Weakly Secure Regenerating Codes for Distributed Storage

Swanand Kadhe and Alex Sprintson

Abstract

We consider the problem of secure distributed data storage under the paradigm of *weak security*, in which no *meaningful information* is leaked to the eavesdropper. More specifically, the eavesdropper cannot get any information about any individual data packet or a small group of packets. The key benefit of the weak security paradigm is that it incurs no loss in the storage capacity, which makes it practically appealing. In this work, we present a coding scheme, using a coset coding based outer code and a Product-Matrix Minimum Bandwidth Regenerating code (proposed by Rashmi et al.), that achieves weak security when the eavesdropper can observe any single storage node. We show that the proposed construction has good security properties and requires small finite field size.

I. Introduction

Distributed storage systems (DSS) have recently received significant research attention due to their important applications in data centers and cloud networks. To achieve reliability in DSS, some form of redundancy is introduced using either replication or erasure coding. Erasure Coding is attractive in terms of storage efficiency, but it requires large amount of data to be downloaded during the recovery of a failed node. To cater this problem, Dimakis et al. [1] introduced a new class of codes, called as Regenerating Codes, which significantly reduce the amount of data downloaded during the node repair process.

Specifically, Dimakis *et al.* [1], [2] showed that there exists a trade-off between storage space per node and repair bandwidth for single node failure, and proposed a class of codes called as *regenerating codes* that optimally achieve this trade-off. The codes on one extreme point of the optimal storage-repair bandwidth trade-off curve that minimize the repair bandwidth are referred to as Minimum Bandwidth Regenerating (MBR) codes; whereas, the codes on the other extreme point that minimize storage per node are referred to as Minimum Storage Regenerating (MSR) codes. Several explicit code constructions have been proposed for exact regenerating codes (see [3] and references therein). In this paper, we focus on product-matrix codes [4], since these codes can be constructed for the entire range of parameters and they require small finite field size.

Another important challenge for a DSS is the security of the stored data. For example, some of the storage nodes in the cloud networks owned by certain private organizations can be eavesdropped. Providing secrecy against eavesdropping is particularly challenging in DSS because of its dynamic nature, with nodes continually failing and being repaired. The eavesdropper *Eve* can observe some nodes from the original set of nodes that the system starts with or even choose some of the replacement nodes added to the system to repair it from failures.

DSS can be secured using either conventional cryptographic techniques or information-theoretic approaches. One major drawback of almost all the secret key based encryption techniques is that they require secret key management mechanisms, which incur severe burden on resources. Therefore, in the distributed setting of DSS, providing information-theoretic secrecy might be advantageous.

Information-theoretic model for securing the regenerating codes was introduced by Pawar *et al.* [5] (see also [6]). Since then, a number of investigations have been carried out on characterizing the outer-bound on secrecy capacity and the associated achievable schemes (see [7], [8], [9]). However, all these results are focussed on the paradigm of information-theoretic *perfect secrecy*. Essentially, perfect secrecy requires that the eavesdropper gains absolutely *no information* about the stored data from its observations. For

Swanand Kadhe and Alex Sprintson are with the Department of Electrical and Computer Engineering at Texas A&M University, USA; Emails:{kswanand1,spalex}@tamu.edu.

many practical storage systems, this condition might be too strong. Moreover, coding schemes that provide perfect secrecy involve mixing the data symbols with the random keys to confuse the eavesdropper, which incurs loss in the storage capacity. Considering these drawbacks of the perfect secrecy notion, we focus on the notion of *weak secrecy* [10].[1]

The notion of weak secrecy requires that an eavesdropper gains no *meaningful information* about the stored data. Specifically, weak secrecy requires that the eavesdropper should only be able to obtain some linear combination(s) of a large number of files, from which it cannot infer any information about any individual file or any small group of files. Essentially, weakly secure coding schemes use data packets as keys, and thus, do not incur loss in capacity.

Despite of its practical benefits, there have been relatively very few attempts on employing weak secrecy for DSS. In [11], Oliveira *et al.* present the construction of weakly secure erasure codes for DSS without considering the regenerating codes. Very recently, Dau *et al.* [12] have analyzed the weak secrecy properties of two families of regenerating codes: regular-graph codes [13] and product-matrix codes [4].

In this paper, going a step ahead from [12], we focus on designing outer codes to improve the weak secrecy properties of the regenerating codes. To be specific, we present explicit construction of a coset coding based outer code for product-matrix (PM) codes operating at MBR point (referred to as PM-MBR codes) [4] for the scenario wherein Eve can observe any single storage node. The proposed coding scheme has numerous advantages. First, it enhances the weak secrecy properties of the PM-MBR codes. In particular, when the size of the stored data is large, the gain achieved by the proposed coding scheme is twofold. Second, the proposed outer codes leverage the elegant structure that is present in the PM codes, and thus, require small finite field size. Finally, the *weak-secrecy capacity* of the proposed coding scheme is *nearly equal*[2] to the non-secure storage capacity. These features make the proposed coding scheme attractive in practical settings.

II. Preliminaries

A. Regenerating Codes

Suppose we need to store a file $S = \{S_1, \ldots, S_B\}$ containing B symbols, each drawn uniformly and independently from a finite field $\mathbb{F}_q$, across n storage nodes, where each node is capable of storing α symbols. A regenerating code encodes the B message symbols into $n\alpha$ symbols in such a way that it satisfies the following two properties. First, a *data collector* (DC) connecting to *any* k out of n nodes should be able to reconstruct the entire file; this is referred to as the *reconstruction property*. Second, when a storage node is failed, it is *regenerated* by adding a new node which downloads β symbols each from any d out of the remaining $n-1$ nodes; this is referred to as the *regeneration property*. A regenerating code with these parameters is referred to as an (n, k, d, α, β) regenerating code.

Under these requirements, the outer bound on the capacity of an (n, k, d, α, β) regenerating code is given as [2]

$$B \leq \sum_{i=0}^{k-1} \min\{\alpha, (d-i)\beta\} \tag{1}$$

For Minimum Bandwidth Regenerating (MBR) codes, first the *repair bandwidth* $d\beta$ is minimized and then the storage per node α is minimized. Specifically, for an $(n, k, d, \alpha, \beta = 1)$ MBR code, we have $B = \sum_{i=0}^{k-1} d-i$, and $\alpha = d$. If the regenerated node is an exact replica of the failed node, then repair model is said to be *exact repair* [13]. In this paper, we focus on a special class of exact minimum bandwidth regenerating (MBR) codes called as the product-matrix codes [4], which are described in section II-D.

[1]Note that the notion of weak secrecy that is introduced in [10] and considered throughout this paper, is different from the conventional notion of information-theoretic weak secrecy, which is defined for asymptotically large block-lengths. The weak secrecy notion considered in this paper is applicable to finite block-lengths as well.

[2]For all parameters, the weakly secure capacity of the proposed scheme is two units below the non-secure storage capacity, which is negligible when the non-secure capacity is large.

B. Eavesdropper Model

The most generalized eavesdropper model for a DSS, called as the (l_1, l_2)-eavesdropper model, is considered in [9] (see also [7]). An (l_1, l_2)-eavesdropper, *Eve*, can access the data stored on any l_1 nodes, and the data downloaded during the regeneration of any l_2 nodes.

Notice that at MBR point, the number of downloaded symbols is equal to the number of stored symbols. Therefore, Eve cannot gain any additional information by observing the data downloaded during the regeneration, and thus, it is sufficient to simply consider the total number of nodes $l := l_1 + l_2$ that Eve has access to.

In this paper, we assume that Eve can access any one storage node. Thus, we have $l = 1$. We assume that Eve is passive, has unbounded computational power, and has the knowledge of the coding scheme being used.

C. Information-theoretic Secrecy

The notion of information-theoretic *perfect secrecy* requires that Eve's observation should not *leak any information* about the stored file. To be precise, suppose we want to store B_s message symbols $S = \{S_1, \ldots, S_{B_s}\}$ in a perfectly secure sense. Let E denote the set of (encoded) symbols that Eve can observe. A DSS is said to be perfectly secure if the mutual information between the message symbols S and the eavesdropped symbols E is zero, *i.e.*, $I(S; E) = 0$. Under this requirement, Pawar *et al.* [5] characterized an upper bound on the *secrecy capacity* as:

$$B_s \leq \sum_{i=l}^{k-1} \min\{\alpha, (d-i)\beta\} \tag{2}$$

Comparing (1) and (2), we can say that in order to make the system perfectly secure, the l nodes that are compromised by the eavesdropper cannot effectively contain any useful information. Consequently, the perfect secrecy requirement results in a loss of storage capacity, *i.e.*, $B_s < B$.

Remark 1. *Shah et al. [7] show that PM-MBR codes can be made perfectly secure against an* (l_1, l_2) *eavesdropper by appropriately mixing random keys with the message symbols. The secure codes achieve the capacity outer bound given in* (2)*, and the loss of capacity incurred due to perfect secrecy requirement is* $B - B_s = ld - \binom{l}{2}$.

In this paper, we focus on a relaxed, yet practically appealing, notion of *weak secrecy* which demands that Eve should not get *any meaningful information* about the stored file [10]. In particular, a DSS is said to be *weakly secure* if $I(S_i; E) = 0$, $\forall i \in [B_s]$, where $[B_s] := \{1, \ldots, B_s\}$. Furthermore, suppose Eve has been able to obtain, as a side information, some g message symbols denoted as $S_{\mathcal{G}} := \{S_i : i \in \mathcal{G}\}$ for some $\mathcal{G} \subset [B_s]$ such that $|\mathcal{G}| = g$. Then, a DSS is said to be *weakly secure against* g *guesses* if we have

$$I(S_i; E|S_{\mathcal{G}}) = 0 \qquad \forall i \in [B_s] \setminus \mathcal{G},\ \forall \mathcal{G} : |\mathcal{G}| \leq g, \tag{3}$$

where $[B_s] := \{1, \ldots, B_s\}$. In [14], it was shown that the above condition is equivalent to

$$I(S_{\mathcal{G}'}; E) = 0 \qquad \forall \mathcal{G}' \subseteq [B_s] : |\mathcal{G}'| \leq g + 1. \tag{4}$$

Essentially, this implies that in a scheme that is weakly secure against g guesses, it is not possible for Eve to obtain any information about *any* subset of $g + 1$ symbols from his observations.[3]

[3]In [12], a scheme that is weakly secure against $g - 1$ guesses is called as a *g-block secure* scheme, following condition (4).

D. Recap of Product-Matrix MBR Codes

Let us review the Product-Matrix framework based MBR Codes (PM-MBR Codes) proposed in [4].The PM-MBR code is obtained by taking the product of an $(n \times d)$ encoding matrix Ψ and a $(d \times \alpha)$ message matrix M that contains the B message symbols arranged in a particular fashion. Specifically, the encoding matrix Ψ and the message matrix M have the following structure

$$\underbrace{\Psi}_{n\times d} = \begin{bmatrix} \underbrace{\Phi}_{n\times k} & \underbrace{\Delta}_{n\times(d-k)} \end{bmatrix}, \quad \underbrace{M}_{d\times d} = \begin{bmatrix} \underbrace{B_1}_{k\times k} & \underbrace{B_2}_{k\times(d-k)} \\ \underbrace{B_2^T}_{(d-k)\times k} & \underbrace{0}_{(d-k)\times(d-k)} \end{bmatrix} \tag{5}$$

In the message matrix M, the component matrix B_1 is a $k \times k$ symmetric matrix which contains $\frac{k(k+1)}{2}$ data symbols in the upper triangular half; whereas, the other component matrix B_2 is a $k \times (d-k)$ matrix which contains the remaining $k(d-k)$ message symbols.

The matrices Φ and Δ are chosen in such a way that any k rows of Φ are linearly independent, and any d rows of Ψ are linearly independent. If Ψ is chosen to be a Vandermonde or a Cauchy matrix, these requirements are satisfied. Note that the choice of the matrix Ψ determines the field size q. For instance, if Ψ is a Vandermonde matrix, then we need to have $q \geq n$.

The α symbols stored on the i-th node are given by $C_i = \Psi_i M$, where Ψ_i denotes the i-th row of Ψ. The regeneration and the reconstruction processes can be found in [4].

Example: Consider a $(n = 5, k = 3, d = 4, \alpha = 4, \beta = 1)$ PM-MBR code. Then, from (1), we have $B = 9$. Suppose the encoding matrix Ψ is a Cauchy matrix. Then, in parametric form, we have

$$\Psi = \left[\frac{1}{a_i + b_j}\right]_{i=1,j=1}^{5,4}, \quad M = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \\ x_2 & x_5 & x_6 & x_7 \\ x_3 & x_6 & x_8 & x_9 \\ x_4 & x_7 & x_9 & 0 \end{bmatrix}, \tag{6}$$

where $a_i, b_j \in \mathbb{F}_q$ such that $a_i \neq b_j$ and $a_i + b_j \neq 0$ for all i, j. Note that, to satisfy these requirements, we need at least $n + d = 9$ distinct elements, and thus, we require $q \geq 9$.

III. Main Results

A. Outer Code Based on Coset Coding

We propose to use an outer code to improve the weak secrecy level of the PM-MBR codes. When outer code is used, the overall encoding consists of two steps. First, the outer code is used to encode the length-B_s message file $S = [S_1 \dots S_{B^{(s)}}] \in \mathbb{F}_q^{B_s}$ into a codeword $X = [X_1 \dots X_B] \in \mathbb{F}_q^B$. Next, the codeword X is encoded using the PM-MBR code by populating the entries of matrix M (see (5)) with codeword symbols X. Notice that the regeneration process remains the same. To obtain the message file S, a user would first decode X using the reconstruction process of the PM-MBR code, and then, decode the outer code to get S.

We design the outer code based on *coset coding* [15]. A coset code is constructed using a $(B, B - B_s)$ linear code $\mathcal{C}$ over $\mathbb{F}_q$ with parity-check matrix $H \in \mathbb{F}_q^{B_s \times B}$. Specifically, the message file S is encoded by selecting uniformly at random some $X \in \mathbb{F}_q^B$ such that $S = HX$. Therefore, the message file can be considered as a syndrome specifying a coset of $\mathcal{C}$ and the codeword X is a randomly chosen element of that coset. Notice that the decoding operation consists of simply computing the syndrome $S = HX$.

To design the matrix H appropriately, we need to transform the weak secrecy condition (4) into a condition on H. For this, we use the following result from [14, Lemma 6], which is a generalization of [16, Theorem 1].

Lemma 1. *Suppose a coset code based on a parity-check matrix $H \in \mathbb{F}_q^{B_s \times B}$ is used as an outer code over a given exact regenerating code. Let $E = GX$ be the μ linearly independent symbols observed by an eavesdropper. Then, for any $\mathcal{G}' \subseteq [B_s] : |\mathcal{G}'| \leq B - \mu$, we have*

$$I\left(S_{\mathcal{G}'}; E\right) = \textit{rank}\, H_{\mathcal{G}'} + \textit{rank}\, G - \textit{rank} \begin{bmatrix} H_{\mathcal{G}'} \\ G \end{bmatrix}, \tag{7}$$

$$G_1 = \begin{bmatrix} \Psi(1,1) & \Psi(1,2) & \Psi(1,3) & \Psi(1,4) & 0 & 0 & 0 & 0 & 0 \\ 0 & \Psi(1,1) & 0 & 0 & \Psi(1,2) & \Psi(1,3) & \Psi(1,4) & 0 & 0 \\ 0 & 0 & \Psi(1,1) & 0 & 0 & \Psi(1,2) & 0 & \Psi(1,3) & \Psi(1,4) \\ 0 & 0 & 0 & \Psi(1,1) & 0 & 0 & \Psi(1,2) & 0 & \Psi(1,3) \end{bmatrix} \tag{9}$$

where $H_{\mathcal{G}'}$ is a sub-matrix of H formed by choosing the rows indexed by the set $\mathcal{G}'$.

Then, weak secrecy would be satisfied by designing H and G such that RHS above is equal to zero.

B. Securing the PM-MBR Codes

As previously mentioned in section II-B, we assume that Eve can observe any single storage node. Let e denote the index of the node that Eve can access. Eve can observe the $\alpha = d$ symbols stored on node e given by $C_e = \Psi_e M$, where Ψ_e is the e-th row of Ψ. To use condition (7), we need to find a matrix G_e such that $E = G_e X$.This is carried out by a a simple linear transformation that guarantees $E = G_e X = (\Psi_e M)^T$. To describe the transformation formally, assume without loss of generality, that the B outer-coded symbols X are filled in matrix M in a lexicographic order for $1 \leq i \leq k$ and $1 \leq j \leq d$. Therefore, if $M_{(i,j)}$ denotes the symbol at i-th row and j-th column of M, then we have $X_1 = M_{(1,1)}$, $X_2 = M_{(1,2)}$, ..., $X_B = M_{(k,d)}$. Equivalently, the set of symbols $\{M_{(1,1)}, \ldots, M_{(1,d)}, M_{(2,2)}, \ldots, M_{(2,d)}, \ldots, M_{(k,k)}, \ldots, M_{(k,d)}\}$ represents the B symbols $X = \{X_1, \ldots, X_B\}$. Further, notice that since M is a symmetric matrix, if $M_{(i,j)} = X_b$, then we have $M_{(j,i)} = X_b$ as well. Using this, the symbols observed by Eve can be written as $E = G_e X$. where the $d \times B$ matrix G_e is a generator matrix of node e whose (i,b)-th entry is given as

$$G_e(i,b) = \begin{cases} \Psi_{(e,j)} & \text{if } M_{(i,j)} = X_b \text{ or } M_{(j,i)} = X_b, \\ 0 & \text{otherwise,} \end{cases} \tag{8}$$

for $1 \leq i \leq d$ and $1 \leq b \leq B$.

Example: For the previous example of $(n = 6, k = 4, d = 5, \alpha = 5, \beta = 1)$ PM-MBR code, if Eve observes the first node then we can write G_1 as (9).

Finally, using (7), it is sufficient to ensure that the following condition is satisfied to ensure weak security against g guesses.

$$\text{rank} \begin{bmatrix} H_{\mathcal{G}'} \\ G_e \end{bmatrix} = \text{rank } H_{\mathcal{G}'} + \text{rank } G_e, \ \ \forall \mathcal{G}' \subset [B_s] : |\mathcal{G}'| \leq g + 1. \tag{10}$$

Remark 2. *Observe that the matrix G_e for each node $e \in [n]$ is sparse. In particular, G_e for each node $e \in [n]$ contains at least one row vector with Hamming weight k. Thus, PM-MBR codes are not secure against $g \geq k - 1$ guesses, when Eve can observe one storage node. This shows the necessity to employ an outer code to improve the the level of weak secrecy.*

Remark 3. *It would be possible to use a random matrix as H, however it would require very large field size. This is because the condition* (10) *must be satisfied for all sub-matrices of H, the number of which are exponentially large. Moreover, for each sub-matrix $H_{\mathcal{G}'}$, we must ensure the above condition for each node $e \in [n]$, since Eve can observe any storage node. Thus, we explicitly construct H that requires small field size.*

Now, our aim is to jointly design a PM-MBR code and a coset code such that (10) is satisfied. However, we do not want to disturb the structure of PM-MBR codes which facilitates efficient regeneration and reconstruction properties. The only one degree of freedom that we have is in choosing the encoding matrix Ψ such that the conditions specified in section II-D are satisfied.

The main idea of our solution is to construct parity-check matrix H such that it has the same structure as the generator matrix G_e of a node. The idea is to leverage the same structure of G_e and H to ensure the condition (10).

To describe the structure present in PM-MBR codes more formally, we introduce the notion of *type*. We say that a length-B row vector $h^{(i)}$ is of type i if it has form as the i-th row of G_e. In particular, the vector of type i has non-zero coefficients at the same locations as in the i-th row of G_e; the values of the non-zero coefficients can be different. Essentially, the type a row vector specifies the locations of the non-zero coefficients of the vector. We call the corresponding set of locations of non-zero coefficients as the *index set of type* i, denoted as $\mathcal{I}(i)$.

For instance, considering our running example, a vector of type 4 has the form $h^{(4)} = \begin{bmatrix}0 & 0 & 0 & \gamma & 0 & 0 & \gamma^2 & 0 & \gamma\end{bmatrix}$ for some $\gamma \in \mathbb{F}_q$. The corresponding index set is $\mathcal{I}(4) = \{4, 7, 9\}$, which corresponds to the indices of elements of fourth column of M (see (6)).

We construct H such that each row of H belongs to one of the d types. Let θ_i denote the number of row vectors of type i, $1 \leq i \leq d$, that are present in H. Define $\theta := [\theta_1 \dots \theta_d]$, which we call the *type cardinality vector*. For each $\theta_i > 0$, let H_i denote the $\theta_i \times B$ sub-matrix of H that is composed of all the row vectors of type i.

Once the type of an row vector is fixed, it is sufficient to give a set of values of non-zero coefficients to fully specify the row vector. For example, the non-zero coefficient values of all the vectors in G_e are specified by the row vector Ψ_e. In a similar manner, we represent the non-zero coefficients of the row vectors in H using a matrix $\hat{\Psi}$. Specifically, a $d \times d$ matrix $\hat{\Psi}$ is defined in such a way that the j-th row of $\hat{\Psi}$ specifies the non-zero coefficient values of the j-th vector of type i that is present in H for $i, j \in [d]$. We call the matrix $\hat{\Psi}$ as the *coefficient matrix*. Observe that the type cardinality vector θ along with the coefficient matrix $\hat{\Psi}$ are sufficient to specify the parity-check matrix H.

In the following, we describe an explicit construction of the encoding matrix Ψ and the parity check matrix H which improves the security performance of the PM-MBR codes beyond $g = k - 1$ guesses.

Construction 1. *First, choose the type cardinality vector θ as follows.*

$$\theta_i = \begin{cases} 0 & \text{if} \quad i = 1, \\ d - k + i & \text{if} \quad 2 \leq i \leq k-1, \\ d - 1 & \text{if} \quad i = k, \\ 1 & \text{if} \quad k+1 \leq i \leq d. \end{cases} \tag{11}$$

Note that $\max_{1 \leq i \leq d} \theta_i = d - 1$.

Next, choose an $n \times d$ encoding matrix Ψ and a $d \times d$ coefficient matrix $\hat{\Psi}$ in such a way that any square sub-matrix of $\tilde{\Psi} := \begin{bmatrix}\Psi \\ \hat{\Psi}\end{bmatrix}$ is non-singular.

Finally, using $\hat{\Psi}$ and θ, construct H as follows. For each θ_i, $2 \leq i \leq d$, the $\theta_i \times B$ sub-matrix H_i of H is given as

$$H_i(p, b) = \begin{cases} \hat{\Psi}_{(p,j)} & \text{if } M_{(i,j)} = X_b \text{ or } M_{(j,i)} = X_b \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

for $1 \leq p \leq \theta_i$ and $1 \leq b \leq B$. The parity-check matrix H is obtained by vertically concatenating the sub-matrices H_i, i.e., $H = \begin{bmatrix}H_2^T & H_3^T & \dots & H_d^T\end{bmatrix}^T$.

Note that the requirement on $\tilde{\Psi}$ mentioned in Construction 1, that any of its square sub-matrices should be non-singular, can be ensured, for example, by choosing $\tilde{\Psi}$ as a Cauchy matrix. Another construction of a matrix that satisfies this requirement can be found in [17]. Both these constructions require that $q \geq n + 2d$.

There are couple of points about this requirement on $\tilde{\Psi}$ that are worth mentioning. First, note that this requirement on $\tilde{\Psi}$ implies that any square sub-matrix of Ψ should also be non-singular. This is a stronger requirement, which guarantees the two requirements on Ψ that are mentioned in section II-D. Second, choosing Ψ as a Vandermonde matrix, which is good enough to meet the requirements of PM-MBR codes, is not sufficient, since a Vandermonde matrix defined over finite field can contain singular square sub-matrices (see [17], also [12]).

$$H = \begin{bmatrix} 0 & \hat{\Psi}(1,1) & 0 & 0 & \hat{\Psi}(1,2) & \hat{\Psi}(1,3) & \hat{\Psi}(1,4) & 0 & 0 \\ 0 & \hat{\Psi}(2,1) & 0 & 0 & \hat{\Psi}(2,2) & \hat{\Psi}(2,3) & \hat{\Psi}(2,4) & 0 & 0 \\ 0 & \hat{\Psi}(3,1) & 0 & 0 & \hat{\Psi}(3,2) & \hat{\Psi}(3,3) & \hat{\Psi}(3,4) & 0 & 0 \\ 0 & 0 & \hat{\Psi}(1,1) & 0 & 0 & \hat{\Psi}(1,2) & 0 & \hat{\Psi}(1,3) & \hat{\Psi}(1,4) \\ 0 & 0 & \hat{\Psi}(2,1) & 0 & 0 & \hat{\Psi}(2,2) & 0 & \hat{\Psi}(2,3) & \hat{\Psi}(2,4) \\ 0 & 0 & \hat{\Psi}(3,1) & 0 & 0 & \hat{\Psi}(3,2) & 0 & \hat{\Psi}(3,3) & \hat{\Psi}(3,4) \\ 0 & 0 & 0 & \hat{\Psi}(1,1) & 0 & 0 & \hat{\Psi}(1,2) & 0 & \hat{\Psi}(1,3) \end{bmatrix} \tag{13}$$

$$T = \begin{bmatrix} H_{\mathcal{G}'} \\ G_e \end{bmatrix} = \begin{bmatrix} 0 & \hat{\Psi}(2,1) & 0 & 0 & \hat{\Psi}(2,2) & \hat{\Psi}(2,3) & \hat{\Psi}(2,4) & 0 & 0 \\ 0 & 0 & \hat{\Psi}(3,1) & 0 & 0 & \hat{\Psi}(3,2) & 0 & \hat{\Psi}(3,3) & \hat{\Psi}(3,4) \\ 0 & 0 & 0 & \hat{\Psi}(1,1) & 0 & 0 & \hat{\Psi}(1,2) & 0 & \hat{\Psi}(1,3) \\ \Psi(e,1) & \Psi(e,2) & \Psi(e,3) & \Psi(e,4) & 0 & 0 & 0 & 0 & 0 \\ 0 & \Psi(e,1) & 0 & 0 & \Psi(e,2) & \Psi(e,3) & \Psi(e,4) & 0 & 0 \\ 0 & 0 & \Psi(e,1) & 0 & 0 & \Psi(e,2) & 0 & \Psi(e,3) & \Psi(e,4) \\ 0 & 0 & 0 & \Psi(e,1) & 0 & 0 & \Psi(e,2) & 0 & \Psi(e,3) \end{bmatrix} \tag{14}$$

Example: For the running example, construction 1 yields $\theta = \{0, 3, 3, 1\}$. Let the 5×4 encoding matrix Ψ be a Cauchy matrix (*cf.* (6)). We choose $\hat{\Psi}$ such that $\hat{\Psi} = \begin{bmatrix} \Psi \\ \hat{\Psi} \end{bmatrix}$ is also a Cauchy matrix. Note that this requires $q \geq 13$. Then, the resulting parity-check matrix H is given in (13).

C. Analysis

First, we characterize the file size that can be stored in a weakly secure sense by using the proposed outer code along with a PM-MBR code.

Theorem 1. *When an outer coset code based on the parity-check matrix H given in Construction 1 is used along with a PM-MBR code, the weakly secure storage capacity is $B_s = B - 2$.*

Proof: See appendix A. ■

Remark 4. *Note that, for perfectly secure PM-MBR codes with $l = 1$, we have $B - B_s = d$ which increases in d (see [7]). On the other hand, proposed scheme suffers a constant loss of two units which is negligible for large B.*

Next, we compute the level of secrecy that can be attained using the proposed outer code along with a PM-MBR code.

Theorem 2. *An outer coset code based on the parity-check matrix H given in Construction 1 makes a PM-MBR code weakly secure against $g \leq d + k - 4$ guesses.*

Proof: See Appendix B. ■

Remark 5. *In [12], it is shown that, when Eve observes l nodes, the PM-MBR codes using Cauchy matrix as their encoding matrix are weakly secure against $k - l - 1$ guesses. Thus, for $l = 1$, it is shown that PM-MBR codes are secure against $k - 2$ guesses. Our proposed encoding enhances the level of security to $d + k - 4$ guesses, which is an improvement of $d - 2$ for all set of parameters (except for $d = k = 2$). Notice that, for any regenerating code, $d \geq k$. Thus, for large k, the enhancement achieved by the proposed scheme is almost twofold.*

APPENDIX A
PROOF OF THEOREM 1

Notice that the message file, which is securely stored, can be considered as the syndrome of the coset code as $S = HX$. Thus, the weak-secrecy capacity is the dimension of matrix H. First, we show that, if H is designed following construction 1, it contains $B-2$ rows. Next, we show that H is full-rank to prove the result.

Now, notice that the total number of rows in H is equal to $\sum_{i=1}^{d}\theta_i$. From (11), we have

$$\begin{aligned}\sum_{i=1}^{d}\theta_i &= 0+\left(\sum_{i=2}^{k-1} d-k+i\right)+(d-1)+(d-k)\\ &= \left(\sum_{i=1}^{k-2} d-i\right)+(d-1)+(d-k)\\ &\overset{(a)}{=} \left(\sum_{i=0}^{k-1} d-i\right)-2\\ &\overset{(a)}{=} B-2 \end{aligned} \tag{15}$$

where (a) can be easily obtained by carrying out simple algebraic manipulations, and (b) follows from (1) and from the fact that at MBR point $\alpha = d$ (we assume that $\beta = 1$).

To prove that H is full-rank, we show that it is possible to append two rows to H in such a way that the resulting $B \times B$ matrix, denoted as H', is non-singular. Specifically, append a type 1 row vector with non-zero coefficients corresponding to the first row of $\hat{\Psi}$, and append a type k row vector with non-zero coefficients corresponding to the d-th row of $\hat{\Psi}$. From (11), it is easy to see that H' contains $d-(k-i)$ rows of type i for $k \geq i \geq 2$, one row of type 1, and one row each of types $k+1$ through d. Without loss of generality, assume that the rows of H are ordered in such a way that first d rows are of type k, next $d-1$ rows are of type $k-1$ and so on up to $d-(k-2)$ rows of type 2. The last $d-k+1$ rows are of types $k+1$ through d and of type 1, respectively.

Now, for proving the non-singularity of H', consider a system of linear equations $Z = H'Y$, where $Y = [Y_1 \cdots Y_B]$ and $Z = [Z_1 \cdots Z_B]$ are length-B vectors of variables Y_1 through Y_B and Z_1 through Z_B. We show that it is possible to *successively decode* variables in Y in terms of variables in Z by leveraging the elegant structure of H'.

To describe the process of successive decoding, we need to introduce some notation. Recall that the type of a row vector specifies the locations of the non-zero coefficients of the vector. The corresponding set of locations of non-zero coefficients is referred to as the index set of type i, $\mathcal{I}(i)$. Define $Y[\mathcal{I}(i)] := \{Y_l : l \in \mathcal{I}(i)\}$, *i.e.*, $Y[\mathcal{I}(i)]$ is the vector of elements of Y that are indexed by the index set of type i.

Suppose we write vector Y in the format of matrix M (see (5)) in a lexicographic order. Notice that the index set of type i is the set of indices of the set of indices of the elements in the i-th column of M. Observing the structure of M, we divide the d types into two groups. The types 1 through k are called as group I, while the types $k+1$ through d are called as group II. For any group I type, the index set consists of d elements, *i.e.*, $|\mathcal{I}(i)| = d$, $\forall i \in [k]$. On the other hand, for any group II type, the corresponding index set has k elements, *i.e.*, $|\mathcal{I}(i)| = k\ \forall i \in \{k+1, \ldots, d\}$. Further, index set corresponding to any group I has one index common with the index sets of all other types, *i.e.*, $|\mathcal{I}(i) \cap \mathcal{I}(j)| = 1\ \forall i < j : i \in [k]$. Whereas, any pair of index sets of group II types has no common symbol, *i.e.*, $|\mathcal{I}(i) \cap \mathcal{I}(j)| = 0\ \forall k < i < j \leq d$.

Let γ_1 and γ_2 denote the number of row vectors in H' of types of group I and II, respectively. Algorithm 1 presented below decodes elements of $Y[\mathcal{I}(i)]$ for each $i \in [d]$ successively.

Claim 1. *Algorithm 1 decodes all the B elements of Y in terms of Z.*

Proof: The algorithm begins with type k, of which there are d row vectors in H'. By the construction of H', the non-zero coefficient values are the elements of the Cauchy matrix $\hat{\Psi}$. Thus, it is possible to solve for $Y[\mathcal{I}(k)]$ by inverting $\hat{\Psi}$. Next, we prove by induction that, for $2 \leq i \leq k$, if the elements of

Algorithm 1 Successive decoding for $Z = H'Y$

1: Set $\gamma_1 = d$, $\gamma_2 = k$
2: **for** $p = k$ **to** 2 **do**
3: Consider set of equations corresponding to γ_1 rows of type p as $Z[\mathcal{I}(p)] = \hat{\Psi}_{1:\gamma_1} Y[\mathcal{I}(p)]$
4: Using perviously decoded elements, decode for elements of Y located at $\mathcal{I}(p) \setminus \left(\bigcup_{l=1}^{k-p} (\mathcal{I}(p) \cap \mathcal{I}(p+l))\right)$
5: $\gamma_1 = \gamma_1 - 1$, $\gamma_2 = \gamma_2 - 1$
6: **end for**
7: Decode for the remaining elements in index sets of types $k+1$ through d
8: Decode for the remaining single element of type 1

$\mathcal{I}(i+1)$ through $\mathcal{I}(k)$ have been decoded, it is possible to decode the elements of $\mathcal{I}(i)$. By construction of H', there are $d-k-i$ rows of type i in H' for $2 \leq i \leq k$ with non-zero coefficients given by $\hat{\Psi}_{1:(d-k-i)}$, respectively. This forms a system of $d-k-i$ linear equations in d variables of $\mathcal{I}(i)$ as $Z[\mathcal{I}(i)] = \hat{\Psi}_{1:(d-k-i)} Y[\mathcal{I}(i)]$. Note that, since type i is a group I type, there is one element common between $\mathcal{I}(i)$ and each of $\mathcal{I}(i+1)$ through $\mathcal{I}(k)$. Thus, there are $d-k-i$ elements in $\mathcal{I}(i)$ that are yet to be decoded. As any square sub-matrix of $\hat{\Psi}$ is non-singular by construction, it is possible to solve for the un-decoded $d-k-i$ variables using $Z[\mathcal{I}(i)] = \hat{\Psi}_{1:(d-k-i)} Y[\mathcal{I}(i)]$.

At the end of the first **for** loop, $k-1$ elements from each $\mathcal{I}(j)$ $j \in [d]$ are decoded. Thus, in each of the index sets of group II types, there is just one element to be decoded. By construction, H' has one row in each of the group II types, and thus, it is possible to decode all elements in group II index sets. At this point, all the elements from index sets of all types except type 1 are decoded.

Finally, since $\mathcal{I}(1)$ has one element common with all the remaining tinder sets, only single element from $\mathcal{I}(1)$ remains to be decoded, which can be decoded using the row of type 1 that is appended to H.

Notice that, as the each index set corresponds to a column of matrix M, $\bigcup_{j=1}^{d} \mathcal{I}(j) = \{Y_1, \ldots, Y_B\}$. Therefore, the algorithm decodes all the B elements of Y. ∎

Note that successively decoding for the variables of a particular type is equivalent to performing Gaussian elimination on the corresponding rows of that particular type. Thus, in essence, the procedure for successive decoding gives the order in which Gaussian elimination can be performed in H'.

Example: Consider H given in (13). Append a row vector of type 3, and decode for variables indexed by $\mathcal{I}(3) = \{2, 5, 6, 7\}$. Notice that $\mathcal{I}(3) \cap \mathcal{I}(2) = 6$. Then, using the three rows of type 2 and already decoded variable at index 6, solve for variables indexed by $\mathcal{I}(2) \setminus (\mathcal{I}(3) \cap \mathcal{I}(2)) = \{3, 8, 9\}$. Then, using a row vector of type 4, decode for $\mathcal{I}(4) \setminus ((\mathcal{I}(4) \cap \mathcal{I}(3)) \cup (\mathcal{I}(4) \cap \mathcal{I}(2))) = \{4\}$. Finally, by appending one row of type one, decode for $\mathcal{I}(1) \setminus ((\mathcal{I}(1) \cap \mathcal{I}(4)) \cup (\mathcal{I}(1) \cap \mathcal{I}(3)) \cup (\mathcal{I}(1) \cap \mathcal{I}(2))) = \{1\}$. The non-zero coefficients of the appended rows are specified by the appropriate rows of $\hat{\Psi}$. The successive decoding uses the property that any square sub-matrix of the Cauchy matrix $\hat{\Psi}$ is non-singular.

APPENDIX B
PROOF OF THEOREM 2

Essentially, we need to prove that condition (10) always holds for the proposed coding scheme as long as $|\mathcal{G}'| \leq d+k-3$. For notational convenience, let $T := \begin{bmatrix} H_{\mathcal{G}'} \\ G_e \end{bmatrix}$. Notice that there are $\binom{B_s}{|\mathcal{G}'|}$ number of ways to choose a particular $|\mathcal{G}'|$, and we ensure (10) for each possible $H_{\mathcal{G}'}$ as long as $|\mathcal{G}'| \leq d+k-3$.

Since H is full-rank as shown in theorem 1, its sub-matrix $H_{\mathcal{G}'}$ will be full rank for any $\mathcal{G}' \subseteq [B_s]$. Further, it has been shown in [7] that for PM-MBR codes each storage node stores α linearly independent symbols, thus, it follows that G_e is full-rank. Therefore, to prove (10), we need to prove that the matrix T is full-rank. We prove divide the proof into three cases: $k \geq 3$, $k = 2$, and $k = 1$.

Case 1: $k \geq 3$.

As in the proof of Theorem 1, we show that, if $|\mathcal{G}'| \leq d-k+1$, it is always possible to append $B - |\mathcal{G}'| - \alpha$ rows of appropriate types to T in such a way that the resulting $B \times B$ matrix is non-singular.

Algorithm 2 Appending T and carrying out successive decoding for $k \geq 3$

Sort λ_j for $j \in [k]$, Let $\lambda_{j_1} \leq \cdots \leq \lambda_{j_k}$
Sort λ_j for $k+1 \leq j \leq d$, Let $\lambda_{j_{k+1}} \leq \cdots \leq \lambda_{j_d}$
Find L such that $\lambda_{j_{d-L+1}} = \lambda_{j_{d-L+2}} = \cdots = \lambda_{j_d} = 2$
$\{$Notice that $0 \leq L \leq d-k\}$
Set $\gamma_1 = d$, $\gamma_2 = k$
for $p = k$ **to** 3 **do**
  if $\lambda_{j_p} > \gamma_1$ **then**
    Declare failure, Exit
  else
    Append T with $\gamma_1 - \lambda_{j_p}$ additional rows of type p with non-zero coefficients as the rows of $\hat{\Psi}$ that are not present in the λ_{j_p} rows of type p
    Using the equations corresponding to the γ_1 rows of type p, decode the un-decoded variables from $Y[\mathcal{I}(p)]$
    $\gamma_1 = \gamma_1 - 1$, $\gamma_2 = \gamma_2 - 1$
  end if
end for
$\{$At this point, $\gamma_1 = d - (k-2)$ and $\gamma_2 = k - (k-2) = 2\}$
if $L > 0$ **then**
  Successively decode the remaining variables from $Y[\mathcal{I}(j_{d-L+i})]$ for $i \in [L]$
  $\gamma_1 = \gamma_1 - L$
end if
$\{$At this point, $\gamma_1 = d - (k-2) - L$ and $\gamma_2 = 2\}$
if $\lambda_{j_2} > \gamma_1$ **then**
  Declare failure, exit
else
  Append T with $\gamma_1 - \lambda_{j_2}$ additional rows of type 2 with non-zero coefficients as the rows of $\hat{\Psi}$ that are not present in the λ_{j_2} rows of type 2
  Decode the un-decoded variables from $Y[\mathcal{I}(2)]$
  $\gamma_1 = \gamma_1 - 1$, $\gamma_2 = \gamma_2 - 1$
  $\{$At this point, $\gamma_1 = d - (k-2) - L - 1$ and $\gamma_2 = 2 - 1 = 1\}$
  Decode for the remaining symbols from $Y[\mathcal{I}(j_{k+1})], Y[\mathcal{I}(j_{k+2})], \ldots, Y[\mathcal{I}(j_{d-L})]$
  Append T with a row of type 1 with non-zero coefficients as the first row of $\hat{\Psi}$
  Decode the un-decoded variable from $Y[\mathcal{I}(1)]$
end if

In the following, for all we present an algorithm which, for any given $H_{\mathcal{G}'}$ and node index $e \in [n]$, appends the row vectors of appropriate types to $T = \begin{bmatrix} H_{\mathcal{G}'} \\ G_e \end{bmatrix}$ in such a way that successive decoding can be carried out. Let λ'_i be the number of encoding vectors of type i, $i \in [d]$, that are present in $H_{\mathcal{G}'}$. Notice that $\lambda'_i \leq \theta_i \ \forall i \in [d]$. Let λ_i denote the number of row vectors of type i, $i \in [d]$, that are present in T. Note that, for any $e \in [n]$, G_e contains single row vector of each of the d types. Thus, $\lambda_i = \lambda'_i + 1$, $\forall i \in [d]$. This implies that $\lambda_i \leq \theta_i + 1 \ \forall i \in [d]$. Further, from (11), we have that $\lambda_i \in \{1, 2\}$ for all group II type $k+1 \leq i \leq d$. Let γ_1 and γ_2 denote the number of equations that are required to decode the variables of group I and group II types, respectively, in a given iteration.

First, we prove the correctness of the algorithm.

Claim 2. *If algorithm 2 does not report a failure, it finds a solution to $Z = T'Y$, where T' is the matrix resulting after appending the rows to T as described in the algorithm.*

Proof: In the same way as in the proof of claim 1, it is easy to prove by induction that in the first for loop, the algorithm decodes for $Y[\mathcal{I}(i)]$ for $3 \leq i \leq k$. Since each pair of group I types has one index

in common, $k-2$ elements of each type are decoded at the end of the first for loop. Note that there are two rows each of types j_{d-L+1} through j_d. Since $k-2$ elements of each of them are already decoded, the remaining two elements are decoded at line 17.

At line 25, all the remaining elements of $\mathcal{I}(2)$ will be decoded. At this point, there is only one un-decoded element in types j_{k+1} through j_{d-L}, which will be decoded at line 28. Note that, at this point, all the $d-1$ types from 2 through d have been decoded. Thus, there remains only one un-decoded element of type 1 which will be decoded as the final step. Throughout the proof, we rely on the fact that for matrix $\tilde{\Psi}_e = \begin{bmatrix} \hat{\Psi} \\ \Psi_e \end{bmatrix}$, any square sub-matrix is singular. Note that this condition hold by construction 1.

Essentially, algorithm covers all the d types in the order (j_k, j_{k-1}, …, j_3), (j_{d-L+1}, …, j_d), (j_{k+1}, …, j_{d-L+1}), j_2, j_1, and decodes all the B elements. ∎

Next, we prove that the algorithm 2 does not declare a failure if the number rows in $H_{\mathcal{G}'}$ is bounded below a threshold.

Claim 3. *If $|\mathcal{G}'| \leq d+k-3$, algorithm 2 always succeeds.*

Proof: The proof is by contradiction. Suppose $|\mathcal{G}'|$ and the algorithm fails. *Case 1:* Algorithm fails in the first iteration ($p=k$) on line 6. This implies that $\lambda_{j_k} > \gamma_1 = d$. However, we can write

$$\begin{aligned} \lambda_{j_k} &= \max_{1\leq l\leq k} \lambda l \\ &\overset{(a)}{\leq} \max_{1\leq l\leq k} \theta_l + 1 \\ &\overset{(b)}{=} d \end{aligned}$$

where (a) follows from the fact that the number of rows of any type i in T is at most θ_i+1, *i.e.*, $\lambda_i \leq \theta_i+1$ $\forall 1 \in [d]$, and (b) is due to (11). This is a contradiction, and the algorithm cannot fail in the first iteration when $p=k$.

Case 2: Algorithm fails in the first for loop for some i-th iteration such that $2 \leq i \leq k-2$. Note that this implies $k \geq$. Also, at the i-th iteration, we have $p = k-(i-1)$. As γ_1 is reduced by 1 in each iteration, on i-th iteration we have $\gamma_1 = d-(i-1)$. This implies that $\lambda_{j_{k-i+1}} > d-(i-1)$. Then, we can write

$$\begin{aligned} \sum_{l=1}^{i} \lambda_{j_{k-l+1}} &\overset{(c)}{\geq} i\lambda_{j_{k-i+1}} \\ &\overset{(d)}{\geq} i(d-i+2) \\ \therefore \quad i + \sum_{l=1}^{i} \lambda'_{j_{k-l+1}} &\geq i(d-i+2) \qquad (16) \\ \sum_{l=1}^{i} \lambda'_{j_{k-l+1}} &\geq i(d-i+1), \qquad (17) \end{aligned}$$

where (c) is due to $\lambda_{j_k} \geq \lambda_{j_{k-1}} \geq \cdots \geq \lambda_{j_{k-i+1}}$ and (d) is due to $\lambda_{j_{k-i+1}} > d-(i-1)$. To get (16), we use $\lambda_l = \lambda'_l + 1$ $\forall l \in [d]$.

First, note that $|\mathcal{G}'| = \sum_{l=1}^{d} \lambda'_l$. Thus, $|\mathcal{G}'| \geq \sum_{l=1}^{i} \lambda'_{j_{k-l+1}}$. Next, notice that $f(i) = i(d-i+2)$ is a concave function in i and thus it will attain minimum over $2 \leq i \leq k-2$ at one of its boundary points. Substituting in (17), we have

$$|\mathcal{G}'| \geq \min\{2(d-1), (d-k+3)(k-2)\}. \qquad (18)$$

Now, we show that both of these boundary points result in contradiction. First, since $d \geq k$ for any regenerating code, clearly, $2(d-1)-(d+k-3) = d-k+1 > 0$. Next, consider the second boundary

point,

$$\begin{aligned}(d-k+3)(k-2)-(d+k-3) &\overset{(e)}{=} (k-3)d-(k-2)^2+1\\ &\overset{(f)}{\geq} (k-3)k-(k-2)^2+1\\ &= k-3,\end{aligned}$$

where (e) follows from algebraic manipulations, and (f) follows because for any regenerating code $d \geq k$ and since we have $k \geq 4$ for this case. Finally, since $k \geq 4$ in this case, we have $(d-k+2)(k-2) > d+k-3$. Therefore, substituting we have $|\mathcal{G}'| \geq \min\{2(d-1), (d-k+3)(k-2)\} > d+k-3$, which is a contradiction.

Case 3: Algorithm fails at line 22, while appending for j_2. It is easy to see that at this point $\gamma_1 = d-(k-2)-L$ and $\gamma_2 = 1$. The failure implies that $\lambda_{j_2} \geq d-(k-2)-L+1$. Now, let us consider the total number of rows in T of types that have been considered so far as follows.

$$\begin{aligned}\sum_{l=2}^{k}\lambda_{j_l}+\sum_{m=d-L+1}^{d}\lambda_{j_m} &\overset{(g)}{\geq} \sum_{l=2}^{k}\lambda_{j_2}+2L\\ &\overset{(h)}{\geq} (k-1)(d-k+L+3)+2L, \end{aligned} \tag{19}$$

where (g) follows from $\lambda_{j_2} \leq \lambda_{j_3} \leq \cdots \leq \lambda_{j_k}$ and $\lambda_{j_{d-L+1}} = \lambda_{j_{d-L+2}} = \cdots = \lambda_{j_d} = 2$, and (h) follows from $\lambda_{j_2} \geq d-(k-2)-L+1$. However, since $\lambda_l = \lambda'_l + 1$ for each $l \in [d]$, from (19), we can write

$$\begin{aligned}\sum_{l=2}^{k}\lambda'_{j_l}+\sum_{m=d-L+1}^{d}\lambda'_{j_m} &\geq (k-1)(d-k+L+3)-(k-1)-L\\ &= (k-1)(d-k-L+2)+L. \end{aligned} \tag{20}$$

After some manipulations, it is straightforward to show that $(k-1)(d-k-L+2)-(d+k-3) = (d-k-L)(k-2)+1$, which is strictly positive for $k \geq 3$ as $d \geq k$ and $0 \leq L \leq d-k$. Therefore, we have

$$|\mathcal{G}'| = \sum_{l=1}^{d}\lambda'_{j_l} \geq \sum_{l=2}^{k}\lambda'_{j_l}+\sum_{m=d-L+1}^{d}\lambda'_{j_m} \overset{(o)}{\geq} (k-1)(d-k-L+2)+L \overset{(r)}{>} d+k-3, \tag{21}$$

where (o) follows from (20) and (r) is proved in the previous point. However, this is a contradiction, which completes the proof for $k \geq 3$.

■

Case 2: $k = 2$.

We present the algorithm for successive decoding as follows.

First, we prove the correctness of the algorithm.

Claim 4. *If algorithm 3 does not report a failure, it finds a solution to $Z = T'Y$, where T' is the matrix resulting after appending the rows to T as described in the algorithm.*

Proof: As showed in the discussion before claim 3, notice that there are two elements each in $\mathcal{I}(l)$ for $k+1 \leq l \leq d$. For the L types, j_{d-L+1} through j_d, there are two rows present in T. Thus, all the variables from $\mathcal{I}(j_l)$ for $d-L+1 \leq l \leq d$ are decoded at line 7.

Algorithm will decode all the un-decoded variables indexed by $\mathcal{I}(k=2)$ at line 16. Then, there remains only one un-decoded element of types j_{k+1} through j_{d-L}, which will be decoded in line 19. At this point, there remains only single un-decoded variable from type 1, ant it is decoded as the final step.

Essentially, algorithm covers all the d types in the order (j_{d-L+1}, …, j_d), ($j_{(k=2)}$), (j_{k+1}, …, j_{d-L+1}), j_1, and decodes all the B elements. ■

Claim 5. *If $|\mathcal{G}'| \leq d+k-3 = d-1$, algorithm 3 always succeeds.*

Algorithm 3 Appending T and carrying out successive decoding for $k = 2$

1: Sort λ_j for $k+1 \leq j \leq d$, Let $\lambda_{j_{k+1}} \leq \cdots \leq \lambda_{j_d}$
2: Find L such that $\lambda_{j_{d-L+1}} = \lambda_{j_{d-L+2}} = \cdots = \lambda_{j_d} = 2$
3: $\{$Notice that $0 \leq L \leq d-k\}$
4: Set $\gamma_1 = d$, $\gamma_2 = k = 2$
5: **if** $L > 0$ **then**
6: **for** $p = 1$ **to** L **do**
7: Using the equations corresponding to the γ_2 rows of type $d-L+p$, decode the variables indexed by $\mathcal{I}(d-L+p)$
8: Set $\gamma_1 = \gamma_1 - 1$
9: **end for**
10: **end if**
11: $\{$At this point, $\gamma_1 = d - L\}$
12: **if** $\lambda_{j_k} > \gamma_1$ **then**
13: Declare failure, exit
14: **else**
15: Append T with $\gamma_1 - \lambda_{j_k}$ additional rows of type $k = 2$ with non-zero coefficients as the rows of $\hat{\Psi}$ that are not present in the λ_{j_2} rows of type 2
16: Decode the un-decoded variables from $Y[\mathcal{I}(2)]$
17: $\gamma_1 = \gamma_1 - 1$, $\gamma_2 = \gamma_2 - 1$
18: $\{$At this point, $\gamma_1 = d - L - 1$ and $\gamma_2 = 2 - 1 = 1\}$
19: Decode for the remaining symbols from $Y[\mathcal{I}(j_{k+1})], Y[\mathcal{I}(j_{k+2})], \ldots, Y[\mathcal{I}(j_{d-L})]$
20: Append T with a row of type 1 with non-zero coefficients as the first row of $\hat{\Psi}$
21: Decode the un-decoded variable from $Y[\mathcal{I}(1)]$
22: **end if**

Proof: The proof is by contradiction. Suppose $|\mathcal{G}'|$ and the algorithm fails. The only possibility of failure is line 13. Consider the total number of rows in T that have been considered till this point as follows.

$$\sum_{m=d-L+1}^{d} \lambda_{j_m} + \lambda_{j_2} \overset{(a)}{\geq} (d-L+1) + 2L$$

$$\therefore \quad \sum_{m=d-L+1}^{d} \lambda'_{j_m} + L + \lambda_{j_2} + 1 \geq d + L + 1, \tag{22}$$

$$\therefore \quad \sum_{m=d-L+1}^{d} \lambda'_{j_m} + \lambda_{j_2} \geq d, \tag{23}$$

where (a) follows because failure implies $\lambda_{j_2} > d - L$. To get (22), we use $\lambda_{j_l} = \lambda'_{j_l} + 1$ for each $l \in [d]$. Now, we can write

$$\begin{aligned} |\mathcal{G}'| &= \sum_{l=1}^{d} \lambda'_{j_l} \\ &\geq \sum_{m=d-L+1}^{d} \lambda'_{j_m} + \lambda_{j_2} \\ &\overset{(b)}{\geq} d, \end{aligned} \tag{24}$$

where (b) is due to (23). However, $|\mathcal{G}'| \geq d$ is a contradiction and the proof follows. ■

Case 3: $k = 1$.

Notice that for $k = 1$, Eve gets the same degrees of freedom as a data collector (which accesses to k nodes to recover the file). Therefore, it is not possible to achieve any form of security since, similar to the data collector, Eve can also decode the complete file

Example: Consider one possible matrix T as shown in (14). First, append one row of type 4 and decode for the variables indexed by $\mathcal{I}(4) = \{4, 7, 9\}$. Then, add one row of type 3 and decode for the variables indexed by $\mathcal{I}(3) \setminus ((\mathcal{I}(4) \cap \mathcal{I}(3))) = \{3, 6, 8\}$. Using the two rows of type 2, decode for the variables indexed by $\mathcal{I}(2) \setminus ((\mathcal{I}(4) \cap \mathcal{I}(2)) \cup (\mathcal{I}(3) \cap \mathcal{I}(2))) = \{2, 5\}$. Finally, using a row of type 1, decode for the variable indexed by $\mathcal{I}(1) \setminus ((\mathcal{I}(1) \cap \mathcal{I}(4)) \cup (\mathcal{I}(1) \cap \mathcal{I}(3)) \cup (\mathcal{I}(1) \cap \mathcal{I}(2))) = \{1\}$. The non-zero coefficients of the appended rows are specified by the appropriate rows of $\hat{\Psi}$, and the successive decoding is feasible due to the property that any square sub-matrix of the Cauchy matrix $\tilde{\Psi} = \begin{bmatrix} \Psi \\ \hat{\Psi} \end{bmatrix}$ is non-singular.

REFERENCES

[1] A. G. Dimakis, P. B. Godfrey, M. Wainwright, and K. Ramachandran, "Network Coding for Distributed Storage Systems," in *Proc. of IEEE INFOCOM*, Urbana-Champaign, May 2007.

[2] ——, "Network Coding for Distributed Storage Systems," *IEEE Trans. Inf. Theory*, vol. 56, no. 9, pp. 4539–4551, Sep. 2010.

[3] A. G. Dimakis, K. Ramchandran, Y. Wu, and C. Suh, "A Survey on Network Codes for Distributed Storage," *Proceedings of the IEEE*, vol. 99, no. 3, pp. 476–489, Mar. 2011.

[4] K. V. Rashmi, N. B. Shah, P. V. Kumar, and K. Ramachandran, "Optimal Exact-Regenerating Codes for Distributed Storage at the MSR and MBR Points via a Product-Matrix Construction," *IEEE Trans. Inf. Theory*, vol. 57, no. 8, pp. 5227–5239, Aug. 2011.

[5] S. Pawar, S. E. Rouayheb, and K. Ramchandran, "On secure distributed data storage under repair dynamics," in *Proc. IEEE ISIT*, Austin, Jun. 2010.

[6] ——, "Securing Dynamic Distributed Storage Systems Against Eavesdropping and Adversarial Attacks," *IEEE Trans. Inf. Theory*, vol. 57, no. 10, pp. 6734–6753, Oct. 2011.

[7] N. B. Shah, K. V. Rashmi, and P. V. Kumar, "Information-Theoretically Secure Regenerating Codes for Distributed Storage," in *Proc. Globecomm*, Houston, Dec. 2011, pp. 1–5.

[8] R. Zhu and W. Guo, "On the secure conditions for distributed storage systems," in *Proc. NetCod*, Calgary, Jun. 2013.

[9] A. S. Rawat, O. O. Koyluoglu, N. Silberstein, and S. Vishwanath, "Optimal locally repairable and secure codes for distributed storage systems," *IEEE Trans. Inf. Theory*, vol. 60, no. 1, pp. 212–236, Jan. 2014.

[10] K. Bhattad and K. R. Narayanan, "Weakly Secure Network Coding," in *Proc. NetCod*, Riva del Garda, Apr. 2005.

[11] P. F. Oliveira, L. Lima, T. T. V. Vinhoza, J. Barros, and M. Mdard, "Coding for trusted storage in untrusted networks," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 6, pp. 1890–1899, Dec. 2012.

[12] S. H. Dau, W. Song, and C. Yuen, "On block security of regenerating codes at the mbr point for distributed storage systems," *arXiv preprint arXiv:1309.2712*, 2013.

[13] K. V. Rashmi, N. B. Shah, P. V. Kumar, and K. Ramachandran, "Explicit Construction of Optimal Exact Regenerating Codes for Distributed Storage," in *Proc. Allerton Conf.*, Urbana-Champaign, Sep. 2009.

[14] D. Silva and F. R. Kschischang, "Universal secure network coding via rank-metric codes," *IEEE Trans. Inf. Theory*, vol. 57, no. 2, pp. 1124–1135, Feb. 2011.

[15] L. H. Ozarow and A. D. Wyner, "The wire-tap channel II," *Bell Syst. Tech. Journ.*, vol. 63, pp. 2135–2157, 1984.

[16] S. E. Rouayheb and E. Soljanin, "On Wiretap Networks II," in *Proc. Int. Symp. Information Theory*, Nice, France, Jun. 2007, pp. 551–555.

[17] J. Lacan and J. Fimes, "Systematic MDS Erasure Codes Based on Vandermonde Matrices," *IEEE Commun. Lett.*, vol. 8, no. 9, pp. 570–572, Sep. 2004.