} \neq \boldsymbol{v}} \mathsf{d}(\boldsymbol{u}, \boldsymbol{v}) \ ,$$

is equal to d. Sometimes we may use the notation $[n, k]_q$ or just $[n, k]$ for the sake of simplicity. The vectors in $\mathscr{C}$ are called codewords. It is easy to see that the minimum weight of a nonzero codeword in a linear code $\mathscr{C}$ is equal to its minimum distance $\mathsf{d}(\mathscr{C})$. A *generator matrix* $\boldsymbol{G}$ of an $[n, k]_q$ code $\mathscr{C}$ is a $k \times n$ matrix whose rows are linearly independent codewords of $\mathscr{C}$. Then $\mathscr{C} = \{\boldsymbol{y}\boldsymbol{G} : \boldsymbol{y} \in \mathbb{F}_q^k\}$. The well-known Singleton bound states that for any $[n, k, d]_q$ code, it holds that $d \leq n - k + 1$. If the equality is attained, the code is called maximum distance separable (MDS).

Let $\boldsymbol{F} = (F_1, F_2, \ldots, F_{\mathcal{M}})$, where F_i's $(i \in [\mathcal{M}])$ are independent and identically uniformly distributed random variables over $\mathbb{F}_q$. We assume that the file to be stored in the system is $\boldsymbol{f} = (f_1, f_2, \ldots, f_{\mathcal{M}}) \in \mathbb{F}_q^{\mathcal{M}}$, a realization of $\boldsymbol{F}$. We call $\mathcal{M}$ the file size and each f_i a (original) data unit.

We denote by $\mathcal{D}(n, k, d)$ a typical DSS with n storage nodes where the file can be recovered from the contents of any k out of n nodes, and to repair a failed node, a new node (referred to as a newcomer) can contact any d nodes to regenerate the content of the failed node. We refer to d as the *repair degree*. Additionally, each node stores α coded units, i.e. α linear combinations of data units f_i's. In the repair process, a newcomer downloads β coded units from each of d live nodes. Suppose that $\beta = 1$ (data striping is used for larger β). At the MBR point, we have [19], [14]

$$\mathcal{M} = kd - \binom{k}{2}, \quad \alpha = d.$$

Following the adversarial attack model proposed by Pawar *et al.* [10], [7], we suppose that the adversary can observe the stored contents of some ℓ nodes. Let $\boldsymbol{C}^i$ denote the random vector over $\mathbb{F}_q^{\alpha}$ that represents the stored content at Node i.

Definition 1 ([10], [7])**.** A DSS $\mathcal{D}(n, k, d)$ together with its coding scheme is called *perfectly secure* against an adversary of strength λ $(\lambda < k)$ if the mutual information

$$\mathsf{I}(\boldsymbol{F}, \cup_{i \in E} \boldsymbol{C}^i) = 0,$$

for all subsets $E \subseteq [n]$, $|E| \leq \lambda$.

Definition 2. A DSS $\mathcal{D}(n, k, d)$ together with its coding scheme is called *b-block secure* against an adversary of strength λ $(\lambda < k)$ if the mutual information

$$\mathsf{I}(\cup_{j \in B} F_j, \cup_{i \in E} \boldsymbol{C}^i) = 0,$$

for all subsets $B \subseteq [\mathcal{M}]$, $|B| \leq b$, and for all subsets $E \subseteq [n]$, $|E| \leq \lambda$.

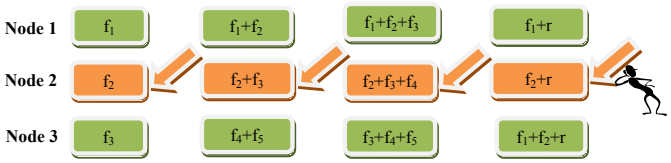

Fig. 3: Different levels of security.

We illustrate in Fig. 3 different levels of security for DSS. For instance, in Fig. 3(c), as long as the adversary accesses the stored content of only one node, it cannot deduce any linear combination of two units. Therefore, the system is 2-block secure against an adversary of strength one. In Fig. 3(d), by using a randomly generated variable r, as long as the adversary accesses the stored content of only one node, it cannot deduce any linear combination of the data units, and hence gains no information at all about the stored file. Hence in this case, the system is perfectly secure against an adversary of strength one.

It is straightforward that $\mathcal{M}$-block security is equivalent to perfect security, where $\mathcal{M}$ is the file size. Hence, perfect security can well be regarded as the highest level of block security. However, perfect security is not given for free. One has to trade some storage capacity for perfect security (see Section I-B). In fact, in the perfectly secure code constructions presented in [10], [7], [11], part of the file has to be replaced by randomly generated variables, hence reducing the useful storage space of the system. By contrast, lower levels of block security can be achieved at essentially no cost, as we later present in Section III and IV. This advantage of block security makes the concept attractive to practical storage systems, where the storage redundancy has to be minimized to maintain a competitive price for the storage service.

Under the assumptions that

- the data units are all independent and identically uniformly distributed random variables over $\mathbb{F}_q$,
- the coding scheme is linear,

the b-block security given in Definition 2 is equivalent to the requirement that no linear combination of at most b data units can be deduced by the adversary. The following lemma specifies a necessary and sufficient condition for the block security of DSS.

Lemma 3. *Let $\boldsymbol{f} = (f_1, f_2, \ldots, f_{\mathcal{M}}) \in \mathbb{F}_q^{\mathcal{M}}$ be the stored file and $\boldsymbol{E}\boldsymbol{f}^{\mathsf{t}}$ represent the coded units that the adversary obtains by observing ℓ storage nodes. Let $\mathscr{C}^{\boldsymbol{E}}$ be the linear error-correcting code generated by the rows of the matrix $\boldsymbol{E}$, and $\mathsf{d}(\mathscr{C}^{\boldsymbol{E}})$ be its minimum distance. Then the adversary cannot deduce any nontrivial linear combination of any group of at most b data units if and only if $b \leq \mathsf{d}(\mathscr{C}^{\boldsymbol{E}}) - 1$.*

A rigorous proof of Lemma 3 can be found in the Appendix. We discuss below the intuition behind the proof of this lemma. As the adversary obtains $\boldsymbol{E}\boldsymbol{f}^{\mathsf{t}}$, it can linearly transform these coded symbols by considering the product $\boldsymbol{\alpha}\boldsymbol{E}\boldsymbol{f}^{\mathsf{t}}$, where $\boldsymbol{\alpha}$ is some coefficient vector. Since $\boldsymbol{\alpha}\boldsymbol{E}$ is a codeword of $\mathscr{C}^{\boldsymbol{E}}$, its weight is at least $\mathsf{d}(\mathscr{C}^{\boldsymbol{E}})$ if it is a nonzero codeword. In other words, if $\boldsymbol{\alpha}\boldsymbol{E} \neq \boldsymbol{0}$ then it has at least $\mathsf{d}(\mathscr{C}^{\boldsymbol{E}})$ nonzero coordinates. As a result, $\boldsymbol{\alpha}\boldsymbol{E}\boldsymbol{f}^{\mathsf{t}}$ is a linear combination of at least $\mathsf{d}(\mathscr{C}^{\boldsymbol{E}})$ data units. Therefore, by linearly transforming the eavesdropped coded units $\boldsymbol{E}\boldsymbol{f}^{\mathsf{t}}$, the adversary cannot produce a nontrivial linear combination of $\mathsf{d}(\mathscr{C}^{\boldsymbol{E}}) - 1$ data units or less.

On the other hand, as the adversary can choose an appropriate vector $\boldsymbol{\alpha}$ so that $\boldsymbol{\alpha}\boldsymbol{E}$ has weight exactly $\mathsf{d}(\mathscr{C}^{\boldsymbol{E}})$, it can always determine the value of a linear combination of a certain group of $\mathsf{d}(\mathscr{C}^{\boldsymbol{E}})$ data units. Thus, the adversary cannot deduce any nontrivial linear combination of any group of at most b data units *only if* $b \leq \mathsf{d}(\mathscr{C}^{\boldsymbol{E}}) - 1$.

III. On the Security of Regular-Graph Codes

We briefly describe the regular-graph codes constructed by Rashmi *et al.* [19] and El Rouayheb *et al.* [15] below.

Let $\mathcal{G}$ be a d-regular graph on n vertices $u_1, u_2, \ldots, u_n$. Then each vertex of $\mathcal{G}$ is adjacent to d edges. Therefore, let $e_1, e_2, \ldots, e_{nd/2}$ be all $nd/2$ edges of $\mathcal{G}$. Let $\boldsymbol{G}$ be an $(nd/2) \times \mathcal{M}$ matrix over $\mathbb{F}_q$ satisfying the MDS property: any $\mathcal{M}$ rows of $\boldsymbol{G}$ are linearly independent. Then $\mathcal{M}$ data units $f_1, f_2, \ldots, f_{\mathcal{M}}$ are encoded into $nd/2$ coded units by the transformation $\boldsymbol{c}^{\mathsf{t}} = (c_1, \ldots, c_{nd/2}) = \boldsymbol{G}\boldsymbol{f}^{\mathsf{t}}$. Node i stores c_j if and only if the edge e_j is adjacent to the vertex u_i. As $\mathcal{G}$ is d-regular, each node stores exactly $\alpha = d$ coded units c_j's. Any set of k nodes together provide $kd - \binom{k}{2} = \mathcal{M}$ distinct coded units c_j's, hence can recover the whole file thank to the MDS property of the encoding matrix $\boldsymbol{G}$. When Node i fails ($i \in [n]$), the newcomer contacts Node j if u_i and u_j are adjacent in $\mathcal{G}$. Since $\mathcal{G}$ is d-regular, there are d such nodes. For such a Node j, the newcomer downloads $\beta = 1$ coded unit c_s if e_s is the edge connected u_i and u_j. These d coded units c_s are distinct as they correspond to d distinct edges adjacent to u_i, and form the content of the failed node. The newcomer simply stores these d coded units and the repair process for Node i is done.

A. Cauchy-Based Regular-Graph Codes Are Block Secure

A Cauchy matrix is a matrix of form $\Big((x_i + y_j)^{-1}\Big)_{m \times n}$, where x_i's and y_j's are elements of $\mathbb{F}_q$ that satisfy $x_i + y_j \neq 0$ for all $i \in [m]$ and $j \in [n]$. A Cauchy matrix has a special property that any submatrix is again a Cauchy matrix. It is well known that any square Cauchy matrix is invertible. Therefore, any square submatrix of a Cauchy matrix is invertible. This is a crucial property that makes codes based on Cauchy matrices block secure.

Theorem 4. *A DSS $\mathcal{D}(n, k, d)$ equipped with a regular-graph regenerating code [15] based on a Cauchy matrix is b-block secure against an adversary of strength ℓ ($\ell < k$), where*

$$b = \Big(kd - \binom{k}{2}\Big) - \Big(\ell d - \binom{\ell}{2}\Big) = (k - \ell)\Big(d - \frac{k + \ell - 1}{2}\Big).$$

Note that nd must be even according to the code construction.

Proof: Note that according to the code construction [19], [15], an adversary that accesses ℓ nodes obtains $\ell d - \binom{\ell}{2}$ distinct coded units. Therefore, the adversary obtains $\boldsymbol{E}\boldsymbol{f}^{\mathsf{t}}$, where $\boldsymbol{E}$ is a submatrix of the encoding matrix $\boldsymbol{G}$, consisting of $\ell d - \binom{\ell}{2}$ rows of $\boldsymbol{G}$. Note that $\boldsymbol{E}$ has $kd - \binom{k}{2}$ columns. As $\boldsymbol{G}$ is a Cauchy matrix, any square submatrix of size $\ell d - \binom{\ell}{2}$ of $\boldsymbol{E}$ is also a Cauchy matrix, and hence invertible. Therefore, the rows of $\boldsymbol{E}$ generates an MDS code of length $kd - \binom{k}{2}$ and dimension $\ell d - \binom{\ell}{2}$ [20, Ch. 11]. It is also well known that such an MDS code has minimum distance $(kd - \binom{k}{2}) - (\ell d - \binom{\ell}{2}) + 1$. Applying Lemma 3, the proof follows. ∎

For instance, according to Theorem 4, for $n = 7$, $k = 5$, $d = 6$, the degradation of the block security level of a Cauchy-based regular-graph code is illustrated in Fig. 4. Note that the file size is $\mathcal{M} = 5 \times 6 - \binom{5}{2} = 20$.

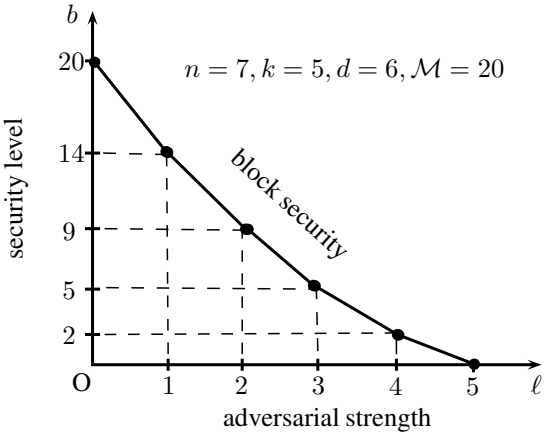

Fig. 4: Degradation of the security level for $\mathcal{D}(7, 5, 6)$ with a Cauchy-based regular-graph code as the adversarial strength increases.

The corresponding construction of perfectly secure codes [10], [7] is completely the same as the one described

at the beginning of this section, except that a subset of data units has to be replaced by a subset of randomly generated variables and that the encoding matrix $\boldsymbol{G}$ has to be either Vandermonde or Cauchy (or more generally, a (transposed) generator matrix of a nested MDS code). Suppose that the Cauchy matrix is used in this construction. Treating the random variables as data units, we can apply Theorem 4 and show that even when the adversarial strength surpasses the specified threshold, although the code is no longer perfectly secure, it is still block secure. Hence we have the following corollary.

Corollary 5. *Consider a DSS $\mathcal{D}(n,k,d)$ equipped with the regenerating code proposed in [10], [7], [15], which is perfectly secure against an adversary of strength at most λ. Suppose that in the construction of the regenerating code, a Cauchy matrix is used. Then when the adversarial strength ℓ ($\ell < k$) surpasses the threshold λ, the code is no longer perfectly secure, but is still b-block secure, where*

$$b = \Big(kd - \binom{k}{2}\Big) - \Big(\ell d - \binom{\ell}{2}\Big) = (k-\ell)\Big(d - \frac{k+\ell-1}{2}\Big).$$

For example, for a DSS $\mathcal{D}(7,5,6)$ as in Fig. 4, suppose that the specified security threshold is $\lambda = 2$, then the degradation of the security is depicted in Fig. 5. Note that now the file size has to be decreased to $\mathcal{M}^{\mathrm{s}} = 9$, according to (2).

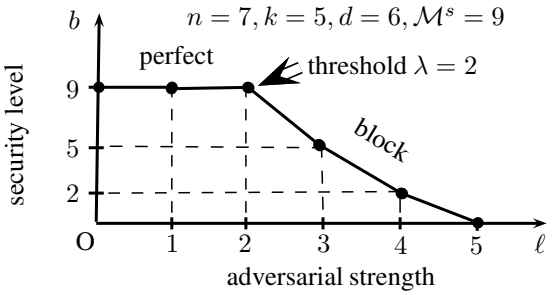

Fig. 5: Degradation of the security level for $\mathcal{D}(7,5,6)$ with a perfectly secure Cauchy-based regular-graph code as the adversarial strength increases.

Remark 6. The construction of optimal perfectly secure codes with uncoded exact repair was proposed by Pawar *et al.* [10], [7] for the case $d = n-1$. We argue below that the same construction, i.e. replacing data units by random units and using nested MDS codes, based on regular-graphs [15] also provides optimal perfectly secure codes for all d with nd even. That is why we mention both constructions in Corollary 5. Indeed, as $\beta = 1$ and $\alpha = d$, the bound (2) reduces to

$$\mathcal{M}^{\mathrm{s}} \le (k-\ell)\big(d - (k+\ell-1)/2\big). \tag{3}$$

According to the construction in [15], eavesdropping on ℓ nodes reveals $R = \ell d - \binom{\ell}{2}$ distinct coded units. Hence, $\ell d - \binom{\ell}{2}$ data units has to be replaced by random units. Therefore, the number of data units that can be stored securely using a nested MDS code is

$$\begin{aligned}\mathcal{M}^{\mathrm{s}} &= \mathcal{M} - R \\ &= \Big(kd - \binom{k}{2}\Big) - \Big(\ell d - \binom{\ell}{2}\Big) \\ &= (k-\ell)\big(d - (k+\ell-1)/2\big),\end{aligned}$$

which matches the bound (3). Hence the regular-graph codes [15] are optimal perfectly secure code for every d with nd even.

B. Vandermonde-Based Regular-Graph Codes Are Not Block Secure

We present below an example of a Vandermonde-based regular-graph code [19], [15] that is not block secure and briefly explain the reason behind it.

Let $n = 4$, $k = 2$, and $d = 3$. As $\beta = 1$, we have $\mathcal{M} = kd - \binom{k}{2} = 5$. The encoding matrix $\boldsymbol{G}$ is chosen as a 6×5 Vandermonde matrix over $\mathbb{F}_{13}$

$$\boldsymbol{G} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 3 & 9 & 1 & 3 \\ 1 & 5 & 12 & 8 & 1 \\ 1 & 7 & 10 & 5 & 9 \\ 1 & 9 & 3 & 1 & 9 \\ 1 & 11 & 4 & 5 & 3 \end{pmatrix}$$

An adversary which accesses one node obtains $d = 3$ distinct coded units. Suppose these three coded units are $\boldsymbol{G}_1\boldsymbol{f}^{\mathrm{t}}$, $\boldsymbol{G}_2\boldsymbol{f}^{\mathrm{t}}$, and $\boldsymbol{G}_5\boldsymbol{f}^{\mathrm{t}}$. In other words, the adversary obtains $\boldsymbol{E}\boldsymbol{f}^{\mathrm{t}}$ where $\boldsymbol{E}$ is the submatrix of $\boldsymbol{G}$ consisting of the first, the second, and the fifth rows of $\boldsymbol{G}$. It is straightforward to verify that the minimum distance of the error-correcting code generated by the rows of $\boldsymbol{E}$ is one. Therefore, according to Lemma 3, the code is not even weakly secure (1-block secure) against an adversary of strength one. More specifically, by applying a linear transformation $\boldsymbol{v}\boldsymbol{E}\boldsymbol{f}^{\mathrm{t}}$ where $\boldsymbol{v} = (9,1,3)$, the adversary obtains $f_3 = \boldsymbol{v}\boldsymbol{E}\boldsymbol{f}^{\mathrm{t}}$ explicitly.

The reason behind this unwanted behavior of Vandermonde-based codes can be explained as follows. The block security of Cauchy-based regular-graph codes (see Proof of Theorem 4) strictly relies on a very special property of a Cauchy matrix: every square submatrix of a Cauchy matrix is invertible. However, a Vandermonde matrix does not have this property. For example, the 3×3 submatrix of the Vandermonde matrix $\boldsymbol{G}$ above that consisting of the entries in the first, the second, and the fifth rows, and in the first, second, and the fifth columns of $\mathcal{G}$ only has rank two. Therefore, the block security of the corresponding code is no longer guaranteed.

IV. On the Security of Product-Matrix Codes

We briefly describe the MBR product-matrix codes constructed by Rashmi *et al.* [14] below.

The file size $\mathcal{M} = kd - \binom{k}{2}$ can be rewritten as $\mathcal{M} = \binom{k+1}{2} - k(d-k)$. Let $\boldsymbol{M}$ be the *message* matrix of the following form

$$\boldsymbol{M} = \begin{pmatrix} \boldsymbol{S} & \boldsymbol{T} \\ \boldsymbol{T}^{\mathrm{t}} & \boldsymbol{0} \end{pmatrix},$$

where $\boldsymbol{S}$ is a $k \times k$ symmetric matrix and $\boldsymbol{T}$ is a $k \times (d-k)$ matrix. The $\binom{k+1}{2}$ entries in the upper-triangular half of $\boldsymbol{S}$ are filled up by $\binom{k+1}{2}$ distinct data units drawn from the set $\{f_i\}_{i\in[\mathcal{M}]}$. The remaining $k(d-k)$ data units are used to fill up the second $k \times (d-k)$ matrix $\boldsymbol{T}$. The encoding matrix is $\Psi = [\Phi \quad \Delta]$, where Φ and Δ are $n \times k$ and $n \times (d-k)$ matrices, respectively, chosen in such a way that

1) any d rows of Ψ are linearly independent,
2) any k rows of Φ are linearly independent.

Then Node i stores d entries of row i of the matrix $\Psi\boldsymbol{M}$. The encoding matrix Ψ is often chosen as a Vandermonde matrix or a Cauchy matrix. Details on the file reconstruction and node repair processes can be found in [14].

A. Cauchy-Based Product-Matrix Codes Are Block Secure

Our main result in this section is to prove that the Cauchy-based product-matrix code [14] is $(k-\ell)$-block secure against an adversary of strength at most $\ell < k$. Note that if k nodes are eavesdropped, the whole file will be reconstructed. Thus the block security level of the Cauchy-based product-matrix codes is $d-(k+\ell-1)/2$ times lower than that of the Cauchy-based regular graph-codes (see Theorem 4).

Theorem 7. *In the construction of an MBR product-matrix code [14] for a DSS $\mathcal{D}(n,k,d)$, if the encoding matrix Ψ is a Cauchy matrix then the code is $(k-\ell)$-block secure against an adversary of strength ℓ $(\ell < k)$.*

Sketch: To facilitate our discussion, we label the data units by the index set

$$\mathcal{I} = \{(i,j) \mid 1 \le i \le k \text{ and } i \le j \le d\}. \tag{4}$$

We assume that the $\mathcal{M}$ elements of $\mathcal{I}$ are always listed in the lexicographic order: $(1,1)$, $(1,2)$, …, $(1,d)$, $(2,2)$, $(2,3)$, …, $(2,d)$, …, (k,k), $(k,k+1)$, …, (k,d). For $\xi = (i,j) \in \mathcal{I}$, we often write f_ξ or $f_{(i,j)}$ interchangeably. Also, for $(i,j) \in \mathcal{I}$, both the (i,j)-entry and the (j,i)-entry of $\boldsymbol{M}$ are $f_{(i,j)}$. Again,

$$\boldsymbol{f} = (f_{(1,1)}, \ldots, f_{(1,d)}, f_{(2,2)}, \ldots, f_{(2,d)}, \ldots, f_{(k,k)}, \ldots, f_{(k,d)})$$

is the vector in $\mathbb{F}_q^{\mathcal{M}}$ that represents the file stored in the system. Suppose the encoding matrix Ψ is a Cauchy matrix. We assume that the adversary can access some ℓ storage nodes and $\boldsymbol{E}$ is the submatrix of Ψ consisting of the corresponding ℓ rows of Ψ. Hence the adversary obtains

$$\boldsymbol{H} = \boldsymbol{E}\boldsymbol{M}. \tag{5}$$

This is regarded as a collection of d linear systems with unknowns f_ξ's, $\xi \in \mathcal{I}$. In order to apply Lemma 3, we have to transform these systems into a single linear system with unknown $\boldsymbol{f}$. Let $\overline{\boldsymbol{E}} = (\boldsymbol{b}_{j,\xi})_{j\in[d],\xi\in\mathcal{I}}$ be a $(d\ell) \times \mathcal{M}$ block matrix where

$$\boldsymbol{b}_{j,\xi} = \begin{cases} \boldsymbol{E}[i], & \text{if } \xi = (i,j) \text{ or } \xi = (j,i), \\ \boldsymbol{0}, & \text{otherwise,} \end{cases} \tag{6}$$

and $\boldsymbol{0}$ denotes the all-zero vector in $\mathbb{F}_q^{\ell}$. Recall that $\boldsymbol{E}[i]$ denotes the column i of $\boldsymbol{E}$. Let $\overline{\boldsymbol{H}} = (\boldsymbol{H}[1]^{\mathrm{t}}, \boldsymbol{H}[2]^{\mathrm{t}}, \ldots, \boldsymbol{H}[d]^{\mathrm{t}})^{\mathrm{t}}$. The proofs of the following two lemmas can be found in the Appendix.

Lemma 8. *The systems (5) are equivalent to the following system*

$$\overline{\boldsymbol{H}} = \overline{\boldsymbol{E}}\boldsymbol{f}^t. \tag{7}$$

Lemma 9. *The linear code generated by the rows of the matrix $\overline{\boldsymbol{E}}$ has minimum distance $k-\ell+1$.*

Combining these two lemmas and Lemma 3 (applied to $\overline{\boldsymbol{E}}$ instead of $\boldsymbol{E}$), we deduce that the code is $(k-\ell)$-block secure against adversaries of strength ℓ $(\ell < k)$. ∎

For instance, according to Theorem 7, for $n = 7$, $k = 5$, $d = 6$, the degradation of the block security level of a Cauchy-based product-matrix code is illustrated in Fig. 6. Note that the file size is $\mathcal{M} = 5 \times 6 - \binom{5}{2} = 20$.

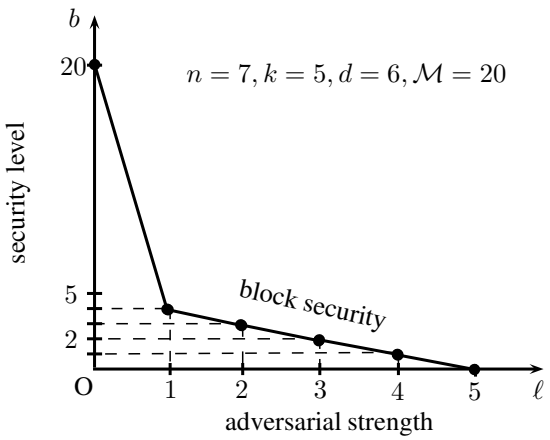

Fig. 6: Degradation of the security level for $\mathcal{D}(7,5,6)$ with a Cauchy-based product-matrix code as the adversarial strength increases.

Similar to Corollary 5, we can establish that if Cauchy matrices are used in the construction of perfectly secure product-matrix codes [11], the codes remain $(k-\ell)$-block secure even when the adversarial strength ℓ surpasses the specified threshold for perfect security $(\ell < k)$. For example, for a DSS $\mathcal{D}(7,5,6)$ as in Fig. 6, suppose that the specified security threshold is $\lambda = 2$, then the degradation of the security is depicted in Fig. 7. Note that now the file size has to be decreased to $\mathcal{M}^{\mathrm{s}} = 9$, according to (2).

B. Vandermonde-Based Product-Matrix Codes Are Not Block Secure

The Vandermonde-based product-matrix codes are not 1-block (weakly) secure against an adversary of strength $k-1$. Indeed, consider the example from the original paper on product-matrix codes in Fig. 1, Section IV [14]. In that example, $k = 3$. By observing two nodes, Node 1 and Node 6, the adversary obtains two linear combinations $f_7 + f_8 + f_9$ and $f_7 + 6f_8 + f_9$. From these two, the adversary can deduce f_8. Hence, the code used in this example, which is based

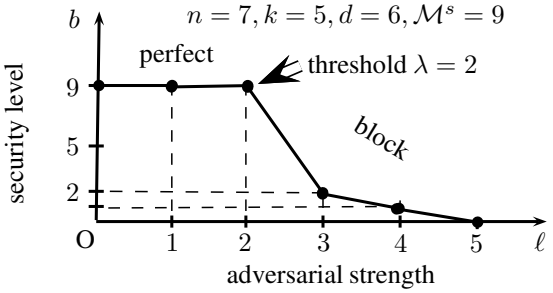

Fig. 7: Degradation of the security level for $\mathcal{D}(7, 5, 6)$ with a perfectly secure Cauchy-based product-matrix code as the adversarial strength increases.

on a Vandermonde matrix, is not weakly secure against an adversary of strength two. The Cauchy-based code, however, is weakly secure against an adversary of strength two.

V. Conclusion

We have evaluated the security levels of the two well-known MBR regenerating codes with exact repair, namely, regular-graph codes and product-matrix codes, employing a more general concept of security - block security. Block security provides weaker levels of security compared with perfect security, however, does not require a trade-off with the storage capacity. We established that Cauchy matrices play a vital role in guaranteeing block security for these MBR codes. Examining the security levels of known MSR regenerating codes is an interesting direction for future work.

References

[1] S. Rhea, C. Wells, P. Eaton, D. Geels, B. Zhao, H. Weatherspoon, and J. Kubiatowicz, "Maintenance-free global data storage," *IEEE Internet Computing*, vol. 5, no. 5, pp. 40–49, 2001.

[2] R. Bhagwan, K. Tati, Y.-C. Cheng, S. Savage, and G. M. Voelker, "Total recall: system support for automated availability management," in *Proceedings of the 1st ACM/USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2004, pp. 25–25.

[3] F. Dabek, J. Li, E. Sit, J. Robertson, M. Kaashoek, and R. Morris, "Designing a dht for low latency and high throughput," in *Proceedings of the 1st ACM/USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2004, pp. 7–7.

[4] H. Weatherspoon and J. Kubiatowicz, "Erasure coding vs. replication: A quantitative comparison," in *Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS)*, 2001, pp. 328–338.

[5] A. Dimakis, P. Godfrey, M. Wainwright, and K. Ramchandran, "Network Coding for distributed storage systems," in *Proceedings of the 26th IEEE International Conference on Computer Communications (INFOCOM)*, 2007, pp. 2000–2008.

[6] S. Pawar, S. E. Rouayheb, and K. Ramchandran, "Securing dynamic distributed storage systems from malicious nodes," in *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, 2011, pp. 1452–1456.

[7] ——, "Securing dynamic distributed storage systems against eavesdropping and adversarial attacks," *IEEE Transactions on Information Theory*, vol. 57, no. 10, pp. 6734–6753, 2011.

[8] K. V. Rashmi, N. B. Shah, and P. V. Kumar, "Regenerating codes for errors and erasures in distributed storage," in *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, 2012, pp. 1202–1206.

[9] T. K. Dikaliotis, A. G. Dimakis, and T. Ho, "Security in distributed storage systems by communicating a logarithmic number of bits," in *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, 2010, pp. 1948–1952.

[10] S. Pawar, S. E. Rouayheb, and K. Ramchandran, "On secure distributed data storage under repair dynamics," in *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, 2010, pp. 2543–2547.

[11] N. B. Shah, K. V. Rashmi, and P. V. Kumar, "Information-theoretically secure regenerating codes for distributed storage," in *Proceedings of the Global Telecommunications Conference (GLOBECOM)*, 2011, pp. 1–5.

[12] A. S. Rawat, O. O. Koyluoglu, N. Silberstein, and S. Vishwanath, "Secure locally repairable codes for distributed storage systems," in *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, 2013, pp. 2224–2228.

[13] S. Goparaju, S. E. Rouayheb, R. Calderbank, and H. V. Poor, "Data secrecy in distributed storage systems under exact repair," in *Proceedings of the International Symposium on Network Coding (NetCod)*, 2013, pp. 1–6.

[14] 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.

[15] S. E. Rouayheb and K. Ramchandran, "Fractional repetition codes for repair in distributed storage systems," in *Proceedings of the 48th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2010, pp. 1510–1517.

[16] K. Bhattad and K. R. Narayanan, "Weakly secure network coding," in *Proceedings of the 1st Workshop on Network Coding, Theory, and Application (NetCod)*, 2005.

[17] D. Silva and F. R. Kschischang, "Universal weakly secure network coding," in *Proceedings of the IEEE Information Theory Workshop (ITW)*, 2009, pp. 281–285.

[18] S. H. Dau, V. Skachek, and Y. M. Chee, "On the security of index coding with side information," *IEEE Transactions on Information Theory*, vol. 58, no. 6, pp. 3975–3988, 2012.

[19] K. V. Rashmi, N. B. Shah, P. V. Kumar, and K. Ramchandran, "Explicit construction of optimal exact regenerating codes for distributed storage," in *Proceedings of the 47th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2009, pp. 1243–1249.

[20] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*. Amsterdam: North-Holland, 1977.

Appendix

A. Proof of Lemma 3

The *only if* direction is proved earlier in the discussion below Lemma 3. It remains to prove the *if* direction.

Let $\tilde{\boldsymbol{f}}$ be the guessed value (by the adversary) for the real file vector $\boldsymbol{f}$. By eavesdropping, the adversary obtains some linear combinations of the coordinates of $\tilde{\boldsymbol{f}}$ and $\boldsymbol{f}$, namely $\boldsymbol{u} = \boldsymbol{E}\tilde{\boldsymbol{f}}^{\mathrm{t}} = \boldsymbol{E}\boldsymbol{f}^{\mathrm{t}}$. We aim to show that for every nontrivial vector $\boldsymbol{v} \in \mathbb{F}_q^{\mathcal{M}}$ of weight less than $\mathsf{d}(\mathscr{C}^{\boldsymbol{E}})$, for the adversary, all possible values for the linear combination $\boldsymbol{v}\tilde{\boldsymbol{f}}^{\mathrm{t}}$ are equally probable. As a consequence, the adversary gains no information about $\boldsymbol{v}\boldsymbol{f}^{\mathrm{t}}$.

Let $\boldsymbol{v} \in \mathbb{F}_q^{\mathcal{M}}$ be any nontrivial vector of weight less than $\mathsf{d}(\mathscr{C}^{\boldsymbol{E}})$ and $\tilde{s}$ an arbitrary element of $\mathbb{F}_q$. It suffices to show that the system of linear equations

$$\begin{cases} \boldsymbol{E}\tilde{\boldsymbol{f}}^{\mathrm{t}} = \boldsymbol{u} \\ \boldsymbol{v}\tilde{\boldsymbol{f}}^{\mathrm{t}} = \tilde{s} \end{cases} . \tag{8}$$

always has the same number of solutions $\tilde{\boldsymbol{f}}$ for every choice of $\tilde{s} \in \mathbb{F}_q$. It is a basic fact from linear algebra that the solution set for the system (8) above, if nonempty, is an affine space, which is the sum of one solution of (8) and the solution space of the corresponding homogeneous system. Therefore, if the system (8) always has at least one solution for every $\tilde{s}$, then it would have the same number of solutions for every $\tilde{s}$.

Therefore, it remains to prove that this system always has a solution for every choice of $\tilde{s} \in \mathbb{F}_q$.

Indeed, let $s = \boldsymbol{v}\boldsymbol{f}^{\mathsf{t}}$. Note that s can be different from $\tilde{s}$. We have

$$\begin{cases} \boldsymbol{E}\boldsymbol{f}^{\mathsf{t}} = \boldsymbol{u} \\ \boldsymbol{v}\boldsymbol{f}^{\mathsf{t}} = s \end{cases}. \tag{9}$$

By subtracting the corresponding equations in the two systems (8) and (9) and let $\boldsymbol{x} = (\tilde{\boldsymbol{f}} - \boldsymbol{f})$ be the new unknowns, we obtain the following system

$$\begin{cases} \boldsymbol{E}\boldsymbol{x}^{\mathsf{t}} = \boldsymbol{0} \\ \boldsymbol{v}\boldsymbol{x}^{\mathsf{t}} = \tilde{s} - s \end{cases}. \tag{10}$$

It is clear that the system (8) has a solution if and only if the system (10) has a solution. Hence, it suffices to show that the system (10) always has a solution for every choice of $\tilde{s} \in \mathbb{F}_q$.

We claim that there exists some $\overline{\boldsymbol{x}} \in \mathbb{F}_q^{\mathcal{M}}$ satisfying $\boldsymbol{E}\overline{\boldsymbol{x}}^{\mathsf{t}} = \boldsymbol{0}$ and $\boldsymbol{v}\overline{\boldsymbol{x}}^{\mathsf{t}} \neq 0$. Then it is obvious that

$$\boldsymbol{x}^* = \frac{\tilde{s} - s}{\boldsymbol{v}\overline{\boldsymbol{x}}^{\mathsf{t}}}\overline{\boldsymbol{x}}$$

would be a solution of (10), and hence the proof would follow. Indeed, if $\boldsymbol{v}\boldsymbol{x}^{\mathsf{t}} = 0$ for every $\boldsymbol{x}$ satisfying $\boldsymbol{E}\boldsymbol{x}^{\mathsf{t}} = \boldsymbol{0}$ then $\boldsymbol{v}$ must belong to the orthogonal complement of the solution space of the system $\boldsymbol{E}\boldsymbol{x}^{\mathsf{t}} = \boldsymbol{0}$, which is precisely the row space of $\boldsymbol{E}$. However, the row space of $\boldsymbol{E}$ contains no nontrivial vector of weight less than $\mathsf{d}(\mathscr{C}^{\boldsymbol{E}})$. This would cause a contradiction since we assume from the beginning that $\boldsymbol{v}$ is nontrivial and has weight less than $\mathsf{d}(\mathscr{C}^{\boldsymbol{E}})$. Thus, there must exist some $\overline{\boldsymbol{x}} \in \mathbb{F}_q^{\mathcal{M}}$ satisfying $\boldsymbol{E}\overline{\boldsymbol{x}}^{\mathsf{t}} = \boldsymbol{0}$ and $\boldsymbol{v}\overline{\boldsymbol{x}}^{\mathsf{t}} \neq 0$, as claimed above.

B. Proof of Lemma 8

We can rewrite the systems (5) as

$$\boldsymbol{H}[j] = \boldsymbol{E}(\boldsymbol{M}[j]), \quad j \in [d].$$

For each $j \in [d]$, the above system is equivalent to

$$\boldsymbol{H}[j] = [\boldsymbol{b}_{j,\xi_1} \mid \boldsymbol{b}_{j,\xi_2} \mid \cdots \mid \boldsymbol{b}_{j,\xi_{\mathcal{M}}}]\boldsymbol{f}, \tag{11}$$

where $\boldsymbol{b}_{j,\xi}$ is given by (6), $\{\xi_1, \xi_2, \ldots, \xi_{\mathcal{M}}\}$ forms the index set $\mathcal{I}$ defined in (4) in the lexicographic order, and $\boldsymbol{f} = (f_{\xi_1}, f_{\xi_2}, \ldots, f_{\xi_{\mathcal{M}}})$ is the stored file. By vertically concatenating the systems (11) for all $j \in [d]$, we obtain (7).

As an example, let $k = 3$ and $d = 5$. Then $\mathcal{M} = 12$ and the index set $\mathcal{I}$ is

$$\begin{aligned} \mathcal{I} &= \{\xi_1, \xi_2, \ldots, \xi_{12}\} \\ &= \big\{(1,1), (1,2), (1,3), (1,4), (1,5), (2,2), (2,3), (2,4), \\ &\qquad (2,5), (3,3), (3,4), (3,5)\big\}. \end{aligned}$$

The the message matrix is

$$\boldsymbol{M} = \begin{pmatrix} f_{(1,1)} & f_{(1,2)} & f_{(1,3)} & f_{(1,4)} & f_{(1,5)} \\ f_{(1,2)} & f_{(2,2)} & f_{(2,3)} & f_{(2,4)} & f_{(2,5)} \\ f_{(1,3)} & f_{(2,3)} & f_{(3,3)} & f_{(3,4)} & f_{(3,5)} \\ f_{(1,4)} & f_{(2,4)} & f_{(3,4)} & 0 & 0 \\ f_{(1,5)} & f_{(2,5)} & f_{(3,5)} & 0 & 0 \end{pmatrix}.$$

The file is

$$\begin{aligned} \boldsymbol{f} = (&f_{(1,1)}, f_{(1,2)}, f_{(1,3)}, f_{(1,4)}, f_{(1,5)}, f_{(2,2)}, f_{(2,3)}, f_{(2,4)}, \\ &f_{(2,5)}, f_{(3,3)}, f_{(3,4)}, f_{(3,5)}). \end{aligned}$$

The matrix $\overline{\boldsymbol{E}}$ is (to save space we replace $\boldsymbol{E}[i]$ by $\boldsymbol{e}_i$, columns are indexed by $\xi_1, \xi_2, \ldots, \xi_{12}$)

$$\overline{\boldsymbol{E}} = \begin{array}{c} \begin{matrix} \xi_1 & \xi_2 & \xi_3 & \xi_4 & \xi_5 & \xi_6 & \xi_7 & \xi_8 & \xi_9 & \xi_{10} & \xi_{11} & \xi_{12} \end{matrix} \\ \begin{pmatrix} \boldsymbol{e}_1 & \boldsymbol{e}_2 & \boldsymbol{e}_3 & \boldsymbol{e}_4 & \boldsymbol{e}_5 & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{e}_1 & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{e}_2 & \boldsymbol{e}_3 & \boldsymbol{e}_4 & \boldsymbol{e}_5 & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{e}_1 & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{e}_2 & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{e}_3 & \boldsymbol{e}_4 & \boldsymbol{e}_5 \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{e}_1 & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{e}_2 & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{e}_3 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{e}_1 & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{e}_2 & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{e}_3 \end{pmatrix} \end{array}.$$

C. Proof of Lemma 9

Let $\mathscr{C}^{\overline{\boldsymbol{E}}}$ be the error-correcting code generated by the rows of $\overline{\boldsymbol{E}}$. We aim to prove that for every codeword of $\mathscr{C}^{\overline{\boldsymbol{E}}}$, each group of $k - \ell$ coordinates can be represented as linear combinations of the other $\mathcal{M} - (k - \ell)$ coordinates, which do not belong to the group. As a consequence, if a codeword $\boldsymbol{c}^{\overline{\boldsymbol{E}}} \in \mathscr{C}^{\overline{\boldsymbol{E}}}$ has weight at most $k-\ell$, i.e. it has some $\mathcal{M}-(k-\ell)$ zero coordinates, then its remaining $k - \ell$ coordinate must also be zero, and hence $\boldsymbol{c}^{\overline{\boldsymbol{E}}} = \boldsymbol{0}$. Therefore, every nonzero codeword in $\mathscr{C}^{\overline{\boldsymbol{E}}}$ has weight at least $k - \ell + 1$. Hence $\mathsf{d}(\mathscr{C}^{\overline{\boldsymbol{E}}}) \geq k-\ell+1$. This constitutes the most challenging part in the proof of Lemma 9. The proof of the other direction, namely $\mathsf{d}(\mathscr{C}^{\overline{\boldsymbol{E}}}) \leq k - \ell + 1$, is almost obvious, as we shall see later.

We now introduce some necessary notations for the proof and establish the relationship among them. Let $\mathscr{C}^{\boldsymbol{E}}$ be the error-correcting code generated by the rows of $\boldsymbol{E}$. Since $\boldsymbol{E}$ is an $\ell \times d$ Cauchy matrix, $\mathscr{C}^{\boldsymbol{E}}$ is a $[d, \ell]$ MDS code (see Proof of Theorem 4). Also, any set of ℓ columns of $\boldsymbol{E}$ generate the whole column space. Therefore, for any ℓ-subset L of $[d]$ and any index $i \in [d] \setminus L$, we have

$$\boldsymbol{E}[i] = \sum_{s \in L} a_{s,i}(L)\boldsymbol{E}[s],$$

for some coefficients $a_{s,i}(L) \in \mathbb{F}_q$. As a consequence, for any codeword $\boldsymbol{c}^{\boldsymbol{E}} = (c_1^{\boldsymbol{E}}, \ldots, c_d^{\boldsymbol{E}}) \in \mathscr{C}^{\boldsymbol{E}}$, we have

$$c_i^{\boldsymbol{E}} = \sum_{s \in L} a_{s,i}(L)c_s^{\boldsymbol{E}} \quad (i \in [d] \setminus L). \tag{12}$$

Note that as any set of ℓ columns of $\boldsymbol{E}$ is linear independent, the coefficients $a_{s,i}(L)$ are all nonzero and uniquely determined by $\boldsymbol{E}$ and L. From now on we only consider the case $L \subset [k]$. We often write $a_{i,j}$ instead of $a_{i,j}(L)$ to simplify the notation.

For $j \in [d]$, let $\mathscr{C}^j$ be the row space of the matrix $\boldsymbol{B}^j = [\boldsymbol{b}_{j,\xi_1} \mid \boldsymbol{b}_{j,\xi_2} \mid \cdots \mid \boldsymbol{b}_{j,\xi_{\mathcal{M}}}]$. Then $\mathscr{C}^{\overline{\boldsymbol{E}}} = \mathscr{C}^1 + \mathscr{C}^2 + \cdots + \mathscr{C}^d$ as a sum of spaces. Note that according to the definition of $\boldsymbol{b}_{j,\xi}$ in (6),

- for $1 \leq j \leq k$: $\boldsymbol{b}_{j,\xi} = \boldsymbol{E}[i]$ if $\xi = (i,j) \in \mathcal{I}$, i.e. $1 \leq i \leq j$, or $\xi = (j,i) \in \mathcal{I}$, i.e. $j + 1 \leq i \leq d$,
- for $k < j \leq d$: $\boldsymbol{b}_{j,\xi} = \boldsymbol{E}[i]$ if $\xi = (i,j) \in \mathcal{I}$, i.e. $1 \leq i \leq k$.

Therefore, when $j \leq k$, the matrix $\boldsymbol{B}^j$ has precisely d nonzero columns, which are the same as d columns of $\boldsymbol{E}$. On the other hand, when $j > k$, $\boldsymbol{B}^j$ has precisely k nonzero columns, which are the same as the first k columns of $\boldsymbol{E}$. Hence, for each codeword $\boldsymbol{c}^j = (c^j_{\xi_1}, \ldots, c^j_{\xi_{\mathcal{M}}}) \in \mathscr{C}^j$ and each $\xi \in \mathcal{I}$, we have

$$c^j_\xi = \begin{cases} c^{\boldsymbol{E}}_{i,j}, & \text{if } \xi = (i,j) \text{ or } \xi = (j,i), \\ 0, & \text{otherwise,} \end{cases} \tag{13}$$

where

- $(c^{\boldsymbol{E}}_{1,j}, c^{\boldsymbol{E}}_{2,j}, \ldots, c^{\boldsymbol{E}}_{d,j})$ is a codeword of $\mathscr{C}^{\boldsymbol{E}}$, if $1 \leq j \leq k$,
- $(c^{\boldsymbol{E}}_{1,j}, c^{\boldsymbol{E}}_{2,j}, \ldots, c^{\boldsymbol{E}}_{k,j})$ is a codeword of $\mathscr{C}^{\boldsymbol{E}}$ restricted on the first k coordinates, if $k < j \leq d$.

Since $\mathscr{C}^{\overline{\boldsymbol{E}}} = \mathscr{C}^1 + \mathscr{C}^2 + \cdots + \mathscr{C}^d$, for each codeword $(c^{\overline{\boldsymbol{E}}}_{\xi_1}, c^{\overline{\boldsymbol{E}}}_{\xi_2}, \ldots, c^{\overline{\boldsymbol{E}}}_{\xi_{\mathcal{M}}}) \in \mathscr{C}^{\overline{\boldsymbol{E}}}$ and each $\xi \in \mathcal{I}$, we have

$$c^{\overline{\boldsymbol{E}}}_\xi = \begin{cases} c^{\boldsymbol{E}}_{j,j}, & \text{if } \xi = (j,j), \\ c^{\boldsymbol{E}}_{i,j} + c^{\boldsymbol{E}}_{j,i}, & \text{if } \xi = (i,j) \text{ and } i < j. \end{cases} \tag{14}$$

Note that $c^{\boldsymbol{E}}_{i,j}$ is the ith coordinate of a codeword $\boldsymbol{c}^{\boldsymbol{E}}_j = (c^{\boldsymbol{E}}_{1,j}, c^{\boldsymbol{E}}_{2,j}, \ldots, c^{\boldsymbol{E}}_{d,j})$ of $\mathscr{C}^{\boldsymbol{E}}$ and $c^{\boldsymbol{E}}_{j,i}$ is the jth coordinate of another codeword $\boldsymbol{c}^{\boldsymbol{E}}_i = (c^{\boldsymbol{E}}_{1,i}, c^{\boldsymbol{E}}_{2,i}, \ldots, c^{\boldsymbol{E}}_{d,i})$ of $\mathscr{C}^{\boldsymbol{E}}$.

Due to (12), for every ℓ-subset L of $[k]$ and for every $j \in [d]$ we have

$$c^{\boldsymbol{E}}_{i,j} = \sum_{s \in L} a_{s,i} c^{\boldsymbol{E}}_{s,j}, \quad \text{for } i \in [d] \setminus L. \tag{15}$$

We now prove that $\mathsf{d}(\mathscr{C}^{\overline{\boldsymbol{E}}}) \leq k - \ell + 1$.

- If $d > k$ then $\mathsf{d}(\mathscr{C}^j) = k - \ell + 1$, for every $j > k$, $j \leq d$. Indeed, according to the definition, $\mathscr{C}^j$ is the row space of the matrix $\boldsymbol{B}^j$. Moreover, according to the discussion above, for $j > k$, the matrix $\boldsymbol{B}^j$ has precisely k nonzero columns, which are the same as the first k columns of $\boldsymbol{E}$. These k nonzero columns form an $\ell \times k$ Cauchy matrix, which in turn generates a $[k, \ell]$ MDS code of minimum distance $k - \ell + 1$ (see Proof of Theorem 4). As other columns of $\boldsymbol{B}^j$ are all-zero columns, we conclude that $\mathsf{d}(\mathscr{C}^j) = k - \ell + 1$. As $\mathscr{C}^{\overline{\boldsymbol{E}}} = \mathscr{C}^1 + \mathscr{C}^2 + \cdots + \mathscr{C}^d$ as a sum of spaces, we deduce that $\mathsf{d}(\mathscr{C}^{\overline{\boldsymbol{E}}}) \leq k - \ell + 1$.
- If $d = k$, then similar argument applies to $\mathscr{C}^j$ for any $j \in [k]$.

Thus, in both cases, $\mathsf{d}(\mathscr{C}^{\overline{\boldsymbol{E}}}) \leq k - \ell + 1$.

We now proceed to the most important part of the proof of Lemma 9. Our goal is to show that $\mathsf{d}(\mathscr{C}^{\overline{\boldsymbol{E}}}) \geq k - \ell + 1$. Let $\boldsymbol{c}^{\overline{\boldsymbol{E}}} = (c^{\overline{\boldsymbol{E}}}_{(1,1)}, c^{\overline{\boldsymbol{E}}}_{(1,2)}, \ldots, c^{\overline{\boldsymbol{E}}}_{(k,d)})$ be any codeword of $\mathscr{C}^{\overline{\boldsymbol{E}}}$. Let U be any subset of $k - \ell$ elements of the index set $\mathcal{I} = \{(1,1), (1,2), \ldots, (k,d)\}$. We prove below that the coordinates of $\boldsymbol{c}^{\overline{\boldsymbol{E}}}$ indexed by the elements of U can be represented as linear combinations of the coordinates indexed by the elements of $\overline{U} = \mathcal{I} \setminus U$. According to the discussion at the beginning of the proof of Lemma 9, this implies that $\mathsf{d}(\mathscr{C}^{\overline{\boldsymbol{E}}}) \geq k - \ell + 1$. It suffices to show that for every index $(s,t) \in U$, the corresponding coordinate $c^{\overline{\boldsymbol{E}}}_{(s,t)}$ can be written as a linear combination of $c^{\overline{\boldsymbol{E}}}_{(i,j)}$'s where $(i,j) \in \overline{U}$. We divide the proof into different cases, depending on whether $s = t$ or $s \neq t$. Three additional lemmas are introduced below to tackle those cases separately. We henceforth drop $\overline{\boldsymbol{E}}$ from the notation $\boldsymbol{c}^{\overline{\boldsymbol{E}}}$ to simplify the presentation.

Lemma 10. *Let $\boldsymbol{c} = (c_{(1,1)}, c_{(1,2)}, \ldots, c_{(k,d)})$ be an arbitrary codeword of $\mathscr{C}^{\overline{\boldsymbol{E}}}$. Suppose that $(t,t) \in U$ ($t \in [k]$). Then there exists an ℓ-subset L of $[k]$ such that*

$$c_{(t,t)} = \sum_{i \in L} a_{i,t}(L) c^*_{(i,t)} - \sum_{\substack{i \leq j \\ i,j \in L}} a_{i,t}(L) a_{j,t}(L) c_{(i,j)}, \tag{16}$$

where

$$c^*_{(i,t)} = \begin{cases} c_{(i,t)}, & \text{if } i \leq t, \\ c_{(t,i)}, & \text{otherwise.} \end{cases}$$

Moreover, none of the indices (i,t) (or (t,i)) and (i,j) are in U for every $i \in L$ and $j \in L$.

According to Lemma 10, $c_{(t,t)}$ can be written as a linear combination of the coordinates of $\boldsymbol{c}^{\overline{\boldsymbol{E}}}$ indexed by the elements in $\overline{U}$, which is precisely what we want to show. Hence the case $(s = t, t) \in U$ is settled.

Proof of Lemma 10: We first construct an appropriate subset L such that none of the sub-indices of the terms in the right-hand side of (16) belong to U. Once such a subset is chosen, we proceed to prove that (16) indeed holds.

	Elements	Count
Starts with t	$(t,u_1), (t,u_2), \ldots, (t,u_a = t)$	a
Ends with t	$(s_1,t), (s_2,t), \ldots, (s_b = t, t)$	b
Not starts nor ends with t	$(p_1,q_1), (p_2,q_2), \ldots, (p_c,q_c)$	c

TABLE I: All elements of U - Case 1.

We list all elements of U in Table I. Note that as the element (t,t) is counted twice, we have $|U| = k - \ell = a + b + c - 1$. It is straightforward that if L does not contain any element in the 'bad' set

$$L^* = \{u_1, \ldots, u_a = t\} \cup \{s_1, \ldots, s_{b-1}\} \cup \{q_1, \ldots, q_c\},$$

then none of the sub-indices of the terms in the right-hand side of (16) belong to U, as desired. As $|L^*| \leq a + (b-1) + c = k - \ell$, the set $[k] \setminus L^*$ has cardinality at least ℓ. Therefore we can choose a legitimate L by taking an arbitrary subset of ℓ elements of $[k] \setminus L^*$.

We now show that (16) holds for L chosen as above. By (14) and (15) we have

$$\begin{aligned} \sum_{i \in L} a_{i,t} c^*_{(i,t)} &\underset{i \neq t}{=} \sum_{i \in L} a_{i,t}(c^{\boldsymbol{E}}_{i,t} + c^{\boldsymbol{E}}_{t,i}) \\ &= \sum_{i \in L} a_{i,t} c^{\boldsymbol{E}}_{i,t} + \sum_{i \in L} a_{i,t} c^{\boldsymbol{E}}_{t,i} \\ &= c^{\boldsymbol{E}}_{t,t} + \sum_{i \in L} a_{i,t} (\sum_{j \in L} a_{j,t} c^{\boldsymbol{E}}_{j,i}) \\ &= c_{(t,t)} + \sum_{i \in L} \sum_{j \in L} a_{i,t} a_{j,t} c^{\boldsymbol{E}}_{j,i} \\ &= c_{(t,t)} + \sum_{i \in L} \sum_{j \in L} a_{i,t} a_{j,t} c^{\boldsymbol{E}}_{i,j}, \end{aligned} \tag{17}$$

where in the last transition, the indices i and j are swapped. Also by (14) we have

$$\begin{aligned}
&\sum_{\substack{i\le j\\ i,j\in L}} a_{i,t}a_{j,t}c_{(i,j)}\\
&= \sum_{\substack{i=j\\ i,j\in L}} a_{i,t}a_{j,t}c_{(i,j)} + \sum_{\substack{i<j\\ i,j\in L}} a_{i,t}a_{j,t}(c^{\boldsymbol{E}}_{i,j} + c^{\boldsymbol{E}}_{j,i})\\
&= \sum_{\substack{i=j\\ i,j\in L}} a_{i,t}a_{j,t}c^{\boldsymbol{E}}_{i,j} + \sum_{\substack{i<j\\ i,j\in L}} a_{i,t}a_{j,t}c^{\boldsymbol{E}}_{i,j} + \sum_{\substack{i<j\\ i,j\in L}} a_{i,t}a_{j,t}c^{\boldsymbol{E}}_{j,i}\\
&= \sum_{\substack{i=j\\ i,j\in L}} a_{i,t}a_{j,t}c^{\boldsymbol{E}}_{i,j} + \sum_{\substack{i<j\\ i,j\in L}} a_{i,t}a_{j,t}c^{\boldsymbol{E}}_{i,j} + \sum_{\substack{j<i\\ i,j\in L}} a_{i,t}a_{j,t}c^{\boldsymbol{E}}_{i,j}\\
&= \sum_{i\in L}\sum_{j\in L} a_{i,t}a_{j,t}c^{\boldsymbol{E}}_{i,j},
\end{aligned} \tag{18}$$

where the second to last transition is done by swapping the indices i and j in the third sum. Combining (17) and (18) we finish the proof of the Lemma 10 . ∎

We now consider $(s,t) \in U$ where $s \neq t$. We examine another two cases, depending on whether $t \le k$ or $t > k$.

Lemma 11. *Let $\boldsymbol{c} = (c_{(1,1)}, c_{(1,2)}, \ldots, c_{(k,d)})$ be an arbitrary codeword of $\mathscr{C}^{\overline{\boldsymbol{E}}}$. Suppose that $(s,t) \in U$ and $t \in [k]$. Then there exists an ℓ-subset L of $[k]$ such that $s \in L$, $t \notin L$, and*

$$c_{(t,t)} = \sum_{i\in L} a_{i,t}(L)c^*_{(i,t)} - \sum_{\substack{i\le j\\ i,j\in L}} a_{i,t}(L)a_{j,t}(L)c_{(i,j)}, \tag{19}$$

where

$$c^*_{(i,t)} = \begin{cases} c_{(i,t)}, & \text{if } i \le t,\\ c_{(t,i)}, & \text{otherwise.}\end{cases}$$

Moreover, for every $i \in L$ and $j \in L$, none of the indices (i,t) (or (t,i)), except (s,t), and (i,j), except (s,s), are in U.

As a consequence of Lemma 11, $c_{(t,t)}$ can be written as a linear combination of $c_{(s,t)}$, $c_{(s,s)}$, and the coordinates of $\boldsymbol{c}^{\overline{\boldsymbol{E}}}$ indexed by the elements in $\overline{U}$. As noted before, the coefficient of $c_{(s,t)}$, namely $a_{s,t}$, is nonzero. Therefore, $c_{(s,t)}$ can also be written as a linear combination of $c_{(t,t)}$, $c_{(s,s)}$, and the coordinates of $\boldsymbol{c}^{\overline{\boldsymbol{E}}}$ indexed by the elements in $\overline{U}$. Moreover, according to Lemma 10, if $(t,t) \in U$ then $c_{(t,t)}$ can be written as a linear combination of the coordinates of $\boldsymbol{c}^{\overline{\boldsymbol{E}}}$ indexed by the elements in $\overline{U}$. Similar assertion holds for $c_{(s,s)}$ if $(s,s) \in U$. Thus, we conclude that $c_{(s,t)}$ can be written as a linear combination of just the coordinates of $\boldsymbol{c}^{\overline{\boldsymbol{E}}}$ indexed by the elements in $\overline{U}$. Hence the case $(s,t) \in U$, $s \neq t \le k$ is settled.

Proof of Lemma 11: We first construct an appropriate subset L such that $s \in L$, $t \notin L$, and none of the sub-indices of the terms in the right-hand side of (19) belong to U, except for (s,t) and (s,s). Once such a subset is chosen, the proof that (19) indeed holds is the same as that of Lemma 10. We divide the proof into four sub-cases, depending on whether $(s,s) \in U$ and/or $(t,t) \in U$.

Sub-case 2-1: $(s,s) \notin U$, $(t,t) \notin U$. We list all elements of U in Table II.

	Elements	Count
Starts with s	$(s,t_1),(s,t_2),\ldots,(s,t_a=t)$	a
Ends with s	$(r_1,s),(r_2,s),\ldots,(r_b,s)$	b
Starts with t	$(t,u_1),(t,u_2),\ldots,(t,u_c)$	c
Ends with t	$(s_1=s,t),(s_2,t),\ldots,(s_d,t)$	d
Not starts/ends with s,t	$(p_1,q_1),(p_2,q_2),\ldots,(p_e,q_e)$	e

TABLE II: All elements of U - Sub-case 2-1.

Note that as the element (s,t) is counted twice, we have $|U| = k - \ell = a + b + c + d + e - 1$. The 'bad' set to be excluded is

$$\begin{aligned}L^* = &\{t_1,\ldots,t_a=t\} \cup \{r_1,\ldots,r_b\} \cup \{u_1,\ldots,u_c\}\\ &\cup \{s_2,\ldots,r_d\} \cup \{q_1,\ldots,q_e\}.\end{aligned}$$

As $|L^*| \le a + b + c + (d-1) + e = k - \ell$, the set $[k] \setminus L^*$ has cardinality at least ℓ. Moreover, $s \in [k] \setminus L^*$. Therefore we can choose a legitimate L by taking an arbitrary subset of ℓ elements of $[k] \setminus L^*$ that contains s.

Sub-case 2-2: $(s,s) \in U$, $(t,t) \in U$. We list all elements of U in Table III. Note that as the elements (s,t), (s,s), and (t,t)

	Elements	Count
Starts with s	$(s,t_1=s),(s,t_2),\ldots,(s,t_a=t)$	a
Ends with s	$(r_1=s,s),(r_2,s),\ldots,(r_b,s)$	b
Starts with t	$(t,u_1),(t,u_2),\ldots,(t,u_c=t)$	c
Ends with t	$(s_1=s,t),(s_2,t),\ldots,(s_d=t,t)$	d
Not starts/ends with s,t	$(p_1,q_1),(p_2,q_2),\ldots,(p_e,q_e)$	e

TABLE III: All elements of U - Sub-case 2-2.

are counted twice, we have $|U| = k - \ell = a + b + c + d + e - 3$. The 'bad' set to be excluded is

$$\begin{aligned}L^* = &\{t_2,\ldots,t_a=t\} \cup \{r_2,\ldots,r_b\} \cup \{u_1,\ldots,u_{c-1}\}\\ &\cup \{s_2,\ldots,s_{d-1}\} \cup \{q_1,\ldots,q_e\}.\end{aligned}$$

As $|L^*| \le (a-1) + (b-1) + (c-1) + (d-2) + e < k - \ell$, the set $[k] \setminus L^*$ has cardinality greater than ℓ. Moreover, $s \in [k] \setminus L^*$. Therefore we can choose a legitimate L by taking an arbitrary subset of ℓ elements of $[k] \setminus L^*$ that contains s.

Sub-case 2-3: $(s,s) \in U$, $(t,t) \notin U$. We list all elements of U in Table IV. Note that as the elements (s,t) and (s,s) are

	Elements	Count
Starts with s	$(s,t_1=s),(s,t_2),\ldots,(s,t_a=t)$	a
Ends with s	$(r_1=s,s),(r_2,s),\ldots,(r_b,s)$	b
Starts with t	$(t,u_1),(t,u_2),\ldots,(t,u_c)$	c
Ends with t	$(s_1=s,t),(s_2,t),\ldots,(s_d,t)$	d
Not starts/ends with s,t	$(p_1,q_1),(p_2,q_2),\ldots,(p_e,q_e)$	e

TABLE IV: All elements of U - Sub-case 2-3.

counted twice, we have $|U| = k - \ell = a + b + c + d + e - 2$. The 'bad' set to be excluded is

$$\begin{aligned}L^* = &\{t_2,\ldots,t_a=t\} \cup \{r_2,\ldots,r_b\} \cup \{u_1,\ldots,u_c\}\\ &\cup \{s_2,\ldots,s_d\} \cup \{q_1,\ldots,q_e\}.\end{aligned}$$

As $|L^*| \leq (a-1)+(b-1)+c+(d-1)+e < k-\ell$, the set $[k] \setminus L^*$ has cardinality greater than ℓ. Moreover, $s \in [k] \setminus L^*$. Therefore we can choose a legitimate L by taking an arbitrary subset of ℓ elements of $[k] \setminus L^*$ that contains s.

Sub-case 2-4: $(s,s) \notin U$, $(t,t) \in U$. We list all elements of U in Table V. Note that as the elements (s,t) and (t,t) are

	Elements	Count
Starts with s	$(s,t_1),(s,t_2),\ldots,(s,t_a=t)$	a
Ends with s	$(r_1,s),(r_2,s),\ldots,(r_b,s)$	b
Starts with t	$(t,u_1),(t,u_2),\ldots,(t,u_c=t)$	c
Ends with t	$(s_1=s,t),(s_2,t),\ldots,(s_d=t,t)$	d
Not starts/ends with s,t	$(p_1,q_1),(p_2,q_2),\ldots,(p_e,q_e)$	e

TABLE V: All elements of U - Sub-case 2-4.

counted twice, we have $|U| = k-\ell = a+b+c+d+e-2$. The 'bad' set to be excluded is

$$L^* = \{t_1,\ldots,t_a=t\} \cup \{r_1,\ldots,r_b\} \cup \{u_1,\ldots,u_{c-1}\} \cup \{s_2,\ldots,s_{d-1}\} \cup \{q_1,\ldots,q_e\}.$$

As $|L^*| \leq a+b+(c-1)+(d-2)+e < k-\ell$, the set $[k] \setminus L^*$ has cardinality greater than ℓ. Moreover, $s \in [k] \setminus L^*$. Therefore we can choose a legitimate L by taking an arbitrary subset of ℓ elements of $[k] \setminus L^*$ that contains s.

Thus, we complete the proof of Lemma 11. ∎

Lemma 12. *Let $\mathbf{c} = (c_{(1,1)}, c_{(1,2)}, \ldots, c_{(k,d)})$ be an arbitrary codeword of $\mathscr{C}^{\overline{E}}$. Suppose that $(s,t) \in U$, $s \neq t$, and $t > k$. Then there exists an ℓ-subset L of $[k]$ such that*

$$c_{(s,t)} = \sum_{i \in L} a_{i,t}(L) c^*_{(i,s)} + \sum_{i \in L} a_{i,s}(L) c_{(i,t)} - \sum_{\substack{i \leq j \\ i,j \in L}} \big(a_{i,t}(L)a_{j,s}(L) + a_{i,s}(L)a_{j,t}(L)\big) c_{(i,j)} \tag{20}$$

where

$$c^*_{(i,s)} = \begin{cases} c_{(i,s)}, & \text{if } i \leq s, \\ c_{(s,i)}, & \text{otherwise.} \end{cases}$$

Moreover, for every $i \in L$ and $j \in L$, none of the indices (i,t), (i,s) (or (s,i)), and (i,j), are in U.

According to Lemma 12, for $(s,t) \in U$, $s \neq t$, $t > k$, the coordinate $c_{(s,t)}$ can be written as a linear combination of the coordinates of $\boldsymbol{c}^{\overline{E}}$ indexed by the elements in $\overline{U}$, which is precisely what we want to show. Hence this very last case is settled. As a consequence, the proof of Lemma 9 also follows.

Proof of Lemma 12: We first construct an appropriate subset L such that none of the sub-indices of the terms in the right-hand side of (20) belong to U. Once such a subset is chosen, we proceed to prove that (20) indeed holds. We consider two sub-cases, namely $(s,s) \in U$ and $(s,s) \notin U$. Note that as $t > k$, according to the definition of $\mathcal{I}$ in (4), (t,t) is not a valid index, and hence we do not have to consider the element (t,t).

Sub-case 3-1: $(s,s) \notin U$. We list all elements of U in Table VI. Since $t > k$, note that there is no valid index of the form (t,u), according to the definition of $\mathcal{I}$ in (4). Note that as the element (s,t) is counted twice, we have

	Elements	Count
Starts with s	$(s,t_1),(s,t_2),\ldots,(s,t_a=t)$	a
Ends with s	$(r_1,s),(r_2,s),\ldots,(r_b,s)$	b
Ends with t	$(s_1=s,t),(s_2,t),\ldots,(s_c,t)$	c
Not starts/ends with s,t	$(p_1,q_1),(p_2,q_2),\ldots,(p_d,q_d)$	d

TABLE VI: All elements of U - Sub-case 3-1.

$|U| = k-\ell = a+b+c+d-1$. It is straightforward that if L does not contain any element in the 'bad' set

$$L^* = \{t_1,\ldots,t_{a-1}\} \cup \{r_1,\ldots,r_b\} \cup \{s_1,\ldots,s_c\} \cup \{q_1,\ldots,q_d\},$$

then none of the sub-indices of the terms in the right-hand side of (20) belong to U, as desired. Note that here we do not have to exclude t, because we are about to choose L as a subset of $[k]$ whereas we already assume that $t \notin [k]$ for this case. As $|L^*| \leq (a-1)+b+c+d = k-\ell$, the set $[k] \setminus L^*$ has cardinality at least ℓ. Therefore we can choose a legitimate L by taking an arbitrary subset of ℓ elements of $[k] \setminus L^*$.

Sub-case 3-2: $(s,s) \in U$. We list all elements of U in Table VII. Since $t > k$, note that there is no valid index of the form (t,u), according to the definition of $\mathcal{I}$ in (4). Since the elements (s,t) and (s,s) are counted twice, we have

	Elements	Count
Starts with s	$(s,t_1=s),(s,t_2),\ldots,(s,t_a=t)$	a
Ends with s	$(r_1=s,s),(r_2,s),\ldots,(r_b,s)$	b
Ends with t	$(s_1=s,t),(s_2,t),\ldots,(s_c,t)$	c
Not starts/ends with s,t	$(p_1,q_1),(p_2,q_2),\ldots,(p_d,q_d)$	d

TABLE VII: All elements of U - Sub-case 3-2.

$|U| = k-\ell = a+b+c+d-2$. The 'bad' set to be excluded is

$$L^* = \{t_1,\ldots,t_{a-1}\} \cup \{r_2,\ldots,r_b\} \cup \{s_2,\ldots,s_c\} \cup \{q_1,\ldots,q_d\}.$$

Note that here we do not have to exclude t, because we are about to choose L as a subset of $[k]$ whereas we already assume that $t \notin [k]$ for this case. As $|L^*| \leq (a-1)+(b-1)+(c-1)+d < k-\ell$, the set $[k] \setminus L^*$ has cardinality larger than ℓ. Therefore we can choose a legitimate L by taking an arbitrary subset of ℓ elements of $[k] \setminus L^*$.

Thus, in both cases we can always find an ℓ-subset L of $[k]$ such that none of the sub-indices of the terms in the right-hand side of (20) belong to U. We now show that (20) holds for L chosen as above. Instead of $a_{i,j}(L)$, we simply write $a_{i,j}$ as

there is no possible confusion. Using (14) and (15) we have

$$\begin{aligned}
&\sum_{i\in L} a_{i,t}c^*_{(i,s)} + \sum_{i\in L} a_{i,s}c_{(i,t)} \\
&= \sum_{i\in L} a_{i,t}(c^{\boldsymbol{E}}_{i,s} + c^{\boldsymbol{E}}_{s,i}) + \sum_{i\in L} a_{i,s}(c^{\boldsymbol{E}}_{i,t} + c^{\boldsymbol{E}}_{t,i}) \\
&= \sum_{i\in L} a_{i,t}c^{\boldsymbol{E}}_{i,s} + \sum_{i\in L} a_{i,t}c^{\boldsymbol{E}}_{s,i} + \sum_{i\in L} a_{i,s}c^{\boldsymbol{E}}_{i,t} + \sum_{i\in L} a_{i,s}c^{\boldsymbol{E}}_{t,i} \\
&= c^{\boldsymbol{E}}_{t,s} + \sum_{i\in L} a_{i,t}c^{\boldsymbol{E}}_{s,i} + c^{\boldsymbol{E}}_{s,t} + \sum_{i\in L} a_{i,s}c^{\boldsymbol{E}}_{t,i} \\
&= c_{(s,t)} + \sum_{i\in L} a_{i,t}c^{\boldsymbol{E}}_{s,i} + \sum_{i\in L} a_{i,s}c^{\boldsymbol{E}}_{t,i} \\
&= c_{(s,t)} + \sum_{i\in L}\sum_{j\in L} a_{i,t}a_{j,s}c^{\boldsymbol{E}}_{j,i} + \sum_{i\in L}\sum_{j\in L} a_{i,s}a_{j,t}c^{\boldsymbol{E}}_{j,i} \\
&= c_{(s,t)} + \sum_{i\in L}\sum_{j\in L} a_{j,t}a_{i,s}c^{\boldsymbol{E}}_{i,j} + \sum_{i\in L}\sum_{j\in L} a_{j,s}a_{i,t}c^{\boldsymbol{E}}_{i,j},
\end{aligned} \tag{21}$$

where in the last transition, we swap the indices i and j. We now calculate the remaining sum in the right-hand side of (20). We first split it into two sums and transform each individually. The first sum is

$$\begin{aligned}
&\sum_{\substack{i\le j\\ i,j\in L}} a_{i,t}a_{j,s}c_{(i,j)} \\
&= \sum_{\substack{i=j\\ i,j\in L}} a_{i,t}a_{j,s}c_{(i,j)} + \sum_{\substack{i<j\\ i,j\in L}} a_{i,t}a_{j,s}c_{(i,j)} \\
&= \sum_{\substack{i=j\\ i,j\in L}} a_{i,t}a_{j,s}c^{\boldsymbol{E}}_{i,j} + \sum_{\substack{i<j\\ i,j\in L}} a_{i,t}a_{j,s}(c^{\boldsymbol{E}}_{i,j} + c^{\boldsymbol{E}}_{j,i}) \\
&= \sum_{\substack{i=j\\ i,j\in L}} a_{i,t}a_{j,s}c^{\boldsymbol{E}}_{i,j} + \sum_{\substack{i<j\\ i,j\in L}} a_{i,t}a_{j,s}c^{\boldsymbol{E}}_{i,j} + \sum_{\substack{j<i\\ i,j\in L}} a_{j,t}a_{i,s}c^{\boldsymbol{E}}_{i,j},
\end{aligned} \tag{22}$$

where in the last transition, we swap the indices i and j. Similarly, the second sum is

$$\begin{aligned}
&\sum_{\substack{i\le j\\ i,j\in L}} a_{i,s}a_{j,t}c_{(i,j)} \\
&= \sum_{\substack{i=j\\ i,j\in L}} a_{i,s}a_{j,t}c^{\boldsymbol{E}}_{i,j} + \sum_{\substack{i<j\\ i,j\in L}} a_{i,s}a_{j,t}c^{\boldsymbol{E}}_{i,j} + \sum_{\substack{j<i\\ i,j\in L}} a_{j,s}a_{i,t}c^{\boldsymbol{E}}_{i,j}.
\end{aligned} \tag{23}$$

Combining (22) and (23) and grouping suitable sums together we deduce

$$\begin{aligned}
&\sum_{\substack{i\le j\\ i,j\in L}} (a_{i,t}a_{j,s} + a_{i,s}a_{j,t})c_{(i,j)} \\
&= \sum_{i\in L}\sum_{j\in L} a_{j,t}a_{i,s}c^{\boldsymbol{E}}_{i,j} + \sum_{i\in L}\sum_{j\in L} a_{j,s}a_{i,t}c^{\boldsymbol{E}}_{i,j}.
\end{aligned} \tag{24}$$

From (21) and (24) we derive (20), hence complete the proof of Lemma 12. ∎

By Lemma 10, Lemma 11, and Lemma 12, we show that for every codeword $\boldsymbol{c}^{\overline{\boldsymbol{E}}} = (c^{\overline{\boldsymbol{E}}}_{(1,1)}, c^{\overline{\boldsymbol{E}}}_{(1,2)}, \ldots, c^{\overline{\boldsymbol{E}}}_{(k,d)})$ of $\mathscr{C}^{\overline{\boldsymbol{E}}}$, every $k-\ell$ coordinates of $\boldsymbol{c}^{\overline{\boldsymbol{E}}}$ can be written as linear combinations of the remaining $\mathcal{M} - (k-\ell)$ coordinates. Thus the code $\mathscr{C}^{\overline{\boldsymbol{E}}}$ has minimum distance at least $k-\ell+1$. As we already establish that this code has minimum distance at most $k-\ell+1$, the proof of Lemma 9 follows.