Secure Clustered Distributed Storage Against Eavesdropping

Beongjun Choi, *Student Member, IEEE*, Jy-Yong Sohn, *Student Member, IEEE*, Sung Whan Yoon, *Member, IEEE*, and Jaekyun Moon, *Fellow, IEEE*

***Abstract*—This paper investigates interplay among storage overhead, bandwidth requirement, and security constraint in distributed storage. In the model used in our analysis, storage nodes are dispersed in multiple clusters. When a node fails, necessary content gets restored by downloading data from different nodes that may possibly be in other clusters. The bandwidth required for transferring data for node repair is assumed more scarce for cluster-to-cluster links than the links connecting intra-cluster nodes. Eavesdropping takes place on links across clusters only, and a fraction of the total number of clusters is assumed compromised. When a cluster is compromised, any repair traffic going in and out of it is eavesdropped. For this clustered model with eavesdroppers, we analyze the security of distributed storage systems (DSSs) and provide guidelines on designing system solutions for securing the data. First, under the setting of functional repair, we derive a general upper bound on the secrecy capacity, the maximum data size that can be stored in DSSs with perfect secrecy. In the practically important bandwidth-limited regime where the node storage size is equal to the repair bandwidth, the upper bound is shown to be achievable through proposed code constructions. Moreover, we obtain a closed-form expression for the required system resources-node storage size and repair bandwidth-to store a given amount of data with perfect secrecy. Second, we investigate the behavior of secrecy capacity as the number of compromised clusters increases. According to our mathematical analysis, the secrecy capacity decreases as a quadratic function until the number of compromised clusters reaches a certain threshold. Finally, based on the fundamental relationship between the system resources and the secrecy capacity, we provide a guideline on balancing intra- and cross-cluster repair bandwidths depending on the given system security level.**

***Index Terms*—Distributed storage system, secrecy capacity, eavesdropper, network coding.**

I. Introduction

A DISTRIBUTED storage system (DSS) is a network of storage nodes to reliably store massive amounts of data under frequent node failure events. Good examples are the large-scale data centers that employ storage nodes widely spread over the Internet [2]–[5]. The file to be stored is erasure-coded and distributed to n storage nodes, so that the legitimate user is able to reconstruct the complete file by contacting any k out of n active storage. In order to support reliable storage, a DSS is typically able to repair nodes in the case of node failure events using predefined coding scheme. If an active node fails, a "newcomer node" joins the system by downloading data from any $d < n$ active helper nodes.

Clusters naturally arise in the form of racks in data centers but may also exist in many different variations that may involve wireless connection links as well. For example, wireless local area networks connected together in wireless links may be seen as forming clusters. Connection speeds within the cluster can be faster than those across clusters. When a node fails, a newcomer node gets established within the same cluster, which promptly downloads necessary data from any node within or outside the same cluster. Repair bandwidth across clusters is generally more restricted than within cluster, which adds significant twist to the trade-off picture of storage overhead versus repair bandwidth [6].

Storage nodes and links in DSSs can be vulnerable to intruder attacks, and guaranteeing system security against possible threats is a major challenge. Intruder models for DSSs are generally a node-based model [7], [8] where the intruder directly invades individual storage nodes. Much research efforts have gone into securing DSSs against these types of intruders [7]–[17]. Considering the clustered nature of storage nodes in many real world DSSs, however, conventional intruder models may not fully capture the behavior of the attackers. In this paper, we utilize what we call the cluster eavesdropper model, where cross-cluster repair links can be vulnerable to eavesdroppers whereas intra-cluster links are assumed secure. We assume that a certain number of clusters are compromised, and when a cluster is compromised any repair traffic that crosses its boundary is eavesdropped. Thus, in our cluster eavesdropper model an eavesdropper cannot access the individual storage nodes residing in the clusters, but can monitor data that go in and out of a certain number of clusters during the node repair process.

Under this model, this paper attempts to provide answers to the following key questions on securing clustered DSSs: What is the maximum amount of information that can be stored securely given the available cross- and intra-cluster

Manuscript received July 30, 2018; revised June 19, 2019; accepted June 19, 2019. Date of publication June 25, 2019; date of current version October 18, 2019. This work was supported by the National Research Foundation of Korea under Grant 2016R1A2B4011298. This paper was presented in part at the IEEE Conference on Communications (ICC), Paris, France, May 2017 [1].

The authors are with the School of Electrical Engineering, Korea Advanced Institute of Science and Technology, Daejeon 305-701, South Korea (e-mail: bbzang10@kaist.ac.kr; jyshon1108@kaist.ac.kr; shyoon8@kaist.ac.kr; jmoon@kaist.edu).

Communicated by A. Menezes, Associate Editor for Complexity and Cryptography.

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIT.2019.2924895

repair bandwidths? As a larger number of clusters become compromised, sending data through other clusters will generally incur a bigger capacity loss. How rapidly would capacity be degraded as a function of the number of compromised clusters? How should the repair burden be balanced through intra-cluster and cross-cluster links in the presence of the given security threats?

A. Contributions

To our best knowledge, secrecy issues reflecting clustering characteristic of DSS have rarely been studied. Compared to previous works in securing DSS, we utilize a new node repair model suitable for a clustered DSS wherein the repair bandwidths assigned to intra- and cross-cluster links are imbalanced [6]. Under the setting of *functional repair*, which does not require the contents of the replacement node to be exactly the same as the failed node, we derive a general upper bound on the *secrecy capacity*, i.e., the maximum data size that can be stored in DSS with perfect secrecy, when clusters could be compromised by eavesdroppers. We demonstrate achievability of the bound in the practically important *bandwidth-limited regime*, where the node storage size α is equal to the repair bandwidth γ, by presenting secure coding schemes achieving the bound. The suggested coding schemes have an *exact repair* property, where regeneration directly restores the exact copies of the failed node, and are applicable for general system parameters (n, k) with maximal helper nodes. The coding schemes are based on a concatenation of the symbol distribution rule in [18] and the nested maximum distance separable (MDS) code in [19]. Based on the analytical results on secrecy capacity, we obtain a closed-form expression for the required amount of resources - node storage size and repair bandwidth - to securely store a given amount of data. Finally, we provide some key insights on the system behavior under the security constraint. In particular, we show that secrecy capacity is represented as a quadratically decreasing function of the number of compromised clusters L_c when L_c is small enough, and remains at a constant value as L_c is greater than a certain threshold. Moreover, we obtain the optimal ratio of cross- to intra-cluster repair bandwidth to maximize secrecy capacity, depending on the eavesdropper's capability. Interestingly, the optimal solution depends largely on whether the portion of the compromised clusters is larger than $1-\sqrt{1-\mathcal{R}(1-\mathcal{R})}$, where $\mathcal{R}=k/n$ is the code rate.

B. Related Work

There have been some suggestions on modeling intruder behaviors in a DSS [7]–[9]. Roughly, intruder models are designed and classified into two types in DSSs. One is an active intruder model in which the attacker can maliciously change stored or transmitted data in the system, while the other is a passive eavesdropper model where the attacker only observes data without altering it. The passive eavesdropper model is classified further into node-based intruders [7], [8] and link-based intruders [9]. The node-based eavesdropper model suggested in [7], [8] allows access to individual storage nodes in the system. The model in [7] has a parameter l which represents the total number of accessible nodes by the eavesdropper. In this model the intruder eavesdrops stored contents and downloaded data (during repair process) from l compromised storage nodes. On the other hand, the authors of [8] divide the compromised storage nodes into two groups. They use two parameters l_1, l_2, such that an eavesdropper observes stored contents up to l_1 nodes and has an access to downloaded data up to l_2 nodes. When $l_1 = l_2 = l$, the two eavesdropper models suggested in [7], [8] become identical. Unlike the node-based eavesdroppers, the link-based eavesdroppers suitable for a DSS have been rarely discussed in the literature. The authors of [9] classify data centers into two groups, local and remote, and assume transmitted data from remote sites to a local site are more likely to be eavesdropped compared to transmitted data within a local site. In this aspect, our cluster eavesdropper model is similar to the model of [9] in that cross-cluster transmission are more likely to be compromised. However, the present paper reflects asymmetric repair more suitable for multiple-cluster scenarios while [9] still uses the conventional symmetric repair model of [20].

In the early work of Dimakis *et al.* [20], a fundamental trade-off relationship between node storage size α and repair bandwidth γ is found to reliably store a file with given size. Repair-efficient codes which follow the fundamental trade-off curve are called *regenerating codes*. In the literature, two extreme points in the trade-off curve have been extensively studied: the minimum storage regenerating (MSR) point (minimize the storage size α) and the minimum bandwidth regenerating (MBR) point (minimize the repair bandwidth γ).

There have been much discussions on how to securely store data against intruders [1], [7]–[17]. The authors of [7] define *secrecy capacity* as the maximum securely stored data in a system against passive eavesdropper with perfect secrecy. The authors of [7] provide a tight upper bound on secrecy capacity and present a coding scheme which achieves the upper bound in the bandwidth-limited regime. In [8], secure codes based on the product matrix codes are proposed. The suggested scheme achieves an upper bound presented in [7] at the MBR point with any set of system parameter values for n, k and d. They also construct low-rate information-theoretically secure MSR codes with a constraint that an eavesdropper can observe stored data but no downloaded data ($l_2 = 0$). An improved upper bound on secrecy capacity in the MSR region and secure codes based on maximum rank distance codes are obtained in [10]. More secure codes following the fundamental trade-off curves are suggested and analyzed in [11]–[13]. The authors of [14] consider multiple node failure scenarios and employ an exact-repair secure coding scheme for the MBR region. Relaxing the perfect secrecy condition, the scenario of eavesdropper accessing a portion of the stored data is considered in [9], [15]. The trade-off between storage size and repair bandwidth in exact-repair secure codes in the presence of eavesdropper is investigated in [16], [17].

Although several researchers have already considered the clustered nature of storage nodes [6], [21]–[23], secrecy problems have not been previously addressed in a sufficient fashion in the context of clustered DSSs. The work of [24] discusses secrecy capacity of a heterogeneous DSS where each storage node has distinct storage size and requires different repair bandwidth. The model, however, does not reflect the clustered structure. Of the several existing DSS models, we basically adopt the model suggested in [6]. This model is suitable to a practical scenario where the available bandwidth associated with cross-cluster traffic is restricted compared to that for intra-cluster traffic. When recovering a failed node, helper nodes provide different amounts of data depending on whether a regenerated node is in the same cluster or not.

C. Organization

This paper is organized as follows. We provide a threat model and an overview of the system for the clustered DSS in Section II. We provide results on secrecy capacity and derive the feasible region of system resources in Section III and IV. In Section V, we analyze secrecy capacity with various system parameters in the bandwidth-limited regime. Finally, secure code construction is provided in Section V-D and we draw conclusions and provide future research directions in Section VII.

II. System and Threat Model

A. Clustered DSS Model [6]

The system model for clustered DSS in [6] is reviewed. Suppose a source wants to store $\mathcal{M}$ symbols of data in a clustered DSS, where each symbol belongs to a finite field $\mathbb{F}_s$. The source encodes the data to ensure reliability and distributes the encoded data to n storage nodes, each of which has capacity to store α symbols over $\mathbb{F}_s$. Active n storage nodes are distributed in L clusters where the number of nodes per cluster is fixed to $n_I = n/L$. A clustered DSS employs erasure coding so that a data collector should be able to reconstruct the entire data by retrieving encoded data from any choice of $k < n$ storage nodes.

In a DSS, individual storage nodes frequently fail to provide a reliable data [25]. In order to store message for a long period of time, system provides the node repair process to keep n active storage nodes. When an active node fails,[1] a replacement node joins the system by receiving β_I symbols from each of the d_I helper nodes in the same cluster, and β_c symbols from each of the d_c helper nodes in all other clusters. A clustered DSS considers a realistic scenario wherein the bandwidth assigned for intra-cluster repair is more abundant than that for cross-cluster repair, it is assumed that

$$\beta_I \geq \beta_c.$$

In case of node failures, a failed node and corresponding newcomer node reside in the identical cluster to keep constant number n_I of active storage nodes per cluster. The total repair bandwidth to regenerate a single replacement node is expressed as $\gamma = \gamma_I + \gamma_c$ where $\gamma_I = d_I\beta_I$ is the intra-cluster repair bandwidth and $\gamma_c = d_c\beta_c$ is the cross-cluster repair bandwidth. Finally, we define q and r as the quotient and remainder of k divided by n_I. We summarize the parameters used in the paper in Table I.

[1]It is assumed that only single node failure occurs in the system since a vast majority of data recovery required in practice is due to single node failure [25].

TABLE I
Parameters Used in Clustered Distributed Storage [6]

n	number of storage nodes
k	number of DC-contacting nodes
L	number of clusters
$n_I = n/L$	number of nodes in a cluster
α	storage capacity of each node
$d = d_I + d_c$	total number of helper nodes
d_I	number of intra-cluster helper nodes
d_c	number of cross-cluster helper nodes
β_I	intra-cluster repair bandwidth (per node)
β_c	cross-cluster repair bandwidth (per node)
$\gamma_I = d_I\beta_I$	intra-cluster repair bandwidth
$\gamma_c = d_c\beta_c$	cross-cluster repair bandwidth
$\gamma = \gamma_I + \gamma_c$	total repair bandwidth
$q = \lfloor k/n_I \rfloor$	quotient of k divided by n_I
$r = mod(k, n_I)$	remainder of k divided by n_I

Unlike other models dealing with a clustered DSS [23], [26], the present model assumes that individual storage nodes does not aggregate data within a cluster. Considering general scenarios where nodes are clustered and connected each other with limited resources, the present model without aggregation is practically well-suited. Now, we summarize a main result of [6] in the following lemma which obtains *capacity*, i.e., the maximum amount of reliably storable data, of a clustered DSS.

Lemma 1 (Theorem 1 of [6]). *The capacity $\mathcal{C}$ of a clustered DSS with maximum number of helper nodes ($d_I = n_I - 1$, $d_c = n - n_I$) is*

$$\mathcal{C} = \sum_{i=1}^{n_I}\sum_{j=1}^{g_i} \min\{\alpha, \rho_i\beta_I + (n - \rho_i - j - \sum_{m=1}^{i-1} g_m)\beta_c\}, \quad (1)$$

where

$$\rho_i = n_I - i,$$

$$g_m = \begin{cases} q+1, & m \leq r \\ q, & otherwise. \end{cases}$$

Remark 1 (Proposition 2 of [6]). *For every (i, j) with $i \in [n_I]$, $j \in [g_i]$, the following inequality holds.*

$$\gamma \geq \rho_i\beta_I + (n - \rho_i - j - \sum_{m=1}^{i-1} g_m)\beta_c$$

B. Cluster Eavesdropper Model

We introduce the cluster eavesdropper model of a clustered DSS. Note that in case of node failures, the necessary content gets restored by downloading data from different nodes that may possibly be in other clusters. A cluster eavesdropper

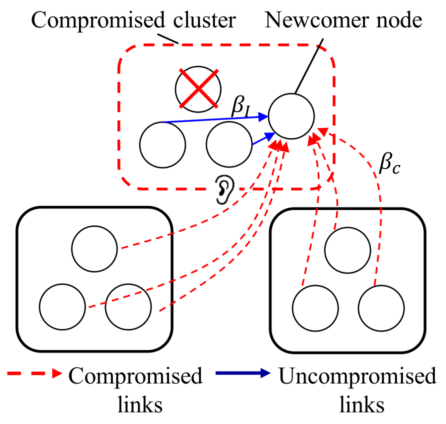


Fig. 1. The cluster eavesdropper model. The eavesdropper can observe incoming and outgoing data from compromised clusters but cannot access intra-cluster links.

compromises a certain number L_c of total clusters, and eavesdrops repair traffic of both incoming and outgoing data of compromised clusters. Repair traffic residing in the clusters and contents stored in individual nodes are not directly observed by a cluster eavesdropper. We assume that a cluster eavesdropper has sufficient computational power and complete knowledge of the repair protocol, like in other eavesdropper models [7], [8] for a DSS. The *secrecy capacity* of a clustered DSS is defined as the maximum amount of data that can be stored in the system with perfect secrecy requirement, i.e., without revealing any data to the cluster eavesdropper. To be more precise, for a given message vector $\mathbf{m}$ to be stored in the system and a vector of revealed symbols $\mathbf{e}$ which the eavesdropper observes, the condition

$$H(\mathbf{m}|\mathbf{e}) = H(\mathbf{m})$$

represents the perfect secrecy requirement, where H is an entropy function. The legitimate data collector can access the stored data by retrieving information from a given number of nodes, while a cluster eavesdropper cannot obtain any information about the data.

In our cluster eavesdropper model, we assume that only the data used for node repair can be observed by the eavesdropper. In other words, the cluster eavesdropper has no access to data when the source stores them to nodes or when the data collector retrieves them from nodes. We made this assumption due to the following reason. In the case of node repair, each node in other clusters helps β_c symbols to regenerate a failed node. In the case of data collection, however, each node provides all of its contents (α symbols) to a data collector, which is far greater than β_c. (Similarly, in the case of data storing, each node receives α symbols from the source.) Thus, in the system designer‘s perspective, securing the channel for data storing/collection has a higher priority, compared to securing the channel for node repair.

A simple example on how the attacker eavesdrops data in a clustered DSS is shown in Fig. 1. There are $n = 9$ storage nodes which are dispersed in $L = 3$ clusters. A single cluster is compromised which is illustrated as a smooth square with dashed line. Suppose a node in the compromised cluster fails. Then, a node is regenerated by receiving β_I units of data from a node within the cluster and β_c units of data from each node in the other clusters. Notice that links from nodes in the uncompromised clusters must pass through the boundary of the compromised cluster, which is observed by eavesdropper. Therefore, $6\beta_c$ symbols are eavesdropped. Here, compromised and uncompromised links are illustrated as dashed and solid links in the figure, respectively.

III. Secrecy Capacity Against Cluster Eavesdropper

In this section, we summarize our main results, the secrecy capacity of a clustered DSS against a cluster eavesdropper. Here, we assume that the number of helper nodes is set to the maximum value ($d_I = n_I - 1, d_c = n - n_I$) since maximizing the number of helper nodes maximizes secrecy capacity as stated in the following proposition. Under this assumption, parameters n, k, L, α, β_I and β_c completely specify a clustered DSS. We denote such system as $(n, k, L, \alpha, \beta_I, \beta_c)$ clustered DSS.

Proposition 1. *Let γ, γ_c be specified for a clustered DSS. Setting d_I and d_c to the maximum possible values maximizes secrecy capacity in the bandwidth-limited regime.*

Proof. See Appendix D-A. □

A. Upper Bound on Secrecy Capacity

Consider a $(n, k, L, \alpha, \beta_I, \beta_c)$ clustered DSS and a cluster eavesdropper who compromises L_c clusters. Under the functional repair setting in the regeneration process, we present a theorem which provides an upper bound on secrecy capacity against a cluster eavesdropper.

Theorem 1. *The secrecy capacity of a $(n, k, L, \alpha, \beta_I, \beta_c)$ clustered DSS against L_c cluster eavesdropping is upper-bounded by*

$$\mathcal{C}_s(\alpha, \beta_I, \beta_c) \le \mathcal{C}_s^U$$

where

$$\mathcal{C}_s^U = \begin{cases} L_c \sum_{i=1}^{n_I} \min\{\alpha, (n_I - i)\beta_I\} + \mathcal{C}_r, & n_I L_c < k \\ q \sum_{i=1}^{n_I} \min\{\alpha, (n_I - i)\beta_I\} \\ + \sum_{i=1}^{r} \min\{\alpha, (n_I - i)\beta_I\}, & \text{otherwise}. \end{cases}$$

Here, q and r are defined in Table I and $\mathcal{C}_r$ is capacity [6] of a reduced $(n', k', L', \alpha', \beta_I, \beta_c)$ clustered DSS with parameters

$$n' = n - n_I L_c,$$
$$k' = k - n_I L_c,$$
$$L' = L - L_c,$$
$$\alpha' = \min\{\alpha, (n_I - 1)\beta_I + (n' - n_I)\beta_c\}.$$

Proof. See Appendix A-A. □

Remark 2. *Note that there are two cases where eavesdropping does not occur. Under these settings, secrecy capacity is equal to capacity obtained in [6] which considers the clustered model without security constraints.*

(a) *When $L_c = 0$, the upper bound $\mathcal{C}_s^U$ obtained in Theorem 1 is reduced to capacity of a $(n, k, L, \alpha, \beta_I, \beta_c)$ clustered DSS.*

(b) *When $\beta_c = 0$, the upper bound $\mathcal{C}_s^U$ reduces to capacity of a $(n, k, L, \alpha, \beta_I, \beta_c = 0)$ clustered DSS. This can be obtained as follows. Note that capacity expression of a $(n, k, L, \alpha, \beta_I, \beta_c)$ clustered DSS in Lemma 1 is reduced to*

$$\mathcal{C} = \sum_{i=1}^{n_I} g_i \min\{\alpha, (n_I - i)\beta_I\},$$

when $\beta_c = 0$. Meanwhile, the upper bound obtained in Theorem 1 is reduced to

$$\begin{aligned}\mathcal{C}_s^U &= L_c \sum_{i=1}^{n_I} \min\{\alpha, (n_I - i)\beta_I\} \\ &\quad + \sum_{i=1}^{n_I} (g_i - L_c) \min\{\alpha', (n_I - i)\beta_I\} \\ &= L_c \sum_{i=1}^{n_I} \min\{\alpha, (n_I - i)\beta_I\} \\ &\quad + \sum_{i=1}^{n_I} (g_i - L_c) \min\{\alpha, (n_I - i)\beta_I\} \\ &= \sum_{i=1}^{n_I} g_i \min\{\alpha, (n_I - i)\beta_I\},\end{aligned}$$

when $\beta_c = 0$, which is the capacity of a $(n, k, L, \alpha, \beta_I, \beta_c = 0)$ clustered DSS.

Remark 3. *The upper bound $\mathcal{C}_s^U$ is obtained based on an information flow graph; as such, the analysis inevitably bears some similarities to that on secrecy capacity of the non-clustered DSS model in [20]. However, there are some significant additional challenges in pursuing analysis for the clustered model. In the non-clustered model that assumes symmetric repair (i.e., $\beta_I = \beta_c$), identifying l compromised nodes for the worst-case scenario (when the intruder eavesdrops the maximum amount of information) suffices to find the least upper bound. On the other hand, in the clustered model with asymmetric repair (i.e., $\beta_I \neq \beta_c$), identifying the order of k failed nodes as well as L_c compromised clusters in the worst-case scenario is required to obtain the least upper bound. It turns out that an optimal solution is obtained by solving a non-trivial combinatorial optimization problem, which is not required for the non-clustered model in [20].*

B. Secrecy Capacity in the Bandwidth-Limited Regime

Suppose a clustered DSS satisfies $\alpha \geq \gamma$. From Remark 1, the capacity $\mathcal{C}$ stated in (1) is expressed as

$$\mathcal{C} = \sum_{i=1}^{n_I} \sum_{j=1}^{g_i} (\rho_i \beta_I + (n - \rho_i - j - \sum_{m=1}^{i-1} g_m)\beta_c).$$

This regime is said to be *bandwidth-limited* since only the repair bandwidths β_I and β_c limit the capacity $\mathcal{C}$. Notice that if node storage size α is strictly larger than γ, α can be reduced without affecting the capacity. Since reducing system resources is desired, we say that the bandwidth-limited regime is the regime satisfying the equality, $\alpha = \gamma$. Based on Theorem 1, the secrecy capacity in the bandwidth-limited regime is upper bounded as follows, with the bound being a function of intra- and cross-cluster repair bandwidths β_I and β_c. Applying the bandwidth-limited condition $\alpha = \gamma = (n_I - 1)\beta_I + (n - n_I)\beta_c$ to Theorem 1 directly completes the proof.

Corollary 1. *Consider a $(n, k, L, \alpha, \beta_I, \beta_c)$ clustered DSS. In the bandwidth-limited regime, the secrecy capacity against L_c cluster eavesdropping is bounded by*

$$\mathcal{C}_s^{BL}(\beta_I, \beta_c) \leq R$$

with

$$R = \begin{cases} L_c \beta_I \frac{n_I(n_I - 1)}{2} + \mathcal{C}_r, & n_I L_c < k \\ q\beta_I \frac{n_I(n_I-1)}{2} \\ + \sum_{i=1}^{r} (n_I - i)\beta_I, & \text{otherwise} \end{cases} \tag{2}$$

where $\mathcal{C}_r$ is the capacity of a reduced $(n', k', L', \alpha', \beta_I, \beta_c)$ clustered DSS with parameters

$$\begin{aligned} n' &= n - n_I L_c, \\ k' &= k - n_I L_c, \\ L' &= L - L_c, \\ \alpha' &= \min\{\alpha, (n_I - 1)\beta_I + (n' - n_I)\beta_c\}. \end{aligned}$$

The upper bound of secrecy capacity in the bandwidth-limited regime obtained in the above corollary turns out to be the exact value of secrecy capacity, as stated in the following theorem.

Theorem 2. *Consider a $(n, k, L, \alpha, \beta_I, \beta_c)$ clustered DSS in the bandwidth-limited regime. The secrecy capacity against L_c cluster eavesdropping is determined as*

$$\mathcal{C}_s^{BL}(\beta_I, \beta_c) = R$$

where R is given by (2).

Proof. As stated in Theorem 5 and 6, we present code constructions where R symbols can be stored with perfect secrecy under the desired setting with the presence of a cluster eavesdropper. Thus, a clustered DSS securely stores R symbols, i.e.,

$$\mathcal{C}_s^{BL}(\beta_I, \beta_c) \geq R.$$

Combining this with Corollary 1 completes the proof. □

IV. Feasible $(\alpha, \beta_I, \beta_c)$ to Store Data With Perfect Secrecy

In the previous section, we derived an upper bound of secrecy capacity under general system parameters and resources $(n, k, L, \alpha, \beta_I, \beta_c)$, and the exact value of secrecy capacity in the bandwidth-limited regime. Using these results, we analyze required resources $(\alpha, \beta_I, \beta_c)$ to store $\mathcal{M}_s$ symbols of data with perfect secrecy, given the system parameters (n, k, L). We introduce an important parameter

$$\epsilon \triangleq \beta_c / \beta_I. \tag{3}$$

which is a ratio of cross- to intra-cluster repair bandwidth. Since we consider practical scenarios where cross-cluster bandwidth is less than intra-cluster repair bandwidth, the range of ϵ is expressed as

$$0 \le \epsilon \le 1.$$

Note that $\epsilon = 0$ is the case that cross-cluster communication does not occur in the system, thus eavesdropping cannot happen. Thus, secrecy capacity is equal to capacity, and its analysis is identical to [6]. We only consider the non-trivial case for

$$0 < \epsilon \le 1.$$

As stated in Table I, total repair bandwidth of a clustered DSS with a maximal number of helper nodes is given as

$$\gamma = (n_I - 1)\beta_I + (n - n_I)\beta_c. \tag{4}$$

Combining (3) with (4), intra- and cross-cluster repair bandwidth can be expressed as functions of γ and ϵ:

$$\beta_I = \frac{\gamma}{(n_I - 1) + (n - n_I)\epsilon}, \tag{5}$$

$$\beta_c = \frac{\gamma\epsilon}{(n_I - 1) + (n - n_I)\epsilon}. \tag{6}$$

Using these expressions, the required resources $(\alpha, \beta_I, \beta_c)$ can be expressed more compactly by using resources (α, γ) for some given ratio ϵ. For example, secrecy capacity $\mathcal{C}_s(\alpha, \beta_I, \beta_c)$ can be expressed as $\mathcal{C}_s(\alpha, \gamma)$ for a given ϵ.

A. Parameters

In this subsection, we define parameters that used in the corollaries as follows. Here, $[k]$ stands for a set $\{1, \cdots, k\}$.

$$n' = n - n_I L_c,$$

$$k' = k - n_I L_c,$$

$$g_t = \begin{cases} q+1, & t \le r \\ q, & otherwise, \end{cases}$$

$$h_t = \min_{s \in [n_I]} \sum_{i=1}^{s} (g_i - L_c) \ge t, \ t \in [k'],$$

$$z_t = \begin{cases} (n' - n_I - t + h_t)\epsilon + (n_I - h_t), & t \in [k'] \\ \infty, & t = 0, \end{cases}$$

$$x_t = \begin{cases} \frac{(n_I - 1) + (n - n_I)\epsilon}{n_I - t}, & t \in [n_I - 1] \\ 0, & t = 0 \\ \infty, & t = n_I, \end{cases}$$

$$y_t = \begin{cases} \frac{(n_I - 1) + (n - n_I)\epsilon}{z_t}, & t \in [k'] \\ 0, & t = 0, \end{cases}$$

$$u_t = \begin{cases} t^{th} \text{ minimum of } \{x_i\}_{i=1}^{n_I} \cup \{y_j\}_{j=1}^{k'}, & t \in [n_I + k'] \\ 0, & t = 0 \\ \infty, & t = n_I + k' + 1, \end{cases}$$

$$p_t = \underset{i \in [n_I - 1] \cup \{0\}}{\arg\max} \ x_i \le \gamma, \gamma \in (u_t \alpha, u_{t+1}\alpha],$$

$$q_t = \underset{j \in [k'] \cup \{0\}}{\arg\max} \ y_j \le \gamma, \gamma \in (u_t \alpha, u_{t+1}\alpha],$$

$$r_t = \sum_{i=t+1}^{n_I} \frac{L_c}{x_i}, \quad s_t = \sum_{j=t+1}^{k'} \frac{1}{y_j},$$

$$a_t = L_c p_t + q_t, \quad b_t = r_{p_t} + s_{q_t},$$

$$\tau = \left\{ \tau \in [k' + n_I - 2] \cup \{0\} : u_\tau \alpha < \gamma \le u_{\tau+1}\alpha \right\},$$

$$c_t = \begin{cases} (q+1)t, & 0 \le t < r \\ qt + r, & r \le t \le n_I - 1, \end{cases}$$

$$d_t = \begin{cases} \sum_{i=t+1}^{n_I} \frac{q}{x_i} + \sum_{i=t+1}^{r} \frac{1}{x_i}, & 0 \le t < r \\ \sum_{i=t+1}^{n_I} \frac{q}{x_i}, & r \le t \le n_I - 1. \end{cases}$$

B. Main Results

We say that a point (α, γ) is feasible to store $\mathcal{M}_s$ symbols with perfect secrecy if and only if $\mathcal{C}_s(\alpha, \gamma) \ge \mathcal{M}_s$. Following Corollaries 2, 3 reveal infeasible (α, γ) points to store $\mathcal{M}_s$ symbols with perfect secrecy. Corollary 2 covers the case for eavesdropping capability satisfying $L_c < k/n_I$ while Corollary 3 handles $L_c \ge k/n_I$ case.

Corollary 2. *Consider a clustered DSS for storing $\mathcal{M}_s$ symbols of data with perfect secrecy against $L_c < k/n_I$ cluster eavesdropping. For any $\gamma < \gamma^*(\alpha)$, it is impossible to store the $\mathcal{M}_s$ symbols with perfect secrecy. The threshold value $\gamma^*(\alpha)$ can be obtained as*

1) if $\frac{1}{n-k} \le \epsilon \le 1$,

$$\gamma^*(\alpha) = \begin{cases} \infty, & \alpha \in (0, \frac{\mathcal{M}_s}{k - L_c}) \\ \frac{\mathcal{M}_s - a_t \alpha}{b_t}, & \alpha \in [\frac{\mathcal{M}_s}{a_{t+1} + b_{t+1}u_{t+1}}, \frac{\mathcal{M}_s}{a_t + b_t u_t}) \\ & \quad (t = k' + n_I - 2, k' + n_I - 3, \cdots, 1) \\ \frac{\mathcal{M}_s}{b_0}, & \alpha \in [\frac{\mathcal{M}_s}{a_1 + b_1 u_1}, \infty). \end{cases} \tag{7}$$

2) if $0 < \epsilon < \frac{1}{n-k}$, $k' < n_I$

$$\gamma^*(\alpha) = \begin{cases} \infty, & \alpha \in (0, \frac{\mathcal{M}_s}{k - L_c}) \\ \frac{\mathcal{M}_s - a_t \alpha}{b_t}, & \alpha \in [\frac{\mathcal{M}_s}{a_{t+1} + b_{t+1}u_{t+1}}, \frac{\mathcal{M}_s}{a_t + b_t u_t}) \\ & \quad (t = k' + n_I - 2, k' + n_I - 3, \cdots, 1) \\ \frac{\mathcal{M}_s}{b_0}, & \alpha \in [\frac{\mathcal{M}_s}{a_1 + b_1 u_1}, \infty). \end{cases} \tag{8}$$

3) if $0 < \epsilon < \frac{1}{n-k}$, $k' \ge n_I$

$$\gamma^*(\alpha) = \begin{cases} \infty, & \alpha \in (0, \frac{\mathcal{M}_s}{a_\tau + n_I - 1 + (n - n_I)\epsilon}) \\ \frac{\mathcal{M}_s - a_\tau \alpha}{b_\tau}, & \alpha \in [\frac{\mathcal{M}_s}{a_\tau + n_I - 1 + (n - n_I)\epsilon} \frac{\mathcal{M}_s}{a_\tau + b_\tau u_\tau}) \\ \frac{\mathcal{M}_s - a_t \alpha}{b_t}, & \alpha \in [\frac{\mathcal{M}_s}{a_{t+1} + b_{t+1}u_{t+1}}, \frac{\mathcal{M}_s}{a_t + b_t u_t}) \\ & \quad (t = \tau - 1, \tau - 2, \cdots, 1) \\ \frac{\mathcal{M}_s}{b_0}, & \alpha \in [\frac{\mathcal{M}_s}{a_1 + b_1 u_1}, \infty). \end{cases} \tag{9}$$

Proof. See Appendix B-A. □

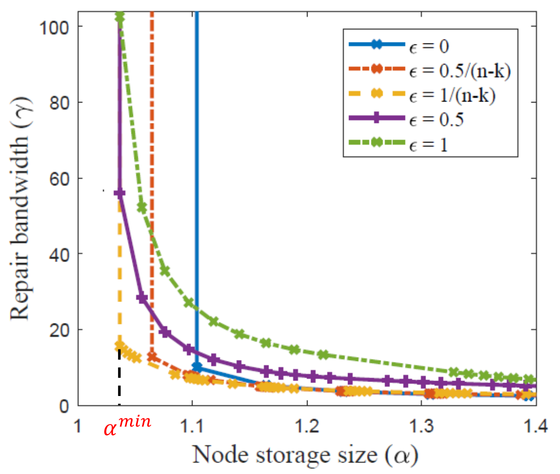


Fig. 2. The set of $(\alpha, \gamma^*(\alpha))$ points to store $\mathcal{M}_s = 85$ symbols with perfect secrecy, under the setting of $n = 100$, $k = 85$, $L = 10$ and $L_c = 3$.

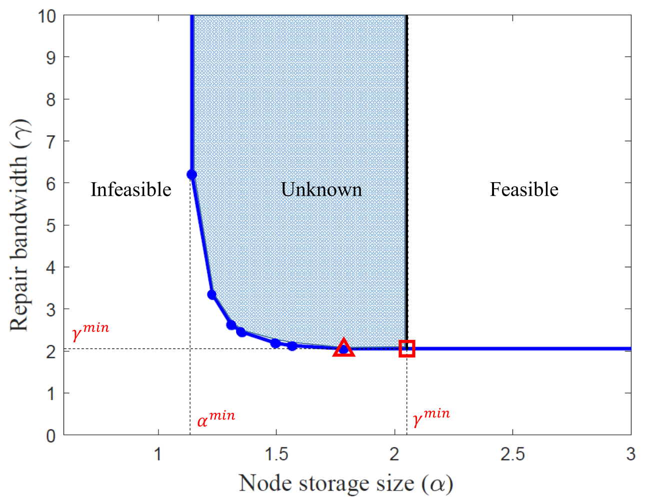


Fig. 3. Feasible and infeasible regions to store $\mathcal{M}_s = 8$ symbols with perfect secrecy, under the setting of $n = 15$, $k = 8$, $L = 3$, $\epsilon = 1/7$ and $L_c = 1$.

Corollary 3. *Consider a clustered DSS for storing $\mathcal{M}_s$ symbols of data with perfect secrecy against $L_c \geq k/n_I$ cluster eavesdropping. For any $\gamma < \gamma^*(\alpha)$, it is impossible to store the $\mathcal{M}_s$ symbols with perfect secrecy. The threshold value $\gamma^*(\alpha)$ can be obtained as*

$$\gamma^*(\alpha) = \begin{cases} \infty, & \alpha \in (0, \frac{\mathcal{M}_s}{k-q}) \\ \frac{\mathcal{M}_s - c_t\alpha}{d_t}, & \alpha \in [\frac{\mathcal{M}_s}{c_{t+1}+d_{t+1}x_{t+1}}, \frac{\mathcal{M}_s}{c_t+d_t x_t}) \\ & (t = n_I - 2, n_I - 3, \cdots, 1) \\ \frac{\mathcal{M}_s}{d_0}, & \alpha \in [\frac{\mathcal{M}_s}{c_1+d_1x_1}, \infty). \end{cases} \quad (10)$$

Proof. See Appendix B-B. □

Remark 4. *Every $(\alpha, \gamma^*(\alpha))$ pair satisfies $\mathcal{C}_s^U(\alpha, \gamma^*(\alpha)) = \mathcal{M}_s$.*

Consider a $(n, k, L, \alpha, \beta_I, \beta_c)$ clustered DSS to store $\mathcal{M}$ symbols without the secrecy concern. It is known that it can store $\mathcal{M}$ symbols with minimum storage overhead $\alpha = \mathcal{M}/k$ at the expense of repair bandwidth if and only if $\epsilon \geq \frac{1}{n-k}$ [6]. However, in a clustered DSS with perfect secrecy constraint, it is impossible to store $\mathcal{M}_s$ symbols with storage overhead $\alpha = \mathcal{M}_s/k$. The lower bound of storage size α to secure $\mathcal{M}_s$ symbols of data is stated as follows.

Theorem 3. *A clustered DSS cannot store $\mathcal{M}_s$ symbols of data with perfect secrecy when $\alpha < \frac{\mathcal{M}_s}{k-L_c}$.*

Proof. See Appendix A-B. □

We illustrate $(\alpha, \gamma^*(\alpha))$ pairs with various ϵ values in Fig. 2. The points below each curve represent an infeasible region to store $\mathcal{M}_s = 85$ with perfect secrecy. Here, the case with $\epsilon = 0$ corresponds to the scenario where a failed node is completely repaired without contacting nodes in other clusters. The $\epsilon = 1$ case becomes a *symmetric repair* where $\beta_I = \beta_c$. For every ϵ, it is shown that the (α, γ) point is infeasible for all $\alpha < \alpha^{min}$ where $\alpha^{min} = \frac{\mathcal{M}_s}{k-L_c} \approx 1.037$.

Now, we provide a feasible region to store $\mathcal{M}_s$ symbols in the following corollary.

Corollary 4. *Consider a clustered DSS for storing $\mathcal{M}_s$ symbols with perfect secrecy for a given $0 < \epsilon \leq 1$. It is possible to securely store data with resources (α, γ) where*

$$\alpha \geq \gamma^{min}, \quad \gamma \geq \gamma^{min}$$

with

$$\gamma^{min} = \begin{cases} \mathcal{M}_s/b_0, & L_c < k/n_I \\ \mathcal{M}_s/d_0, & L_c \geq k/n_I. \end{cases} \quad (11)$$

Proof. First, we show that $(\alpha, \gamma) = (\gamma^{min}, \gamma^{min})$ is a feasible point to store $\mathcal{M}_s$ symbols with perfect secrecy. In the bandwidth-limited regime where $\alpha = \gamma$, secrecy capacity is explicitly derived in Theorem 2, achieving the upper bound proposed in Theorem 1. Therefore,

$$\mathcal{C}_s(\alpha, \gamma) = \mathcal{C}_s(\gamma^{min}, \gamma^{min}) = \mathcal{C}_s^U(\gamma^{min}, \gamma^{min}) = \mathcal{M}_s.$$

Trivially, it is possible to store $\mathcal{M}_s$ symbols with perfect secrecy for an arbitrary (α, γ) point where $\alpha \geq \gamma^{min}$ and $\gamma \geq \gamma^{min}$, which utilizes more system resources than $(\alpha, \gamma) = (\gamma^{min}, \gamma^{min})$. □

An example of $(\alpha, \gamma^*(\alpha))$ points and corresponding feasible/infeasible regions are illustrated in Fig. 3. The set of $(\alpha, \gamma^*(\alpha))$ points and the infeasible region are obtained from Corollary 3, while the feasible region is obtained from Corollary 4. Whether the blue shaded region is feasible or not remains unknown.

Here we shed light on the difference between the *minimum bandwidth point* in this paper and the Minimum Bandwidth Regenerating (MBR) point in [6], [20]. Note that both points have the same definition, the point having the minimum storage given the minimum bandwidth constraint, while the functional regeneration is not guaranteed for the former point. It is well known that $\alpha = \gamma$ generally holds for the MBR points of [6], [20]. However, considering the set of $(\alpha, \gamma^*(\alpha))$ pairs obtained from the above corollaries, $\alpha = \gamma$ does not hold at the minimum bandwidth point. The reason for this difference is that the present paper targets a secure data storage

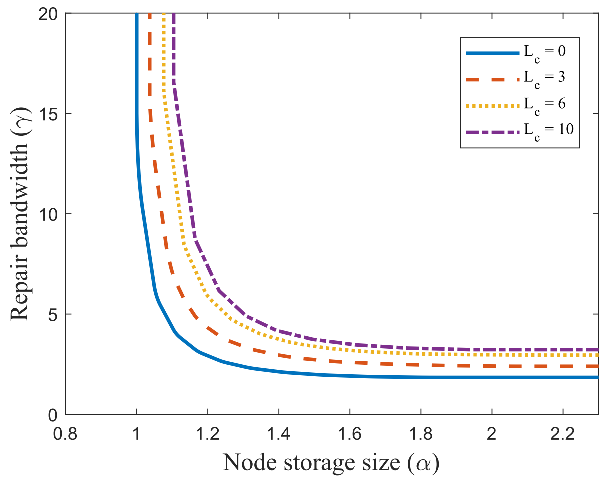


Fig. 4. The set of $(\alpha, \gamma^*(\alpha))$ points to store $\mathcal{M}_s = 85$ symbols with perfect secrecy, under the setting of $n = 100, k = 85, L = 10$ and $\epsilon = 1/15$.

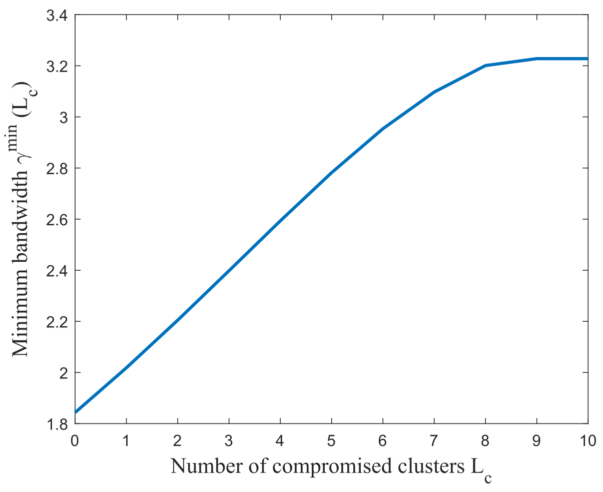


Fig. 5. Minimum required bandwidth $\gamma^{min}(L_c)$ to store $\mathcal{M}_s = 85$ symbols with perfect secrecy, under the setting of $n = 100, k = 85, L = 10$ and $\epsilon = 1/15$.

against eavesdroppers, not a fault-tolerant data storage as in [20] and [6]. The minimum bandwidth point is depicted as a triangle mark while the point with minimum bandwidth satisfying $\alpha = \gamma$ is shown as a square mark in Fig. 3.

In Fig. 4, we illustrate the set of $(\alpha, \gamma^*(\alpha))$ points for a various number of compromised clusters L_c. Note that $L_c = 0$ corresponds to the case where there is no eavesdropper. When $L_c = 0$, every communication link is secure, and thus the least amount of resources are required. As the power of eavesdropper becomes stronger, the more resources are required to secure data. The $L_c = 10$ case is a system where every cluster is compromised, i.e., every cross-cluster communication is monitored by a cluster eavesdropper. Fig. 5 illustrates γ^{min}, the minimum required bandwidth to achieve the secrecy capacity, defined in (11) as a function of L_c. It is shown that γ^{min} gradually increases with L_c and then gets saturated.

V. Implications in the Bandwidth-Limited Regime

In this section, we focus on the bandwidth-limited regime and answer key questions related to securely storing data in a clustered DSS against cluster eavesdropping. The bandwidth γ required for communication is generally the more scarce resource than storage node size α, given the abundance of cheap commodity disks. Since the required bandwidth for repairing failed nodes is minimized in this regime, we feel this is a practical important scenario, reflecting the real world.

A. Secrecy Capacity When $L_c < k/n_I$

As the number of compromised clusters becomes larger, the amount of securely storable data in a system is clearly decreased. Let us first see how rapidly secrecy capacity degrades as a function of L_c. For fair comparison, we fixed system parameters $(n, k, L, \alpha, \gamma)$ and use (ϵ, L_c) as variables. We define the secrecy capacity loss in the bandwidth-limited regime as

$$\Delta\mathcal{C}_s^{BL}(L_c) := \mathcal{C}_s^{BL}(L_c + 1) - \mathcal{C}_s^{BL}(L_c). \tag{12}$$

where $\mathcal{C}_s^{BL}(L_c)$ is the secrecy capacity against the cluster eavesdropper who observes L_c clusters. The secrecy capacity loss is a loss of secrecy capcaity due to an additional compromised cluster. $\Delta\mathcal{C}_s^{BL}(L_c)$ is derived in the following corollary.

Corollary 5. *The secrecy capacity loss $\Delta\mathcal{C}_s^{BL}(L_c)$ is expressed as*

$$\Delta\mathcal{C}_s^{BL}(L_c) = \begin{cases} -n_I^2(L - L_c - 1)\frac{\gamma\epsilon}{n_I - 1 + (n - n_I)\epsilon}, & L_c < k/n_I \\ 0, & L_c \geq k/n_I. \end{cases}$$

Proof. See Appendix B-C. □

Notice that $\Delta\mathcal{C}_s^{BL}(L_c)$ linearly decreases with L_c; thus secrecy capacity is a quadratic function of L_c. It directly provides the following proposition, the proof of which is available on Appendix D-B.

Proposition 2. *For $L_c < k/n_I$, $\mathcal{C}_s^{BL}(L_c)$ is expressed as a quadratic function of L_c,*

$$\mathcal{C}_s^{BL}(L_c) = a\big(L_c - (L - 1/2)\big)^2 + c - a(L - 1/2)^2$$

where

$$a = \frac{\gamma}{2}\frac{n_I^2\epsilon}{(n_I - 1) + (n - n_I)\epsilon},$$
$$c = k\gamma - \frac{\gamma}{2}\frac{(1 - \epsilon)(qn_I^2 + r^2 - k) + k(k - 1)\epsilon}{(n_I - 1) + (n - n_I)\epsilon}.$$

We illustrate the secrecy capacity for given parameters with various (ϵ, L_c) in Fig. 6. In the case with $\epsilon = 0$ where cross-cluster communication does not occur, the secrecy capacity remains constant regardless of L_c. On the other hand, the case with $\epsilon = 1$ is a symmetric repair which utilizes more cross-cluster bandwidth than $0 \leq \epsilon < 1$. Therefore, data revealed to the eavesdropper is the largest, and secrecy capacity drops very quickly with increasing L_c.

B. Secrecy Capacity When $L_c \geq k/n_I$

According to Theorem 2, the secrecy capacity is saturated and remains as a constant value for $L_c \geq k/n_I$. The following corollary explicitly gives the saturation capacity.

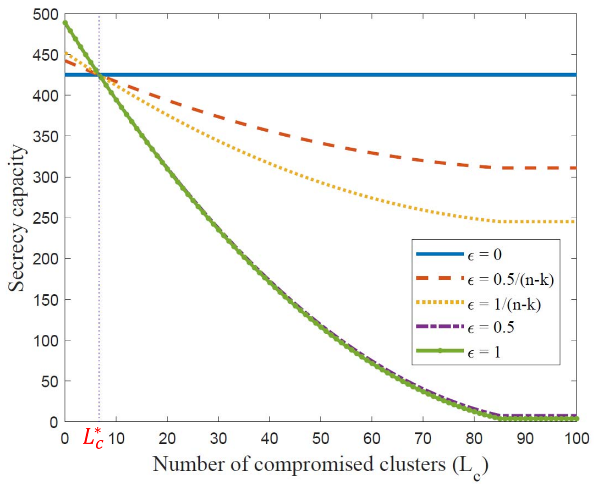


Fig. 6. Secrecy capacity as a function of L_c, under the setting of $n = 1000$, $k = 850$, $L = 100$ and $\alpha = \gamma = 1$.

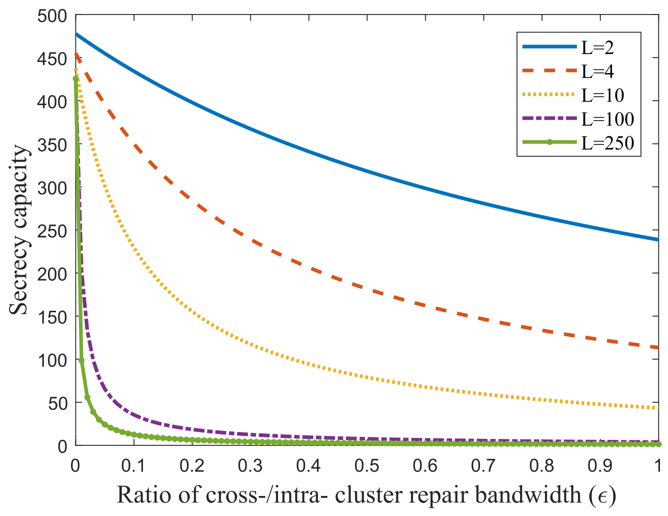


Fig. 7. Secrecy capacity as a function of ϵ with $L_c \geq k/n_I$, under the setting of $n = 1000$, $k = 850$ and $\alpha = \gamma = 1$.

Corollary 6. *Suppose enough clusters are compromised such that $L_c \geq k/n_I$. Then, secrecy capacity is independent of L_c and is given by*

$$\mathcal{C}_s^{BL} = \frac{\gamma}{2} \frac{(q+1)(n_I-1)n_I - (n_I-r-1)(n_I-r)}{(n_I-1)+(n-n_I)\epsilon}.$$

Proof. From Theorem 2, the secrecy capacity with $L_c \geq k/n_I$ is expressed as

$$\begin{aligned}\mathcal{C}_s^{BL} &= q\sum_{i=1}^{n_I}(n_I-i)\beta_I + \sum_{i=1}^{r}(n_I-i)\beta_I \\ &= (q+1)\sum_{i=1}^{n_I}(n_I-i)\beta_I + \sum_{i=r+1}^{n_I}(n_I-i)\beta_I \\ &= \frac{(q+1)(n_I-1)n_I-(n_I-r-1)(n_I-r)}{2}\beta_I. \quad (13)\end{aligned}$$

Combining (13) with (5) completes the proof. □

Fig. 7 illustrates the secrecy capacity as a function of ϵ with various L. The graph with $L = 100$ is consistent with the previous example in Fig. 6. Note that every cross-cluster communication can be observed by Eve. Consider a scenario of allocating storage nodes to multiple clusters to securely store data. A key question relates to the number of clusters to have in the system to ensure a certain security level. For every ϵ, minimizing the number of clusters is beneficial to secrecy capacity. In other words, the secrecy capacity degrades when we distribute data into more clusters, which is not surprising. However, distributing data to a small number of clusters poses a risk such as cluster failure scenarios, which may cause a permanent lost of stored data. This is the price we need to pay for increasing secrecy capacity.

In the case of $\epsilon = 0$, increasing the number of clusters L hardly affects the capacity of the entire system, which is also observed in [6]. In contrast, the secrecy capacity dramatically drops as a function of L with a given small positive ϵ. Thus, considering untrusted scenarios where every cross-cluster communication can be eavesdropped, setting zero cross-cluster repair bandwidth would be much helpful to secure data. Also, it is observed that the loss of secrecy capacity as a function of ϵ highly depends on the number of clusters. When distributing data into large number of clusters, a large ϵ would be a disaster in terms of secrecy. However, we should be able to assign cross-cluster bandwidth more freely when L is small.

C. Optimal ϵ to Maximize Secrecy Capacity

Suppose that there exists no eavesdropper ($L_c = 0$) in the system (wherein the secrecy capacity is equal to capacity). It is known that capacity is an increasing function of ϵ [6], also implied in Fig. 6. This is consistent with the previous work on asymmetric repair in [24] which shows that the symmetric repair maximizes capacity of the system. As L_c becomes larger, the secrecy capacity degrades with different speeds depending on ϵ, and it is seen that the curves intersect at one point, as seen in Fig. 6. Denote this point as L_c^*. As L_c exceeds this number, repair across clusters always degrade secrecy capacity; when L_c is below this critical number, cross-cluster repair always helps secrecy capacity. L_c^* can be explicitly derived as a function of system parameters as follows.

Theorem 4. *With various ϵ, $\mathcal{C}_s^{BL}(L_c)$ intersects on one point L_c^*. The intersection point is given by*

$$L_c^* = (L-1/2) - \sqrt{(L-1/2)^2 - \frac{\omega}{n_I^2(n_I-1)}} \quad (14)$$

where

$$\omega = qn_I(n_I-1)(n-qn_I-2r) + r(r-1)(n-n_I).$$

Proof. See Appendix A-C. □

The intersection point L_c^* is further approximated in the following corollary. We observed that the approximation was very accurate as L grows.

Corollary 7. *The intersection point (14) is approximated as*

$$L_c^* \approx L_{c,approx}^* = L(1-\sqrt{1-\mathcal{R}(1-\mathcal{R})})$$

where $\mathcal{R} = k/n$ is the code rate.

Proof. See Appendix B-D. □

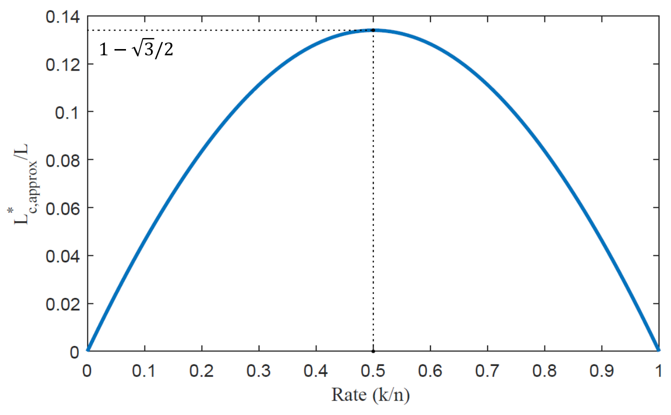


Fig. 8. $L^*_{c,approx}/L$ versus code rate k/n.

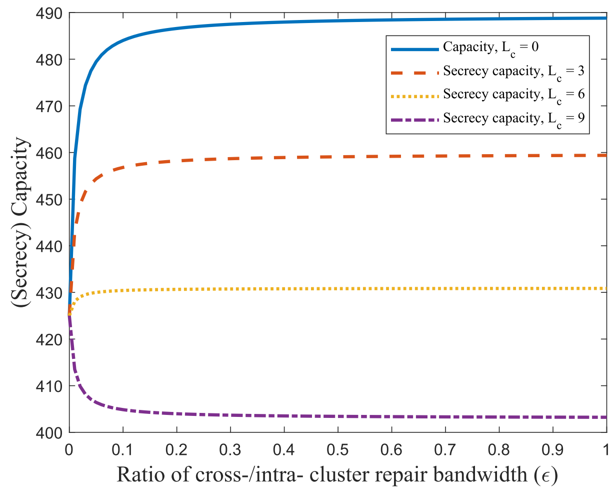


Fig. 9. Capacity and secrecy capacity as a function of ϵ, under the setting of $n = 1000, k = 850, L = 100$ and $\alpha = \gamma = 1$. In this case, $L_c^* = 6.629$ holds.

Fig. 8 shows the approximated ratio of L_c^* versus code rate k/n. For given values of L_c and L, suppose ϵ is a design parameter to maximize secrecy capacity. In this case, Figs. 7 and 8 provide useful guideline. From Fig. 8, we would know L_c^*/L for a given code rate. If L_c/L is smaller than this, then it is recommended to increase ϵ to 1 to maximize secrecy capacity. Otherwise, minimizing ϵ to 0 would maximize secrecy capacity. It is shown that as the rate is closer to 0.5, the system can tolerate a larger number of compromised clusters before cross-cluster should be disallowed. Notice that $L^*_{c,approx}$ is upper bounded as

$$L^*_{c,approx} = L(1-\sqrt{1-\mathcal{R}(1-\mathcal{R})}) \leq L(1-\sqrt{3}/2) \approx 0.134L.$$

If a cluster eavesdropper occupies more than 13.4% of clusters, reducing ϵ is always beneficial to maximize the secrecy capacity.

Moreover, we depict secrecy capacity as a function of ϵ in Fig. 9 where the system parameters are consistent with Fig. 6. If L_c is lower than the critical $L_c^* = 6.629$, the secrecy capacity is an increasing function of ϵ. Otherwise, the secrecy capacity is a decreasing function of ϵ.

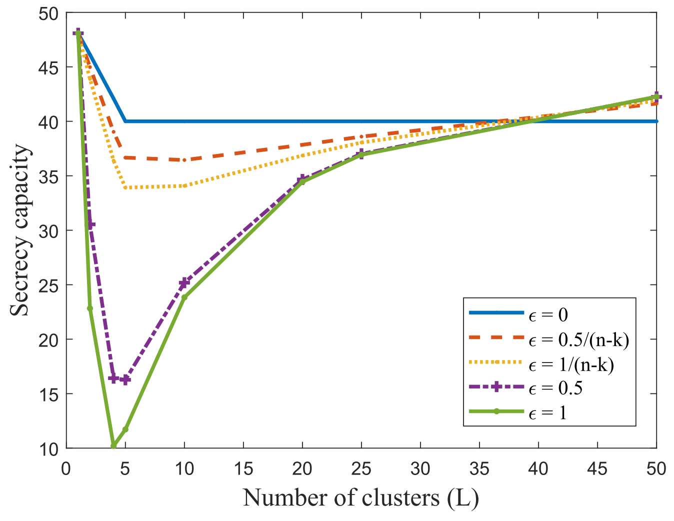


Fig. 10. Secrecy capacity as a function of L, under the setting of $n = 100$, $k = 80$, $\alpha = 1$, $\gamma = 1$ and $L_c = 3$.

D. Secrecy Capacity as a Function of L

Suppose L_c is set to a constant value. Now see what happens to secrecy capacity as L changes. Fig. 10 illustrates secrecy capacity as a function of L for different values of ϵ. Since we assume that all clusters have the identical number $n_I = n/L$ of nodes, we only plotted for the set of L values which divide n. As L becomes larger, the secrecy capacity tends to decrease up to a certain point, and then increases, for all positive values of ϵ. When L is small enough, storage nodes are distributed to a few number of clusters. Data transmission using intra-cluster links occupies a large portion of total communication. Thus, Eve can access a smaller amount of data, which increases secrecy capacity in small L. When L is large enough, an eavesdropper with fixed L_c observes only a fraction of total clusters. Accessibility of Eve to communication links is limited in large L, which again increases secrecy capacity. Here, again note that distributing data to nodes in a small number of clusters is susceptible to cluster failure scenarios which may cause permanent data loss. This would mean that when L_c is fixed, increasing the number of clusters is highly beneficial.

VI. Secure Code Constructions at the Bandwidth-Limited Regime

In this section, we provide explicit coding schemes to securely store data comprising of $\mathcal{C}_s^{BL}$ symbols in the bandwidth-limited regime, where each symbol belongs to a finite field $\mathbb{F}_s$ of size s. In the case of $\beta_c = 0$, note that a cluster eavesdropper cannot observe any data regardless of L_c, since cross-cluster communication does not occur in this setting. Thus, secrecy capacity is equal to the capacity. An explicit code construction achieving the capacity in the bandwidth-limited regime is obtained in [18], thus it trivially achieves the secrecy capacity when $\beta_c = 0$. Hereafter, we focus on nontrivial cases where $\beta_c \neq 0$. The following secure codes are applied to clustered DSS with arbitrary system parameters

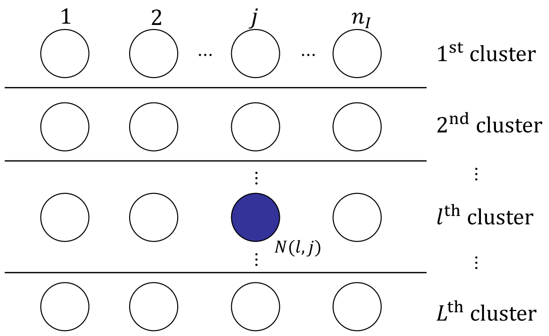


Fig. 11. Two dimensional representation of a clustered DSS.

$(n, k, L, \alpha, \beta_I, \beta_c)$ and eavesdropping capability L_c when the following conditions are guaranteed: 1) the number of helper nodes is maximized, i.e., $d_I = n_I - 1$ and $d_c = n - n_I$, 2) n is divisible by L, 3) $\alpha = \gamma$ (bandwidth-limited condition), 4) $\chi := \beta_c/\beta_I$ is an integer, and 5) field size s is greater than or equal to θ, where θ is defined in (15). From now on, we set $\beta_c = 1$ and $\beta_I = \chi$ without a loss of generality. The coding for arbitrary $\beta_c > 1$ is completed by repeatedly applying the code with $\beta_c = 1$ in a parallel manner.

Before providing the code construction rules, we introduce a two dimensional representation of nodes in a clustered DSS (Fig. 11). Consider a (n, k, L) clustered DSS where each cluster has $n_I = n/L$ storage nodes. For $l \in [L]$ and $j \in [n_I]$, the j^{th} node in l^{th} cluster is represented as node $N(l, j)$. The set of active n storage nodes is represented as

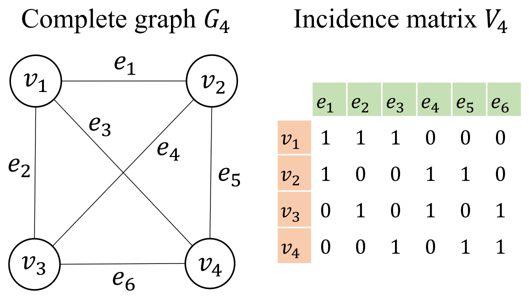
$$\mathcal{N} = \{N(l, j) : l \in [L], j \in [n_I]\}.$$

A. Code Construction on $n_I L_c < k$

For a $(n, k, L, \alpha, \beta_I = \chi, \beta_c = 1)$ clustered DSS, we present a secure code construction to store data in the bandwidth-limited regime, when the capability of a cluster eavesdropper is restricted to $L_c < k/n_I$. Note that storage size α of each node in the bandwidth-limited regime is determined as

$$\alpha = \gamma = d_I \beta_I + d_c \beta_c = (n_I - 1)\chi + (n - n_I).$$

We first introduce the main concept of the wiretap channel problem [27], a solution to which will be used in our code construction. Consider the following situation. Alice wants to convey R message symbols to Bob where each symbol is independent and uniformly chosen in the field $\mathbb{F}_s$. Alice uses an erasure channel with transmission length $\theta > R$, and knows that Bob receives arbitrary $\mathcal{M}$ among θ encoded symbols and Eve wiretaps $\mathcal{K} \leq \mathcal{M}$ out of θ symbols. This channel is denoted as $h_{wt}(\theta, \mathcal{M}, \mathcal{K})$. The role of Alice is to encode R message symbols into θ symbols in a way that the channel can convey all the message symbols to Bob, while Eve cannot retrieve any information on the message symbols. We summarize the result of [19] for securing R symbols with perfect secrecy in the following lemma. Note that a random vector of size $\mathcal{K}$ is used in the encoding process to guarantee perfect secrecy.



	e_1	e_2	e_3	e_4	e_5	e_6
v_1	1	1	1	0	0	0
v_2	1	0	0	1	1	0
v_3	0	1	0	1	0	1
v_4	0	0	1	0	1	1

Fig. 12. Complete graph G_4 and corresponding incidence matrix V_4.

Lemma 2 (Main result of [19]). *Consider a wiretap channel $h_{wt}(\theta, \mathcal{M}, \mathcal{K})$. The secrecy capacity of this channel, i.e., the maximum number of message symbols that can be conveyed with perfect secrecy, is $R = \mathcal{M} - \mathcal{K}$. Moreover, for a given message vector $\mathbf{s} = [s_1, \cdots, s_R] \in \mathbb{F}_s^R$, the secrecy capacity is achievable by conveying a vector of encoded symbols $\mathbf{c} = [c_1, \cdots, c_\theta] \in \mathbb{F}_s^\theta$ generated by using a nested MDS code:*

$$\mathbf{c} = \begin{bmatrix}\mathbf{k} & \mathbf{s}\end{bmatrix}\begin{bmatrix}G_K \\ G_S\end{bmatrix} = \mathbf{k}G_K + \mathbf{s}G_S,$$

where $\mathbf{k} = [k_1, \cdots, k_{\mathcal{K}}] \in \mathbb{F}_s^{\mathcal{K}}$ is a random vector evenly distributed over $\mathbb{F}_s^{\mathcal{K}}$, while G_K and $G = \begin{bmatrix}G_K \\ G_S\end{bmatrix}$ are generator matrices of MDS codes with size $\mathcal{K}\times\theta$ and $\mathcal{M}\times\theta$, respectively.

In the construction of the suggested coding scheme, we use several parameters

$$\theta \triangleq (\chi - 1)\binom{n_I}{2}L + \binom{n}{2}, \tag{15}$$

$$\mathcal{M} \triangleq k\alpha - \frac{1}{2}(\chi - 1)(qn_I^2 + r^2 - k) - \frac{k(k-1)}{2}, \tag{16}$$

$$\mathcal{K} \triangleq n_I^2 L_c(L - \frac{L_c + 1}{2}), \tag{17}$$

and a matrix V_t defined as follows. Consider a fully connected graph G_t with t vertices. Then, V_t, the incidence matrix of G_t is a $t \times \binom{t}{2}$ matrix given by:

$$V_t(j, i) = \begin{cases} 1, & \text{if } i^{th} \text{ edge is connected to } j^{th} \text{ node} \\ 0, & \text{otherwise.} \end{cases} \tag{18}$$

An example of incidence matrix V_4 is given in Fig. 12.

Using the parameters defined above, the explicit code construction rule for securely storing $\mathcal{M} - \mathcal{K}$ message symbols is provided in Algorithm 1. According to Theorem 5, the code in Algorithm 1 is shown to achieve the secrecy capacity of $(n, k, L, \alpha, \chi, 1)$ clustered DSS under the condition of $L_c < k/n_I$. Here, $\mathcal{K}$ random symbols are used when generating encoded symbols.

Before stating Theorem 5, we provide some physical insights into two important parameters, $\mathcal{M}$ and $\mathcal{K}$ used in Algorithm 1. First, the properties of the symbol distribution rule in Step 2 of Algorithm 1 is described in Lemma 3. Second, the physical meaning of $\mathcal{K}$ is stated in Lemma 4.

Algorithm 1 Secure Code Construction on $n_I L_c < k$

```
Input: System parameters $n, k, L, L_c, \chi$,
       Message vector $\mathbf{s} \triangleq [s_1, \cdots, s_{\mathcal{M}-\mathcal{K}}] \in \mathbb{F}_s^{1\times(\mathcal{M}-\mathcal{K})}$
Output: Symbols stored on each node in $\mathcal{N}$
  Step 1. Generate encoded symbols $\{c_1, \cdots, c_\theta\}$
    Applying the code in Lemma 2 to secure $\mathbf{s}$ in $h_{wt}(\theta, \mathcal{M}, \mathcal{K})$, generate an encoded vector $\mathbf{c} \triangleq [c_1, \cdots, c_\theta] = \begin{bmatrix}\mathbf{k} & \mathbf{s}\end{bmatrix}\begin{bmatrix}G_K \\ G_S\end{bmatrix}$.
  Step 2. Distribute encoded symbols $\{c_1, \cdots, c_\theta\}$ to nodes under the following rules (Construction 2 in [18]):
    · Node $N(l, j)$ stores a symbol $c_{i_1}$ if and only if $V_n(n_I(l-1)+j, i_1) = 1$.
    · For $t \in [\chi - 1]$, node $N(l, j)$ stores $c_{\binom{n}{2}+(\chi l-\chi-l+t)\binom{n_I}{2}+i_2}$ if and only if $V_{n_I}(j, i_2) = 1$.
```

Lemma 3 (Theorem 2 and Lemma 2 in [18])**.** *Consider a $(n, k, L, \alpha, \chi, 1)$ clustered DSS in the bandwidth-limited regime. The symbol distribution rule stated in Step 2 of Algorithm 1 satisfies the following properties:*

(a) Each encoded symbol c_i is duplicated and stored in two distinct nodes.
(b) Any choice of two storage nodes in different clusters share one encoded symbol.
(c) Any choice of two storage nodes in the same cluster share χ encoded symbols.
(d) Any failed node is exactly regenerated by retrieving a given number of symbols (χ or 1) from helper nodes.
(e) When a data collector receives data from k nodes, at least $\mathcal{M}$ encoded symbols stored in nodes are accessible.

Lemma 4. *Consider a $(n, k, L, \alpha, \chi, 1)$ clustered DSS and a cluster eavesdropper who compromises L_c clusters. When using the distribution rule stated in Step 2 of Algorithm 1, the maximum*[2] *number of encoded symbols observed by a cluster eavesdropper is*

$$\mathcal{K} = n_I^2 L_c(L - \frac{L_c+1}{2}).$$

Proof. Recall that a coded symbol stored in both the compromised cluster and another cluster can be observed by a cluster eavesdropper. Note that any two storage nodes in different clusters share a single encoded symbol according to property (b) in Lemma 3. Thus, the number of coded symbols stored in a compromised cluster and another compromised cluster is obtained as $\binom{L_c}{2}n_I^2$. Similarly, the number of coded symbols stored in a compromised cluster and a non-compromised cluster is obtained as $L_c(L-L_c)n_I^2$. Combining these numbers results in $\mathcal{K} = \binom{L_c}{2}n_I^2 + L_c(L-L_c)n_I^2 = n_I^2 L_c(L - \frac{L_c+1}{2})$. □

According to property (e) in Lemma 3, a data collector observes at least $\mathcal{M}$ among θ encoded symbols by contacting k nodes. Similarly, a cluster eavesdropper observes at most $\mathcal{K}$ out of θ encoded symbols according to Lemma 4. Thus, from the statement of Lemma 2, maximally $\mathcal{M} - \mathcal{K}$ message symbols can be stored in the system with perfect secrecy by applying a nested MDS code. By showing that $\mathcal{M} - \mathcal{K}$ is equal to R in (2), Theorem 5 below proves that the code in Algorithm 1 achieves the upper bound on secrecy capacity.

[2]Note that depending on the set of failed nodes, some of the $\mathcal{K}$ symbols may not be eavesdropped. For example, when only $N(2,1)$ node is failed in Fig. 13, a cluster eavesdropper can observe only two symbols, c_2 and c_6. On the other hand, when $N(1,1)$ and $N(1,2)$ nodes are successively failed, a cluster eavesdropper observes $\mathcal{K} = 8$ symbols.

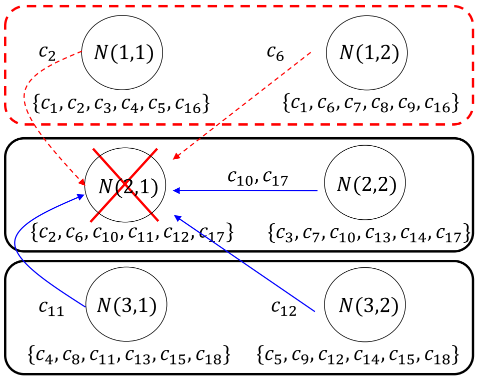

c_2
$N(1,1)$
c_6
$N(1,2)$
$\{c_1, c_2, c_3, c_4, c_5, c_{16}\}$
$\{c_1, c_6, c_7, c_8, c_9, c_{16}\}$
$N(2,1)$
c_{10}, c_{17}
$N(2,2)$
$\{c_2, c_6, c_{10}, c_{11}, c_{12}, c_{17}\}$
$\{c_3, c_7, c_{10}, c_{13}, c_{14}, c_{17}\}$
c_{11}
$N(3,1)$
c_{12}
$N(3,2)$
$\{c_4, c_8, c_{11}, c_{13}, c_{15}, c_{18}\}$
$\{c_5, c_9, c_{12}, c_{14}, c_{15}, c_{18}\}$

Fig. 13. Secure code for a $(6, 4, 3, 6, 2, 1)$ clustered DSS against a cluster eavesdropper when $L_c = 1$.

Theorem 5. *Consider a $(n, k, L, \alpha, \chi, 1)$ clustered DSS and a cluster eavesdropper who compromise L_c clusters under condition $n_I L_c < k$. The code constructed by Algorithm 1 achieves the secrecy capacity.*

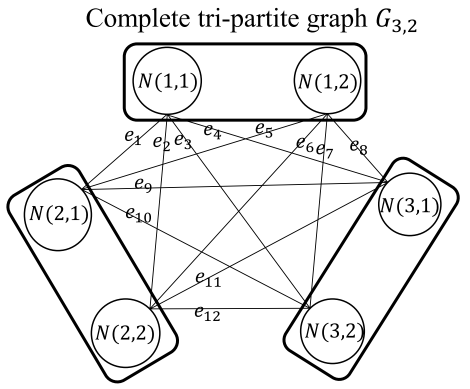
Proof. See Appendix A-D. □

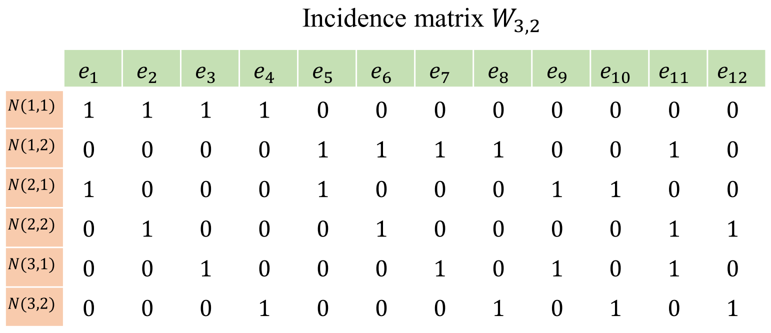
In addition, we summarize properties of the suggested capacity-achieving code in the following remark.

Remark 5. *The code constructed by Algorithm 1 satisfies the following.*

(a) Exact regeneration: Any failed node is exactly regenerated by retrieving a given number of symbols (χ or 1) from helper nodes.
(b) Data reconstruction: A data collector can reconstruct the message vector **s** *by contacting any k among n nodes.*
(c) Perfect secrecy: A cluster eavesdropper cannot obtain any information on the message **s**.

We provide an example of the suggested code in a $(6, 4, 3, 6, 2, 1)$ clustered DSS against a cluster eavesdropper with eavesdropping capability $L_c = 1$. Note that this regime is bandwidth-limited since $\alpha = \gamma = \beta_I + 4\beta_c$ holds. From (15), (16) and (17), we obtain design parameters $\theta = 18, \mathcal{M} = 16$ and $\mathcal{K} = 8$, respectively. In order to store a message vector $\mathbf{s} = [s_1, \cdots, s_8]$ of size $\mathcal{M} - \mathcal{K}$ with perfect secrecy, the encoded vector $\mathbf{c} = [c_1, \cdots, c_{18}] = \begin{bmatrix}\mathbf{k} & \mathbf{s}\end{bmatrix}\begin{bmatrix}G_K \\ G_S\end{bmatrix}$ of size θ is generated in Step 1 of Algorithm 1 by applying a code in Lemma 2. Then, the coded symbols are distributed to nodes under the rule described in Step 2. Fig. 13 shows that how the coded symbols are distributed to nodes. Here we illustrate a special case where the cluster containing nodes υ_1 and υ_2 is compromised by a cluster eavesdropper. Now, we show that this code satisfies three properties stated in Remark 5.




Incidence matrix $W_{3,2}$

	e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8	e_9	e_{10}	e_{11}	e_{12}
$N(1,1)$	1	1	1	1	0	0	0	0	0	0	0	0
$N(1,2)$	0	0	0	0	1	1	1	1	0	0	1	0
$N(2,1)$	1	0	0	0	1	0	0	0	1	1	0	0
$N(2,2)$	0	1	0	0	0	1	0	0	0	0	1	1
$N(3,1)$	0	0	1	0	0	0	1	0	1	0	1	0
$N(3,2)$	0	0	0	1	0	0	0	1	0	1	0	1

Fig. 14. Complete tripartite graph $G_{3,2}$ and corresponding incidence matrix $W_{3,2}$.

(a) *Exact regeneration*: Suppose a node $N(2,1)$ fails. Then, the node can be exactly repaired by receiving one symbol from nodes $N(1,1)$, $N(1,2)$, $N(3,1)$ and $N(3,2)$ while $\chi = 2$ symbols from $N(2,2)$.

(b) *Data reconstruction*: A data collector observes at least $\mathcal{M} = 16$ coded symbols by contacting $k = 4$ nodes. From property (e) in Lemma 3, the message $\mathbf{s}$ can be reconstructed from $\mathcal{M}$ symbols.

(c) *Perfect secrecy*: Notice that $\mathcal{K} = 8$ encoded symbols $\{c_2, c_3, c_4, c_5, c_6, c_7.c_8, c_9\}$ are at risk of being observed by a cluster eavesdropper. This is because those encoded symbols are stored in both the compromised cluster and another cluster, so they have chance to be transmitted from/to the compromised cluster when regenerating failed nodes. From Lemma 2, an eavesdropper cannot obtain any message from $\mathcal{K}$ encoded symbols.

```
Algorithm 2 Secure Code Construction on $n_I L_c \geq k$
Input: System parameters $n, k, L, \chi$
        Message vector $\mathbf{s} = [s_1, \cdots, s_R] \in \mathbb{F}_s^{1\times R}$ where $R$ is
        obtained from system parameters as in (2)
Output: Symbols stored on each node in $\mathcal{N}$
  Step 1. Generate encoded symbols $\{c_1, \cdots, c_\eta\}$ and dummy
  symbols $\{c_{\eta+1}, \cdots, c_\theta\}$
    Generate an encoded vector $\mathbf{c} \triangleq [c_1, \cdots, c_\eta] = \mathbf{s}G$ where
  $G$ is a generator matrix of a $(\eta, R)$ MDS code.
  Step 2. Distribute encoded symbols $\{c_1, \cdots, c_\theta\}$ to nodes
  under the following rules:
  · For $t \in [\chi]$, node $N(l, j)$ stores
  $c_{(l-1)\chi\binom{n_I}{2}+(t-1)\binom{n_I}{2}+i_1}$ if and only if $V_{n_I}(j, i_1) = 1$.
  · Node $N(l, j)$ stores symbol $c_{\eta+i_2}$ if and only if
  $W_{L,n_I}(j, i_2) = 1$.
```

B. Code Construction on $n_I L_c \geq k$

Consider a $(n, k, L, \alpha, \beta_I = \chi, \beta_c = 1)$ clustered DSS in the bandwidth-limited regime. We present a code construction to securely store data when sufficiently large clusters are compromised by a cluster eavesdropper, i.e., $L_c \geq k/n_I$. First we define

$$\eta \triangleq L\chi\binom{n_I}{2} \tag{19}$$

and a matrix W_{L,n_I} used in the code construction. Consider a complete L-partite graph G_{L,n_I} where each partition has n_I vertices. An incidence matrix W_{L,n_I} of G_{L,n_I} is a $n_I L \times n_I^2 L(L-1)/2$ matrix given by

$$W_{L,n_I}(j,i) = \begin{cases} 1, & \text{if } i^{th} \text{ edge is connected to } j^{th} \text{ node} \\ 0, & \text{otherwise.} \end{cases} \tag{20}$$

An example of incidence matrix W_{L,n_I} when $L = 3$ and $n_I = 2$ is provided in Fig. 14.

Using these definitions, we provide an explicit code construction in Algorithm 2, which store R message symbols with perfect secrecy.

Theorem 6 below shows that the code in Algorithm 2 achieves the secrecy capacity of the system with $L_c \geq k/n_I$, which can be proved as follows. First, a cluster eavesdropper cannot access any of η encoded symbols $\{c_1, \cdots, c_\eta\}$, which is proved in Lemma 5. Next, a data collector can access at least R out of η encoded symbols $\{c_1, \cdots, c_\eta\}$, which is proved in Lemma 6. Note that a data collector can obtain message vector $\mathbf{s}$ of size R by decoding (η, R) MDS codes defined in Step 1 of Algorithm 2. Combining these results , we conclude that a message vector $\mathbf{s}$ of size R can be stored in the system with perfect secrecy, which completes the proof of Theorem 6. The formal statements of Lemmas 5, 6 and Theorem 6 are provided as follows.

Lemma 5. *In Algorithm 2, each encoded symbol in $\{c_1, \cdots, c_\eta\}$ is stored on two distinct nodes in a single cluster. Therefore, such symbols are stored with perfect secrecy against a cluster eavesdropper since they cannot be eavesdropped during any repair events.*

Proof. When distributing an encoded symbol c_t such that $t \leq \eta$, we use the first rule in Step 2. From the symbol allocation rule, it is directly obtained that symbol c_t is stored on nodes in l^{th} cluster where $l = \lfloor \frac{t}{\chi\binom{n_I}{2}} \rfloor$. □

Lemma 6. *Consider θ symbols $\{c_1, \cdots, c_\theta\}$ are distributed to n nodes following the rule specified at step 2 in Algorithm 2. Then, a data collector observes at least R symbols in $\{c_1, \cdots, c_\eta\}$ by contacting k out of n nodes.*

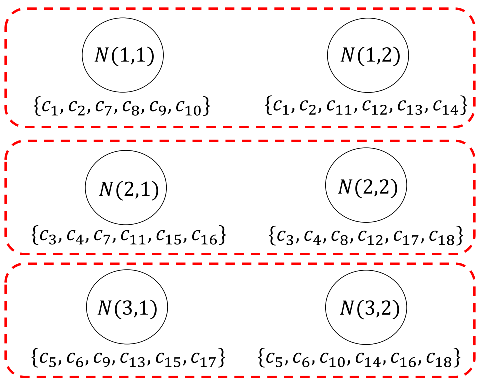

$N(1,1)$
$\{c_1, c_2, c_7, c_8, c_9, c_{10}\}$
$N(1,2)$
$\{c_1, c_2, c_{11}, c_{12}, c_{13}, c_{14}\}$
$N(2,1)$
$\{c_3, c_4, c_7, c_{11}, c_{15}, c_{16}\}$
$N(2,2)$
$\{c_3, c_4, c_8, c_{12}, c_{17}, c_{18}\}$
$N(3,1)$
$\{c_5, c_6, c_9, c_{13}, c_{15}, c_{17}\}$
$N(3,2)$
$\{c_5, c_6, c_{10}, c_{14}, c_{16}, c_{18}\}$

Fig. 15. Symbol allocation rule of a $(6, 4, 3, 6, 2, 1)$ clustered DSS against a cluster eavesdropper with $L_c = 3$.

Proof. See Appendix C-A. □

Theorem 6. *Consider a $(n, k, L, \alpha, \chi, 1)$ clustered DSS in the bandwidth-limited regime and a cluster eavesdropper who compromise L_c clusters under condition $n_I L_c \geq k$. The code construction described in Algorithm 2 achieves the secrecy capacity.*

As an example, Fig. 15 represents the distribution rule of Algorithm 2 to store $\theta = 18$ symbols $\{c_1, \cdots, c_{18}\}$ in a $(6, 4, 3, 6, 2, 1)$ clustered DSS. You may easily find that the first $\eta = 6$ coded symbols $\{c_1, \cdots, c_6\}$ are stored in a single cluster, thus they cannot be observed by a cluster eavesdropper. Finally, we summarize the properties of the suggested code in Remark 6.

Remark 6. *The code constructed by Algorithm 2 satisfies the following.*

(a) Exact regeneration: Any failed node is exactly regenerated by retrieving a given number of symbols (χ or 1) from helper nodes.
(b) Data reconstruction: A data collector can reconstruct the message vector **s** *by contacting any k among n nodes.*
(c) Perfect secrecy: A cluster eavesdropper cannot obtain any information on the message **s**.

VII. Conclusion

We have presented a cluster eavesdropper model for practical clustered DSSs. Our model covers scenarios where data is distributed to nodes in multiple clusters and a cluster eavesdropper observes links from cluster to cluster. Based on an analysis of information flow graph, we derived the maximum amount of data stored in the system with perfect secrecy. Using the results, we obtained feasible points of resources - node storage size and repair bandwidth - to securely store data with given size. We provide some key insights into how to best allocate resources in order to secure data. Furthermore, an explicit coding scheme to achieve the theoretical upper bound of secrecy capacity is suggested for the practically important bandwidth-limited regime. In realistic scenarios where data fragments are dispersed in multiple clusters, security of the system can be evaluated using our results. The present paper only deals maximal helper nodes in the cluster system. Due to potential latency issues and multiple node failures of storage nodes, expanding our results to some fixed number of helper nodes would be meaningful future work.

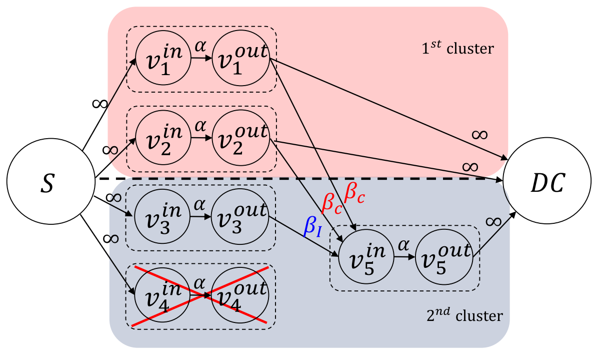


Fig. 16. An information flow graph of a $(4, 3, 2, \alpha, \beta_I, \beta_c)$ clustered DSS.

Appendix A
Proof of Theorems

A. Proof of Theorem 1

Before proving the theorem, we introduce *information flow graph*, an useful mathematic tool used in [20] to analyze dynamic node failure events and repair processes. Information flow graph is a directed acyclic graph consisting of three types of nodes: the source node S, the data collector node DC and the storage nodes v_i, where i is the node label. In the graph, storing encoded data into n storage nodes is represented by n directed edges from the source S to n storage nodes. Each storage node v_i consists of the input node v_i^{in} and the output node v_i^{out}, connected via a link whose capacity is α, the node storage size. In the case of node failures, a replacement node joins the system by retrieving data from remaining nodes, each contributes to given amount (β_I or β_c) of edge capacity. Finally, the retrieval process from the data collector DC is illustrated by the k directed edges from the v_i^{out} nodes to DC. An example of information flow graph in a $(4, 3, 2, \alpha, \beta_I, \beta_c)$ clustered DSS is shown in Fig. 16.

Due to the dynamic node failure and repair processes in a DSS, the number of possible flow graphs can be infinite. We define $\mathcal{G}(n, k, L, \alpha, \beta_I, \beta_c)$ as the set of all possible flow graphs in a $(n, k, L, \alpha, \beta_I, \beta_c)$ clustered DSS. An information flow graph $G \in \mathcal{G}(n, k, L, \alpha, \beta_I, \beta_c)$ can be expressed as $G = (V(G), E(G))$ where $V(G)$ is the set of all vertices and $E(G)$ is the set of all directed edges in G. For a given $G \in \mathcal{G}(n, k, L, \alpha, \beta_I, \beta_c)$, a *cut* is defined as a set $c \subset E(G)$ of edges which satisfies the following: every directed path from the source to the data collector must pass at least one edge in c. We define a *cut-value* $w(c)$ as the sum of edge capacities for the edges in c. Note that a cut c can be alternatively expressed as a pair of two exclusive vertex sets $(U, \bar{U})$ where $U \subset V(G)$ and $\bar{U} = V(G) \backslash U$ (i.e., V exclusive of U). In the $(U, \bar{U})$ representation, the elements of a cut are determined by the set of all directed edges from U to $\bar{U}$.

As in Fig. 17, data revealed to a cluster eavesdropper are fully characterized by a set of directed edges E_e in

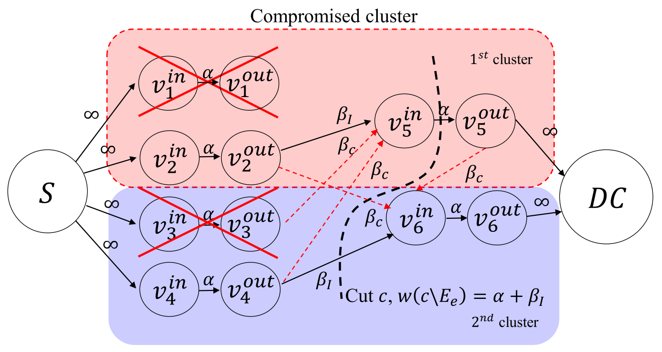


Fig. 17. A possible information flow graph of a $(4, 2, 2, \alpha, \beta_I, \beta_c)$ clustered DSS against $L_c = 1$ cluster eavesdropping. From a given cut c, we can observe that the source cannot store more than $w(c\backslash E_e) = \alpha + \beta_I$ symbols with perfect secrecy.

the information flow graph. For a given flow graph $G \in \mathcal{G}(n, k, L, \alpha, \beta_I, \beta_c)$ and a set of compromised data links E_e, the upper bound on secrecy capacity of a $(n, k, L, \alpha, \beta_I, \beta_c)$ clustered DSS can be obtained as

$$\text{Secrecy capacity} \leq w(c\backslash E_e)$$

where $w(c\backslash E_e)$ is the sum of edge capacities of a cut without compromised edges. It is intuitive to see that the system cannot store data more than $w(c\backslash E_e)$ with perfect secrecy, since DC can retrieve a maximum amount of secure data by ignoring all data from the compromised links.

Now we prove our main theorem by specifying an information flow graph $G \in \mathcal{G}(n, k, L, \alpha, \beta_I, \beta_c)$, a cut $c \in E(G)$ and a set of compromised links $E_e \in E(G)$. Here, we use two dimensional representation of nodes as in Fig. 11. Consider a $(n, k, L, \alpha, \beta_I, \beta_c)$ clustered DSS and a cluster eavesdropper who compromises L_c clusters. The source node stores data in nodes v_i, $i \in [n]$, where the location of each node is expressed as

$$v_i = N(x_i, y_i), \qquad t \in [n]$$

where

$$x_i = \lceil i/n_I \rceil,$$
$$y_i = mod(i-1, n_I) + 1.$$

Without a loss of generality, we assume that $1^{st}, \cdots, L_c^{th}$ clusters are compromised by a cluster eavesdropper. First, we describe obtaining a flow graph G and a set of compromised data links E_e. Consider k successive node failures and repair events. k storage nodes among $\{v_i\}_{i=1}^n$ are successively failed and regenerated to $v_{n+1}, \cdots, v_{n+k}$ where node $v_{n+i}, i \in [k]$ indicates i^{th} regenerated node. The data collector DC contacts k regenerated nodes $v_{n+1}, \cdots, v_{n+k}$ to retrieve data. The two dimensional location of i^{th} failed and corresponding regenerated node is specified as $N(l_i, j_i)$ where

$$l_i = \begin{cases} \lceil i/n_I \rceil, & \text{if } i \leq n_I L_c \\ (i - n_I L_c) - \sum_{m=1}^{j_i-1}(g_m - L_c) + L_c, & \text{otherwise,} \end{cases} \tag{A.1}$$

$$j_i = \begin{cases} mod(i-1, n_I) + 1, & \text{if } i \leq n_I L_c \\ \min\{v : \sum_{m=1}^{v}(g_m - L_c) \geq i - n_I L_c\}, & \text{otherwise,} \end{cases} \tag{A.2}$$

and

$$g_m = \begin{cases} q+1, & \text{if } m \leq r \\ q, & \text{otherwise.} \end{cases}$$

We provide an example to obtain the location of k regenerated nodes in Fig. 19. A node with an index i in the figure stands for the location of i^{th} regenerated node, v_{n+i}. For each index $i \leq n_I L_c$, v_{n+i} is located in the $\lceil i/n_I \rceil^{th}$ compromised clusters. Also, for an index $i > n_I L_c$, note that selection of remaining $k - n_I L_c$ failed nodes follows the identical selection rule defined in Definition 3, 4 of [6]. This procedure provides an unique information flow graph G and the a set of compromised links E_e by a cluster eavesdropper.

Finally, we specify a cut $c = (U, \bar{U})$ of G. Notice that there exists $2 + 2(n+k)$ vertices in G which are the source node S, the data collector node DC and the $(n+k)$ input and output storage nodes $\{v_i^{in}, v_i^{out}\}_{i=1}^{n+k}$. Define w_i^* as the sum of non-compromised incoming edge capacities from $\{v_i^{out}\}_{i=1}^n$ to i^{th} regenerated input node v_{n+i}^{in}, and a set T as

$$T = \{i \in [k] : w_i^* \geq \alpha\}.$$

We split the set of vertices into two disjoint groups U and $\bar{U}$ to specify a cut c as follows.

$$U = \{S, v_i^{in}, v_i^{out}, v_{n+t}^{in} : i \in [n], t \in T\}$$
$$\bar{U} = \{DC, v_{n+t}^{in}, v_{n+i}^{out} : i \in [k], t \notin T\}$$

Then a cut $c = (U, \bar{U})$ satisfies

$$w(c\backslash E_e) = \sum_{i=1}^{k} \min\{\alpha, w_i^*\}. \tag{A.3}$$

Obtaining a graph G, compromised edges E_e and cut c is illustrated in Fig. 18.

What remains is obtaining a value of $w(c\backslash E_e)$ with a function of system parameters. For convenience, we write $\mathcal{C}_s^{(1)}$ and $\mathcal{C}_s^{(2)}$ as follows, with an understanding that $w(c\backslash E_e) = \mathcal{C}_s^{(1)} + \mathcal{C}_s^{(2)}$. If $k \leq n_I L_c$, we regard as $\mathcal{C}_s^{(2)} = 0$.

$$\mathcal{C}_s^{(1)} = \sum_{i=1}^{\min\{n_I L_c, k\}} \min\{\alpha, w_i^*\} \tag{A.4}$$

$$\mathcal{C}_s^{(2)} = \sum_{i=n_I L_c+1}^{k} \min\{\alpha, w_i^*\} \tag{A.5}$$

First, we specify w_i^* for $i \leq n_I L_c$. Consider w_1^* when $L_c \geq 1$. Notice that the first n_I replacement nodes $v_{n+i}, i \in [n_I]$ are located in the first cluster, which is compromised by a cluster eavesdropper. When the first replacement node v_{n+1} connects to the remaining $n-1$ helper nodes to retrieve data, every link coming from other clusters is observed by a cluster eavesdropper. Therefore, links from nodes $\{v_i^{out}\}_{i=2}^{n_I}$ to v_{n+1}^{in} (each node contributes β_I) are the only secure links, while links from $\{v_i^{out}\}_{i=n_I+1}^{n}$ to v_{n+1}^{in} (each node contributes β_c) are not. Thus, $w_1^* = (n_I - 1)\beta_I$. Next, consider w_2^* where the second replacement node v_{n+2} is located in the first cluster. v_{n+2}^{in} connects to $n-2$ active output nodes from the set U and v_{n+1}^{out} from $\bar{U}$. Since v_{n+2}^{in} and $\{v_i^{out}\}_{i=3}^{n_I}$ are located in the

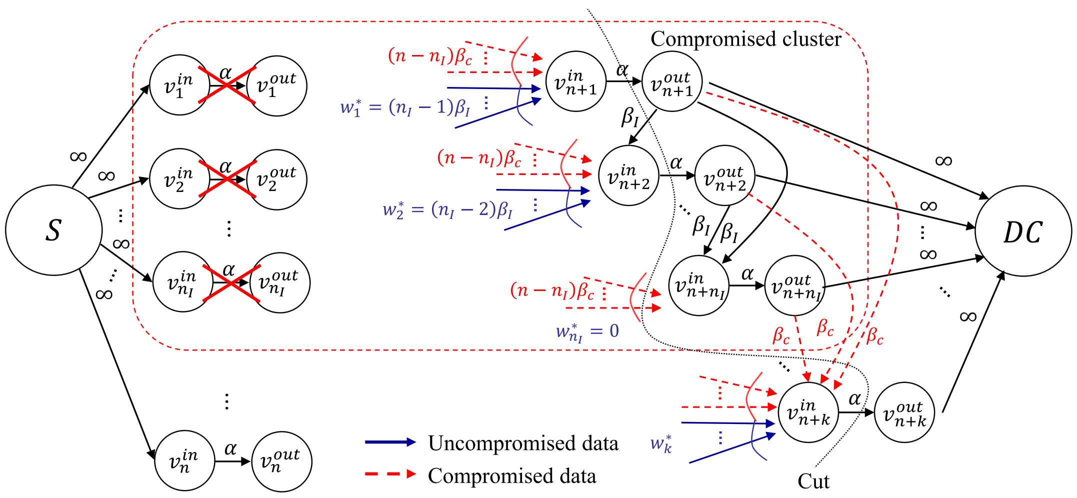


Fig. 18. The information flow graph used to obtain $w(c\backslash E_e)$.

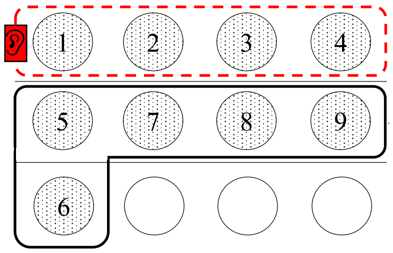


Fig. 19. Failure ordering of a clustered DSS against a cluster eavesdropper where $n = 12, k = 9, L = 3$ and $L_c = 1$.

same cluster, v_{n+2}^{in} receives β_I from $\{v_i^{out}\}_{i=3}^{n_I}$ and β_c from $\{v_i^{out}\}_{i=n_I+1}^{n}$. Since links from nodes in the $j \in [L]\backslash\{1\}^{th}$ cluster to v_{n+2}^{in} are compromised by the eavesdropper, links from $\{v_i^{out}\}_{i=n_I+1}^{n}$ to v_{n+2}^{in} are compromised. Therefore,

$$w_2^* = \gamma_I - \beta_I = (n_I-2)\beta_I.$$

By pursuing a similar analysis, w_t^* is generally derived as

$$w_t^* = (n_I - 1 - mod(t-1, n_I))\beta_I, \quad t \in [n_I L_c]. \tag{A.6}$$

Substituting (A.6) to (A.4) explicitly gives $C_s^{(1)}$. For $k \le n_I L_c$,

$$\begin{aligned}
\mathcal{C}_s^{(1)} &= \sum_{i=1}^{n_I q} \min\{\alpha, mod(n_I - t, n_I)\beta_I\} \\
&\quad + \sum_{i=n_I q+1}^{k} \min\{\alpha, mod(n_I - t, n_I)\beta_I\} \\
&= q\sum_{i=1}^{n_I} \min\{\alpha, (n_I - i)\beta_I\} \\
&\quad + \sum_{i=1}^{r} \min\{\alpha, (n_I - i)\beta_I\}.
\end{aligned}$$

and for $k > n_I L_c$,

$$\begin{aligned}
\mathcal{C}_s^{(1)} &= \sum_{i=1}^{n_I L_c} \min\{\alpha, mod(n_I - t, n_I)\beta_I\} \\
&= L_c \sum_{i=1}^{n_I} \min\{\alpha, (n_I - i)\beta_I\}.
\end{aligned}$$

Now we obtain $\mathcal{C}_s^{(2)}$ for $k > n_I L_c$. For given i where $i \in [k]\backslash[n_I L_c]$, the i^{th} replacement node v_{n+i}^{in} is connected to $\{v_{i+1}^{out}, \cdots, v_n^{out}\}$ in set U and $\{v_{n+1}^{out}, \cdots, v_{n+i-1}^{out}\}$ in set $\bar{U}$. Notice that $\{v_{i+1}^{out}, \cdots, v_n^{out}\}$ in U are located in clusters not observed by a cluster eavesdropper. Therefore, incoming edges from U to v_{n+i}^{in} are secure links. An important observation is that selection of i^{th} failed nodes $i \in [k]\backslash[n_I L_c]$ among remaining $(n - n_I L_c)$ nodes follows an ordering rule suggested in Definition 3, 4 of [6]. Thus, value of $\mathcal{C}_s^{(2)}$ is the cut-value (which turns out to capacity) in a clustered DSS without nodes in the compromised clusters. Therefore, $\mathcal{C}_s^{(2)}$ can be rewritten as

$$\mathcal{C}_s^{(2)} = \mathcal{C}_r$$

where $\mathcal{C}_r$ is capacity of the $(n - n_I L_c, k - n_I L_c, L - L_c, \alpha, \beta_I, \beta_c)$ clustered DSS. Note that repaired bandwidth of a reduced clustered DSS is $(n_I - 1)\beta_I + (n - n_I L_c - n_I)\beta_c$. Since the feasible node storage size should be less than or equal to repair bandwidth, $\mathcal{C}_r$ can be rewritten as a capacity of $(n - n_I L_c, k - n_I L_c, L - L_c, \alpha', \beta_I, \beta_c)$ clustered DSS where $\alpha' = \min\{\alpha, (n_I - 1)\beta_I + (n - n_I L_c - n_I)\beta_c\}$. An upper bound of secrecy capacity $w(c\backslash E_e)$ is obtained by combining $\mathcal{C}_s^{(1)}$ and $\mathcal{C}_s^{(2)}$, which completes the proof.

B. Proof of Theorem 3

Case 1) $1 \le L_c < k/n_I$: From (B.8) and (B.10), we know

$$a_{k'+n_I-1} + b_{k'+n_I-1}u_{k'+n_I-1} = k - L_c$$

and

$$a_\tau \alpha + b_\tau \bar{\gamma} = \{a_\tau + n_I - 1 + (n - n_I)\epsilon\}\alpha.$$

From (B.6), notice that

$$\mathcal{C}_s^U(\alpha,\gamma) = \begin{cases} a_t\alpha + b_t\gamma, & \gamma \in (u_t\alpha, u_{t+1}\alpha], \\ & t \in [k'+n_I-2] \cup \{0\} \\ 0, & \gamma = 0 \end{cases}$$

is a continuous increasing function of γ and $\{b_t\}$ is a decreasing sequence of t where a_t, b_t and τ are defined in Section IV-A.

Therefore, the following inequality holds

$$a_\tau\alpha + b_\tau\bar{\gamma} < a_{k'+n_I-1}\alpha + b_{k'+n_I-1}\bar{\gamma} = (k-L_c)\alpha,$$

which results in

$$\frac{\mathcal{M}_s}{k-L_c} < \frac{\mathcal{M}_s}{a_\tau + n_I - 1 + (n-n_I)\epsilon}.$$

Thus, every (α,γ) pair with $\alpha < \frac{\mathcal{M}_s}{k-L_c}$ is infeasible for all $\gamma \in [0,\infty)$ from Corollary 2.

Case 2) $L_c \geq k/n_I$: Since $q = \lfloor k/n_I \rfloor$, the following inequality trivially holds.

$$\frac{\mathcal{M}_s}{k-L_c} \leq \frac{\mathcal{M}_s}{k-q}$$

Therefore, $\alpha < \frac{\mathcal{M}_s}{k-L_c}$ is infeasible for all $\gamma \in [0,\infty)$ from Corollary 3.

C. Proof of Theorem 4

Define $\mathcal{C}_s^*$ as below, which is a constant with respect to L_c.

$$\mathcal{C}_s^* = \frac{\gamma}{2}\frac{(q+1)(n_I-1)n_I - (n_I-r-1)(n_I-r)}{(n_I-1)}$$

Combining with Proposition 2 which says

$$\mathcal{C}_s^{BL}(L_c) = a(L_c - (L-1/2))^2 + c - a(L-1/2)^2,$$

an intersection point L_c^* satisfying equality $\mathcal{C}_s^{BL}(L_c^*) = \mathcal{C}_s^*$ is expressed as

$$L_c^* = (L-1/2) - \sqrt{(L-1/2)^2 - \frac{c-\mathcal{C}_s^*}{a}}.$$

Note that parameters a and c are function of ϵ. What remains is to prove the following equality

$$\frac{c-\mathcal{C}_s^*}{a} = \frac{w}{n_I^2(n_I-1)},$$

which implicitly shows that intersection point L_c^* is independent of ϵ. This is shown in (A.8), as shown at the top of the next page, which completes the proof.

D. Proof of Theorem 5

A message vector with size $\mathcal{M}-\mathcal{K}$ can be stored with perfect secrecy in a $(n,k,L,(n_I-1)\chi+(n-n_I),\chi,1)$ clustered DSS by using Algorithm 1. All the remaining to prove is that $\mathcal{M}-\mathcal{K}$ achieves the upper bound R of secrecy capacity. In (2), R is expressed as

$$R = \chi L_c \frac{n_I(n_I-1)}{2} + \mathcal{C}_r,$$

where $\mathcal{C}_r$ is the capacity of a reduced $(n',k',L',\alpha',\chi,1) = (n-n_IL_c, k-n_IL_c, L-L_c, (n_I-1)\chi+(n'-n_I),\chi,1)$ clustered DSS. We state the following remark, which explicitly expresses capacity in the bandwidth-limited regime.

Remark 7 (Proposition 2 in [18]). *Capacity $\mathcal{C}$ of a $(n,k,L,\alpha,\chi,1)$ clustered DSS in the bandwidth-limited regime is*

$$\mathcal{C} = k\alpha - \frac{1}{2}(\chi-1)(qn_I^2 + r^2 - k) - \frac{k(k-1)}{2}, \qquad \text{(A.7)}$$

where $q = \lfloor k/n_I \rfloor$ and $r = mod(k,n_I)$.

From Lemma 7, the capacity of a reduced clustered DSS $\mathcal{C}_r$ is expressed as

$$\mathcal{C}_r = k'\alpha' - \frac{1}{2}(\chi-1)(q'n_I^2 + r'^2 - k') - \frac{k'(k'-1)}{2},$$

where $q' = \lfloor k'/n_I \rfloor$ and $r' = mod(k',n_I)$. We can numerically derive an equality

$$\mathcal{M} - \mathcal{K} = R$$

with a simple calculation, where $\mathcal{M}$ and $\mathcal{K}$ are defined in (16) and (17), respectively.

Appendix B
Proof of Corollaries

A. Proof of Corollary 2

Consider a $(n,k,L,\alpha,\beta_I,\beta_c)$ clustered DSS. Using the expressions of ϵ and γ given in (3) and (4), the upper bound of secrecy capacity is expressed as

$$\mathcal{C}_s^U(\alpha,\gamma) = L_c\sum_{i=1}^{n_I}\min\{\alpha, \frac{(n_I-i)\gamma}{n_I-1+(n-n_I)\epsilon}\} + \mathcal{C}_r \qquad \text{(B.1)}$$

where $\mathcal{C}_r$ is the capacity of a $(n',k',L',\alpha',\beta_I,\beta_c)$ clustered DSS.

From the proof of Theorem 1 in [6], note that capacity of a clustered distributed storage can be expressed as

$$\mathcal{C}_r = \sum_{i=1}^{k'}\min\{\alpha, \frac{(n'-n_I-i+h_i)\epsilon + (n_I-h_i)}{(n_I-1)+(n-n_I)\epsilon}\gamma\},$$

where

$$\begin{aligned} n' &= n - n_IL_c, \\ k' &= k - n_IL_c, \\ h_t &= \min_{s\in[n_I]}\sum_{i=1}^{s}(g_i - L_c) \geq t, \;\; t \in [k']. \end{aligned}$$

Define sequences $\{x_t\}$, $\{y_t\}$, $\{z_t\}$ and $\{u_t\}$ as in Section IV-A. Then, the upper bound in (B.1) is simply reduced to

$$\mathcal{C}_s^U(\alpha,\gamma) = L_c\sum_{t=1}^{n_I}\min\{\alpha, \frac{\gamma}{x_t}\} + \sum_{t=1}^{k'}\min\{\alpha, \frac{\gamma}{y_t}\}, \qquad \text{(B.2)}$$

which is a continuous function of γ.

Remark 8. *$\{x_t\}, \{y_t\}$ and $\{u_t\}$ are non-negative increasing sequences, while $\{z_t\}$ is a non-negative decreasing sequence.*

$$
\begin{aligned}
\frac{c-\mathcal{C}_s^*}{a} &= \frac{k\gamma - \frac{\{(1-\epsilon)(qn_I^2+r^2-k)+k(k-1)\}\gamma\epsilon}{2\{n_I-1+(n-n_I)\epsilon\}} - \frac{(q+1)(n_I-1)n_I-(n_I-r-1)(n_I-r)}{2(n_I-1)}}{\frac{n_I^2\gamma\epsilon}{2\{n_I-1+(n-n_I)\epsilon\}}} \\
&= \frac{2k(n_I-1)\{n_I-1+(n-n_I)\epsilon\}-(1-\epsilon)(qn_I^2+r^2-k)(n_I-1)-k(k-1)\epsilon(n_I-1)}{\epsilon n_I^2(n_I-1)} \\
&\quad - \frac{\{(q+1)(n_I-1)n_I-(n_I-r-1)(n_I-r)\}\{n_I-1+(n-n_I)\epsilon\}}{\epsilon n_I^2(n_I-1)} \\
&= \frac{2k(n_I-1)^2-(qn_I^2+r^2-k)(n_I-1)-\{(q+1)(n_I-1)n_I-(n_I-r-1)(n_I-r)\}(n_I-1)}{\epsilon n_I^2(n_I-1)} \\
&\quad + \frac{2k(n_I-1)(n-n_I)+(qn_I^2+r^2-k)(n_I-1)-k(k-1)(n_I-1)(n-n_I)}{n_I^2(n_I-1)} \\
&\quad - \frac{\{(q+1)(n_I-1)n_I-(n_I-r-1)(n_I-r)\}}{n_I^2(n_I-1)} \\
&= \frac{qn_I(n_I-1)(n-qn_I-2r)+r(r-1)(n-n_I)}{n_I^2(n_I-1)}
\end{aligned}
\tag{A.8}
$$

Proof. It is trivial to show $\{x_t\}$ is a non-negative increasing sequence and $\{u_t\}$ is an increasing sequence. $\{z_t\}$ is a non-negative sequence since it satisfies

$$
\begin{aligned}
z_t &= (n'-n_I-t+h_t)\epsilon+(n_I-h_t) \\
&\geq (n'-n_I-t+h_t)\epsilon+(n_I-h_t)\epsilon \\
&= (n'-t)\epsilon > 0
\end{aligned}
$$

for all $t \in [k']$. It follows that $\{y_t\}$ and $\{u_t\}$ are non-negative sequences.

By definition, note that h_t satisfies

$$
h_{t+1} = \begin{cases} h_t+1, & t \in T \\ h_t, & t \in [k'-1]\backslash T \end{cases} \tag{B.3}
$$

where

$$
T = \Big\{ \sum_{i=1}^{l}(g_i - L_c) : l \in [n_I] \Big\}.
$$

Combining with a definition of sequence $\{z_t\}$,

$$
z_{t+1} - z_t = \begin{cases} -1, & t \in T \\ -\epsilon, & t \in [k'-1]\backslash T. \end{cases}
$$

Thus, $\{z_t\}$ is a decreasing sequence, which implies $\{y_t\}$ is an increasing sequence. □

Next, we specify a valid region of repair bandwidth γ. Since intra-cluster bandwidth per node β_I cannot exceed a node storage size α, inequality $\beta_I \leq \alpha$ is trivially satisfied. Therefore, γ is bounded by

$$
\begin{aligned}
\gamma &= (n_I-1)\beta_I + (n-n_I)\beta_c \\
&= \{n_I-1+(n-n_I)\epsilon\}\beta_I \leq \{n_I-1+(n-n_I)\epsilon\}\alpha.
\end{aligned}
$$

and we define threshold $\bar{\gamma}$ of repair bandwidth as

$$
\bar{\gamma} \triangleq \{n_I-1+(n-n_I)\epsilon\}\alpha. \tag{B.4}
$$

Here, notice that the sequence $\{x_t\}$ satisfies

$$
x_t\alpha \leq \bar{\gamma}. \quad \forall t \in [n_I-1].
$$

The rest of the proof depends on the range of ϵ.

Case 1) $\frac{1}{n-k} \leq \epsilon \leq 1$*:* Each element of sequence $\{z_t\}$ satisfies

$$
\begin{aligned}
z_t &\geq z_{k'} \\
&\geq (n'-n_I-k'+h_t)\epsilon+(n_I-h_t)\epsilon \\
&= (n-k)\epsilon \geq 1.
\end{aligned}
\tag{B.5}
$$

Therefore, sequence $\{y_t\}$ satisfies

$$
y_t\alpha \leq \{n_I-1+(n-n_I)\epsilon\}\alpha = \bar{\gamma}, \quad \forall t \in [k'].
$$

For given $\gamma \leq \bar{\gamma}$, there exist unique p, q which satisfy

$$
\begin{aligned}
x_p\alpha &\leq \gamma < x_{p+1}\alpha, \quad p \in [n_I-1] \cup \{0\}, \\
y_q\alpha &\leq \gamma < y_{q+1}\alpha, \quad q \in [k'] \cup \{0\}.
\end{aligned}
$$

Let p, q be an integer that satisfying the above inequalities. Then, the upper bound $\mathcal{C}_s^U$ (B.2) is simplified to

$$
\mathcal{C}_s^U(\alpha,\gamma) = (L_c p + q)\alpha + (r_p + s_q)\gamma, \quad \gamma \leq \bar{\gamma}
$$

where r_p and s_q are defined in Section IV-A.

We can express $\mathcal{C}_s^U$ on different ranges of γ,

$$
\mathcal{C}_s^U(\alpha,\gamma) = \begin{cases} a_t\alpha + b_t\gamma, & \gamma \in (u_t\alpha, u_{t+1}\alpha], \\ & \quad t \in [k'+n_I-2] \cup \{0\} \\ 0, & \gamma = 0 \end{cases} \tag{B.6}
$$

where a_t, b_t are defined in Section IV-A. Notice that $u_{k'+n_I-1} = \bar{\gamma}$, since $u_{k'+n_I-1}$ is the second largest element of $\{x_i\}_{i=1}^{n_I}, \{y_j\}_{j=1}^{k'}$. and x_i, y_j satisfy

$$
\begin{aligned}
x_i, y_j &\leq \bar{\gamma}, \quad i \in [n_I-1], j \in [k'], \\
x_{n_I-1} &= \bar{\gamma}, \\
x_{n_I} &= \infty.
\end{aligned}
$$

Remark 9. *$\{b_t\}$ is a non-negative decreasing sequence.*

Proof. First, every element of sequence $\{b_t\}$ is nonnegative since r_p and s_q are summations of non-negative elements.

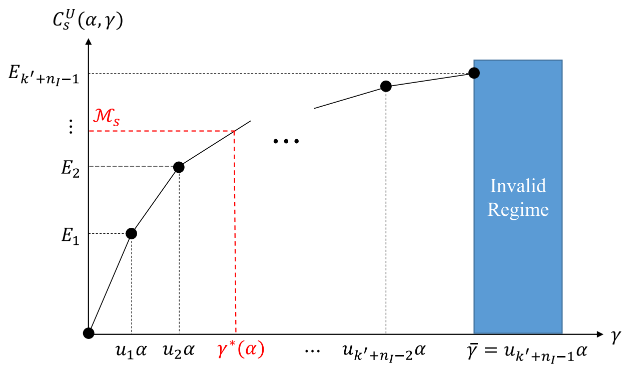


Fig. 20. $\mathcal{C}_s^U(\alpha,\gamma)$ as a function of γ when $L_c < k/n_I$ and $\epsilon \geq \frac{1}{n-k}$.

In order to prove $\{b_t\}$ is a decreasing sequence, let

$$b_t = r_{p'} + s_{q'},$$
$$b_{t+1} = r_{p''} + s_{q''}.$$

Then, inequalities

$$p' \leq p'', \quad q' \leq q''$$

are satisfied. Combining with a definition of r_p and s_q in Section IV-A,

$$b_t - b_{t+1} = \sum_{t=p'+1}^{p''} \frac{1}{x_i} + \sum_{t=q'+1}^{q''} \frac{1}{y_i} \geq 0.$$

Thus, $\{b_t\}$ is a non-negative decreasing sequence. □

Since $\mathcal{C}_s^U$ is a continuous function of γ and b_t is non-negative decreasing sequences, $\mathcal{C}_s^U$ is shown in Fig. 20 as a function of γ. Notice that $\mathcal{C}_s^U(\alpha,\gamma) < \mathcal{M}_s$ if and only if $\gamma < \gamma^*(\alpha)$ from the figure. Consider a (α,γ) pair with $\gamma < \gamma^*(\alpha)$; Since $\mathcal{C}_s^U(\alpha,\gamma)$ is a monotonic increasing function of γ,

$$\mathcal{C}_s(\alpha,\gamma) \leq \mathcal{C}_s^U(\alpha,\gamma) < \mathcal{M}_s$$

holds. Thus, it is impossible to securely store $\mathcal{M}_s$ with given resources (α,γ).

Using (B.6), the threshold value $\gamma^*(\alpha)$ is expressed as

$$\gamma^*(\alpha) = \begin{cases} 0, & \mathcal{M}_s = 0 \\ \frac{\mathcal{M}_s - a_t\alpha}{b_t}, & \mathcal{M}_s \in (E_t, E_{t+1}], \\ & \quad t = [n_I + k' - 2] \cup \{0\} \\ \infty, & \mathcal{M}_s \in (E_{n_I+k'-1}, \infty) \end{cases}$$

where

$$E_t = \mathcal{C}_s^U(\alpha, u_t\alpha) = (a_t + b_t u_t)\alpha. \tag{B.7}$$

$\gamma^*(\alpha)$ is alternatively expressed as a function of α as in (7). Notice that the following equality holds for $t = k' + n_I - 1$.

$$\begin{aligned} a_{k'+n_I-1} + b_{k'+n_I-1}u_{k'+n_I-1} &= L_c(n_I-1) + k' \\ &= k - L_c \end{aligned} \tag{B.8}$$

Case 2) $0 < \epsilon < \frac{1}{n-k}$, $k' < n_I$*:* Under the constraint $k' < n_I$, inequality $h_{k'} \leq n_I - 1$ holds. Therefore, each element of $\{z_t\}_{t=0}^{k'}$ satisfies

$$z_t \geq z_{k'} \geq (n_I - h_t) \geq 1.$$

The remaining proof is identical to the case $\frac{1}{n-k} \leq \epsilon \leq 1$.

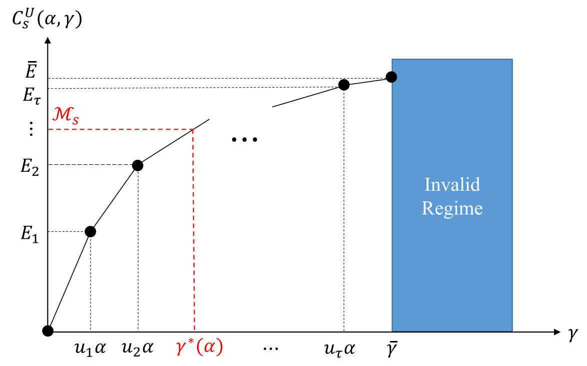


Fig. 21. $\mathcal{C}_s^U(\alpha,\gamma)$ as a function of γ when $L_c < k/n_I, 0 < \epsilon < \frac{1}{n-k}$ and $k' \geq n_I$.

Case 3) $0 < \epsilon < \frac{1}{n-k}, k' \geq n_I$*:* Notice that $h_{k'} = n_I$ holds for $k' \geq n_I$. Since $\{z_t\}$ is a decreasing sequence and $z_{k'} = (n-k)\epsilon < 1$, there exists an unique $\tau \in [k'+n_I-2] \cup \{0\}$ such that

$$u_\tau\alpha < \bar{\gamma} \leq u_{\tau+1}\alpha.$$

From an analysis similar to the $\frac{1}{n-k} < \epsilon < 1$ case, we obtain

$$\mathcal{C}_s^U(\alpha,\gamma) = \begin{cases} 0, & \gamma = 0 \\ a_t\alpha + b_t\gamma, & \gamma \in (u_t\alpha, u_{t+1}\alpha], \\ & \quad t \in [\tau - 1] \cup \{0\} \\ a_\tau\alpha + b_\tau\gamma, & \gamma \in (u_\tau\alpha, \bar{\gamma}] \end{cases} \tag{B.9}$$

for τ such that

$$u_\tau\alpha < \gamma \leq u_{\tau+1}\alpha, \quad \tau \in [k' + n_I - 2] \cup \{0\}$$

where parameters are given in Section IV-A.

Fig. 21 illustrates (B.9) as a function of γ. From the figure, $\mathcal{C}_s^U(\alpha,\gamma) < \mathcal{M}_s$ holds if and only if $\gamma < \gamma^*(\alpha)$. Threshold $\gamma^*(\alpha)$ is obtained as

$$\gamma^*(\alpha) = \begin{cases} 0, & \mathcal{M}_s = 0 \\ \frac{\mathcal{M}_s - a_t\alpha}{b_t}, & \mathcal{M}_s \in (E_t, E_{t+1}], \quad t \in [\tau-1] \cup \{0\} \\ \frac{\mathcal{M}_s - a_\tau\alpha}{b_\tau}, & \mathcal{M}_s \in (E_\tau, \bar{E}] \\ \infty, & \mathcal{M}_s \in (\bar{E}, \infty) \end{cases}$$

where E_t is defined in (B.7) and $\bar{E}$ is expressed as

$$\begin{aligned} \bar{E} \triangleq \mathcal{C}_s^U(\alpha,\bar{\gamma}) &= a_\tau\alpha + b_\tau\bar{\gamma} \\ &= \{a_\tau + n_I - 1 + (n - n_I)\epsilon\}\alpha. \end{aligned} \tag{B.10}$$

Threshold $\gamma^*(\alpha)$ is alternatively expressed as a function of α as in (9).

B. Proof of Corollary 3

From Theorem 1, the upper bound of secrecy capacity is expressed as

$$\begin{aligned} \mathcal{C}_s^U(\alpha,\gamma) = q \sum_{i=1}^{n_I} \min\{\alpha, \frac{(n_I - i)\gamma}{n_I - 1 + (n - n_I)\epsilon}\} \\ + \sum_{i=1}^{r} \min\{\alpha, \frac{(n_I - i)\gamma}{n_I - 1 + (n - n_I)\epsilon}\}. \end{aligned}$$

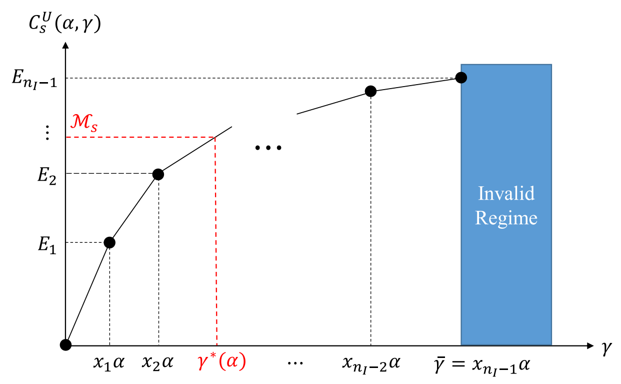


Fig. 22. $C_s^U(\alpha,\gamma)$ as a function of γ when $L_c \geq k/n_I$.

Let $\{x_t\}$ be defined in Section IV-A. The upper bound is simply reduced to

$$C_s^U(\alpha,\gamma) = q\sum_{i=1}^{n_I}\min\{\alpha, \frac{\gamma}{x_i}\} + \sum_{t=1}^{r}\min\{\alpha, \frac{\gamma}{x_i}\}, \tag{B.11}$$

which is a continuous function of γ. Notice that $\{x_t\}$ is a non-negative increasing function of t and satisfies inequality $x_t\alpha \leq \bar{\gamma}$ for all $t \in [n_I-1]$ where $\bar{\gamma}$ defined on (B.4). Therefore, for given $0<\gamma\leq\bar{\gamma}$, there exists an unique t such that

$$x_t\alpha < \gamma \leq x_{t+1}\alpha, \quad t\in[n_I-1]\cup\{0\}.$$

From (B.11), the upper bound C_s^U can be expressed as

$$C_s^U(\alpha,\gamma) = \begin{cases} c_t\alpha + d_t\gamma, & \gamma \in (x_t\alpha, x_{t+1}\alpha], \\ & \qquad t\in[n_I-2]\cup\{0\} \\ 0, & \gamma = 0 \end{cases} \tag{B.12}$$

where $\{c_t\}$ and $\{d_t\}$ are defined in Section IV-A. As illustrated in Fig. 22, C_s^U is a continuous function of γ and $\{v_t\}$ is a non-negative decreasing sequence. Monotonicity of the function leads to: $C_s^U(\alpha,\gamma) < \mathcal{M}_s$ if and only if $\gamma < \gamma^*(\alpha)$, where $\gamma^*(\alpha)$ is expressed as

$$\gamma^*(\alpha) = \begin{cases} \frac{\mathcal{M}_s\alpha}{d_0}, & \mathcal{M}_s \in [0, E_1] \\ \frac{\mathcal{M}_s - c_t\alpha}{d_t}, & \mathcal{M}_s \in (E_t, E_{t+1}] \\ \infty, & \mathcal{M}_s \in (E_{n_I-1},\infty) \end{cases}$$

and

$$E_t \triangleq C_s^U(\alpha, x_t\alpha) = (c_t + d_t x_t)\alpha.$$

Alternatively, $\gamma^*(\alpha)$ is expressed as (10).

C. Proof of Corollary 5

Case 1) $L_c < k/n_I$: From Theorem 2, C_s^{BL} is expressed as

$$C_s^{BL} = L_c\beta_I\frac{n_I(n_I-1)}{2} + C_r^{BL},$$

where C_r^{BL} is capacity of a reduced $(n',k',L',\alpha',\beta_I,\beta_c) = (n-n_IL_c, k-n_IL_c, L-L_c, (n_I-1)\beta_I + (n-n_IL_c-n_I)\beta_c, \beta_I,\beta_c)$ clustered DSS. Thus, secrecy capacity loss can be rewritten as

$$\begin{aligned}\Delta C_s^{BL}(L_c) &= -C_s^{BL}(L_c) + C_s^{BL}(L_c+1) \\ &= \frac{\beta_I n_I(n_I-1)}{2} + C_r^{BL}(L_c+1) - C_r^{BL}(L_c) \\ &= -\beta_c n_I^2(L-L_c-1) \end{aligned} \tag{B.13}$$

where (B.13) comes from the following remark.

Remark 10. *Capacity C of a $(n,k,L,\alpha,\beta_I,\beta_c)$ clustered DSS in the bandwidth-limited regime is*

$$C = k\alpha - \frac{1}{2}(\beta_I-\beta_c)(qn_I^2 + r^2 - k) - \beta_c\frac{k(k-1)}{2}$$

where $q = \lfloor k/n_I \rfloor$ and $r = mod(k,n_I)$.

Proof. The capacity of $(n,k,L,\alpha,\chi,1)$ clustered DSS in the bandwidth-limited regime is explicitly obtained in Remark 7. Since capacity is a linear function of γ with a given ϵ in this regime, C is obtained as β_c times a capacity C' of $(n,k,L,\alpha/\beta_c,\beta_I/\beta_c,1)$ clustered DSS where

$$C' = k\alpha/\beta_c - \frac{1}{2}(\beta_I/\beta_c - 1)(qn_I^2 + r^2 - k) - \frac{k(k-1)}{2}.$$

□

Case 2) $L_c \geq k/n_I$: From Theorem 2, secrecy capacity is independent of L_c when $L_c \geq k/n_I$. Thus, $\Delta C_s^{BL}(L_c) = 0$.

D. Proof of Corollary 7

The exact expression of intersection point L_c^* is given in (14). We write

$$\begin{aligned}\frac{\omega}{n_I^2(n_I-1)} &= \frac{q(n-qn_I-2r)}{n_I} + \frac{r(r-1)(L-1)}{n_I(n_I-1)} \\ &= \frac{q(n-k)}{n_I} - \frac{qr}{n_I} + \frac{r(r-1)(L-1)}{n_I(n_I-1)} \\ &= \frac{(qn_I+r)(n-k)}{n_I^2} - \frac{r(n-k)}{n_I^2} - \frac{qr}{n_I} \\ &\quad + \frac{r(r-1)(L-1)}{n_I(n_I-1)} \\ &= \frac{k(n-k)}{n^2}L^2 - \Delta_1 \\ &= \mathcal{R}(1-\mathcal{R})L^2 - \Delta_1, \end{aligned}$$

where

$$0 \leq \Delta_1 = \frac{r(n-1)(n_I-r)}{n_I^2(n_I-1)} \leq \frac{n-1}{4(n_I-1)}.$$

Notice that $\Delta_1 = o(L)$. Therefore,

$$\begin{aligned} L_c^* &\approx L - \sqrt{L^2 - \frac{\omega}{n_I^2(n_I-1)}} \\ &= L\left(1 - \sqrt{1-\mathcal{R}(1-\mathcal{R}) + o(1/L)}\right) \\ &\approx L(1-\sqrt{1-\mathcal{R}(1-\mathcal{R})}). \end{aligned}$$

Appendix C
Proof of Lemmas

A. Proof of Lemma 6

Before proving Lemma 6, we provide Remark 11, which says that symbol distribution rules of Algorithms 1 and 2 are identical by properly relabeling the indices of coded symbols.

Remark 11. *By properly relabeling indices of encoded symbols $\{c_1,\cdots,c_\theta\}$, the distribution rule of encoded symbols stated in Algorithm 2 is identical to the rule stated in Algorithm 1. Thus, the distribution rule of Algorithm 2 satisfies all properties provided in Lemma 3.*

Proof. Recall the encoded symbol distribution rules:

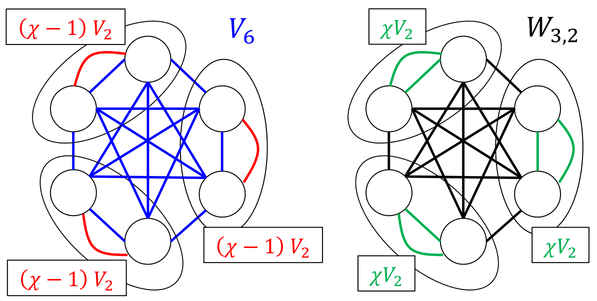


Fig. 23. Graphs corresponding to the encoded symbol distribution rules when $n = 6$, $L = 3$ and $\chi = 2$.

Step 2 of Code 1. Distribute encoded symbols $\{c_1, \cdots, c_\theta\}$ to nodes under the following rules (Construction 2 in [18]):

- First rule: Node $N(l, j)$ stores a symbol c_{i_1} if and only if $V_n(n_I(l-1)+j, i_1) = 1$.

- Second rule: For $t \in [\chi-1]$, node $N(l, j)$ stores a symbol $c_{\binom{n}{2}+(\chi l-\chi-l+t)\binom{n_I}{2}+i_2}$ if and only if $V_{n_I}(j, i_2) = 1$.

Step 2 of Code 2. Distribute encoded symbols $\{c_1, \cdots, c_\theta\}$ to nodes under the following rules:

- First rule: For $t \in [\chi]$, node $N(l, j)$ stores $c_{(l-1)\chi\binom{n_I}{2}+(t-1)\binom{n_I}{2}+i_1}$ if and only if $V_{n_I}(j, i_1) = 1$.

- Second rule: Node $N(l, j)$ stores symbol $c_{\eta+i_2}$ if and only if $W_{L,n_I}(j, i_2) = 1$.

We will show that the distribution rules of codes 1 and 2 can be made identical by properly relabeling the indices. Note that the rule of distributing encoded symbols comes from the incidence matrix of a given graph. We prove the statement of Theorem 5 by showing that two graphs corresponding to the rules of codes 1 and 2 are identical.

The encoded symbol distribution rule of code 1 is obtained from a single complete graph V_n (obtained from the first rule) and $(\chi-1)$ complete graphs V_{n_I} per cluster (obtained from the second rule). On the other hand, that of code 2 comes from χ complete graphs V_{n_I} per cluster (obtained from the first rule) and a single complete L-partite graph W_{L,n_I} (obtained from the second rule). We conclude that two graphs are identical for the following reasons: any two nodes in the same cluster share χ edges, while any two nodes in other clusters share a single edge. In Fig. 23, we provide an example of graphs corresponding to the encoded symbol distribution rules. A graph on the left side corresponds to the rule of code 1, while the right side corresponds to that of code 2. It is easy to see that the two graphs are the same. □

Consider a scenario that θ coded symbols $\{c_1, \cdots, c_\theta\}$ are distributed to a $(n, k, L, (n_I-1)\chi+n-n_I, \chi, 1)$ clustered DSS under the distribution rules described in Algorithm 2. Then, by using Lemma 5 and Remark 11, each symbol in set $T = \{c_1, \cdots, c_\eta\}$ is stored on two distinct nodes in a single cluster, while each of the rest symbols $\{c_{\eta+1}, \cdots, c_\theta\}$ is stored on two distinct nodes in different clusters. A data collector selects totally k nodes ($k_1, k_2, \cdots, k_L$ nodes from 1st, 2nd, $\cdots$, L^{th} cluster) to retrieve coded symbols. Without a loss of generality, we assume

$$k_1 \geq k_2 \geq \cdots \geq k_L.$$

Here, we define a contacting vector as $\mathbf{v} = [k_1, \cdots, k_L]$. A set of feasible contacting vectors is obtained as

$$V = \Big\{[k_1, \cdots, k_L] : \sum_{i=1}^{L} k_i = k, \\ n_I \geq k_1 \geq k_2 \geq \cdots \geq k_L \geq 0\Big\}.$$

From properties (a) and (b) in Lemma 3, we know that any two nodes in a single cluster share χ encoded symbols in set T. Consider a j^{th} cluster with k_j contacting nodes by a data collector. The number of coded symbols in set T retrieved by a data collector from j^{th} cluster is expressed as

$$\mathcal{T}_j = \sum_{i=1}^{k_j} (n_I - i)\chi.$$

Therefore, the minimum number $\mathcal{S}$ of encoded symbols in set T retrieved by a data collector among possible contacting vector $\mathbf{v}$ is expressed as

$$\mathcal{S} = \min_{\mathbf{v}\in V} \sum_{j=1}^{L} \mathcal{T}_j = \min_{\mathbf{v}\in V} \sum_{j=1}^{L}\sum_{i=1}^{k_j} (n_I - i)\chi.$$

Consider an arbitrary contacting vector $\mathbf{v} = [k_1, \cdots, k_L] \in V$. Assume there exists j such that

$$0 < k_{j+1} \leq k_j < n_I.$$

Then, selecting a node in the j^{th} cluster instead of $(j+1)^{th}$ cluster strictly decreases the number of symbols in set T retrieved by a data collector. Therefore, contacting vector $\mathbf{v}' = [k'_1, \cdots, k'_L]$ minimizes retrieved symbols in set T where

$$k'_j = \begin{cases} n_I, & \text{if } j \leq \lfloor k/n_I \rfloor \\ mod(k, n_I), & \text{if } j = \lfloor k/n_I \rfloor + 1 \\ 0, & \text{otherwise.} \end{cases}$$

The value of $\mathcal{S}$ is derived as follow, which is equal to R stated in (2).

$$\begin{aligned} \mathcal{S} &= \sum_{j=1}^{L}\sum_{i=1}^{k'_j} (n_I - i)\chi \\ &= \lfloor k/n_I \rfloor \sum_{i=1}^{n_I} (n_I - i)\chi + \sum_{i=1}^{mod(k,n_I)} (n_I - i)\chi \\ &= \lfloor k/n_I \rfloor \frac{n_I(n_I-1)}{2}\chi + \sum_{i=1}^{mod(k,n_I)} (n_I - i)\chi \\ &= R \end{aligned}$$

Appendix D
Proof of Propositions

A. Proof of Proposition 1

From Theorem 1, the upper bound of secrecy capacity with the maximum number of helper nodes is obtained by the cut-value of secure links (A.3), where a flow graph, compromised clusters and a cut are determined as in the proof of Theorem 1. Consider an information flow graph with a general number of helper nodes in the same setting. Then, a cut-value of secure links with arbitrary d_I, d_c intra- and cross- helper nodes is expressed as

$$W(d_I, d_c) \triangleq \sum_{i=1}^{k} \min\{\alpha, w_i^*(d_I, d_c)\},$$

where $w_i^*(d_I, d_c)$ is defined as the sum of non-compromised incoming edge capacities from $\{v_i^{out}\}_{i=1}^n$ to i^{th} regenerated input node v_{n+i}^{in} with d_I and d_c helper nodes from the same and other clusters. Then, $W(d_I, d_c)$ is the upper bound of secrecy capacity with general d_I and d_c.

In the bandwidth-limited regime where $\alpha = \gamma$, the cut-value is simply expressed as

$$W(d_I, d_c) = \sum_{i=1}^{k} w_i^*(d_I, d_c).$$

The exact value of secrecy capacity with the maximum helper nodes, $\mathcal{C}_{s,max}$, is obtained in the present paper as follows, which is the direct result of Theorem 2:

$$\mathcal{C}_{s,max} = W(d_{I,max}, d_{c,max}),$$

where

$$\begin{aligned} d_{I,max} &= n_I - 1, \\ d_{c,max} &= n - n_I. \end{aligned}$$

We prove the proposition by showing

$$w_i^*(d_I, d_c) \le w_i^*(d_{I,max}, d_{c,max}), \quad \forall d_I, d_c \tag{D.1}$$

for all $i \le n_I L_c$ and

$$\sum_{i=n_I L_c+1}^{k} w_i^*(d_I, d_c) \le \sum_{i=n_I L_c+1}^{k} w_i^*(d_{I,max}, d_{c,max}), \quad \forall d_I, d_c \tag{D.2}$$

which leads to the inequality

$$W(d_I, d_c) \le \mathcal{C}_{s,max} \ \ \forall d_I, d_c. \tag{D.3}$$

The physical meaning of (D.3) is that it is impossible to securely store data more $\mathcal{C}_{s,max}$ symbols with arbitrary d_I and d_c. Thus, setting d_I and d_c to the maximum possible values maximizes secrecy capacity.

Case 1) $i \le n_I L_c$: Notice that the i^{th} replacement node is located in the compromised cluster. Thus, every cross-cluster traffic coming to the replacement node is observed by a cluster eavesdropper. Generally, w_i^* is expressed as

$$\begin{aligned} w_{x+yn_I}^*(d_I, d_c) &= \max\{(d_I + 1 - x), 0\}\beta_I \\ &= \max\{1 - \frac{x-1}{d_I}, 0\}\gamma_I \end{aligned}$$

where $x \in [n_I]$ and $y \in [L_c - 1] \cup \{0\}$. For fixed x, y and γ_I, $w_{x+yn_I}^*(d_I, d_c)$ is a decreasing function of d_I and independent of d_c. Therefore, (D.1) holds for $i \le n_I L_c$.

Case 2) $n_I L_c < i \le k$: As stated in the proof of Theorem 1, the right hand side of (D.2) corresponds to the capacity $\mathcal{C}_r$ of a reduced $(n', k', L', \alpha', \beta_I, \beta_c)$ clustered DSS with maximal helper nodes,

$$\mathcal{C}_r(d_I, d_c) = \sum_{i=n_I L_c+1}^{k} w_i^*(d_{I,max}, d_{c,max})$$

where $n' = n - n_I L_c, k' = k - n_I L_c$ and $L' = L - L_c$. Since maximizing the number of helper nodes maximizes capacity of a clustered DSS, as stated in Proposition 1 in [6], (D.2) directly holds.

B. Proof of Proposition 2

Note that $\mathcal{C}_s^{BL}(0)$ corresponds to the case without an eavesdropper, which is the capacity of a clustered DSS in the bandwidth-limited regime. The exact value of capacity is obtained as below from Remark 10, by setting $\epsilon = \beta_c/\beta_I$ and bandwidth-limited condition $\alpha = \gamma$:

$$\mathcal{C}_s^{BL}(0) = k\gamma - \frac{\gamma}{2} \frac{(1-\epsilon)(qn_I^2 + r^2 - k) + k(k-1)\epsilon}{(n_I - 1) + (n - n_I)\epsilon}.$$

Combining with Corollary 5, which says $\mathcal{C}_s^{BL}(L_c)$ is a quadratic function of L_c with the axis of symmetry $L_c = L - 1/2$, gives the unique expression of $\mathcal{C}_s^{BL}$.

References

[1] B. Choi, J.-Y. Sohn, S. W. Yoon, and J. Moon, "Secure clustered distributed storage against eavesdroppers," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–6.

[2] R. Bhagwan, K. Tati, Y.-C. Cheng, S. Savage, and G. M. Voelker, "Total recall: System support for automated availability management," in *Proc. NSDI*, vol. 4, 2004, p. 25.

[3] F. Dabek, J. Li, E. Sit, J. Robertson, M. F. Kaashoek, and R. Morris, "Designing a DHT for low latency and high throughput," in *Proc. NSDI*, vol. 4, 2004, pp. 85–98.

[4] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, "The Hadoop distributed file system," in *Proc. IEEE 26th Symp. Mass Storage Syst. Technol. (MSST)*, May 2010, pp. 1–10.

[5] S. Muralidhar *et al.*, "f4: Facebook's warm BLOB storage system," in *Proc. 11th USENIX Conf. Operating Syst. Design Implement.*, 2014, pp. 383–398.

[6] J.-Y. Sohn, B. Choi, S. W. Yoon, and J. Moon, "Capacity of clustered distributed storage," *IEEE Trans. Inf. Theory*, vol. 65, no. 1, pp. 81–107, Jan. 2019.

[7] S. Pawar, S. El Rouayheb, and K. Ramchandran, "Securing dynamic distributed storage systems against eavesdropping and adversarial attacks," *IEEE Trans. Inf. Theory*, vol. 57, no. 10, pp. 6734–6753, Oct. 2011.

[8] N. B. Shah, K. Rashmi, and P. V. Kumar, "Information-theoretically secure regenerating codes for distributed storage," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, Dec. 2011, pp. 1–5.

[9] Y.-J. Chen, L.-C. Wang, and C.-H. Liao, "Eavesdropping prevention for network coding encrypted cloud storage systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 8, pp. 2261–2273, Aug. 2016.

[10] 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, Nov. 2013.

[11] A. S. Rawat, "Secrecy capacity of minimum storage regenerating codes," in *Proc. IEEE Inf. Symp. Inf. Theory (ISIT)*, Jun. 2017, pp. 1406–1410.

[12] I. Samy, G. Calis, and O. O. Koyluoglu, "Secure regenerating codes for hybrid cloud storage systems," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2017, pp. 2208–2212.

[13] S. Kadhe and A. Sprintson, "Security for minimum storage regenerating codes and locally repairable codes," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2017, pp. 1028–1032.

[14] O. O. Koyluoglu, A. S. Rawat, and S. Vishwanath, "Secure cooperative regenerating codes for distributed storage systems," *IEEE Trans. Inf. Theory*, vol. 60, no. 9, pp. 5228–5244, Sep. 2014.

[15] J. Liu, H. Wang, M. Xian, and K. Huang, “A secure and efficient scheme for cloud storage against eavesdropper,” in *Proc. Int. Conf. Inf. Commun. Secur.* Cham, Switzerland: Springer, 2013, pp. 75–89.

[16] R. Tandon, S. Amuru, T. C. Clancy, and R. M. Buehrer, “Toward optimal secure distributed storage systems with exact repair,” *IEEE Trans. Inf. Theory*, vol. 62, no. 6, pp. 3477–3492, Jun. 2016.

[17] F. Ye, K. W. Shum, and R. W. Yeung, “The rate region for secure distributed storage systems,” *IEEE Trans. Inf. Theory*, vol. 63, no. 11, pp. 7038–7051, Nov. 2017.

[18] J.-Y. Sohn and J. Moon, “Explicit construction of MBR codes for clustered distributed storage,” 2018, *arXiv:1801.02287*. [Online]. Available: https://arxiv.org/abs/1801.02287

[19] A. Subramanian and S. W. McLaughlin, “MDS codes on the erasure-erasure wiretap channel,” 2009, *arXiv:0902.3286*. [Online]. Available: https://arxiv.org/abs/0902.3286

[20] 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.

[21] Y. Hu *et al.*, “Optimal repair layering for erasure-coded data centers: From theory to practice,” *ACM Trans. Storage*, vol. 13, no. 4, p. 33, 2017.

[22] B. Gastón, J. Pujol, and M. Villanueva, “A realistic distributed storage system that minimizes data storage and repair bandwidth,” in *Proc. Data Compress. Conf.*, Mar. 2013, p. 491.

[23] N. Prakash, V. Abdrashitov, and M. Médard, “The storage versus repair-bandwidth trade-off for clustered storage systems,” *IEEE Trans. Inf. Theory*, vol. 64, no. 8, pp. 5783–5805, Aug. 2018.

[24] T. Ernvall, S. El Rouayheb, C. Hollanti, and H. V. Poor, “Capacity and security of heterogeneous distributed storage systems,” *IEEE J. Sel. Areas Commun.*, vol. 31, no. 12, pp. 2701–2709, Dec. 2013.

[25] T. Benson, A. Akella, and D. A. Maltz, “Network traffic characteristics of data centers in the wild,” in *Proc. 10th ACM SIGCOMM Conf. Internet Meas.*, Nov. 2010, pp. 267–280.

[26] Y. Hu, P. P. C. Lee, and X. Zhang, “Double regenerating codes for hierarchical data centers,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jul. 2016, pp. 245–249.

[27] L. H. Ozarow and A. D. Wyner, “Wire-tap channel II,” *AT T Bell Lab. Tech. J.*, vol. 63, no. 10, pp. 2135–2157, 1984.

Beongjun Choi (S’17) received the B.S. and M.S. degrees in mathematics and electrical engineering from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 2014 and 2017. He is currently pursuing the electrical engineering Ph.D. degree in KAIST. His research interests include blockchain, coding for distributed storage system and information theory. He is a co-recipient of the IEEE international conference on communications (ICC) best paper award in 2017.

Jy-Yong Sohn (S’15) received the B.S. and M.S. degrees in electrical engineering from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 2014 and 2016. He is currently pursuing the Ph.D. degree in KAIST. His research interests include coding for distributed storage/computing, distributed learning and information theory. He received the KAIST EE Best Research Achievement Award in 2018, the IEEE International Conference on Communications (ICC) Best Paper Award in 2017, and the Qualcomm Innovation Award in 2015.

Sung Whan Yoon (M’17) received the M.S. and Ph.D. degrees in electrical engineering from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 2013 and 2017 respectively. He is currently a postdoctoral researcher in KAIST from 2017. His research interests are in the area of coding theory, distributed system and deep learning algorithms. Recently, his primary interest is focused on meta-learning and continual learning of deep neural networks. Also, he is highly interested in hardware-friendly learning algorithms based on distributed resources (data, computation and communication). He was a co-recipient of the IEEE International Conference on Communications best Paper Award in 2017.

Jaekyun Moon (F’05) received the Ph.D. degree in electrical and computer engineering at Carnegie Mellon University, Pittsburgh, PA, USA. He is currently a Professor of electrical engineering at KAIST. From 1990 through early 2009, he was with the faculty of the School of Electrical and Computer Engineering at the University of Minnesota, Twin Cities. He consulted as Chief Scientist for DSPG, Inc. from 2004 to 2007. He also worked as Chief Technology Officer at Link-A-Media Devices Corporation. His research interests are in the area of channel characterization, signal processing and coding for data storage and digital communication. Prof. Moon received the McKnight Land-Grant Professorship from the University of Minnesota. He received the IBM Faculty Development Awards as well as the IBM Partnership Awards. He was awarded the National Storage Industry Consortium (NSIC) Technical Achievement Award for the invention of the maximum transition run (MTR) code, a widely used error-control/modulation code in commercial storage systems. He served as Program Chair for the 1997 IEEE Magnetic Recording Conference. He is also Past Chair of the Signal Processing for Storage Technical Committee of the IEEE Communications Society. He served as a guest editor for the 2001 IEEE JSAC issue on Signal Processing for High Density Recording. He also served as an Editor for IEEE TRANSACTIONS ON MAGNETICS in the area of signal processing and coding for 2001–2006. He is an IEEE Fellow.