Secure Regenerating Codes for Hybrid Cloud Storage Systems

Islam Samy, Gokhan Calis, and O. Ozan Koyluoglu
Department of Electrical and Computer Engineering, The University of Arizona
Email: {islamsamy, gcalis, ozan}@email.arizona.edu

***Abstract*—We study the scenario of hybrid cloud storage where the client utilizes both an off-site and a local storage. The former is a distributed storage system (DSS) with the presence of an eavesdropper that has access to the content stored in and downloaded to some subset of nodes. The latter (local) storage is utilized to store a secret key to secure the stored file against the eavesdropper. We introduce two possibilities to utilize local storage (secret key) in enhancing the DSS. First, the key can be used to increase the maximum file size stored in the DSS. We propose an upper bound for this scenario and show constructions achieving it. Second, the key can be used to decrease the number of contacted nodes required to reconstruct the file at the client. We extend the product matrix (PM) framework and construct codes that enables efficient data access. Our analysis includes both minimum repair bandwidth regenerating (MBR) and minimum storage regenerating (MSR) codes.**

I. Introduction

The massive increase in the demand of reliable data storage mechanisms brings into focus the study of distributed storage systems (DSS). Recently, Dimakis et al. [1] proposed a new class of codes called regenerating codes, and derived a bound on the maximum file size that can be stored in the DSS. The performance of regenerating codes is evaluated by two metrics, the storage capacity per node and the total repair bandwidth to regenerate a failed node. Dimakis et al. [1] further established a trade-off between these two metrics, introducing two special cases of regenerating codes: Minimum storage regenerating (MSR) codes and minimum repair bandwidth regenerating (MBR) codes. Several explicit codes have been proposed to achieve these points recently, see, e.g., [2]–[4], and references therein.

Besides the reliability of DSS, the security of these systems also arises as an important concern, as the data stored increase not only in size but also regarding its confidentiality requirements. Utilizing an information theoretic security approach, a bound on the secure file size, in the presence of certain number of eavesdropped nodes (l), is shown in [5]. Examples of code constructions that achieve the secrecy requirements in addition to regenerating code features are introduced in [6]–[8]. In an extended model, [8] considers the presence of eavesdropper $\mathcal{E}$ of type (l_1, l_2) that has access to the content of l_1 nodes, while it can observe all stored and downloaded symbols (during repair) of other l_2 nodes. In these works, to assure that the stored message file ($\mathcal{F}_s$) is totally secure against the eavesdropper, one has to add some randomness ($\mathcal{R}$) in the encoding of the file before storing in DSS.

We assume a scenario where the client of DSS has a local storage in addition to an off-site (cloud) storage. This is a typical scenario in existing cloud deployments, and it is referred to as the hybrid cloud storage. We consider that the local storage is secure from the adversary eavesdropping the cloud storage, and can be used to store data or secret key ($\mathcal{K}$). In particular, local secret key at the client side can be used in addition to $\mathcal{R}$ to guarantee stored file security. Although this setup imposes extra storage at the client, one can benefit from its exclusive availability in two directions. First, one can use such secret key $\mathcal{K}$ to increase the size of stored secured file as compared to the case with the absence of such a key. Second, it can be used to decrease the number of nodes required for data reconstruction (k). We note that the main limiting factor of the previous formulations is to have $k > l$, and our model allows to relax this requirement at the expense of local storage.

The main contributions of this paper are as follows. We introduce the model of hybrid cloud where a secret key stored at the client, which plays the role of the traditionally used randomness to secure the stored file. We derive a bound on the stored secure file size, in the presence of secret key. Then, we utilize the product matrix framework introduced in [3], [8], to achieve the derived bound. We then propose a technique to decrease the number of required nodes for file reconstruction. The main advantage of this part is that it enables the DSS to tolerate a number of eavesdropped nodes (l) that may be greater than the required number for data reconstruction at the client. We show that the proposed construction achieves the same file size as that of the system without local storage, providing an immediate trade-off between the local storage capacity and the data access bandwidth, defined as the required number of downloaded symbols for file reconstruction.

II. System model

We consider a file $\mathcal{F}_s$ of length F_s, which needs to be stored in a DSS consisting of n storage nodes. Each node has a capacity of α symbols, and is under the risk of failure. After each failure, a node called newcomer replaces the failed one. To regenerate the failed α symbols, the newcomer node contacts any d live nodes and downloads β symbols from each. This regeneration process requires a total repair bandwidth of $d\beta$. The client or data collector (DC) can reconstruct all file symbols by contacting any k nodes from the available

This work is supported in part by NSF awards CCF-1563622 and CNS-1617335.

n nodes. We define $\eta = k\alpha$ as the data access bandwidth, represented by the number of symbols required for file reconstruction. We represent such a system as (n, k, d, α, β) DSS. We consider a secret key $\mathcal{K}$, stored locally at the client and used to encode/encrypt the file to guarantee security against the eavesdropper. Note that $H(\mathcal{K})$ is the local storage capacity.

For any node i, we denote its content and the symbols downloaded to repair it as S_i and D_i, respectively. We use the same notation for any set of nodes $\mathcal{A}$, as $S_\mathcal{A}$ and $D_\mathcal{A}$. We use $\mathcal{C}$ to represent the set of all nodes contacted by the client, such that $|\mathcal{C}| = k$. In the presence of an eavesdropper of type (l_1, l_2), we use $\mathcal{E}_1$ and $\mathcal{E}_2$ to denote the set of nodes observed by the adversary, where $\mathcal{E}_1$ is the set of nodes that leak stored content only, whereas $\mathcal{E}_2$ is the set of nodes that leak both content stored and downloaded. The system, to be secure, has to satisfy the following requirements:

- Secrecy property: $\mathcal{F}_s$ is independent of all symbols accessed by the eavesdropper, $S_{\mathcal{E}_1}$ and $D_{\mathcal{E}_2}$, i.e., $H(\mathcal{F}_s) = H(\mathcal{F}_s|S_{\mathcal{E}_1}, D_{\mathcal{E}_2})$.
- Reconstruction property: The client should be able to reconstruct all file symbols from the content of any k nodes, with the availability of the secret key, i.e., $H(\mathcal{F}_s|S_\mathcal{C}, \mathcal{K}) = 0$.
- Regeneration property: The regenerated content of any failed node is a function of the downloaded repair symbols, i.e., $H(S_i|D_i) = 0$.

III. Increasing the secure file size

In this section, we show how to utilize the presence of the secret key to increase the size of stored file. We consider storing a coded version of the file and the secret key in DSS. The presence of the key in the system fulfills the security requirements, while it does not represent any additional randomness or increased entropy for the client. Assume we have a secure DSS with parameters $(n, k, d, \alpha, \beta, \tilde{\mathcal{F}}_s)$. Using the secret key $\mathcal{K}$, we can have another $(n, k, d, \alpha, \beta, \mathcal{F}_s, \mathcal{K})$ DSS, with $\mathcal{F}_s \geq \tilde{\mathcal{F}}_s$. The increase in the secured file size depends on the local storage capacity, where the secret key is stored. For the MBR case, the amount of information each newcomer node downloads during repair is the same as per-node storage ($d\beta = \alpha$), which implies that all eavesdropped nodes can be considered of type l_1 and $l_2 = 0$ can be assumed.

Theorem 1. *The upper bound on the file size with using secret key $\mathcal{K}$ for MBR case is*

$$F_s \leq H(\mathcal{K}) + \sum_{i=l_1+1}^{k} \min\{\alpha, (d-i+1)\beta\}. \tag{1}$$

Proof. Consider $\mathcal{C} = \mathcal{E}_1 \cup \mathcal{G}$, with some set of $|\mathcal{G}| = k - l_1$ nodes, i.e., eavesdropped nodes are subset of the nodes observed by DC.

$$\begin{aligned}
F_s &= H(\mathcal{F}_s) \\
&\overset{(a)}{=} H(\mathcal{F}_s|S_{\mathcal{E}_1}) \\
&\overset{(b)}{=} H(\mathcal{F}_s|S_{\mathcal{E}_1}) - H(\mathcal{F}_s|S_\mathcal{G}, S_{\mathcal{E}_1}, \mathcal{K}) \\
&= I(\mathcal{F}_s; S_\mathcal{G}, \mathcal{K}|S_{\mathcal{E}_1}) \\
&= H(S_\mathcal{G}, \mathcal{K}|S_{\mathcal{E}_1}) - H(S_\mathcal{G}, \mathcal{K}|S_{\mathcal{E}_1}, \mathcal{F}_s) \\
&\leq H(S_\mathcal{G}, \mathcal{K}|S_{\mathcal{E}_1}) \\
&= H(S_\mathcal{G}|S_{\mathcal{E}_1}) + H(\mathcal{K}|S_\mathcal{G}, S_{\mathcal{E}_1}) \\
&\leq H(S_\mathcal{G}|S_{\mathcal{E}_1}) + H(\mathcal{K}) \\
&\overset{(c)}{\leq} \sum_{i=l_1+1}^{k} \min\{\alpha, (d-i+1)\beta\} + H(\mathcal{K}),
\end{aligned} \tag{2}$$

where (a) comes from the secrecy property, (b) follows from the reconstruction property, and (c) is from the classical secure DSS bound as derived in [5]. □

Remark 2. *A code that can achieve this bound can be constructed by replacing the randomness in the PM framework used in [8] with a summation of data symbols and secret key. Due to space limitation, we omit the details.*

Remark 3. *Separation approach: Instead of secret keys, one can use the local storage directly to store secret data, separate from the data encoded with randomness and stored DSS. The derived bound can be achieved here as well. However, this separation approach has an asymmetric data access for the symbols stored in DSS, whereas the secret key based approach allows uniformity in access and security guarantees for the stored symbols.*

Theorem 4. *The upper bound on the file size with using secret key $\mathcal{K}$ for MSR case is*

$$\begin{aligned}
F_s \leq \sum_{i=l_1+l_2+1}^{k} &\min\{\alpha, (d-i+1)\beta\} \\
&- H(D_{\mathcal{E}_2}|S_{\mathcal{E}_1}, S_{\mathcal{E}_2}) + H(\mathcal{K}).
\end{aligned} \tag{3}$$

The proof is omitted due to space limitation.

IV. Lowering access bandwidth: MBR case

Consider storing a coded version of a file and the secret key. As the client already knows the secret key, we can utilize this to decrease the required number of nodes to be contacted for data reconstruction. We consider a DSS with parameters $(n, k, d, \alpha, \beta, \mathcal{F}_s)$. In the presence of the secret key $\mathcal{K}$, we can have another DSS denoted as $(n, k, k', d, \alpha, \beta, \mathcal{F}_s, \mathcal{K})$ (which has two additional parameters compared to previous DSS), where k' is the required number of nodes for file reconstruction with the availability of $\mathcal{K}$, and k is the required number of nodes in case that the secret key is not available. We remark that the key-based system can be used to increase l, the tolerable number of eavesdropped nodes, where we can tolerate $l = l_1 + l_2 < k$, and at the same time we can reconstruct the file from $k' < k$ nodes.

In this section, we extend the product matrix (PM) framework, building on the one provided in [8] and [3], which is shown to be optimal for the MBR case. We omit discussing the regeneration property and the security of our construction against the eavesdropper as these properties follow from the analysis given in [8]. Instead, we detail the analysis of the reconstruction property. Note that, in the MBR case, all nodes accessed by the eavesdropper are of type l_1, and $l_2 = 0$. Therefore, we have $l = l_1$.

A. Secure PM-MBR construction [8]

For any $(n, k, d, \alpha, \beta = 1, l)$ DSS, let $C = \Psi M$ be an $n \times \alpha$ code matrix where each α symbols in row i represent the content of node i. Ψ represents the $n \times d$ fixed encoding matrix used to encode the symmetric $d \times \alpha$ matrix M that includes all file symbols. To guarantee the regenerating and reconstruction properties, M and Ψ are constructed as follows

$$M = \begin{bmatrix} \underset{k\times k}{S} & \underset{k\times(d-k)}{T} \\ \underset{(d-k)\times k}{T^t} & \underset{(d-k)\times(d-k)}{0} \end{bmatrix}, \qquad \Psi = [\underset{n\times k}{\Phi} \quad \underset{n\times(d-k)}{\Delta}], \tag{4}$$

where matrix S has to be symmetric, and Ψ should have the following properties: i) Any d rows of Ψ are independent, ii) any k rows of Φ are independent, iii) restricted to the first l columns, any l rows are linearly independent. Any Vandermonde matrix (Ψ_V) can be used to satisfy these properties. To make this construction secure, the random symbols should be stored in the first l rows (and columns due to symmetry) of M.

B. Decreasing the number of required nodes for file reconstruction utilizing $\mathcal{K}$

We consider enhancing the reliability of DSS by making it possible to reconstruct the file from k' nodes instead of k nodes $(k' < k)$. Note that, if we want to decrease k to k', we should have the message matrix M' (in case of contacting k' nodes) as

$$M' = \begin{bmatrix} \underset{k'\times k'}{S} & \underset{k'\times(d-k')}{T} \\ \underset{(d-k')\times k'}{T^t} & \underset{(d-k')\times(d-k')}{0} \end{bmatrix}. \tag{5}$$

For this construction, we must have an encoding matrix Ψ that fulfills the requirements mentioned above. We remark that Vandermonde matrix and all of its permuted column versions represent a valid choice for Ψ. Accordingly, we take $\Psi = \Psi_V P$, where P is any permutation matrix, with the identity matrix I being a special case of it.

Construction I: Consider $\left(n, k, d, \alpha, \beta = 1, l, F_s, K = ld - \binom{l}{2}\right)$ DSS, where F_s is at least the same maximum secure file size, see, e.g., [8]. In the PM-MBR-based construction [8], the authors replace the first l rows (and columns) with randomness. Instead of that, in our construction, we consider replacing the data symbols in certain positions in these rows and columns. More specifically, we modify M as follows:

$$\begin{gathered} m_{ij} = m_{ji} = 0 \\ \forall i \le k - k', \quad \forall j \le k - k' \quad \text{and} \quad \forall j > k. \end{gathered} \tag{6}$$

Also, we introduce the symmetric $d \times d$ matrix A^κ, where the secret key $\mathcal{K}$ is stored in its first l rows (and columns).

We construct the symmetric permutation matrix P as follows:

$$P_{ij} = \begin{cases} 1, & \forall i \le k - k' \quad \text{or} \quad \forall k' + 1 \le i \le k, \\ & \quad \text{and} \quad j = k + 1 - i, \\ 1, & \forall k - k' < i \le k' \quad \text{or} \quad \forall i > k, \\ & \quad \text{and} \quad j = i, \\ 0, & \text{otherwise}, \end{cases} \tag{7}$$

such that it enables us to guarantee that the received content at the client can be later transformed to $\Psi_V M'$. Define $M^S = M + A^\kappa$, then code matrix C can be written as

$$C = \Psi M^S = \Psi_V P M^S = \Psi_V P (M + A^\kappa). \tag{8}$$

Reconstruction: From any k' nodes, the DC can access the symbols X'_{DC} such that,

$$\begin{aligned} X'_{DC} &= \Psi'_{V_{DC}} P M^S = \Psi'_{V_{DC}} P (M + A^\kappa) \\ &= \Psi'_{V_{DC}} P M + \Psi'_{V_{DC}} P A^\kappa, \end{aligned} \tag{9}$$

where $\Psi'_{V_{DC}}$ is the submatrix related to the contacted k' nodes. Since the secret key is known at the DC, the second part of the summation in (9) can be subtracted from X'_{DC}, then

$${X'_{DC}}^{(1)} = X'_{DC} - \Psi'_{V_{DC}} P A^\kappa = \Psi'_{V_{DC}} P M. \tag{10}$$

By multiplying the same P matrix from the right, we have

$${X'_{DC}}^{(2)} = {X'_{DC}}^{(1)} P = \Psi'_{V_{DC}} P M P = \Psi'_{V_{DC}} M_P, \tag{11}$$

where $M_P = PMP$. Now, we are able to write M_P as

$$M_P = \begin{bmatrix} \underset{k'\times k'}{S} & \underset{k'\times(d-k')}{T} \\ \underset{(d-k')\times k'}{T^t} & \underset{(d-k')\times(d-k')}{0} \end{bmatrix}. \tag{12}$$

As any k rows of Ψ_V are independent, then we are certain that any k' are also independent. The remaining part $\Psi'_{V_{DC}} M_P$ enables us to reconstruct M_P the same way as in the PM-MBR code construction, and finally we can get M by $M = P M_P P$.

Remark 5. *This approach allows us to decrease the data access bandwidth η, for the same file size reconstruction, from $k\alpha$ to $k'\alpha$ with an enhancement depending on the ratio of k'/k.*

Remark 6. *$k - k' \le l$: We cannot decrease k by more than l, as this construction depends on increasing the dimension of the low right corner zero submatrix of message matrix M using permutation matrix. Also, we are only able to replace the data symbols in the first l rows (and columns) with zeros, thus we cannot increase the dimension of the corresponding zero submatrix by more than l.*

Example 7. *For $(n = 6, k = 4, d = 5, \alpha = 5, \beta = 1, l = 2, K = 9)$, in order to reduce k to $k' = 2$, from (6) and (7), we can construct M and P as follows*

$$M = \begin{bmatrix} 0 & 0 & u_3 & u_4 & 0 \\ 0 & 0 & u_7 & u_8 & 0 \\ u_3 & u_7 & u_{10} & u_{11} & u_{12} \\ u_4 & u_8 & u_{11} & u_{13} & u_{14} \\ 0 & 0 & u_{12} & u_{14} & 0 \end{bmatrix}. \tag{13}$$

$$P = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{14}$$

and A^κ will be

$$A^\kappa = \begin{bmatrix} \kappa_1 & \kappa_2 & \kappa_3 & \kappa_4 & \kappa_5 \\ \kappa_2 & \kappa_6 & \kappa_7 & \kappa_8 & \kappa_9 \\ \kappa_3 & \kappa_7 & 0 & 0 & 0 \\ \kappa_4 & \kappa_8 & 0 & 0 & 0 \\ \kappa_5 & \kappa_9 & 0 & 0 & 0 \end{bmatrix}. \tag{15}$$

From (13) and (15), M^S can be written as

$$M^S = M + A^\kappa = \begin{bmatrix} \kappa_1 & \kappa_2 & u_3+\kappa_3 & u_4+\kappa_4 & \kappa_5 \\ \kappa_2 & \kappa_6 & u_7+\kappa_7 & u_8+\kappa_8 & \kappa_9 \\ u_3+\kappa_3 & u_7+\kappa_7 & u_{10} & u_{11} & u_{12} \\ u_4+\kappa_4 & u_8+\kappa_8 & u_{11} & u_{13} & u_{14} \\ \kappa_5 & \kappa_9 & u_{12} & u_{14} & 0 \end{bmatrix}. \tag{16}$$

At the DC, we can access X'_{DC}, and we subtract $\Psi'_{V_{DC}} P A^\kappa$ from it, then multiply it by P.

$${X'_{DC}}^{(2)} = \Psi'_{V_{DC}} \begin{bmatrix} u_{13} & u_{11} & u_8 & u_4 & u_{14} \\ u_{11} & u_{10} & u_7 & u_3 & u_{12} \\ u_8 & u_7 & 0 & 0 & 0 \\ u_4 & u_3 & 0 & 0 & 0 \\ u_{14} & u_{12} & 0 & 0 & 0 \end{bmatrix}. \tag{17}$$

Then, we can recover M_P using any 2 nodes, and M equals

$$M = P M_P P. \tag{18}$$

V. Lowering access bandwidth: MSR case

In this section, we revisit the secure construction of MSR codes using the product matrix form in [8], for the case $d = 2k-2$. In [9], it is mentioned that this construction is optimal for $n = d+1$ and any (l_1, l_2) eavesdropper. Hence, we study any DSS with $(n = 2k-1, k, d = 2k-2, \alpha = k-1, \beta = 1)$.

A. Secure PM-MSR construction [8]

In this construction, we have an $n \times \alpha$ code matrix C, such that $C = \Psi M$, where Ψ is the $n \times d$ encoding matrix, which can be represented by a Vandermonde matrix. On the other hand, M is the $d \times \alpha$ message matrix, written in the form

$$M = \begin{bmatrix} S_1 \\ S_2 \end{bmatrix}, \tag{19}$$

where S_1 and S_2 are two $(\alpha \times \alpha)$ symmetric matrices. All file symbols are stored in the upper triangles of both S_1 and S_2. Regeneration and reconstruction properties of this construction are shown in [3]. This construction is shown to be secure against (l_1, l_2) eavesdropper. In this case, the secure file size is $F_s = (k - l_1 - l_2)(\alpha - l_2)$. In order to secure the file, we need to add R random symbols, such that $R = (l_1 + l_2)\alpha + (k - l_1 - l_2)l_2$. These R symbols are stored in M as follows: i) $(l_1 + l_2)\alpha - \binom{l_1+l_2}{2}$ symbols in the first $l_1 + l_2$ rows of S_1, ii) $\binom{l_1+l_2}{2}$ symbols in the intersection of the first $l_1 + l_2 - 1$ rows and columns of S_2, iii) the remaining $(k - l_1 - l_2)l_2$ symbols in the remaining elements of the first l_2 rows of S_2.

B. Decreasing the number of required nodes for file reconstruction utilizing $\mathcal{K}$

We utilize the secret key $\mathcal{K}$ in place of randomness in order to decrease k. Unlike the MBR case, k here affects our PM construction, as it affects both d and α directly. Therefore, we need to take that into consideration when we reconstruct the message matrix using k' nodes instead of k nodes.

First, suppose we have an $(n', k', d' = 2k'-2, \alpha' = k'-1, \beta' = 1)$ PM-MSR code, with encoding matrix Ψ'. We know that we can reconstruct the message matrix from $X''_{DC} = \Psi'_{k' \times d'} M_{d' \times \alpha'}$, where X''_{DC} is the content of any k' nodes in such a coded system. While for an $(n, k, d = 2k-2, \alpha = k-1, \beta = 1)$ PM-MSR code, we need to reconstruct the file from X'_{DC}, since DC only contacts k' nodes. In our construction, we will be able to reconstruct M from $X'_{DC} = \Psi_{k' \times d} M_{d \times \alpha}$. In order to guarantee that, we need to get X''_{DC} from X'_{DC}, then we get M from X''_{DC}. Note that Ψ' is different from Ψ, but both of them are represented by Vandermonde matrix. That is, e.g., $\Psi = \underset{n \times \alpha}{[\Phi \quad \Lambda \quad \Phi]}_{(n \times n)(n \times \alpha)}$, which must have the following properties: i) Any d rows of Ψ are independent, ii) any k rows of Φ are independent, iii) the elements of Λ are distinct. The first property is to guarantee the regeneration property, while the other two properties are for data reconstruction. Thus, we use Φ as a Vandermonde matrix, whose i^{th} row is $[1, i, i^2 \dots]$. As we do not modify the stored code matrix, the construction readily satisfies the regeneration property, and we here focus on the other two properties when we reconstruct the file from k' nodes instead of k.

Construction II: Consider $\Big(n, k, d, \alpha, \beta = 1, l_1, l_2, F_s, K = (l_1 + l_2)\alpha + (k - l_1 - l_2)l_2\Big)$ DSS, where F_s is at least the same maximum secure file size, using traditional randomness, achieved in [8]. First, we construct matrix M, such that we replace the first $k - k'$ rows (and columns) of S_1 and S_2 by zeros, those are already included in the rows that are replaced with randomness in the secure PM-MSR in [8]. Then, M can be written as

$$\underset{d \times \alpha}{M} = \begin{bmatrix} 0 & \dots 0 \\ \vdots & \underset{\alpha' \times \alpha'}{S'_1} \\ 0 & \dots 0 \\ \vdots & \underset{\alpha' \times \alpha'}{S'_2} \end{bmatrix}, \tag{20}$$

where $\alpha' = \alpha - (k - k') = k' - 1$. We can construct the matrix A^κ as

$$A^\kappa = \begin{bmatrix} A_1^\kappa \\ A_2^\kappa \end{bmatrix}, \tag{21}$$

where A_1^κ and A_2^κ are two symmetric matrices, such that the secret key $\mathcal{K}$ is stored in all places that originally have randomness in the scheme of [8]. That is, we have that the first $k - k'$ rows (columns) of A_1^κ and A_2^κ must be filled by

$\mathcal{K}$. Then, we have $M^S = M + A^\kappa$, and the code matrix C can be written as

$$C = \Psi M^S = \Psi(M + A^\kappa). \tag{22}$$

Reconstruction: Let Ψ'_{DC} be the submatrix of Ψ, which represents the k' nodes connected to the DC. Thus, the DC can access X'_{DC} symbols, such that

$$\begin{aligned} X'_{DC} &= \Psi'_{DC} M^S = \Psi'_{DC}(M + A^\kappa) \\ &= \Psi'_{DC} M + \Psi'_{DC} A^\kappa. \end{aligned} \tag{23}$$

As the secret key is already stored at the DC, then it can construct A^κ. Then, it can subtract $\Psi'_{DC} A^\kappa$ to get

$$X'_{DC}{}^{(1)} = X'_{DC} - \Psi'_{DC} A^\kappa = \Psi'_{DC} M. \tag{24}$$

Now, let ϕ_i be the i^{th} column in matrix Φ', then we can write $X'_{DC}{}^{(1)}$ as in (24),

$$\begin{aligned} X'_{DC}{}^{(1)} &= \begin{bmatrix}[\phi_1 \ \phi_2 \ \dots \ \phi_\alpha] & \Lambda\,[\phi_1 \ \phi_2 \ \dots \ \phi_\alpha]\end{bmatrix} \begin{bmatrix} 0 & \dots 0 \\ \vdots & S'_1 \\ & {\scriptstyle \alpha' \times \alpha'} \\ 0 & \dots 0 \\ \vdots & S'_2 \\ & {\scriptstyle \alpha' \times \alpha'} \end{bmatrix} \\ &= \begin{bmatrix}[\phi_{k-k'+1} \ \dots \ \phi_\alpha] & \Lambda\,[\phi_{k-k'+1} \ \dots \ \phi_\alpha]\end{bmatrix} \begin{bmatrix} 0 & S'_1 \\ & {\scriptstyle \alpha' \times \alpha'} \\ \vdots & S'_2 \\ & {\scriptstyle \alpha' \times \alpha'} \end{bmatrix}. \end{aligned} \tag{25}$$

As, we use Φ as a Vandermonde matrix, whose i^{th} row is $[1, i, i^2 \dots]$, we can use the fact that

$$\phi_i = \begin{bmatrix} 1 & & & \\ & 2 & & \\ & & \ddots & \\ & & & k' \end{bmatrix} \phi_{i-1} = \begin{bmatrix} 1 & & & \\ & 2^{i-1} & & \\ & & \ddots & \\ & & & (k')^{i-1} \end{bmatrix} \phi_1 \quad \forall i > 1. \tag{26}$$

Now, we can write (25) as

$$\begin{aligned} X'_{DC}{}^{(1)} = &\begin{bmatrix} 1 & & & \\ & 2^{k-k'} & & \\ & & \ddots & \\ & & & (k')^{k-k'} \end{bmatrix} \Big[[\phi_1 \ \dots \ \underbrace{\phi_{\alpha-(k-k')}}_{\alpha'}] \cdots \\ &\cdots \Lambda\,[\phi_1 \ \dots \ \underbrace{\phi_{\alpha-(k-k')}}_{\alpha'}]\Big] \begin{bmatrix} 0 & S'_1 \\ & {\scriptstyle \alpha' \times \alpha'} \\ \vdots & S'_2 \\ & {\scriptstyle \alpha' \times \alpha'} \end{bmatrix}, \end{aligned} \tag{27}$$

For simplicity, we suppose that DC is connected to the first k' nodes. We can see that the first diagonal matrix is invertible, so by multiplying by its inverse from the left we get

$$\begin{aligned} X'_{DC}{}^{(2)} &= \begin{bmatrix}[\phi_1 \ \dots \ \phi_{\alpha'}] & \Lambda\,[\phi_1 \ \dots \ \phi_{\alpha'}]\end{bmatrix} \begin{bmatrix} 0 & S'_1 \\ & {\scriptstyle \alpha' \times \alpha'} \\ \vdots & S'_2 \\ & {\scriptstyle \alpha' \times \alpha'} \end{bmatrix} \\ &= \begin{bmatrix} 0 & X''_{DC} \\ {\scriptstyle k' \times (k-k')} & {\scriptstyle k' \times \alpha'} \end{bmatrix}, \end{aligned} \tag{28}$$

where

$$\begin{aligned} X''_{DC} &= \begin{bmatrix}[\phi_1 \ \dots \ \phi_{\alpha'}] & \Lambda\,[\phi_1 \ \dots \ \phi_{\alpha'}]\end{bmatrix} \begin{bmatrix} S'_1 \\ {\scriptstyle \alpha' \times \alpha'} \\ S'_2 \\ {\scriptstyle \alpha' \times \alpha'} \end{bmatrix} \\ &= \begin{bmatrix} \Phi' & \Lambda & \Phi' \\ {\scriptstyle k' \times \alpha'} & {\scriptstyle (k' \times k')(k' \times \alpha')} \end{bmatrix} \begin{bmatrix} S'_1 \\ {\scriptstyle \alpha' \times \alpha'} \\ S'_2 \\ {\scriptstyle \alpha' \times \alpha'} \end{bmatrix}. \end{aligned} \tag{29}$$

Now, we satisfy the two required properties for data reconstruction (any α' rows of Φ' are independent, and the diagonal elements of Λ are distinct). Hence, we can consider X''_{DC} as the content of any k' nodes of an $(n, k', d' = 2k' - 2, \alpha' = k' - 1)$ PM-MSR code. Finally, we can restore the non-zero elements of M from X''_{DC} as the same as in PM-MSR codes.

VI. Conclusion

In this work, we studied the hybrid cloud storage systems, where the client has a local storage in addition to off-site DSS. Although the local storage can be used for storing data, separate from that in DSS, we introduced the idea of utilizing this storage as secret key. The main idea underlying our approach is to replace the randomness used to secure the stored file against eavesdropper, with a linear combination of the file and secret key. Furthermore, we proposed two approaches on how to utilize the secret key. First, we used the secret key to increase the secure file size stored in DSS, and we derived an upper bound on the file size. We also used the secret key to decrease the required number of contacted nodes to reconstruct the stored file at the client. Our constructions are based on extending the PM framework [3] utilized in [8]. An avenue for future work is to characterize the trade-offs between access bandwidth, local storage and file size.

References

[1] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," *IEEE Trans. Inf. Theory*, vol. 56, no. 9, pp. 4539–4551, Sep. 2010.

[2] I. Tamo, Z. Wang, and J. Bruck, "Zigzag codes: MDS array codes with optimal rebuilding," *IEEE Trans. Inf. Theory*, vol. 59, no. 3, pp. 1597–1616, Mar. 2013.

[3] 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 Trans. Inf. Theory*, vol. 57, p. 5227, 2011.

[4] A. G. Dimakis, K. Ramchandran, Y. Wu, and C. Suh, "A survey on network codes for distributed storage," *Proc. IEEE*, vol. 99, no. 3, pp. 476–489, Mar. 2011.

[5] S. Pawar, S. El Rouayheb, and K. Ramchandran, "On secure distributed data storage under repair dynamics," in *Proc. 2010 IEEE International Symposium on Information Theory (ISIT 2010)*, Austin, TX, Jun. 2010.

[6] S. Goparaju, S. El Rouayheb, R. Calderbank, and H. Poor, "Data secrecy in distributed storage systems under exact repair," in *Proc. International Symposium on Network Coding (NetCod)*, Calgary, AB, CA, June 2013.

[7] 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, Jan. 2014.

[8] N. Shah, K. Rashmi, and P. Kumar, "Information-theoretically secure regenerating codes for distributed storage," in *Proc. 2011 IEEE Global Communications Conference*, Houston, TX, Dec 2011.

[9] K. Huang, U. Parampalli, and M. Xian, "On secrecy capacity of minimum storage regenerating codes," *CoRR*, vol. abs/1505.01986, May 2015.