Optimal Coding and Allocation for Perfect Secrecy in Multiple Clouds

Ping Hu, Chi Wan Sung, *Member, IEEE*, Siu-Wai Ho, *Senior Member, IEEE*, and Terence H. Chan, *Member, IEEE*

***Abstract*— For a user to store data in the cloud, using services provided by multiple cloud storage providers (CSPs) is a promising approach to increase the level of data availability and confidentiality, as it is unlikely that different CSPs are out of service at the same time or collude with each other to extract information of a user. This paper investigates the problem of storing data reliably and securely in multiple CSPs constrained by given budgets with minimum cost. Previous works, with variations in problem formulations, typically tackle the problem by decoupling it into sub-problems and solve them separately. While such a decoupling approach is simple, the resultant solution is suboptimal. This paper is the first one which considers the problem as a whole and derives a jointly optimal coding and storage allocation scheme, which achieves perfect secrecy with minimum cost. The analytical result reveals that the optimal coding scheme is the nested maximum-distance-separable code and the optimal amount of data to be stored in the CSPs exhibits a certain structure. The exact parameters of the code and the exact storage amount to each CSP can be determined numerically by simple 2-D search.**

***Index Terms*— Cloud storage, perfect secrecy, information-theoretic security, storage allocation.**

I. Introduction

CLOUD STORAGE providers (CSPs), such as Amazon Cloud, SkyDrive, Dropbox, Google Drive and Sugarsync, offer storage space for their customers. When storing data over a cloud, users are free from the costs of setting up their own servers, electric power charges, space expenses and maintenance costs [1]. Furthermore, they can download their stored data anywhere, provided that the CSP to which they have subscribed is accessible. This brings a lot of convenience in this information era. However, this technology gives rise to security concerns [2]. To address this issue, many techniques have been developed in the literature. For example, auditing protocols in [3] and [4] provide security and privacy for cloud storage and computation. Methodology proposed in [5] secures the data against insider threats among a group of users. The protocol proposed in [6] can identify faulty links in data center under practical assumptions. The system designed in [7] can detect and trace back attacks on encrypted protocols. There are also approaches for other systems, which we can borrow to solve the security problem in cloud storage effectively [8]–[10].

With respect to the features of cloud storage, there are some specific concerns. First, CSPs may be unavailable temporarily or even permanently due to various reasons including disk failure, hacker attack, network disconnection, natural disaster, or even political influence. Second, from users' perspective, data stored in a CSP is not confidential, since the CSP has full access to its customers' data [11]. To ensure data availability and confidentiality, one way is for users to subscribe storage services from multiple CSPs [12], [13]. By encoding and distributing their data to more than one CSPs, users can increase the level of data availability and prevent a CSP to extract information from their data.

While there are many CSPs offering cloud storage services, users have to determine how much storage space should be requested from each of them. This depends on various factors including pricing plans, download and upload rates of their servers, and so on [14], [15]. Naturally, a user would like to minimize the total cost of subscription to several CSPs. Besides, he or she would like to minimize the time to retrieve a file stored in multiple clouds. In this respect, the storage server with lowest download rate becomes the bottleneck of the whole download process. To address this issue, we formulate the problem in a way such that there is a maximum limit on the amount of data blocks that can be stored in each server, depending on the upload/download rate of that server. We call it storage budget of a server, which is one of the three main considerations in this work:

1) data availability and confidentiality;
2) storage cost minimization;
3) data allocation among multiple clouds under storage budget constraints.

In the literature, data availability and confidentiality have been addressed from many different perspectives. One central idea is the seminal work on secret sharing [16], which considers the sharing of a secret among a group of people. When the number of gathered people reaches a threshold t, they can reveal the secret. When there are fewer than t people, no information is disclosed in the information-theoretic sense. This is now commonly called perfect secrecy [17], [18]. Following this approach, there are some classic works from the communications perspective. The secrecy capacity of the

Manuscript received March 29, 2015; revised September 28, 2015; accepted October 29, 2015. Date of publication November 12, 2015; date of current version December 10, 2015. This work was supported in part by the Research Grants Council, University Grants Committee, Hong Kong, under Project AoE/E-02/08 and in part by the Discovery Project Australian Research Council under Grant DP150103658. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Athanasios Vasilakos.

P. Hu and C. W. Sung are with the Department of Electronic Engineering, City University of Hong Kong, Hong Kong (e-mail: irenehu2011@gmail.com; albert.sung@cityu.edu.hk).

S.-W. Ho and T. H. Chan are with the Institute for Telecommunications Research, University of South Australia, Adelaide, SA 5095, Australia (e-mail: siuwai.ho@unisa.edu.au; terence.chan@unisa.edu.au).

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

Digital Object Identifier 10.1109/TIFS.2015.2500193

so-called wiretap channel II is shown to be achievable by coset coding in [19], and that of the erasure-erasure wiretap channel, a generalization of the former channel, is shown to be achievable by nested maximum-distance-separable (MDS) codes in [20]. These works form a basis for the study of perfect secrecy in other applications. For example, the secret sharing approach is adopted to consider secure distributed storage in sensor networks in [21]. For distributed storage systems (DSSs) supporting repair [22], [23], nested MDS codes are used when considering its secrecy capacity [24]. Secrecy capacities of other DSS under different repair requirements are studied in [25] and [26].

Along this line, the most relevant one to our work is [27]. It considers secrecy in multiple clouds but under weak security requirement [28]. It means that the security level is to ensure only that an eavesdropper cannot decode any of the original symbols, which is much weaker than perfect secrecy. On the other hand, it is mentioned in [27] that their result can be extended to perfect secrecy if a specific number of random keys are inserted before encoding. The major difference between [27] and our work is that we also address the issues of storage cost minimisation and data allocation among multiple clouds under storage budget constraints, which are issues (2) and (3) stated above.

For some storage applications, the costs of storing data in different servers may not be equal, which justifies the consideration of (1) and (2) together. In [29], the problem is tackled by decoupling it into two sub-problems. In the first sub-problem, an existing code is adopted and the file is encoded into a pre-determined number of pieces. In the second sub-problem, allocation of those encoded pieces are performed. Since the encoding parameters are pre-determined, the proposed solution is in general sub-optimal. Another approach to tackle the same problem has been considered in our previous work in [30]. The system-level security requirement is that the stored data should be confidential under potential collusion of a certain number of CSPs. Besides, instead of adopting existing coding schemes and focusing only on the optimization aspects, the problem is considered as a whole and the jointly optimal storage allocation and coding scheme is determined.

Extending [30], this paper performs a more comprehensive study of the problem and address the storage budget issue. Under this additional constraint, the problem scope becomes wider and the analysis in [30] is significantly generalized. In the literature, there are also works which consider the coding and allocation problem but with different considerations. For example, the total storage budget required to satisfy a given set of deterministic recovery requirements in a network is minimized in [31] and [32]. Data allocation for maximizing the successful recovery probability within a given storage budget is derived in [33]. How to minimize total storage cost with functional repair mechanics is studied in [34]. Joint coding and allocation problem for heterogeneous storage systems is considered in [35].

In summary, to design the storage application using multiple clouds with consideration of (1), (2) and (3), we need to find a jointly optimal coding and allocation scheme. A fundamental study of the problem based on information theory and optimization techniques is carried out. Our contributions are as follows:

- *Optimal Coding*: A lower bound on the total storage cost for keeping availability and confidentiality with budget constraint is derived. A coding scheme which achieves the bound is identified, showing that the coding scheme is optimal.
- *Optimal Allocation*: The optimal storage amounts of the coded data on the CSPs are shown to exhibit a nice intricate structure, regardless of the differences in prices and budgets. The reveal of this structure enables the design of a fast algorithm for determining the coding parameters.
- *Low-Complexity Algorithm*: A fast two-dimensional search algorithm is designed. It is able to solve the original optimization problem, which has an exponential number of constraints, with a time complexity of $O(N^2 \log N)$, where N is the number of CSPs under consideration.

Since our proposed algorithm has low time complexity, it can be applied to systems involving large number of storage nodes. Apart from the multi-cloud storage scenario, our work may also find applications to other DSSs, such as peer-to-peer cloud storage systems [36], wireless device-to-device systems [37], or wired caching systems [38], as the capacities of the storage nodes in these systems are different, thus imposing different budget constraints.

The rest of this paper is organized as follows. In Section II we present our model. In Section III, we derive a lower bound on the storage cost. A coding scheme which providing privacy and availability within given budgets is presented in Section IV, and the optimality of the coding scheme is shown in Section V. In Section VI, we solve the solution of the optimal allocation parameters. In Section VII, we present the steps for parameters calculation, encoding and allocation. Some numerical examples are given in VIII, and Section IX is the conclusion.

II. System Model

We consider the scenario in which a user wants to store a file securely to multiple clouds. The file consists of B blocks of data, each of which is represented by a symbol drawn uniformly at random from the finite field $\mathbb{F}$ of size q. The whole file is then denoted by a vector $\boldsymbol{x} \triangleq (x_1, x_2, \ldots, x_B)$. The file size, or equivalently, the entropy of $\boldsymbol{x}$, $H(\boldsymbol{x})$, is equal to $B \log_2 q$ bits.

Let N be the number of available CSPs for the user to store data. Before storing the file, the user encodes the B blocks of data into n blocks. We use

$$f : \mathbb{F}^B \to \mathbb{F}^n,$$

which maps $\boldsymbol{x}$ into $\boldsymbol{y}$, to denote the *encoding function*. The n-dimensional vector $\boldsymbol{y}$ is regarded as the concatenation of N sub-vectors, i.e., $\boldsymbol{y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_N)$. For $i = 1, 2, \ldots, N$, let $\boldsymbol{y}_i \triangleq (y_{i,1}, y_{i,2}, \ldots, y_{i,n_i}) \in \mathbb{F}^{n_i}$ be the data stored on CSP i. Clearly, the total number of encoded blocks is equal to the sum of number of encoded blocks stored in each CSP, i.e., $n = \sum_{i=1}^{N} n_i$.

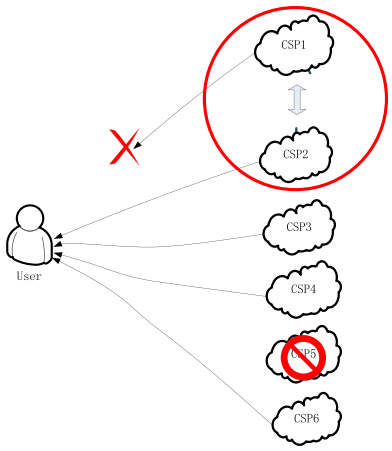

Fig. 1. Example when $N = 6$, $K = 4$ and $T = 2$. The user can retrieve the file from any four CSPs and no information is exposed to any two CSPs. The figure shows the scenario in which CSPs 1 and 2 collude to break the user's file, CSP 5 is failed, and CSP 1 is temporarily unreachable from the user. According to the given parameters, the user can get the file from CSPs 2, 3, 4 and 6, and no information of the file is leaked to CSPs 1 or 2.

Let V_i for $i \in \mathcal{N}$ be the amount of blocks which can be downloaded from CSP i within a predefined time delay, it is required that

$$n_i \leq V_i. \tag{1}$$

In this work we assume V_i's are integers, $V_i \geq 1$ and distinct for $i \in \mathcal{N}$. We call V_i the *budget* of the stored data on CSP i. Let C_i be the cost for storing one block of data on CSP i and C_i's are all distinct. The total storage cost is given by

$$C = \sum_{i=1}^{N} C_i n_i. \tag{2}$$

After encoding and allocation, the user should be able to retrieve the file at any time and the file should be kept private. However, some CSPs may not be available when the user wants to retrieve the file and some CSPs may not be secure. Assume that the maximum number of unavailable CSPs at one time is $N - K$ with $K \leq N$ and the potential number of colluding CSPs is $T < K$. The user has the knowledge of N, K and T. The colluding CSPs have full knowledge about the encoding and decoding algorithms. They can also cooperate to break the scheme. It is required that the user should be able to retrieve the file through any K CSPs and no information of the stored file is exposed against any T colluding CSPs. An example of the model is shown in Fig. 1.

For any $\mathcal{S} \subseteq \mathcal{N} \triangleq \{1, 2, \ldots, N\}$, define $\boldsymbol{y}(\mathcal{S})$ as the sub-vector of $\boldsymbol{y}$ obtained by retaining only $\boldsymbol{y}_i$ for $i \in \mathcal{S}$. To fulfill the *reconstruction requirement* that the customer can reconstruct its file from "any K out of N" CSPs, for any $\mathcal{S} \subset \mathcal{N}$ of cardinality K, there should exist a decoding function $g_{\mathcal{S}}$, such that $g_{\mathcal{S}}(\boldsymbol{y}(\mathcal{S})) = \boldsymbol{x}$. This requirement can also be expressed as

$$H(\boldsymbol{x}|\boldsymbol{y}(\mathcal{S})) = 0, \quad \forall \mathcal{S} \subset_K \mathcal{N}, \tag{3}$$

where we have used the shorthand notation $\mathcal{A} \subset_k \mathcal{B}$ to mean that $\mathcal{A}$ is a k-subset of $\mathcal{B}$.

The secrecy criterion is the *perfect secrecy* [17], [18], which means that the eavesdropper should get no information (in information-theoretic sense) about the message. In other words, to ensure that the file is perfectly secure against any T colluding CSPs, we must have

$$H(\boldsymbol{x}|\boldsymbol{y}(\mathcal{T})) = H(\boldsymbol{x}), \quad \forall \mathcal{T} \subset_T \mathcal{N}. \tag{4}$$

We call the above requirement the *perfect secrecy requirement*.

We call $(n_1, n_2, \ldots, n_N)$ the *allocation parameters*. The problem is to determine the encoding function f and the allocation parameters $(n_1, n_2, \ldots, n_N)$ so as to minimize the total storage cost, C, in (2) with respect to the reconstruction requirement (3), perfect secrecy requirement (4), and budget constraints (1). Note that the code length, n, is a variable depending on the allocation parameters n_i's. The encoding function or allocation parameters is called optimal if there is no other encoding function or allocation papameters results in lower total storage cost.

Throughout this paper, for notational convenience, we use $\mathcal{A}_1 \setminus \mathcal{A}_2 \setminus \mathcal{A}_3$ to represent $(\mathcal{A}_1 \setminus \mathcal{A}_2) \setminus \mathcal{A}_3$ for any given sets $\mathcal{A}_1, \mathcal{A}_2$ and $\mathcal{A}_3$. The notation generalizes naturally to any number of sets.

III. A Lower Bound on the Storage Cost

In this section, we derive a lower bound on the minimum storage cost by elementary manipulations of Shannon's information measures.

Theorem 1: The cost C is bounded below by

$$C_{LB} \triangleq \min \sum_{i=1}^{N} C_i n_i, \tag{5}$$

where n_i's are integers subject to

$$0 \leq n_i \leq V_i, \quad \text{for } i = 1, 2, \ldots, N, \tag{6}$$

$$\min_{\mathcal{T} \subset_T \mathcal{S} \subset_K \mathcal{N}} \Big(\sum_{i \in \mathcal{S}} n_i - \sum_{i \in \mathcal{T}} n_i\Big) \geq B. \tag{7}$$

Proof: Consider an arbitrary $\mathcal{S} \subset_K \mathcal{N}$ and an arbitrary $\mathcal{T} \subset_T \mathcal{N}$. From the perfect secrecy requirement (4) and the reconstruction requirement (3), we have

$$H(\boldsymbol{x}) = H(\boldsymbol{x}|\boldsymbol{y}(\mathcal{T})) - H(\boldsymbol{x}|\boldsymbol{y}(\mathcal{S})).$$

Denote $\mathcal{I} \triangleq \mathcal{T} \cap \mathcal{S}$. We can then rewrite the above equation as

$$\begin{aligned} H(\boldsymbol{x}) &= H(\boldsymbol{x}|\boldsymbol{y}(\mathcal{I}), \boldsymbol{y}(\mathcal{T}\setminus\mathcal{I})) - H(\boldsymbol{x}|\boldsymbol{y}(\mathcal{I}), \boldsymbol{y}(\mathcal{S}\setminus\mathcal{I})) \\ &= I(\boldsymbol{x}; \boldsymbol{y}(\mathcal{S}\setminus\mathcal{I})|\boldsymbol{y}(\mathcal{I})) - I(\boldsymbol{x}; \boldsymbol{y}(\mathcal{T}\setminus\mathcal{I})|\boldsymbol{y}(\mathcal{I})) \\ &\leq I(\boldsymbol{x}; \boldsymbol{y}(\mathcal{S}\setminus\mathcal{I})|\boldsymbol{y}(\mathcal{I})) \\ &\leq H(\boldsymbol{y}(\mathcal{S}\setminus\mathcal{I})|\boldsymbol{y}(\mathcal{I})) \\ &\leq H(\boldsymbol{y}(\mathcal{S}\setminus\mathcal{I})) \\ &\leq \sum_{i \in \mathcal{S}\setminus\mathcal{I}} H(\boldsymbol{y}_i) \\ &\leq \log_2 q \sum_{i \in \mathcal{S}\setminus\mathcal{I}} n_i \\ &= \log_2 q \Big(\sum_{i \in \mathcal{S}} n_i - \sum_{i \in \mathcal{I}} n_i\Big). \end{aligned} \tag{8}$$

The above bound on $H(\boldsymbol{x})$ holds for any $\mathcal{S} \subset_K \mathcal{N}$ and any $\mathcal{T} \subset_T \mathcal{N}$. The tightest ones are the cases when $\mathcal{T}$ is a subset of $\mathcal{S}$. Therefore, (8) can be re-written as

$$H(\boldsymbol{x}) \leq \log_2 q \min_{\mathcal{T}\subset_T\mathcal{S}\subset_K\mathcal{N}} \left(\sum_{i\in\mathcal{S}} n_i - \sum_{i\in\mathcal{T}} n_i\right).$$

As $H(\boldsymbol{x}) = B\log_2 q$, we have shown that (7) is a necessary condition to be met. Besides, thresholds constraints (6) is also a necessary condition. Therefore, if we minimize the total cost subject to (7) and the threshold constraints (6), the result so obtained becomes a lower bound for the storage cost. □

Note that the lower bound in Theorem 1 is well defined if and only if the minimization problem is feasible. It is clear that if the values of V_i's are too small, the problem may become infeasible. To guarantee feasibility, the following condition is necessary. We will show in the next section that the condition is also sufficient.

Lemma 2: The coding and allocation problem is feasible only if

$$\textit{min-sum}_{K-T}(V_1, V_2, \ldots, V_N) \geq B, \tag{9}$$

where $\textit{min-sum}_n(V_1, V_2, \ldots)$ *represents the sum of the n smallest terms of the given sequence,* $V_1, V_2, \ldots$.

Proof: As shown in the proof of Theorem 1, (7) is a necessary condition to be met. In other words, for all $\mathcal{T} \subset_T \mathcal{S} \subset_K \mathcal{N}$, it requires that

$$\sum_{i\in\mathcal{S}\setminus\mathcal{T}} n_i \geq B.$$

In particular, choose $\mathcal{S}$ as the K CSPs with smallest V_i's and choose $\mathcal{T}$ as the T CSPs in $\mathcal{S}$ with largest V_i's. Due to the budget requirement in (1), feasible solution exists only if the inequality in (9) holds. □

IV. A Coding Scheme Providing Perfect Secrecy

In this section, we construct a coding scheme based on the *nested maximum-distance-separable (MDS) code* to provide perfect secrecy for the system and obtain the corresponding storage cost.

In [19], the authors proposed the classical *wiretap channel II*. In wiretap channel II, k data bits are encoded into $n > k$ bits and transmitted to the legitimate receiver without loss. An eavesdropper can observe an arbitrary subset of μ bits. The maximum amount of data that can be sent without any leakage of information to the eavesdropper is called the *secrecy capacity* of the channel. Recall that an (n, k) MDS code is a class of linear codes that meet the Singleton bound, i.e., the minimum Hamming distance of the code, $d_{\min}$, equals $n - k + 1$. The generator matrix of an (n, k) MDS code has the property that all its $k \times k$ submatrices are full rank. A coding scheme based on the cosets of MDS codes was shown be able to achieve the secrecy capacity of wiretap channel II [19].

A modified version of wiretap channel II is the *erasure-erasure wiretap channel* introduced in [20], in which the bits transmitted to the legitimate receiver also experience erasures. In an erasure-erasure wiretap channel with parameters (θ, M, μ), the transmitter sends out θ symbols and the legitimate receiver and the eavesdropper receive M and μ symbols respectively. The secrecy capacity of the erasure-erasure wiretap channel can be achieved by *nested MDS code* [20], [24]. We present the definition of nested MDS code following the lines in [24]:

Definition 3: If the generator matrix G of an (n, k) MDS code can be written as $G = \begin{bmatrix} G_1 \\ G_2 \end{bmatrix}$, *where G_1 itself is a generator matrix of an (n, k_0) MDS code, then we refer to the code defined by G as nested MDS code with parameters (n, k, k_0).*

According to [20] and [24], the secrecy capacity of the erasure-erasure wiretap channel with parameters (θ, M, μ) is equal to $M - \mu$, which can be achieved by a nested MDS code with parameters (θ, M, μ).

Take an example with finite field size $q = 7$,

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

is a nested MDS code with parameters $(5, 4, 2)$ since it is a generator matrix of a $(5, 4)$ MDS code and its sub-matrix

$$G_1 = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 6 \end{bmatrix}$$

is a generator matrix of a $(5, 2)$ MDS code. The secrecy capacity of erasure-erasure wiretap channel with parameters $(5, 4, 2)$ can be achieved by encoding as

$$[y_1, y_2, y_3, y_4, y_5] = [k_1, k_2, x_1, x_2]G, \tag{10}$$

where x_1, x_2 are the messages and k_1, k_2 are the random keys, which are i.i.d and independent from source messages. The encoded symbols are $k_1+2k_2+x_1$, $2k_1+3k_2+2x_2$, $3k_1+4k_2+2x_2$, $4k_1+5k_2+x_2$, $5k_1+6k_2+5x_1+x_2$. In the erasure-erasure wiretap channel $(5, 4, 2)$, the transmitter sends five encoded symbols, the legitimate receiver receives four of them and the eavesdropper receives two of them. The legitimate receiver can decode the messages since any 4×4 submatrix of G is full rank. The eavesdropper gets no information about x_1 or x_2 since it cannot eliminate the random keys.

In general, an (n, k, k_0) nested MDS code can be constructed by the use of a $k \times n$ Vandermonde matrix, whose (i, j)-th element is given by α_i^{j-1} where $i = 1, \ldots, k$, $j = 1, \ldots, n$, and all α_i's shall be distinct. Alternatively, one may use Cauchy matrix, whose (i, j)-th element is given by $1/(x_i - x_j)$ and $x_i \neq x_j$. Therefore, nested MDS code exists for any values of (n, k, k_0), provided that the field size is large enough. In our multiple cloud storage system, we can use it to achieve the following storage cost:

Theorem 4: Suppose (9) *holds. The following storage cost is achievable by the use of nested MDS code:*

$$C^* \triangleq \min \sum_{i=1}^{N} C_i n_i, \tag{11}$$

where n_i's are integers subject to

$$0 \leq n_i \leq V_i, \quad \textit{for } i = 1, 2, \ldots, N. \tag{12}$$

$$\min_{\mathcal{S}\subset_K\mathcal{N}, \mathcal{T}\subset_T\mathcal{N}} \left(\sum_{i\in\mathcal{S}} n_i - \sum_{i\in\mathcal{T}} n_i\right) \geq B. \tag{13}$$

Proof: Since (9) holds, the feasible set as defined by (12) and (13) is non-empty. To see this, we can simply let $n_i = V_i$ for all i. This setting of n_i's satisfies constraints (12) and (13) simultaneously, and thus is a feasible solution. Therefore, optimal solution exists.

Let $(n_1^*, n_2^*, \ldots, n_N^*)$ be an optimal solution to the minimization problem (11). Define $n^* \triangleq \sum_{i=1}^N n_i^*$,

$$\nu \triangleq \min_{\mathcal{S}\subset_K\mathcal{N}} \sum_{\mathcal{S}} n_i^*, \text{ and } \mu \triangleq \max_{\mathcal{T}\subset_T\mathcal{N}} \sum_{\mathcal{T}} n_i^*.$$

The user then generates a random key $\boldsymbol{k} = (k_1, k_2, \ldots, k_\mu)$, in which the k_i's are uniformly and independently distributed over the finite field $\mathbb{F}_q$. The encoder output is defined by

$$\boldsymbol{y} = \begin{bmatrix} \boldsymbol{k} & \boldsymbol{x} \end{bmatrix} \begin{bmatrix} G_k \\ G_x \end{bmatrix}, \tag{14}$$

where $G \triangleq \begin{bmatrix} G_k \\ G_x \end{bmatrix}$ is the generator matrix of a nested MDS code with parameter (n^*, ν, μ).

Since the user is allowed to retrieve the file through any K CSPs, the channel can be seen as a collection of erasure-erasure wiretap channels with parameters (n^*, ν', μ'), where

$$\nu' \in \left\{ \sum_{i\in\mathcal{S}} n_i^* : \mathcal{S} \subset_K \mathcal{N} \right\} \text{ and } \mu' \in \left\{ \sum_{i\in\mathcal{T}} n_i^* : \mathcal{T} \subset_T \mathcal{N} \right\}.$$

Among all these channels, the worst one is the case where the user receives the least number of blocks while the eavesdropper observes the most number of blocks, that is, the channel with parameters (n^*, ν, μ). If the secrecy capacity of this channel is no less than B, i.e.,

$$\nu - \mu \geq B, \tag{15}$$

then the code generated by (14) can store B blocks of data with perfect secrecy. It is obvious that constraints (13) and (15) are equivalent. Constraint (12) ensures that the allocated numbers of data blocks satisfy the budget requirement. □

Remark 1: (13) is different from (7) in that $\mathcal{T}$ is not necessarily a subset of $\mathcal{S}$. As a result, $C^* \geq C_{LB}$. We will prove in the next section that equality indeed holds.

Remark 2: The minimization problem in Theorem 4 can be solved by integer linear programming (ILP). The number of constraints in (13) is

$$\binom{N}{K}\binom{N}{T} = \frac{(N!)^2}{K!(N-K)!T!(N-T)!},$$

which can be very large. For example, when $N = 15$, $K = 10$, and $T = 4$, the above value equals $4,099,095$. Later in this paper, we will present an efficient way to solve the problem.

Corollary 5: The coding and allocation problem is feasible if and only if (9) *holds.*

Proof: The statement follows directly from Lemma 2 and Theorem 4. □

V. Optimality

In Theorems 1 and 4, we provide a lower bound on the storage cost C_{LB} and an achievable storage cost by nested MDS code C^* respectively. Since (13) is different from (7) in that $\mathcal{T}$ is not necessarily a subset of $\mathcal{S}$, we have $C^* \geq C_{LB}$. In this section, we prove that nested MDS code as described in the last section is optimal as its achievable storage cost, C^*, is equal to C_{LB}.

For any non-negative integer sequence $n_1, n_2, \ldots, n_N$, define its decreasing ordered sequence by $n_{(1)}, n_{(2)}, \ldots, n_{(N)}$. In other words,

$$n_{(1)} \geq n_{(2)} \geq \cdots \geq n_{(N)}. \tag{16}$$

The following theorem gives the format of the optimal allocation parameters.

Theorem 6: The solution to the minimization problem in Theorem 4, if exists, must have the following form:

$$n_{(1)} = n_{(2)} = \cdots = n_{(N-K+T+1)}. \tag{17}$$

Furthermore,

$$n_{(N-K+T+1)} + n_{(N-K+T+2)} + \cdots + n_{(N)} = B. \tag{18}$$

Proof: Firstly, it is easy to see that any non-negative integer numbers $n_1, n_2, \ldots, n_N$ satisfy (13), if and only if the decreasing ordered sequence $n_{(1)}, n_{(2)}, \ldots, n_{(N)}$ satisfy

$$(n_{(N-K+1)} + \cdots + n_{(N)}) - (n_{(1)} + \cdots + n_{(T)}) \geq B. \tag{19}$$

In other words, (13) and (19) are equivalent.

Secondly, we show that the solution to the minimization problem must observe (17). Consider some given allocation parameters $n_1, n_2, \ldots, n_N$ that satisfying (12) and (13). Let $n_{(1)}, n_{(2)}, \ldots, n_{(N)}$ be the corresponding ordered sequence. We can do the following manipulations until (17) is satisfied, on the allocation parameters or equivalently the corresponding ordered sequence, to reduce the cost.

For $i < T$, if $n_{(i)} > n_{(i+1)}$, we can reduce the value of $n_{(i)}$ without violating (12) and (19).

For $T \leq i \leq N-K+T$ (Note that $T < N-K+T$ because $N > K$), if there exists $n_{(i)} > n_{(i+1)}$, let $\delta \triangleq n_{(i)} - n_{(i+1)}$. We can subtract δ from each of $n_{(1)}, n_{(2)}, \ldots, n_{(i)}$. When doing the substraction, since there are T terms being subtracted in the second bracket of (19), the value in the second bracket is reduced by $T\delta$. The value reduced in the first bracket of (19) depends on the value of i. It is

$$\begin{cases} 0, & \text{if } i < N-K+1 \\ (i-(N-K+1)+1)\,\delta, & \text{if } N-K+1 \leq i \end{cases}$$

Since the maximum value of i is $N-K+T$, the value reduced in the first bracket is no more than $T\delta$. So subtracting δ from each of $n_{(1)}, n_{(2)}, \ldots, n_{(i)}$ does not violate (19).

In summary, for any $n_1, n_2, \ldots, n_N$, we can always reduce their values without violating the constraints (12) and (19) until the corresponding ordered sequence $n_{(1)}, n_{(2)}, \ldots, n_{(N)}$ satisfying (17).

Finally, with (17), since $K > T$, it can be seen that (19) is equivalent to

$$n_{(N-K+T+1)} + n_{(N-K+T+2)} \cdots + n_{(N)} \geq B. \tag{20}$$

We can further reduce the value of $n_{(j)}$ for $j \geq N-K+T+1$ until (18) holds.

The ordered sequence after the above manipulations keeps satisfying the ordering (16), and constraints (12) and (13).

Since subtractions can reduce the storage cost, we can see that the optimal allocation parameters must satisfy (17) and (18). □

Based on the lower bound of the cost given in Section III and Theorem 6, we can show the optimality of the nested MDS coding:

Theorem 7: Suppose the coding and allocation problem is feasible. Nested MDS code achieves the minimum cost, i.e., $C^* = C_{LB}$.

Proof: By Theorem 6, the optimal allocation parameters satisfy (17) and (18). With these two criteria, (19) must be satisfied and (13) becomes redundant. Hence, in the minimization problem in Theorem 4, (13) can be replaced by (17) and (18).

The same analysis can be applied to the minimization problem in Theorem 1. Denote a set of non-negative integers satisfying (6) and (7) by $n_1^{LB}, \ldots, n_N^{LB}$ and the corresponding decreasing ordered sequence by $n_{(1)}^{LB}, \ldots, n_{(N)}^{LB}$. For the ordered sequence, (7) can be replaced by

$$\left(n_{(N-K+1)}^{LB} + \cdots + n_{(N)}^{LB}\right) - \left(n_{(1)}^{LB} + \cdots + n_{(T)}^{LB}\right) \geq B.$$

Then we can reduce the value of $n_{(1)}^{LB}, \ldots, n_{(N-K+T)}^{LB}$ until

$$n_{(1)}^{LB} = \cdots = n_{(N-K+T+1)}^{LB} \tag{21}$$

and reduce the value of $n_{(j)}$'s for $j \geq N-K+T+1$ until

$$n_{(N-K+T+1)}^{LB} + n_{(N-K+T+2)}^{LB} \cdots + n_{(N)}^{LB} = B. \tag{22}$$

without violating the decreasing order, and constraints (6) (7). Since reducing the values of $n_{(i)}^{LB}$'s will only reduce the total cost, we can see that the solution to the problem in Theorem 1 must have the form (21) and (22). In the minimization problem in Theorem 1, (13) can be replaced by (21) and (22). Hence, $C^* = C_{LB}$. □

VI. Properties of the Optimal Solution

In this section, we will continue to analyze the properties of the optimal solution. This enables us to construct an efficient algorithm in the next section to perform coding and allocation.

According to Theorem 6, the $N-K+T+1$ largest values of the allocation parameters must be equal and we denote this value by x. Recall that we use $n_{(1)}, n_{(2)}, \ldots, n_{(N)}$ to denote the ordered sequence of allocation parameters. With slight abuse of notation, we use (i) to denote the index of the corresponding CSP, i.e. $j = (i)$ if the amount of data blocks stored in CSP j is in the i-th position in the ordered list. Furthermore, $V_{(i)}$ is defined as the budget constraint on that CSP, i.e., $n_{(i)} \leq V_{(i)}$. According to the solution form in Theorem 6, we partition all the CSPs into two sets:

$$\mathcal{N}_1 \triangleq \{(1), (2), \ldots, (N-K+T+1)\}$$

and

$$\mathcal{N}_2 \triangleq \{(N-K+T+2), \ldots, (N)\}.$$

Now we consider the $K-T-1$ CSPs in $\mathcal{N}_2$ a step further. Label the indices of these $K-T-1$ CSPs by $[1], [2], \ldots, [K-T-1]$ according to their storage costs ascendingly, i.e.,

$$C_{[1]} < C_{[2]} < \cdots < C_{[K-T-1]}.$$

Note that by the above definition, we have $\mathcal{N}_2 = \{[1], [2], \ldots, [K-T-1]\}$.

By definition of x and by (17), $n_{(N-K+T+1)} = x$. Because of (18), we have $x \leq B$ and (18) becomes

$$n_{(N-K+T+2)} + \cdots + n_{(N)} = B - x \geq 0. \tag{23}$$

To minimize the storage cost, it is clear that one would put more coded blocks of data to CSP [1], and then CSP [2], and so on. Due to the constraints on ordering in (16), the budget constraint in (12), and the sum constraint in (23), to maximize the number of blocks stored into CSP [1], we have

$$n_{[1]} = \min\{V_{[1]}, x, B-x\}. \tag{24}$$

Similarly, for CSP indexed by [2] to $[K-T-1]$, we have

$$n_{[2]} = \min\{V_{[2]}, x, B-x-n_{[1]}\}, \tag{25}$$

$$\vdots$$

$$n_{[K-T-1]} = \min\{V_{[K-T-1]}, x, B-x-\sum_{i=1}^{K-T-2} n_{[i]}\}. \tag{26}$$

Define

$$y'_{[i]} \triangleq \begin{cases} B-x, & i=1, \\ B-x-\sum_{j=1}^{i-1} n_{[j]}, & 2 \leq i \leq K-T-1. \end{cases}$$

From (23), there must exist one CSP indexed by $[i^*], 1 \leq i^* \leq K-T-1$ stores at value $y'_{[i^*]}$, and the number of blocks assigned to CSP indexed by $[j]$ for $j > i^*$ must be 0. For $j < i^*$, the number of blocks to be stored should be constrained by either x or $V_{[j]}$. In summary, for CSPs in $\mathcal{N}_2$, we have

$$n_{[j]} = \begin{cases} \min\{x, V_{[j]}\}, & j < i^*, \\ y'_{[j]}, & j = i^*, \\ 0, & j > i^*, \end{cases} \tag{27}$$

where $1 \leq i^* \leq K-T-1$.

The above analysis shows the storage allocation in $\mathcal{N}_2$. Note that in (27), $y'_{[j]}$ can be equal to x or $V_{[j]}$. To fulfill further calculation, we need to do some definitions. Let l be the first index in the sequence $\mathcal{N}_2 = \{[1], [2], \ldots, [K-T-1]\}$ such that the number of blocks stored in the corresponding CSP is strictly smaller than x and its budget, i.e.,

$$\{l\} \triangleq \begin{cases} \emptyset, \text{ if } n_{[j]} = \min\{x, V_{[j]}\} \text{ for all } j \in \mathcal{N}_2 \\ \{[j^*]\}, \text{ where } j^* = \arg\min_j \{n_{[j]} < \min\{x, V_{[j]}\}\}. \end{cases} \tag{28}$$

Denote the number of blocks stored in CSP l by y, i.e., $n_l = y$. By definition, we have

$$0 \leq y < \min\{x, V_l\}. \tag{29}$$

We further define two subsets of $\mathcal{N}_2$ as follows: First, denote the subset of CSPs who store nothing, except l, by $\mathcal{Z}$, i.e.,

$$\mathcal{Z} \triangleq \{[j] \in \mathcal{N}_2 \setminus \{l\} : n_{[j]} = 0\}.$$

Second, denote the subset of CSPs whose storage budget is fully used as $\mathcal{B}$, i.e.,

$$\mathcal{B} \triangleq \{[j] \in \mathcal{N}_2 : n_{[j]} = V_{[j]} < x\}.$$

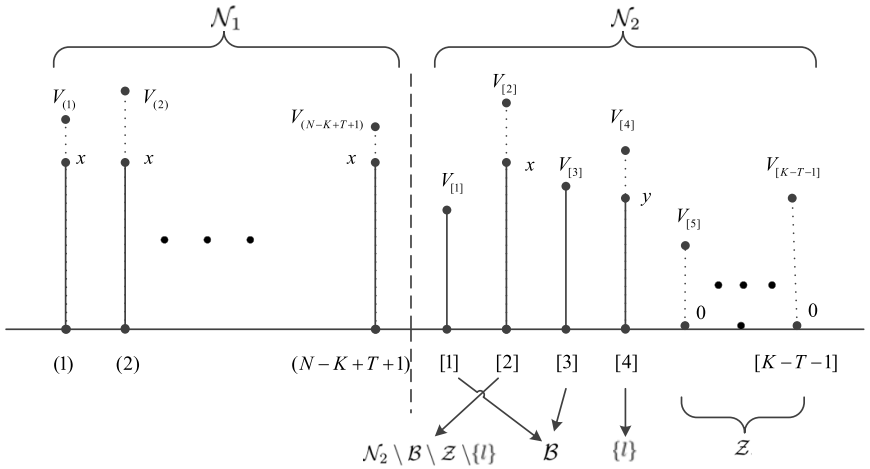

Fig. 2. An example of the allocation parameters.

Then within all the CSPs $\mathcal{N}$, the set of CSPs store at value x can be expressed as

$$\mathcal{X} \triangleq \mathcal{N} \setminus \mathcal{Z} \setminus \{l\} \setminus \mathcal{B}.$$

By definition, we have

$$\max\{V_i, i \in \mathcal{B}\} < x \leq \min\{V_i, i \in \mathcal{X}\}. \tag{30}$$

An illustration of the allocation parameters is shown in Fig. 2. Solid lines represent the solution while dashed lines represent the budgets of the CSPs.

Furthermore, let $z \triangleq |\mathcal{Z}|$ and $b \triangleq |\mathcal{B}|$. The following result specifies the form of the optimal solution:

Theorem 8: The solution to the minimization problem in Theorem 4, if exists, must have the following form:

a) $\mathcal{Z}$ contains the z most expensive CSPs in $\mathcal{N}$;

b) If $\{l\} \neq \emptyset$, CSP l is the most expensive CSP in $\mathcal{N} \setminus \mathcal{Z}$;

c) $\mathcal{B}$ contains the b CSPs with smallest budgets in $\mathcal{N} \setminus \mathcal{Z} \setminus \{l\}$.

Since the proof is tedious, we put it in the Appendix.

Note that the amount of blocks stored in a CSP can be divided into four classes. Either nothing, an amount of x, an amount of y, or an amount equal to the budget is stored. In the special case when there is no budget requirement, then $\mathcal{B}$ becomes empty, and the solution reduces to the three-level format in [30]. No matter the budget constraints exist or not, y, which is the amount stored in CSP l, is actually an artifact due to the indivisibility of $B - \sum_{i\in\mathcal{B}} V_i$ by $K-T-z-b-1$, which can be seen from (32) in the next section. Should x and y be allowed to be real numbers, the solution becomes two-leveled if without budget constraint. It means that coded blocks of data should be allocated equally among a certain subset of cheaper CSPs.

VII. Allocation Parameters Calculation, Encoding and Allocation

Based on the properties of the solution revealed in Theorems 6 and 8, we can develop an efficient algorithm to solve the minimization problem and obtain the optimal allocation parameters. After that, we can determine the encoding parameters and finish the encoding and allocation.

A. Allocation Parameters Calculation

According to Theorem 8, there are z CSPs which are not used to store data and b CSPs which use up all their storage budgets. We first consider the case when $\{l\} \neq \emptyset$. In this case, b and z should be non-negative integers satisfying

$$b + z + 1 \leq |\mathcal{N}_2| = K - T - 1.$$

Since the CSPs who are not in use must be the z most expensive CSPs and the CSPs who use up all their budgets must be the b CSPs with smallest budgets in $\mathcal{N} \setminus \mathcal{B} \setminus \{l\}$, we can search the values of z and b through a two-dimensional search.

We first re-arrange the CSPs according to their prices such that $C_1 < C_2 < \cdots < C_N$. Then

$$\mathcal{Z} = \{N - z + 1, N - z + 2, \ldots, N\}$$

and

$$l = N - z.$$

For CSPs in the set $\mathcal{N} \setminus \mathcal{Z} \setminus \{l\} = \{1, 2, \ldots, N-z-1\}$, sort V_i's in increasing order. Recall that $\mathcal{B}$ is the set of b CSPs with smallest budgets in $\mathcal{N} \setminus \mathcal{Z} \setminus \{l\} = \{1, 2, \ldots, N - z - 1\}$. For any given z and b, we can calculate the storage cost by

$$C_{z,b} = \min_{x,y} \left(\sum_{i\in\mathcal{B}} C_i V_i + C_l y + \sum_{i\in\mathcal{X}} C_i x \right), \tag{31}$$

where x and y are non-negative integers subject to

$$(K - T - z - b - 2)x + y + \sum_{i\in\mathcal{B}} V_i = B - x, \tag{32}$$

(30), and (29). The above constraint (32) is from (23). Due to (32), we can re-write (31) as

$$C_{z,b} = \sum_{i\in\mathcal{B}} C_i V_i + C_l (B - \sum_{i\in\mathcal{B}} V_i) + \min_x \left(\sum_{i\in\mathcal{X}} C_i - C_l (K - T - 1 - b - z) \right) x.$$

From (32), (30), and (29), if $x \le V_l$ we have

$$\left\lceil \frac{B - \sum_{i\in\mathcal{B}} V_i + 1}{K - T - b - z} \right\rceil \le x \le \left\lfloor \frac{B - \sum_{i\in\mathcal{B}} V_i}{K - T - 1 - b - z} \right\rfloor,$$

and if $x \ge V_l + 1$, we have

$$\left\lceil \frac{B - \sum_{i\in\mathcal{B}} V_i + 1 - V_l}{K - T - b - z - 1} \right\rceil \le x \le \left\lfloor \frac{B - \sum_{i\in\mathcal{B}} V_i}{K - T - 1 - b - z} \right\rfloor.$$

After checking the feasibility range of x, we can get the maximum value and minimum value of x. To obtain $C_{z,b}$, if

$$\sum_{i\in\mathcal{X}} C_i - C_l(K - T - 1 - b - z) < 0,$$

optimal x should be the maximum value and otherwise, optimal x should be its minimum value. So given the values of z and b, $C_{z,b}$ can be determined by calculating the value of x. By comparing $C_{z,b}$ for all possible choices of z and b, we can obtain smallest cost for the case when $\{l\} \neq \emptyset$, i.e., $\min_{z,b} C_{z,b}$.

Now we consider the second case when $\{l\} = \emptyset$. According to the definition of $\{l\}$ in (28), if $\{l\} = \emptyset$, then the amount of allocation for all CSPs in $\mathcal{N}_2$ are all non-zero. Therefore, we can set $z = 0$ in this case. The problem reduces to a one-dimensional search for the value of b over the range $0 \le b \le K - T - 1$. The storage cost becomes

$$C'_b = \min_x \left(\sum_{i\in\mathcal{B}} C_i V_i + \sum_{i\in\mathcal{X}} C_i x \right),$$

where x should be a non-negative integer satisfying

$$(K - T - 1 - b)x + \sum_{i\in\mathcal{B}} V_i = B - x$$

and (30). If there is no feasible value of x, we set C'_b to be ∞. The smallest cost for the case when $\{l\} = \emptyset$ is $\min_b C'_b$.

Combining the two cases, we can find the minimum storage cost $C^* = \min\{\min_{z,b} C_{z,b}, \min_b C'_b\}$ and the optimal values of z and b.

Now we analyze the time complexity of calculating the allocation parameters. For the first case when $\{l\} \neq \emptyset$, given any value of z, we need to sort V_i's in increasing order with time complexity of $O(N \log N)$, and search through all possible choices of b with time complexity of $O(N)$. So the time complexity to find the solution for a given value of z is $O(N \log N + N) = O(N \log N)$. Then we need to find the minimum cost within the costs obtained by all possible values of z, with time complexity of $O(N)$. Thus $\min_{z,b} C_{z,b}$ can be determined with overall time complexity of $O(N^2 \log N)$. For the second case when $\{l\} = \emptyset$, since z is fixed to 0, the time complexity for determining $\min_b C'$ is $O(N \log N)$. Therefore, the whole minimization problem can be solved with an overall time complexity of $O(N^2 \log N)$.

B. Encoding and Allocation

After obtaining the allocation parameters $n_1, n_2, \ldots, n_N$, the encoding parameters n, ν and μ can be determined. Sort $n_1, n_2, \ldots, n_N$ in decreasing order and obtain $n_{(1)}, n_{(2)}, \ldots, n_{(N)}$. The encoding parameters are then given by

$$\begin{aligned} n &= n_1 + n_2 + \cdots + n_N, \\ \nu &= n_{(N-K+1)} + n_{(N-K+2)} + \cdots + n_{(N)}, \\ \mu &= n_{(1)} + n_{(2)} + \cdots + n_{(T)}. \end{aligned}$$

To encode the file, the generator matrix G of a nested MDS code with parameter (n, ν, μ) is needed. As mentioned before, G can be constructed by Vandermonde matrix or Cauchy matrix. It is also possible to design the nested MDS code based on special application requirements, such as small field size or sparse generator matrix.

Encoding can be done as follows:

$$\boldsymbol{y} = [\boldsymbol{k} \quad \boldsymbol{x}]\, G,$$

where $\boldsymbol{k} = (k_1, k_2, \ldots, k_\mu)$ is a vector of random keys. The encoding complexity of using Vandermonde matrix or Cauchy matrix is $O(n \log^2 n)$ [39].

The encoded vector $\boldsymbol{y}$ is then divided into $\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_N$ according to n_i's, which are stored into the N CSPs, respectively. The file, which has B blocks of data, is now securely stored and the storage cost is exactly the value of C^* obtained in the last subsection.

To retrieve the file, the user performs decoding by

$$\boldsymbol{x}' = \boldsymbol{y}_B G_B^{-1},$$

where $\boldsymbol{y}_B$ is any B-length vector obtained from $\boldsymbol{y}(\mathcal{S})$ and G_B is the corresponding $B \times B$ sub-matrix of G. The inverse of Vandermonde matrix or Cauchy matrix takes time complexity $O(n^3)$ or $O(n^2)$ [40], [41] and multiplication takes complexity $O(n \log^2 n)$. Thus the decoding complexity is $O(n^3)$ or $O(n^2)$.

VIII. Numerical Examples

In this section, we give some numerical examples on the allocation parameters. In the following examples, we assume that the file size is not large, so the whole file is regarded as a single chuck, which consists of B blocks of data. For a large file, it will be divided into smaller chucks first, and the encoding operations are applied to each chuck repeatedly. For each chunk, we determine the amount of encoded data to be stored in each CSP according to Section VII-A. After that, the code length is determined and the chunk can be encoded according to the method in Section VII-B. The encoded data for a CSP, commonly called a *share*, can then be stored in the CSPs.

Note that our system architecture is similar to that in [13], which presents also a practical software implementation of the multiple-cloud system. Our proposed system design can be implemented in a similar way as that in [13, Sec. 5]. On the other hand, the naming method for each share in our system is different from that in [13]. In our system, it is named by the index of the CSP to which it is stored and this information does not need to be encrypted, since the locations cannot reveal any information of the original file in our model. Besides, the indices of the CSPs to which the shares belong, the lengths of the shares, encoding parameters (n, ν, μ) and encoding matrix

TABLE I
STORAGE ALLOCATION PARAMETERS

CSP	C_i	V_i	$n_i(T=0)$	$n_i(T=1)$	$n_i(T=2)$
1	10	80	65	60	57
2	11	60	60	60	57
3	12	100	65	60	57
4	15	150	65	60	57
5	16	50	50	50	50
6	17	189	65	60	57
7	19	520	65	60	57
8	22	518	65	60	57
9	25	517	65	60	57
10	26	516	65	60	57
11	27	515	65	60	57
12	31	514	0	60	57
13	32	513	0	30	57
14	35	63	0	0	51
15	37	512	0	0	0

TABLE II
STORAGE ALLOCATION PARAMETERS

CSP	C_i	V_i	$n_i(T=1)$	$n_i(T=2)$	$n_i(T=4)$
1	10	101	75	57	65
2	11	102	75	57	65
3	12	100	75	57	65
4	15	150	75	57	65
5	16	50	50	50	50
6	17	189	75	57	65
7	19	520	75	57	65
8	22	518	75	57	65
9	25	517	75	57	65
10	26	516	75	57	65
11	27	515	75	57	65
12	31	514	0	57	65
13	32	513	0	57	65
14	35	63	0	51	63
15	37	512	0	0	62

are also stored in the metadata. For details of file uploading and downloading, we refer the readers to [13].

In the first example shown in Table I, we assume that there are $N = 15$ CSPs and they are listed according to their charges in increasing order. The charge of each CSP is shown in the second column. The third column shows the budgets of the CSPs. The user wants to store $B = 500$ blocks of data. It is required that the file retrieval can be done through any $K = 12$ CSPs, which means that the data is reliable even $N - K = 3$ CSPs are disconnected. Table I lists the allocation parameter on each CSP when the number of colluding CSPs, T, increases from 0 to 2.

When $T = 0$, all the CSPs are trustful. This corresponds to data allocation problem without security concern. We use MDS code $(695, 500)$ to encode. The optimal allocation is that CSPs 2 and 5 stores at their budgets while CSPs 12 to 15 are not used. This code and its allocation is sufficient for file reconstruction through any $K' = K - 4 = 8$ CSPs out of the $N' = N - 4 = 11$ CSPs that are in use. When $T = 1$, we use nested MDS code with parameters $(740, 560, 60)$ to encode. In this case, CSPs 2 and 5 reach their budgets while CSPs 14 and 15 are not used. This code and its allocation is sufficient for file reconstruction through any $K' = K - 2 = 10$ CSPs out of the $N' = N - 2 = 13$ CSPs that are in use. CSP 13 stores 30 blocks of data, which is strictly less than both its budget and the storage amount of other CSPs that are in use. This corresponds to the solution at which $l = 13$ and $y = 30$. When $T = 2$, we use nested MDS code with parameters $(785, 614, 114)$ to encode. In this case, only CSP 5 reaches its budget. Besides, fewer CSPs store nothing.

In Table I, it seems that when T increases, the encoded block length n increases and the storage amount in each CSP decreases. In general, however, they are not true. Consider another example, whose parameters are shown in Table II. It differs from the previous example in that the budgets of CSPs 1 and 2 are both increased to 101 and 102. When $T = 1$, we use nested MDS code with parameters $(800, 575, 75)$ to encode and when $T = 2$, we use nested MDS code with parameters $(785, 614, 114)$ to encode. The encoded block length reduces by 15. Furthermore, when T increases to 4, the number of allocated blocks on each CSP increases.

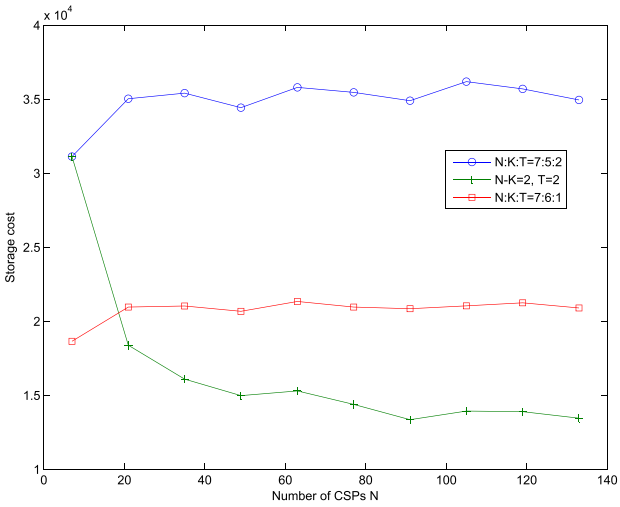

Fig. 3. Storage cost versus number of CSPs N.

From these two example, it can be seen that when T becomes large, more CSPs are used. This is because when we want to keep privacy against more colluding CSPs, we should disperse the data into more blocks and store them in a more scattering way.

Next we investigate the effect of N, K, and T on the storage cost. Let $B = 500$. The charge of each CSP is Gaussian distributed with mean 30 and variance 5, truncated between 10 and 50. The budget of each CSP is Gaussian distributed with mean 70 and variance 10. To ensure that there is a feasible solution, each budget is lower bounded by $\lceil \frac{500}{K-T} \rceil$. We can see in Fig. 3 that the storage cost is relatively steady when the ratio $N : K : T$ is constant. Comparing the case when the ratio is 7:5:2 with the other case when it is 7:6:1, it

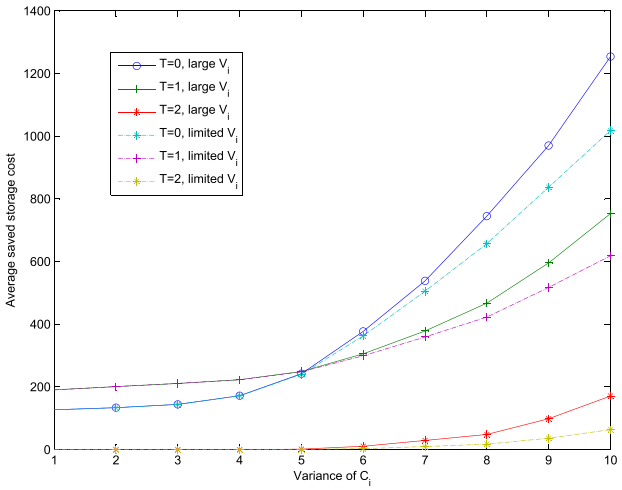

Fig. 4. Saved storage cost of optimal allocation.

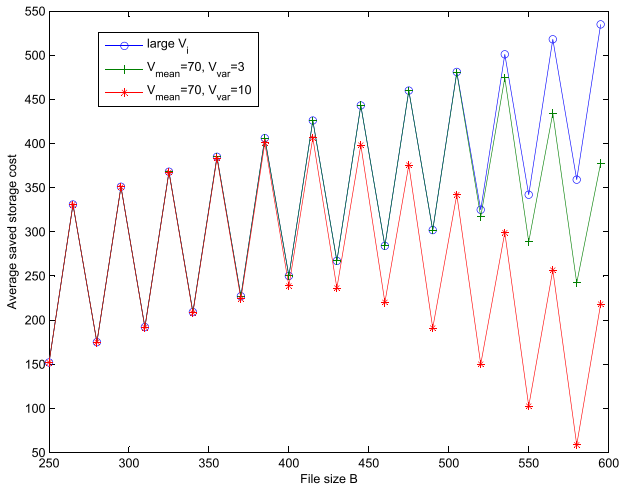

Fig. 5. Saved storage cost of optimal allocation.

can be seen that the storage cost is higher if it is required to retrieve the file from fewer CSPs or there are more colluding CSPs. When T and $N-K$ are fixed, the cost drops when N increases.

To see the benefit of storing unequal amount of data blocks in different CSPs, we compare the cost between equal allocation and optimal allocation. For equal allocation, if we want to satisfy the perfect secrecy and reconstruction requirements, the coding parameter should be $(n'N, n'K, n'T)$, where n' is the number of allocated data on each CSP. Like (15), we have

$$n'(K-T) \geq B.$$

So for equal allocation, the number of data allocated on each CSP should be

$$n' = \left\lceil \frac{B}{K-T} \right\rceil,$$

and the budget on every CSP should be large enough, i.e., $V_i \geq n' = \lceil \frac{B}{K-T} \rceil$ for $i \in \mathcal{N}$.

In the example shown in Fig. 4, we show the comparison of costs between equal allocation and optimal allocation, with $N = 15$ and $K = 12$. The solid lines correspond to the cases when the budgets are set to be very large, so that there is essentially no maximum constraint on the storage amount of each CSP. The dashed lines correspond to the cases when the budgets of the CSPs are the same as those in Table I. The prices of the CSPs are Gaussian distributed with mean 30 and variance changes from 1 to 10, truncated within the range $[10, 50]$. We plot the the saved storage cost against the variance of the price of each CSP, where the saved storage cost is defined as the cost difference between equal allocation and optimal allocation. For every given variance, we run the program 200 times to get the average saved storage cost. It can be seen that optimal allocation has larger saving when the variance of C_i increases. Furthermore, when T decreases or the budgets increase, the saved cost also increases. This is because in these cases, the range of feasible allocations grows and optimal allocation can select the best allocation within a larger set of feasible allocations.

In Fig. 5, we show the saved storage cost against the file size, B. In this example we set $N = 15$, $K = 12$, and $T = 2$. The CSPs' prices are the same as those in Table I. We plot the saved storage cost under three different settings of V_i: (i) the values of V_i's are very large so that the storage amount of each CSP has essentially no constraint, (ii) Gaussian distributed with mean 70 and variance 3, (iii) Gaussian distributed with mean 70 and variance 10. To ensure a non-empty feasible solution set, the budget of each CSP is lower bounded by $\lceil \frac{B}{K-T} \rceil$. For each setting of V_i, we run 100 times to get the average saved storage cost. From the figure, it can be seen that the curves exhibit a sawtooth structure, which is due to the integer constrains of n_i's. Despite this, we can still see the trends with the growth of B. When V_i is large, the growth is linear, since the amount of saving per unit size of the file should be constant. On the other hand, when the budget is limited as in the other two cases, the saved cost eventually drops when B exceeds a certain threshold. The reason is that the optimal allocation amount of some cheaper CSPs reach their budgets, thus reducing the cost difference between optimal allocation and equal allocation. Furthermore, when the variance of V_i is large, CSPs with small budgets becomes more, the benefits of optimal allocation is confined by these CSPs with small budgets.

IX. Conclusion

In this paper, we investigate the minimization of storage cost when the user stores its data in multiple untrustful and unreliable clouds. We give a lower bound on the cost and present a coding scheme that can achieve this bound. This optimal scheme can be solved through two-dimensional search, which has very low computational complexity. Since the computational complexity is low, the result is also applicable to large-scale storage systems.

While this work considers only data confidentiality and availability, it is also interesting to address the issue of data integrity in multiple clouds. A user should be able to detect any changes in data that may occur as a result of transmission errors during file uploading or alteration by

unauthorized persons. How to simultaneously ensure data confidentiality, availability and integrity by coding and allocation with minimum cost is a challenging problem to be tackled in the future.

APPENDIX

The proof of Theorem 8 is shown below:

Proof: We first consider the CSPs in $\mathcal{Z}$. From (27), the CSPs in set $\mathcal{Z}$ must be the z most expensive CSPs in set $\mathcal{N}_2$, i.e.,

$$\min\{C_i, i \in \mathcal{Z}\} > \max\{C_i, i \in \mathcal{N}_2 \setminus \mathcal{Z}\}. \tag{33}$$

We claim that any CSP in $\mathcal{Z}$ must be more expensive than all the CSPs in $\mathcal{N}_1$:

$$\min\{C_i, i \in \mathcal{Z}\} > \max\{C_i, i \in \mathcal{N}_1\}. \tag{34}$$

Let

$$n_j = \begin{cases} x, & j \in \mathcal{N}_1, \\ x, & j \in \mathcal{N}_2 \setminus \mathcal{B} \setminus \mathcal{Z} \setminus \{l\}, \\ y, & j = l, \\ V_j, & j \in \mathcal{B}, \\ 0, & j \in \mathcal{Z}, \end{cases} \tag{35}$$

be the optimal solution. Denote the most expensive CSP in $\mathcal{N}_1$ by t. If there exists one CSP indexed by $p \in \mathcal{Z}$ satisfy

$$C_p < C_t,$$

so that (34) does not hold, we can always find another feasible solution leading to a lower cost.

For the trivial case $x = 1$, we have $n_t = 1$. It is easy to see that since $V_i \geq 1$ for all $i \in \mathcal{N}$, we can reduce the cost by exchange the value of n_p and n_t without violating the budgets.

For the case $x \geq 2$, if $\mathcal{N}_2 \setminus \mathcal{B} \setminus \mathcal{Z} \setminus \{l\} = \emptyset$, we can assign new allocation parameters as

$$n_j = \begin{cases} x - 1, & j \in \mathcal{N}_1, \\ y, & j = l, \\ V_j, & j \in \mathcal{B}, \\ 1, & j = p, \\ 0, & j \in \mathcal{Z} \setminus \{p\}. \end{cases} \tag{36}$$

Since $y < x$ and $V_j < x$ for $j \in \mathcal{B}$, we have $x - 1 \geq y$ and $x - 1 \geq V_j$ for $j \in \mathcal{B}$. By re-ordering the CSPs in $\mathcal{N}_2$ according to the newly assigned parameters, we can see that the re-ordered newly assigned allocation parameters satisfy (16), (17) and (18), and hence they satisfy (19). Since any non-negative integer numbers $n_1, n_2, \ldots, n_N$ satisfy (13), if and only if the ordered sequence $n_{(1)}, n_{(2)}, \ldots, n_{(N)}$ satisfy (20), we can see n_j's in (36) satisfy (19). Furthermore, they all satisfy (12). So it is a feasible solution of the minimization problem in Theorem 4. The new allocation parameters result in a lower cost because the cost change

$$\begin{aligned} C_\triangle &= -\sum_{i=(1)}^{(N-K+T+1)} C_i + C_p \\ &\leq -C_t + C_p \\ &< 0. \end{aligned}$$

If $\mathcal{N}_2 \setminus \mathcal{B} \setminus \mathcal{Z} \setminus \{l\} \neq \emptyset$, we can assign new allocation parameters as

$$n_p = 1, \quad n_t = x - 1,$$

and then exchange the position of t and any one of CSP in set $\mathcal{N}_2 \setminus \mathcal{B} \setminus \mathcal{Z} \setminus \{l\}$. The exchange will not violate the budgets since $V_t \geq x$. Thus the CSPs in $\mathcal{N}_1$ store the same amount of data x. Again, by re-ordering the newly assigned parameters in $\mathcal{N}_2$, we can see they satisfy (16), (17) and (18). Similarly as above, it is also a feasible solution and the cost change is also less than 0.

The above results contradict with the assumption that (35) is the optimal solution. Thus (34) must hold. Combining (33) and (34), we can see that the CSPs in set $\mathcal{Z}$ must be the z most expensive CSPs in $\mathcal{N}$.

For CSP l, we have $y + 1 \leq V_l$ and $y + 1 \leq x$ by definition of l. Similar to the case when we consider $\mathcal{Z}$, if $x = y + 1$, we can exchange n_t and n_l without violating the budgets. If $x \geq y + 2$, similar to the case when we increase $n_p = 0$ to $n_p = 1$ when considering $\mathcal{Z}$, we can increase y to $y + 1$, and the analysis is exactly the same. Thus l is the most expensive CSP in $\mathcal{N} \setminus \mathcal{Z}$.

Lastly, we consider the budgets of CSPs in set $\mathcal{B}$. From (30), we can see that the budget of any CSP in $\mathcal{B}$ is smaller than the budget of any CSP in $\mathcal{N} \setminus \mathcal{Z} \setminus \{l\} \setminus \mathcal{B}$. So $\mathcal{B}$ contains the b CSPs with smallest budgets in $\mathcal{N} \setminus \mathcal{Z} \setminus \{l\}$. □

REFERENCES

[1] M. Armbrust *et al.*, "A view of cloud computing," *Commun. ACM*, vol. 53, no. 4, pp. 50–58, Apr. 2010.

[2] M. Ali, S. U. Khan, and A. V. Vasilakos, "Security in cloud computing: Opportunities and challenges," *Inf. Sci.*, vol. 305, pp. 357–383, Jun. 2015.

[3] L. Wei, H. Zhu, Z. Cao, W. Jia, and A. V. Vasilakos, "SecCloud: Bridging secure storage and computation in cloud," in *Proc. IEEE 30th Int. Conf. Distrib. Comput. Syst. Workshops*, Jun. 2010, pp. 52–61.

[4] L. Wei *et al.*, "Security and privacy for storage and computation in cloud computing," *Inf. Sci.*, vol. 258, pp. 371–386, Feb. 2014.

[5] M. Ali *et al.*, "SeDaSC: Secure data sharing in clouds," *IEEE Syst. J.*, to be published, doi: 10.1109/JSYST.2014.2379646.

[6] X. Zhang *et al.*, "DFL: Secure and practical fault localization for datacenter networks," *IEEE/ACM Trans. Netw.*, vol. 22, no. 4, pp. 1218–1231, Aug. 2014.

[7] Z. M. Fadlullah, T. Taleb, A. V. Vasilakos, M. Guizani, and N. Kato, "DTRAB: Combating against attacks on encrypted protocols through traffic-feature analysis," *IEEE/ACM Trans. Netw.*, vol. 18, no. 4, pp. 1234–1247, Aug. 2010.

[8] Z. Yan, P. Zhang, and A. V. Vasilakos, "A survey on trust management for Internet of Things," *J. Netw. Comput. Appl.*, vol. 42, pp. 120–134, Jun. 2014.

[9] Q. Jing, A. V. Vasilakos, J. Wan, J. Lu, and D. Qiu, "Security of the Internet of Things: Perspectives and challenges," *Wireless Netw.*, vol. 20, no. 8, pp. 2481–2501, Nov. 2014.

[10] N. Xiong *et al.*, "Comparative analysis of quality of service and memory usage for adaptive failure detectors in healthcare systems," *IEEE J. Sel. Areas Commun.*, vol. 27, no. 4, pp. 495–509, May 2009.

[11] M. Van Dijk and A. Juels, "On the impossibility of cryptography alone for privacy-preserving cloud computing," in *Proc. 5th USENIX Conf. Hot Topics Secur.*, Oct. 2010, pp. 1–8.

[12] A. Bessani, M. Correia, B. Quaresma, F. André, and P. Sousa, "DepSky: Dependable and secure storage in a cloud-of-clouds," *ACM Trans. Storage*, vol. 9, no. 4, Nov. 2013, Art. ID 12.

[13] J. Y. Chung, C. Joe-Wong, S. Ha, J. W.-K. Hong, and M. Chiang, "CYRUS: Towards client-defined cloud storage," in *Proc. 10th EuroSys*, Bordeaux, France, Apr. 2015, Art. ID 17.

[14] A. Li, X. Yang, S. Kandula, and M. Zhang, "CloudCmp: Comparing public cloud providers," in *Proc. 10th ACM SIGCOMM Conf. Internet Meas.*, Melbourne, VIC, Australia, Nov. 2010, pp. 1–14.

[15] K. Krawczyk. (2013). *What is the Fastest Cloud Storage Service?*. [Online]. Available: http://blog.laptopmag.com/whats-the-fastest-cloud-storage-service

[16] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979.

[17] C. E. Shannon, "Communication theory of secrecy systems," *Bell Syst. Tech. J.*, vol. 28, no. 4, pp. 656–715, Oct. 1949.
[18] J. L. Massey, "An introduction to contemporary cryptology," *Proc. IEEE*, vol. 76, no. 5, pp. 533–549, May 1988.
[19] 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.
[20] A. Subramanian and S. McLaughlin. (2009). "MDS codes on erasure-erasure wire-tap channel." [Online]. Available: http://arxiv.org/abs/0902.3286
[21] A. Parakh and S. Kak, "A distributed data storage scheme for sensor networks," in *Security and Privacy in Mobile Information and Communication Systems*, vol. 17. Berlin, Germany: Springer-Verlag, 2009, pp. 14–22.
[22] 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.
[23] P. Hu, K. W. Shum, and C. W. Sung, "The fundamental theorem of distributed storage systems revisited," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Hobart, TAS, Australia, Nov. 2014, pp. 65–69.
[24] 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.
[25] 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.
[26] 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.
[27] P. F. Oliveira, L. Lima, T. T. V. Vinhoza, J. Barros, and M. Médard, "Coding for trusted storage in untrusted networks," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 6, pp. 1890–1898, Dec. 2012.
[28] K. Bhattad and K. R. Narayanan, "Weakly secure network coding," in *Proc. 1st Workshop Netw. Coding, Theory, Appl. (NetCod)*, Apr. 2005.
[29] Y. Singh, F. Kandah, and W. Zhang, "A secured cost-effective multi-cloud storage in cloud computing," in *Proc. IEEE Conf. Comput. Commun. Workshop*, Shanghai, China, Apr. 2011, pp. 619–624.
[30] P. Hu, C. W. Sung, S.-W. Ho, and T. H. Chan, "Three-level storage and nested MDS codes for perfect secrecy in multiple clouds," in *Proc. IEEE Int. Symp. Inf. Theory*, Honolulu, HI, USA, Jun./Jul. 2014, pp. 1356–1360.
[31] M. Naor and R. M. Roth, "Optimal file sharing in distributed networks," *SIAM J. Comput.*, vol. 24, no. 1, pp. 158–183, Feb. 1995.
[32] A. Jiang and J. Bruck, "Network file storage with graceful performance degradation," *ACM Trans. Storage*, vol. 1, no. 2, pp. 171–189, May 2005.
[33] D. Leong, A. G. Dimakis, and T. Ho, "Distributed storage allocations," *IEEE Trans. Inf. Theory*, vol. 58, no. 7, pp. 4733–4751, Jul. 2012.
[34] Q. Yu, K. W. Shum, and C. W. Sung, "Tradeoff between storage cost and repair cost in heterogeneous distributed storage systems," *Trans. Emerg. Telecommun. Technol.*, vol. 26, no. 10, pp. 1201–1211, Oct. 2015.
[35] Q. Yu, C. W. Sung, and T. H. Chan, "Irregular fractional repetition code optimization for heterogeneous cloud storage," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 5, pp. 1048–1060, May 2014.
[36] J. Kubiatowicz *et al.*, "OceanStore: An architecture for global-scale persistent storage," *ACM SIGPLAN Notices*, vol. 35, no. 11, pp. 190–201, Nov. 2000.
[37] N. Golrezaei, A. G. Dimakis, and A. F. Molisch, "Wireless device-to-device communications with distributed caching," in *Proc. IEEE Int. Symp. Inf. Theory*, Cambridge, MA, USA, Jul. 2012, pp. 2781–2785.
[38] S. Pawar, S. El Rouayheb, H. Zhang, K. Lee, and K. Ramchandran, "Codes for a distributed caching based video-on-demand system," in *Proc. Conf. Rec. 45th Asilomar Conf. Signals, Syst. Comput.*, Pacific Grove, CA, USA, Nov. 2011, pp. 1783–1787.
[39] I. Gohberg and V. Olshevsky, "Fast algorithms with preprocessing for matrix-vector multiplication problems," *J. Complex.*, vol. 10, no. 4, pp. 411–427, Dec. 1994.
[40] A. Eisinberg and G. Fedele, "On the inversion of the Vandermonde matrix," *Appl. Math. Comput.*, vol. 174, no. 2, pp. 1384–1397, Mar. 2006.
[41] J. Blömer, M. Kalfane, R. Karp, M. Karpinski, M. Luby, and D. Zuckerman, "An XOR-based erasure-resilient coding scheme," California Comput. Sci. Division, Int. Comput. Sci. Inst., Berkeley, CA, USA, Univ. California, Berkeley, CA, SA, Tech. Rep. TR-59-048, Aug. 1995.

Ping Hu received the B.Eng. degree in telecommunications engineering from Xidian University, Xi'an, China, in 2006, and the M.Phil. degree from the City University of Hong Kong, in 2009, where she is currently pursuing the Ph.D. degree with the Department of Electronic Engineering. She was with Alcatel-Lucent Shanghai Bell, Shanghai, China, from 2010 to 2012. Her research interests include wireless communication, information-theoretic security, and distributed storage systems.

Chi Wan Sung (M'98) received the B.Eng., M.Phil., and Ph.D. degrees from the Chinese University of Hong Kong, in 1993, 1995, and 1998, respectively, all in information engineering. He joined the City University of Hong Kong in 2000, as a Faculty Member, where he is currently an Associate Professor with the Department of Electronic Engineering. He is an Adjunct Associate Research Professor with the University of South Australia, and is on the Editorial Boards of the *ETRI Journal* and the *Transactions on Emerging Telecommunications Technologies*. His research interests include wireless communications, resource allocation, network coding, and cloud storage systems.

Siu-Wai Ho (S'05–M'07–SM'15) received the B.Eng., M.Phil., and Ph.D. degrees from The Chinese University of Hong Kong, in 2000, 2003, and 2006, respectively, all in information engineering. From 2006 to 2008, he was a Postdoctoral Research Fellow with the Department of Electrical Engineering, Princeton University, Princeton, NJ. Since 2009, he has been with the Institute for Telecommunications Research, University of South Australia (UniSA), Adelaide, Australia, where he is currently a Senior Research Fellow. His current research interests include Shannon theory, visible light communications, information-theoretic security, and biometric security systems.

Dr. Ho was a recipient of the Croucher Foundation Fellowship for 2006–2008, the 2008 Young Scientist Award from the Hong Kong Institution of Science, UniSA Research SA Fellowship for 2010–2013, and the Australian Research Council Australian Postdoctoral Fellowship for 2010–2013.

Terence H. Chan received the B.Sc.(Math.), master's, and Ph.D. degrees from The Chinese University of Hong Kong, in 1996, 1998, and 2000, respectively, all in information engineering. In 2001, he was a Visiting Assistant Professor with the Department of Information Engineering, The Chinese University of Hong Kong. From 2002 to 2004, he was a Postdoctoral Fellow with the Department of Electrical and Computer Engineering, University of Toronto. He was an Assistant Professor with the University of Regina from 2004 to 2006. He is currently an Associate Professor with the Institute for Telecommunications Research, University of South Australia. He received the Croucher Foundation Fellowship and Sir Edward Youde Fellowship, in 2002 and 2000, respectively. He is the IEEE Information Theory Society, Joint South Australia/ACT/VIC/NSW/QLD Sections Chapter Chair. He serves as the Technical Cochair for the 2011 and 2015 IEEE International Symposium on Network Coding.