Device-to-Device Secure Coded Caching

Ahmed A. Zewail, *Member, IEEE*, and Aylin Yener, *Fellow, IEEE*

***Abstract*— This paper studies device to device (D2D) coded-caching with information theoretic security guarantees. A broadcast network consisting of a server, which has a library of files, and end users equipped with cache memories, is considered. Information theoretic security guarantees for confidentiality are imposed upon the files. The server populates the end user caches, after which D2D communications enable the delivery of the requested files. Accordingly, we require that a user must not have access to files it did not request, i.e., secure caching. First, a centralized coded caching scheme is provided by jointly optimizing the cache placement and delivery policies. Next, a decentralized coded caching scheme is developed that does not require the knowledge of the number of active users during the caching phase. Both schemes utilize non-perfect secret sharing and one-time pad keying, to guarantee secure caching. Furthermore, the proposed schemes provide secure delivery as a side benefit, i.e., any external entity which overhears the transmitted signals during the delivery phase cannot obtain any information about the database files. The proposed schemes provide the achievable upper bound on the minimum delivery sum rate. Lower bounds on the required transmission sum rate are also derived using cut-set arguments indicating the multiplicative gap between the lower and upper bounds. Numerical results indicate that the gap vanishes with increasing memory size. Overall, the work demonstrates the effectiveness of D2D communications in cache-aided systems even when confidentiality constraints are imposed at the participating nodes and against external eavesdroppers.**

***Index Terms*— Device-to-device communications, coded caching, secure caching, secure delivery, secret sharing.**

I. Introduction

OVER the past decade, wireless communication systems have transformed from being limited to serving low data rates, e.g., voice calls and text messages, to offering dependable high data rate services, notably, for video content. The demand in massive amounts of data will only increase going forward, leading to potential bottlenecks. Two potential solutions offered towards alleviating network congestion are *device-to-device (D2D) communications* and *caching*. The former shifts some of the traffic load from the core network to the end users, while the later shifts it from the peak to off-peak hours, i.e., to when the network resources is underutilized. More specifically, *D2D communications* utilize the radio channel for end users to communicate directly instead of routing via the network infrastructure [1]–[3], while *caching* stores partial content that may be requested by users in the network in off-peak hours so as to reduce delivery rates to these users during peak-traffic hours [4], [5]. The seminal reference [6] introduced coded caching and demonstrated that, designing the downloading of partial data in off-peak hours, and the delivery signal in peak-hours in a manner to serve multiple users' file demands simultaneously, provides gains that are above and beyond simply placing some partial content in the caches. In particular, it has been shown that jointly designing the cache placement and delivery phases provides a *global caching gain* that results from the ability of serving multiple users by a single transmission, in addition to the *local caching gain* that results from the fact that some of the requested data is locally available in the user's cache. There has been significant recent interest in caching systems, notably in designing coded caching strategies demonstrating gains in various network settings beyond the broadcast network setting of the original reference, see for example, [7]–[15].

Caching in D2D communications have been pioneered in reference [15]. In particular, a network where a server, with database of N files, each with size F bits, connected to K users, each equipped with a cache memory of size MF bits, has been considered. In the cache placement phase, the server populates the cache memories of the users with partial content from the server's database. During the delivery phase, in contrast with the communication model in [6], the server remains inactive and the users' requests are to be satisfied via D2D communications only. Both centralized and decentralized schemes were provided. In the centralized schemes, the cache placement and delivery phases are jointly optimized, which requires the knowledge of the number of active users in the system while performing cache placement. In decentralized scheme, this knowledge is not necessary. The fundamental limits of coded caching in device-to-device networks have been further investigated in [16]–[20]. For instance, references [16]–[18] have studied the impact of coded caching on throughput scaling laws of D2D networks under the protocol model in [21].

Beside the need of reducing the network load during the peak hours, maintaining secure access and delivery is also essential in several applications, e.g., subscription services. These concerns can be addressed by the so called secure caching and secure delivery requirements studied in server based models. For *secure delivery* [14], [22]–[24], any external entity that overhears the signals during the delivery phase must

Manuscript received August 21, 2018; revised May 1, 2019 and July 21, 2019; accepted August 21, 2019. Date of publication September 11, 2019; date of current version December 19, 2019. This research was supported in part by the National Science Foundation under Grant CNS 13-14719 and Grant CCF 17-49665. This work was presented in part at the Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, CA, USA, November 2016. This research was performed when the first author was with the Pennsylvania State University as a Ph.D. student. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Lejla Batina. *(Corresponding author: Ahmed A. Zewail.)*

A. A. Zewail was with the Department of Electrical Engineering, Pennsylvania State University, State College, PA 16802 USA. He is now with Wireless R&D, Qualcomm, Inc., San Diego, CA 92121 USA (e-mail: azewail@qti.qualcomm.com).

A. Yener is with the Department of Electrical Engineering, Pennsylvania State University, University Park, PA 16802 USA (e-mail: yener@ee.psu.edu).

Digital Object Identifier 10.1109/TIFS.2019.2940885

not obtain any information about the database files. In particular, reference [23] has studied a device-to-device caching system with secure delivery. Utilizing one-time padding, a centralized scheme has been proposed by jointly optimizing the cache placement and delivery phase. The order-optimality of this scheme has been shown in [25], i.e., the multiplicative gap between the achievable delivery load, in [23] and the developed lower bound, in [25], can be bounded by a constant that is independent from the system's parameters. For *secure caching* [14], [24], [26], each user should be able to recover its requested file, but must not gain any information about the contents of the files it did not request.

In this paper, we investigate the fundamental limits of secure caching in *device-to-device* networks. That is, unlike the settings in [14], [24], and [26], the server disengages during the delivery phase, and users' requests must be satisfied via D2D communications only. By the end of the delivery phase, each user must be able to reconstruct its requested file, while not being able to obtain any information about the remaining $N-1$ files. For this D2D model, we derive lower and upper bounds on the rate-memory trade-off. We propose a centralized caching scheme, where the server encodes each file using proper *non-perfect secret sharing schemes* [27]–[30], and generates a set of random keys [31]. Then, the server carefully places these file shares and keys in the cache memories of the users. During the delivery phase, each user maps the contents of its cache memory into a signal transmitted to the remaining users over a shared multicast link. Next, motivated by the proposed schemes in [32] under no secrecy requirements, we provide a semi-decentralized scheme, using a grouping-based approach that guarantees secure caching, and does not require the knowledge of the number of active users in the system while populating the users' cache memories. To evaluate the performance of these achievable schemes, we also develop a lower bound on the required transmission sum rate based on cut-set arguments. We show that the multiplicative gap between the lower and upper bounds is bounded by a constant. Furthermore, we observe numerically that this gap vanishes as the memory size increases.

By virtue of the D2D model, the delivery load has to be completely transferred from the server to the end users during cache placement in this network, so that no matter what file is demanded by a user, it can be delivered from other users. As such, imposing secure caching requirement will also facilitate secure delivery as we shall see in the sequel. In other words, for the proposed schemes, secure delivery [22]–[24] is also satisfied as a byproduct.

This work demonstrates that D2D communications can effectively replace a server with full database access despite the fact that each user accesses only a portion of the database and that this is possible with a negligible transmission overhead, while keeping the users ignorant about the database contents. That is to say that, the performance of the system under investigation and the one in [26] are very close to one another for realistic values of the system parameters. We note that while the centralized scheme and its performance were presented in brief in the conference paper [33], the decentralized coded caching scheme and the order-optimality results are presented for the first time in this paper, along with proof details of all results.

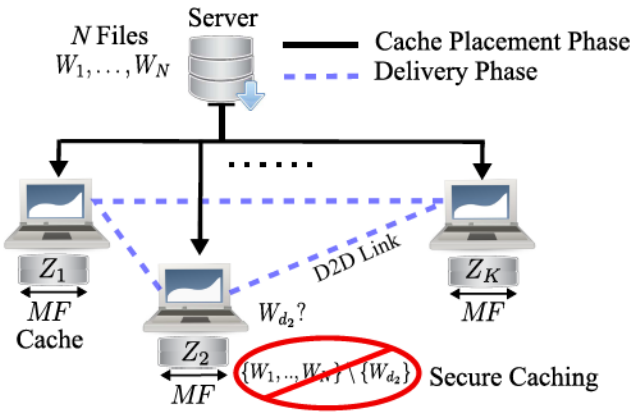

Fig. 1. Device-to-device secure coded caching system.

The remainder of the paper is organized as follows. In Section II, we describe the system model. In Sections III and IV, we detail the centralized and decentralized coded caching schemes, respectively. Section V contains the derivation of the lower bound. In Section VI, we demonstrate the system performance by numerical results. Section VII summarizes our conclusions. In the following, we will use the notation $[L] \triangleq \{1, \ldots, L\}$, for a positive integer L.

II. System Model

Consider a network where a server, with a database of N files, $W_1, \ldots, W_N$, is connected to K users. The files are N independent random variables, each has size F bits and is uniformly distributed over $[2^F]$. Each user equipped with a cache memory with size MF bits, i.e., each user is capable of storing M files. We denote by M the normalized cache memory size and define Z_k to represent the contents of the cache memory at user k, where $k \in \{1, 2, \ldots, K\}$. The system operates over two consecutive phases, as depicted in Fig. 1.

A. Cache Placement Phase

In this phase, the server allocates functions of its database into the users' cache memories without the knowledge of file demands. These possible allocations are designed to preserve the memory capacity constraint at each user. This is made precise by the following definition.

Definition 1 (Cache Placement): In the cache placement phase, the server maps the files of its database to the cache memories of the users. In particular, the content of the cache memory at user k is given by

$$Z_k = \phi_k(W_1, W_2, \ldots, W_N), \quad k = 1, 2, \ldots, K, \tag{1}$$

where $\phi_k : [2^F]^N \to [2^F]^M$. ■

In this work, aligned with the caching literature, e.g., [14], [22]–[24], [26], we assume that the cache placement phase is secure, i.e., it is not overheard by any unauthorized entity and the cache contents of each user is not accessible to any other user.

B. Delivery Phase

During peak traffic, each user requests a file. The indices of the requested files are represented by random variables. In particular, we assume that the demand distribution is uniform for all users, and independent from one user to another [6], [15], i.e., each user can request each file with equal probability, $\frac{1}{N}$ which is independent from the remaining users' requests. The index of the file requested by user k is $d_k \in [N]$, and $\boldsymbol{d} = (d_1, \ldots, d_K)$ represents the demand vector of all users. Similar to [15], we require that the delivery phase is carried our by D2D communications only, i.e., the server participates only in the cache placement phase. Therefore, we need the caches at the users to be able to store the whole library, collectively. Without secrecy requirements, we would need $KM \geq N$ to accomplish this. In Section III, we will see that a larger total memory constraint will be required in order to satisfy the secrecy requirements. With the knowledge of the demand vector $\boldsymbol{d}$, user k maps the contents of its cache memory, Z_k, into a signal that is transmitted to all network users over a noiseless interference-free multicast link. From the K–1 received signals and Z_k, user k must be able to decode its requested file, W_{d_k}, with negligible probability of error. We have the following definition for encoding and decoding at each user.

Definition 2 (Coded Delivery): The transmitted signal by user k is given by

$$X_{k,\boldsymbol{d}} = \psi_k(Z_k, \boldsymbol{d}), \tag{2}$$

where $\psi_k : [2^F]^M \times [N]^K \rightarrow [2^F]^{R_k}$ is the encoding function, R_k is the normalized rate of the transmitted signal by user k and $k \in [K]$. In addition, user k recovers its requested file as

$$\hat{W}_{d_k} = \mu_k(Z_k, \boldsymbol{d}, X_{1,\boldsymbol{d}}, \ldots, X_{k-1,\boldsymbol{d}}, X_{k+1,\boldsymbol{d}}, \ldots, X_{K,\boldsymbol{d}}), \tag{3}$$

where $\mu_k : [2^F]^M \times [N]^K \times [2^F]^{\sum_{i\neq k} R_i} \rightarrow [2^F]$ is the decoding function, and $k \in [K]$. ■

Let $R_T = \sum_{i=1}^{K} R_i$ be the normalized sum rate of the transmitted signals by all users.

C. System Requirements

During the delivery phase, the server remains silent, and all users' requests must be satisfied via D2D communications. Therefore, we have the following reliability requirement.

Definition 3 (Reliability): For each user to recover its requested file from its received signals and the contents of its cache memory, we need

$$\max_{\boldsymbol{d}, k\in[K]} Pr(\hat{W}_{d_k} \neq W_{d_k}) \leq \epsilon, \tag{4}$$

for any $\epsilon > 0$. ■

We impose secure caching constraints on the system. In particular, we require that each user must be able to decode only its requested file, and not be able to obtain any information about the content of the remaining $N - 1$ files.

Definition 4 (Secure caching): For any $\delta_1 > 0$, we have

$$\max_{\boldsymbol{d}, k\in[K]} I(W_{-d_k}; X_{-k,\boldsymbol{d}}, Z_k) \leq \delta_1, \tag{5}$$

where $W_{-d_k} = \{W_1, \ldots, W_N\} \backslash \{W_{d_k}\}$, i.e., the set of all files except the one requested by user k and $X_{-k,\boldsymbol{d}} = \{X_{1,\boldsymbol{d}}, \ldots, X_{K,\boldsymbol{d}}\} \setminus \{X_{k,\boldsymbol{d}}\}$, i.e., the set of all received signals by user k. ■

We aim to minimize the sum rate during the delivery phase under reliability and secure caching requirements. Formally, we have the following definition.

*Definition 5: The secure memory-rate pair (M, R_T) is said to be **achievable** if $\forall \epsilon, \delta_1 > 0$ and $F \rightarrow \infty$, there exists a set of caching functions, $\{\phi_k\}_{k=1}^K$, encoding functions, $\{\psi_k\}_{k=1}^K$, and decoding functions, $\{\mu_k\}_{k=1}^K$, such that (4) and (5) are satisfied. The optimal secure memory-rate trade-off is defined as $R_T^* = \inf\{R_T : (M, R_T)$ is achievable$\}$.* ■

We are also interested in the secure delivery requirement, defined below.

Definition 6 (Secure Delivery): Any eavesdropper that overhears the transmitted signals during the delivery phase must not obtain any information about the contents of the data phase files. Therefore, we have

$$\max_{\boldsymbol{d}} I(W_1, \ldots, W_N; X_{1,\boldsymbol{d}}, \ldots, X_{K,\boldsymbol{d}}) \leq \delta_2, \tag{6}$$

for any $\delta_2 > 0$. ■

Remark 1: We will see that for the D2D setting we consider, our proposed schemes for secure caching will automatically satisfy the secure delivery requirement. ■

Remark 2: In general, secure delivery and secure caching requirements do not have to imply one another. For instance, if $M \geq N$, secure delivery is trivially satisfied by storing the entire database at each user during the cache placement phase violating the secure caching requirements. An example for the reverse scenario, i.e., where secure caching does not imply the secure delivery can be found in subsection III-F. ■

We aim to minimize the total delivery load, i.e., the total transmission rate, by designing the cache contents and the delivery strategy while maintaining the secure caching requirement. The system design requires two steps.

1) Cache placement: The cached contents by each user, by itself, must not reveal any information about the system files. This makes the cache placement problem relevant to the problem of multiple assignment in secret sharing [27]–[29], in the sense that, we aim to distribute the library over the set of end users such that the shares assigned to each of them cannot reveal any information about the files. One main factor that distinguishes the placement strategy from the classical multiple assignment in secret sharing is that the size of shares allocated to each user must not exceed its cache storage capacity, MF.
2) Delivery: Each user must be able to decode only its requested file. With the D2D model, we require the system to maintain self-sustainability without the participation of the server during the delivery phase. Thus, the caches' contents at all users collectively must be able to regenerate the entire library.

In the following two sections, we provide two schemes that minimize the delivery load while maintaining the systems' requirements.

III. Centralized Coded Caching Scheme

In this section, we consider a scenario where the server is able to perform cache placement in a centralized manner. That is, the server knows the total number of users in the system, K, at the beginning of the cache placement phase. We utilize non-perfect secret sharing schemes [30], [34], [35] to encode the database files. The basic idea of the non-perfect secret sharing schemes is to encode the secret in such a way that accessing a subset of shares does not reduce the uncertainty about the secret, and only accessing all shares does. For instance, if the secret is encoded into the scaling coefficient of a line equation, the knowledge of one point on the line does not reveal any information about the secret as there remain infinite number of possibilities to describe the line. One can learn the secret only if two points on the line are provided, and then can do so precisely. We will utilize *non-perfect secret sharing schemes*, formally defined as follows.

Definition 7 [34]: For a file W with size F bits, an (m, d, n) non-perfect secret sharing scheme generates n shares, $S_1, S_2, \ldots, S_n$, such that accessing any $m-d$ shares does not reveal any information about the file W, i.e.,

$$I(W; \mathcal{S}) = 0, \quad \forall \mathcal{S} \subseteq \{S_1, S_2, \ldots, S_n\}, |\mathcal{S}| \leq m - d. \quad (7)$$

Furthermore, the knowledge of any m shares is sufficient to reconstruct the secret, i.e.,

$$H(W|\mathcal{A}) = 0, \quad \forall \mathcal{A} \subseteq \{S_1, S_2, \ldots, S_n\}, |\mathcal{A}| \geq m. \quad (8)$$

■

For large enough F, an (m, d, n) non-perfect secret sharing scheme exists with shares of size equals to $\frac{F}{d}$ bits [30], [34], [35]. We chose to use schemes from this class as they give shares with sizes equal to the secret size divided by the gap, $\frac{F}{d}$ bits. By contrast, perfect secret sharing schemes [36] give shares of size equal to the secret size, F bits. Therefore, the non-perfect secret sharing schemes are more efficient in our case in terms of storage and delivery load.

A. Cache Placement Phase

First, we present a scheme that works for $M = \frac{Nt}{K-t} + \frac{1}{t} + 1$, and $t \in [K-1]$, noting that the remaining values of M can be achieved by memory sharing [6]. That is, for any value of M, we pick the most two adjacent values, M_1 and M_2, such that $M_i = \frac{Nt}{K-t_i} + \frac{1}{t_i} + 1$, $i = 1, 2$, $t_i \in [K-1]$, and $M_1 \leq M \leq M_2$. We determine the sharing parameter $\alpha \in [0, 1]$, by solving the equation $M = \alpha M_1 + (1-\alpha)M_2$. Then, each file, W_n, is divided into two subfiles, W_n^1 and W_n^2, of sizes αF and $(1-\alpha)F$ bits, respectively. The achievability scheme is obtained by applying the scheme designed for the system with memory M_i on the subfiles W_n^i, and $i = 1, 2$.

As a first step, the server encodes each file in the database using a non-perfect secret sharing scheme [30], [34], [35]. In particular, a file, W_n, is encoded using a $\left(t\binom{K}{t}, t\binom{K}{t} - t\binom{K-1}{t-1}, t\binom{K}{t}\right)$ non-perfect secret sharing scheme. Each share, with size F_s bits, is denoted $S_{n,\mathcal{T}}^j$, where $j = 1, \ldots, t$, and $\mathcal{T} \subseteq [K]$ with $|\mathcal{T}| = t$, and

$$F_s = \frac{F}{t\binom{K}{t} - t\binom{K-1}{t-1}} = \frac{F}{(K-t)\binom{K-1}{t-1}}. \quad (9)$$

Algorithm 1 Cache Placement Procedure

Require: $\{W_1, \ldots, W_N\}$
Ensure: $Z_k, k \in [K]$
1: **for** $l \in [N]$ **do**
2: Encode W_l using an $(t\binom{K}{t}, t\binom{K}{t} - t\binom{K-1}{t-1}, t\binom{K}{t})$ non-perfect secret sharing scheme $\rightarrow S_{l,\mathcal{T}}^j$, $j = 1, \ldots, t$, and $\mathcal{T} \subseteq [K]$ with $|\mathcal{T}| = t$.
3: **end for**
4: **for** $\mathcal{T}_K \subseteq [K]$ with $|\mathcal{T}_K| = t+1$ **do**
5: Generate independent keys $K_{\mathcal{T}_K}^i$, $i = 1, \ldots, t+1$.
6: **end for**
7: **for** $k \in [K]$ **do**
8: $Z_k \leftarrow \left\{K_{\mathcal{T}_K}^i : k \in \mathcal{T}_K, \forall i\right\} \cup \bigcup_{l \in [N]} \left\{S_{l,\mathcal{T}}^j : k \in \mathcal{T}, \forall j\right\}$
9: **end for**

We refer to the set $\mathcal{T}$ by the *allocation set* as it determines how the shares will be allocated in the users' caches. In particular, the server places the shares $S_{n,\mathcal{T}}^j$, $\forall j, n$ in the cache of user k whenever $k \in \mathcal{T}$. Also, the parameter t can be see as number of users that will store the same share.

Additionally, the server generates $(t+1)\binom{K}{t+1}$ independent keys, i.e., they are independent from one another and independent from the library files. In particular, each key is uniformly distributed over $[2^{F_s}]$, and is denoted by $K_{\mathcal{T}_K}^i$, where $i = 1, \ldots, t+1$, and $\mathcal{T}_K \subseteq [K]$ with $|\mathcal{T}_K| = t+1$. The server places the keys $K_{\mathcal{T}_K}^i$, $\forall i$, in user k's cache if $k \in \mathcal{T}_K$, i.e., $\mathcal{T}_K$ represents the key allocation set. Therefore, the cached content by user k at the end of the cache placement phase is given by

$$Z_k = \left\{S_{n,\mathcal{T}}^i, K_{\mathcal{T}_K}^j : k \in \mathcal{T}, \mathcal{T}_K, \text{ and } \forall i, n, j\right\}. \quad (10)$$

We summarize the cache placement procedure in Algorithm 1. In the following remark, we verify that this placement satisfies the cache memory capacity constraint.

Remark 3: In the aforementioned placement scheme, each user stores $t\binom{K-1}{t-1}$ shares of each file and $(t+1)\binom{K-1}{t}$ distinct keys, thus the accumulated number of cached bits is given by

$$Nt\ \binom{K-1}{t-1}F_s + (t+1)\binom{K-1}{t}F_s = \frac{Nt}{K-t}F + (1+\tfrac{1}{t})F = MF. \quad (11)$$

It follows that we have

$$t = \frac{1 + (M-1)K + \sqrt{(1-(M-1)K)^2 - 4KN}}{2(N+M-1)}. \quad (12)$$

Clearly, the proposed allocation scheme satisfies the cache memory capacity constraint at each user. ■

Remark 4: We note that the minimum value of the normalized cache size, M, that is need to apply the proposed scheme is $M_{min} = M|_{t=1} = 2 + \frac{N}{K-1}$. For a system without secrecy requirements [15], we need $M \geq \frac{N}{K}$, while with secure delivery, the scheme in [23] requires $M \geq 2 + \frac{N-2}{K}$. It is evident that, with secrecy requirements, more memory is required, as the users not only cache from the data but also cache the secure keys. ■

Algorithm 2 Delivery Procedure

Require: d
Ensure: $X_{k,d}, k \in [K]$
1: **for** $k \in [K]$ **do**
2: **for** $\mathcal{S} \in [K], |\mathcal{S}| = t+1, k \in \mathcal{S}$ **do**
3: $X^{\mathcal{S}}_{k,d} \leftarrow \oplus_{l \in \mathcal{S}\setminus\{k\}} S^{j}_{d_l, \mathcal{S}\setminus\{l\}} \oplus K^{i}_{\mathcal{S}}$, for some choice of i and j
4: **end for**
5: $X_{k,d} \leftarrow \bigcup_{\mathcal{S}\subseteq[\hat{K}], k\in\mathcal{S}} \{X^{\mathcal{S}}_{k,d}\}$
6: **end for**

B. Coded Delivery Phase

At the beginning of the delivery phase, each user requests one of the N files and the demand vector is known to all network users. To derive an upper bound on the required transmission sum rate, we focus our attention on the worst case scenario. We concentrate on the more relevant scenario of $N \geq K$.

The delivery procedure consists of $\binom{K}{t+1}$ transmission instances. At each transmission instance, we consider a set of users $\mathcal{S} \subseteq [K]$, where $|\mathcal{S}| = t+1$. We refer to $\mathcal{S}$ as the transmission set. For $k \in \mathcal{S}$, user k multicasts the following signal of length F_s bits

$$X^{\mathcal{S}}_{k,d} = \oplus_{l \in \mathcal{S}\setminus\{k\}} S^{j}_{d_l, \mathcal{S}\setminus\{l\}} \oplus K^{i}_{\mathcal{S}}. \tag{13}$$

Note that the index i is chosen such that each key is used only once, while the index j is chosen to ensure that each transmission is formed by shares that had not been transmitted in previous transmissions by the other users in $\mathcal{S}$. For example, they can be chosen as the relative order of the user's index with respect to the indices of the remaining users in $\mathcal{S}$. Thus, in total, the transmitted signal by user k can be expressed as

$$X_{k,d} = \bigcup_{\mathcal{S}:\, k\in\mathcal{S},\, \mathcal{S}\subseteq[K], |\mathcal{S}|=t+1} \{X^{\mathcal{S}}_{k,d}\}. \tag{14}$$

Observe that the cache memories of the users from any subset, $\mathcal{S}_t \subset \mathcal{S}$, with $|\mathcal{S}_t| = t$ contain t shares of the file requested by the user in $\mathcal{S} \setminus \mathcal{S}_t$, as can be seen from (10). Thus, utilizing its cache contents, each user in $\mathcal{S}$ obtains t shares from its requested file during this instance of transmission, i.e., the user in $\mathcal{S} \setminus \mathcal{S}_t$ obtains the shares $S^{j}_{d_{\mathcal{S}\setminus\mathcal{S}_t}, \mathcal{S}_t}$ $\forall j$.

Observe also that user k belongs to $\binom{K-1}{t}$ different choices of such subsets of the users, thus at the end of the delivery phase, user k obtains $t\binom{K-1}{t}$ *new* shares of its requested files, in addition to the cached $t\binom{K-1}{t-1}$ shares. Therefore, user k can decode its requested file from its $t\binom{K}{t}$ shares, i.e., the reliability requirement (4) is satisfied. Delivery procedure is summarized in Algorithm 2.

C. Rate Calculation

Now, we focus our attention on calculating the required transmission rate. Note that there are $\binom{K}{t+1}$ different choices of the set $\mathcal{S}$. For each choice, $t+1$ signals of length F_s bits are transmitted, thus the total number of the transmitted bits is given by

$$R_T F = (t+1)\binom{K}{t+1} F_s = \frac{K}{t} F. \tag{15}$$

Consequently, we can achieve the following normalized sum rate

$$R_T = \frac{2K(N+M-1)}{1+(M-1)K+\sqrt{(1-(M+1)K)^2 - 4KN}}. \tag{16}$$

Therefore, we can state the following theorem.

Theorem 1: Under centralized placement, for $M = \frac{Nt}{K-t} + \frac{1}{t} + 1$, *and* $t \in [K-1]$, *the secure sum transmission rate is upper bounded by*

$$R^*_T \leq R^C_T \leq \frac{2K(N+M-1)}{1+(M-1)K+\sqrt{(1-(M-1)K)^2 - 4KN}}. \tag{17}$$

In addition, using memory sharing [6], we can achieve the convex envelope of the points given by the values $M = \frac{Nt}{K-t} + \frac{1}{t} + 1$, *and* $t \in [K-1]$. ■

D. Secrecy Analysis

For user k, the cache's contents, Z_k given by (10), contains only $t\binom{K-1}{t-1}$ shares, from each file, resulting from a $\left(t\binom{K}{t}, t\binom{K}{t} - t\binom{K-1}{t-1}, t\binom{K}{t}\right)$ non-perfect secret sharing scheme. Therefore, Z_k, by itself, cannot reveal any information about the files to user k.

During the delivery phase, if at any instance, user k belongs to the transmission set, $\mathcal{S}$, then the transmitted signals are formed from the shares of the requested file by user k, W_{d_k}, and shares that have been already placed in the cache of user k during the cache placement phase, i.e., from Z_k. When user k does not belong to the transmission set, all the transmitted signals are encrypted using one-time pads, unknown to user k, thus, user k cannot gain any information from these signals [31]. Therefore, the secure caching constraint, (5), is satisfied.

We observe that the server has generated $(t+1)\binom{K}{t+1}$ independent keys with lengths equal to the share size. Thus, with a proper selection of the encrypting key for each transmission, we can ensure a unique use of each key, i.e., one-time padding. The above discussion implies that the secrecy of the transmitted signals, from any external wiretapper that accesses the network links during the delivery phase, is also guaranteed [31]. One-time pads are essential to ensure the secure caching requirement in (5), whereas the secure delivery requirement in (6), is satisfied as a byproduct.

E. An Illustrative Example

Consider a system with four users and a library consists of four files, W_1, W_2, W_3, W_4, i.e., $K = N = 4$. Each user has a normalized memory size $M = \frac{11}{2}$, which gives us $t = 2$ and indicates that each of the resulting shares will be cached by two different users. The server encodes each file using (12, 6, 12) non-perfect secret sharing scheme. For a file, W_n, the server generates 12 shares, which we label by $S^{i}_{n,\mathcal{T}}$ where $i = 1, 2$, $\mathcal{T} \subset \{1, 2, 3, 4\}$ and $|\mathcal{T}| = 2$, each of size $F/6$ bits.

Furthermore, the server generates the set of keys $K^j_{\mathcal{T}_K}$, uniformly distributed over $\{1,\dots,2^{F/6}\}$, where $j=1,2,3$, $\mathcal{T}_K \subset \{1,2,3,4\}$ and $|\mathcal{T}_K|=3$.

User k stores the shares $S^i_{n,\mathcal{T}}$, and the keys $K^j_{\mathcal{T}_K}$ whenever $k\in\mathcal{T}$ and $k\in\mathcal{T}_K$, $\forall n,j,i$, respectively. Therefore, the cache contents at the users are given by

$$Z_1=\left\{S^i_{n,12},S^i_{n,13},S^i_{n,14},\ \forall n,i,K^j_{123},K^j_{124},K^j_{134},\ \forall j\right\},$$
$$Z_2=\left\{S^i_{n,12},S^i_{n,23},S^i_{n,24},\forall n,i,K^j_{123},K^j_{124},K^j_{234},\ \forall j\right\},$$
$$Z_3=\left\{S^i_{n,13},S^i_{n,23},S^i_{n,34},\forall n,i,K^j_{123},K^j_{134},K^j_{234},\ \forall j\right\},$$
$$Z_4=\left\{S^i_{n,14},S^i_{n,24},S^i_{n,34},\forall n,i,K^j_{124},K^j_{134},K^j_{234},\ \forall j\right\}.$$

Each user caches 6 shares of each file. We observe that the caches will not be able to reveal any information about the unrequested files thanks to the non-perfect secret sharing encoding. Also, note that the cache capacity constraints at all the users are satisfied.

Now, consider the delivery phase, where user k requests the file W_k, i.e., $\boldsymbol{d}=(1,2,3,4)$. In this case, the users transmit the following signals.

$$X_{1,\boldsymbol{d}}=\left\{\begin{array}{c}S^1_{2,13}\oplus S^1_{3,12}\oplus K^1_{123},S^1_{4,13}\oplus S^1_{3,14}\oplus K^1_{134},\\ S^1_{2,14}\oplus S^1_{4,12}\oplus K^1_{124}\end{array}\right\},$$
$$X_{2,\boldsymbol{d}}=\left\{\begin{array}{c}S^2_{1,23}\oplus S^2_{3,12}\oplus K^2_{123},S^2_{4,23}\oplus S^2_{3,24}\oplus K^2_{234},\\ S^2_{1,24}\oplus S^2_{4,12}\oplus K^2_{124}\end{array}\right\},$$
$$X_{3,\boldsymbol{d}}=\left\{\begin{array}{c}S^1_{1,23}\oplus S^2_{2,13}\oplus K^3_{123},S^2_{4,13}\oplus S^1_{1,34}\oplus K^2_{134},\\ S^1_{2,34}\oplus S^1_{4,23}\oplus K^3_{234}\end{array}\right\},$$
$$X_{4,\boldsymbol{d}}=\left\{\begin{array}{c}S^1_{1,24}\oplus S^2_{2,14}\oplus K^3_{124},S^2_{1,34}\oplus S^2_{3,14}\oplus K^3_{134},\\ S^2_{2,34}\oplus S^1_{3,24}\oplus K^1_{234}\end{array}\right\}.$$

From its received signals, $X_{2,\boldsymbol{d}}$, $X_{3,\boldsymbol{d}}$ and $X_{4,\boldsymbol{d}}$, and utilizing its cached content, user 1 gets $S^1_{1,23}$, $S^2_{1,23}$, $S^1_{1,24}$, $S^2_{1,24}$, $S^1_{1,34}$ and $S^2_{1,34}$. Thus, user 1 can reconstruct its requested file, W_1, from its 12 shares. Similarly, users 2, 3 and 4 are able to decode files W_2, W_3 and W_4, respectively.

We observe that user k will only obtain new shares of its requested file W_k, thus it cannot gain any information about the remaining files, $\{W_1,W_2,W_3,W_4\}\setminus\{W_k\}$. This is done by proper selection of the keys so that each user cannot gain any information about the remaining three files. In addition, each signal is encrypted using one-time pad which ensures the secrecy of the database files from any external eavesdropper as in [23]. In this delivery procedure, each user participates by 3 distinct transmissions, each of size $F/6$ bits, thus $R^C_T=2$. Comparing with the system in [26], where the server is responsible for the delivery phase, we see that a normalized secure rate $\simeq 1.3$ is achievable, for the same system parameters. This difference is due to limited access of the shares at each user, unlike the case in [26] where the server can access all shares during the delivery phase, i.e., the cost of having D2D delivery.

F. Secure Caching Without Secure Delivery for $M=N(K-1)$

The scheme described above ensures that the requirements in (5) (and (6)) are satisfied. The encryption keys are essential to achieve both. In the following, we study a special case where we can provide a scheme that achieves secure caching, i.e., satisfy (5), without the necessity of satisfying the secure delivery constraint, i.e., (6). More specifically, when $M=N(K-1)$, we can achieve a normalized rate equals to $\frac{K}{K-1}$ without utilizing encryption keys. In particular, each file is encoded using $(K(K-1),K-1,K(K-1))$ non-perfect secret sharing scheme. The resulting shares, each of size $F_s=\frac{F}{K-1}$ bits, are indexed by $S^j_{n,i}$, where n is the file index, $j=1,\dots,K-1$, and $i=1,\dots,K$. The server allocates the shares $S^j_{n,i}$, $\forall j,n$ and $i\neq k$ in the memory of user k, i.e.,

$$Z_k=\{S^j_{n,i}:\forall j,n \text{ and } i\neq k\}. \tag{18}$$

Thus, each user stores $N(K-1)^2$ shares, which satisfies the memory capacity constraint.

At the beginning of the delivery phase, each user announces its request. Again, we assume that the users request different files. User k multicasts the following signal to all other users

$$X_{k,\boldsymbol{d}}=\oplus_{l\in[K]\setminus\{k\}}S^j_{d_l,l}, \tag{19}$$

where j is chosen to ensure that each transmission is formed by fresh shares which had not been included in the previous transmissions. From its received $K-1$ signals, user k can extract the shares $S^j_{d_k,k}$, $\forall j$. By combining these shares with the ones in its memory, user k recovers its requested file, W_{d_k}. The total number of bits transmitted under this scheme is $R_TF=KF_s$. Thus, the following normalized sum rate, under the secure caching constraint (5), is achievable for $M=N(K-1)$,

$$R_T=\frac{K}{K-1}. \tag{20}$$

This rate matches the cut set bound as in Section V.

G. Discussion

The above scheme, in subsection III-F, satisfies only the secure caching constraint (5), without ensuring the protection from any external eavesdropper that overhears the transmitted signals during the delivery phase. On the other hand, the general scheme, presented in subsections III-A and III-B, achieves the same rate, i.e., $R^C_T=\frac{K}{K-1}$, when $M=N(K-1)+\frac{K}{K-1}$, while satisfying the secure caching constraint (5) and the secure delivery constraint (6), simultaneously. In other words, an additional memory at each user with size $\frac{K}{K-1}F$ bits is required to ensure the additional requirement of secure delivery.

We observe that the encryption keys serve to satisfy both the secure caching and secure delivery requirements. Therefore, one can think about the satisfaction of the secure delivery requirement as a byproduct of the general scheme in subsections III-A and III-B, i.e., the secure delivery comes for free while satisfying the secure caching constraint, whenever $M\leq\frac{N(K-2)}{2}+\frac{1}{K-2}+1$. Under different network topologies, secure delivery may require additional cost. For example, in recent reference [14], we have shown that there is no need to use encryption keys to satisfy the secure caching requirements in the setting of combination networks. This is possible due

to the unicast nature of the network links, which is not the case in the system under investigation, as we assume that the users communicate with each other via multicast links.

IV. Decentralized Coded Caching Scheme

In this section, we provide a decentralized coded caching scheme, [37], for our setup. The proposed scheme is motivated by the ones in [32] for multicast coded caching setup without secrecy requirements [6], [37]. It does not require the knowledge of the number of active users of the delivery phase during cache placement. This scheme operates over two phases as follows.

A. Cache Placement Phase

The main idea of the cache placement scheme is to design the cache contents for a number of users L that is less than the number of users in the system during the delivery phase, i.e., K. L is in effect a lower bound on the expected number of active users in the system.

For a given L and $M = \frac{Nt}{L-t}+\frac{2}{t}+1$, and $t \in [L-1]$, each file in the database is encoded using a suitable non-perfect secret sharing scheme. In particular, a file, W_n, is encoded using an $\left(t\binom{L}{t}, t\binom{L}{t} - t\binom{L-1}{t-1}, t\binom{L}{t}\right)$ non-perfect secret sharing scheme. We obtain $t\binom{L}{t}$ shares, each with size $\bar{F}_s$, where

$$\bar{F}_s = \frac{F}{t\binom{L}{t} - t\binom{L-1}{t-1}} = \frac{F}{(L-t)\binom{L-1}{t-1}}. \tag{21}$$

Each share is denoted by $S^j_{n,\mathcal{T}}$, where n is the file index, i.e., $n \in [N]$, $j = 1, \ldots, t$, and $\mathcal{T} \subseteq [L]$ with $|\mathcal{T}| = t$. The server prepares the following set of cache contents, $\bar{Z}_l$,

$$\bar{Z}_l = \{S^j_{n,\mathcal{T}} : l \in \mathcal{T},\ \forall j, n\}, \quad l = 1, 2, \ldots, L. \tag{22}$$

Once user k joins the system, it caches the content $\bar{Z}_{l_k}$ where $l_k = k \mod L$. Such allocation results in dividing the set of active users into $\lceil \frac{K}{L} \rceil$ virtual groups. In particular, we group the first L users to join the system in group 1, and the users from $L+1$ to $2L$ in group 2 and so on. Note that each group from 1 to $\lceil \frac{K}{L} \rceil - 1$ contains L users, and the group $\lceil \frac{K}{L} \rceil$ contains $K - (\lceil \frac{K}{L} \rceil - 1)L$ users. These groups are formed sequentially in time.

As explained in Section III, we require the server to generate a set of random keys to be shared between the users. For group u, $u = 1, \ldots, \lceil \frac{K}{L} \rceil - 1$, the server generates the keys $K^i_{u,\mathcal{T}_K}$, where $i = 1, \ldots, t+1$, $\mathcal{T}_K \subseteq [L]$ and $|\mathcal{T}_K| = t+1$. Each key is uniformly distributed over $[2^{\bar{F}_s}]$. User l_k from group u stores the keys $K^i_{u,\mathcal{T}_K}$, $\forall i$, whenever $l_k \in \mathcal{T}_K$.

In addition, the server generates the keys $K^i_{u^*,\mathcal{T}_K}$, where $i = 1, \ldots, t+1$, and $\mathcal{T}_K \subseteq [L]$, $|\mathcal{T}_K| = t+1$, and allocates these keys in the cache memories of the users in groups 1 and $\lceil \frac{K}{L} \rceil$ as follows. The keys $\{K^i_{u^*,\mathcal{T}_K},\ \forall i\}$ are cached by user l_k from group $\lceil \frac{K}{L} \rceil$, as long as $l_k \in \mathcal{T}_K$. User l_k from group 1 stores the keys $K^j_{u^*,\mathcal{T}_K}$ for only one specific j whenever $l_k \in \mathcal{T}_K$. This index j is chosen such that the users from group 1 store different keys.

In summary, at the end of cache placement, cache contents of user k are given by

$$Z_k = \begin{cases} \{\bar{Z}_{l_k}, K^i_{1,\mathcal{T}_K}, K^j_{u^*,\mathcal{T}_K} : l_k \in \mathcal{T}_K, \forall i,\ \text{for a specific } j\}, \\ \qquad\qquad \text{if } 1 \le k \le L, \\ \{\bar{Z}_{l_k}, K^i_{u,\mathcal{T}_K} : u = \lceil \frac{k}{L} \rceil, l_k \in \mathcal{T}_K, \forall i\}, \\ \qquad\qquad \text{if } L+1 \le k \le K - (\lceil \frac{K}{L} \rceil - 1)L, \\ \{\bar{Z}_{l_k}, K^i_{u^*,\mathcal{T}_K} : l_k \in \mathcal{T}_K, \forall i\}, \\ \qquad\qquad \text{if } K - (\lceil \frac{K}{L} \rceil - 1)L + 1 \le k \le K. \end{cases} \tag{23}$$

Remark 5: We need to ensure that this allocation procedure does not violate the memory capacity constraint at each user. Observe that each user stores the same amount of the encoded file shares, however, the users from group 1 stores more keys than the other users. Thus, satisfying the memory constraint at the users in group 1 implies satisfying the memory constraint at all network users. Each user in group 1 stores $Nt\binom{L-1}{t-1}$ shares and $(t+2)\binom{L-1}{t}$ keys. Thus, the total number of the stored bits is given by

$$Nt\binom{L-1}{t-1}\bar{F}_s + (t+2)\binom{L-1}{t}\bar{F}_s = \frac{Nt}{L-t}F + (1+\frac{2}{t})F = MF, \tag{24}$$

and from (24), we get

$$t = \frac{2 + (M-1)L + \sqrt{(2-(M-1)L)^2 - 8LN}}{2(N+M-1)}. \tag{25}$$

Therefore, the proposed scheme satisfies the cache capacity constraint at each user. ■

B. Coded Delivery Phase

We focus our attention on the worst case demand, where K users request K different files. The delivery phase is divided into $\lceil \frac{K}{L} \rceil$ stages. At each stage, we focus on serving the users of one group. For any stage u, where $u = 1, \ldots, \lceil \frac{K}{L} \rceil - 1$, the delivery process during stage u is performed in a way similar to the one described in subsection III-B with $K = L$ to serve the requests of users in group u. In particular, at each transmission instance, we consider $\mathcal{S} \subseteq [L]$, where $|\mathcal{S}| = t+1$. User k, with $l_k \in \mathcal{S}$, multicasts a signal, of length $\bar{F}_s$ bits, given by

$$K^i_{u,\mathcal{S}} \oplus_{l_v \in \mathcal{S}\setminus\{l_k\}} S^j_{d_v,\mathcal{S}\setminus\{l_v\}}, \tag{26}$$

where the index i is chosen in way that guarantees the uniqueness of the key utilized for each transmission. From the cache placement phase, we observe that any t users belong to the set $\mathcal{S}$ share t shares of the file requested by the remaining user that is in $\mathcal{S}$. Thus, each user in $\mathcal{S}$ obtains t shares from its requested file during this instance of transmission. At the end of stage u, each user from group u can decode its requested file from its $t\binom{L}{t}$ shares.

Since, there are $\binom{L}{t+1}$ different choices of the set $\mathcal{S}$, and for each choice $t+1$ signals of length $\bar{F}_s$ are transmitted, the total

number of the transmitted bits to serve the users from group u is

$$R_u F = (t+1)\binom{L}{t+1}\bar{F}_s = \frac{L}{t}F, \quad u = 1, \ldots, \lceil \frac{K}{L} \rceil - 1. \quad (27)$$

Now, we focus on serving the users of the last group, i.e., group $\lceil \frac{K}{L} \rceil$. First, recall that the number of users in this group is $p \triangleq K - (\lceil \frac{K}{L} \rceil - 1)L < L$, thus these users cannot satisfy their requests via device-to-device communications between them only. We require some of the users from group 1 to participate in this last stage of the delivery phase. In particular, the users indexed by l_k, with $l_k = p+1, \ldots, L$, from group 1 forms a virtual group with the users from group $\lceil \frac{K}{L} \rceil$, such that the resulting group contains L users. Note that, at this stage, the requests of the users from group 1 have been already served, during stage 1. Therefore, at each transmission instance, we consider only the sets $\mathcal{S} \subseteq [L]$, where $|\mathcal{S}| = t+1$ with $l_k \in \mathcal{S}$ and $l_k \in [p]$. We define the sets $\mathcal{S}_{u^*}$ and $\mathcal{S}^c_{u^*}$ to represent the subset of $\mathcal{S}$ that contains the users from group $\lceil \frac{K}{L} \rceil$ and group 1, respectively, i.e., $\mathcal{S}_{u^*} \cup \mathcal{S}^c_{u^*} = \mathcal{S}$. Since, we only care now about serving the users in group $\lceil \frac{K}{L} \rceil$, we neglect any set $\mathcal{S}$ with $\mathcal{S}_{u^*} = \{\}$. For the sets that contain only one user, user k, from group $\lceil \frac{K}{L} \rceil$, i.e., $l_k \in \mathcal{S}_{u^*}$, $|\mathcal{S}_{u^*}| = 1$, and $|\mathcal{S}^c_{u^*}| = t$, each user in the set $\mathcal{S}^c_{u^*}$ transmits

$$K^j_{u^*,\mathcal{S}} \oplus S^i_{d_k,\mathcal{S}\setminus\{l_k\}}, \quad (28)$$

where i is chosen to ensure that from every transmission user k obtains a different share from its requested file.

For the sets that contain more than one user from group $\lceil \frac{K}{L} \rceil$, i.e., $|\mathcal{S}_{u^*}| \geq 2$, each user in the set $\mathcal{S}$ multicasts a signal of length $\bar{F}_s$ given by

$$K^j_{u^*,\mathcal{S}} \oplus_{l_v \in \mathcal{S}_{u^*}\setminus\{l_k\}} S^i_{d_v,\mathcal{S}_{u^*}\setminus\{l_v\}}. \quad (29)$$

By taking into account all possible sets with $|\mathcal{S}_{u^*}| \geq 1$, the total number of the transmitted bits during this stage is given by

$$R_{u^*}F = pt\binom{L-p}{t}\bar{F}_s + \sum_{u=2}^{\min(p,t)}(t+1)\binom{L-p}{t-u+1}\bar{F}_s. \quad (30)$$

Consequently, we can obtain the following upper bound on the normalized sum rate

$$R^D_T = R_{u^*} + \left(\lceil \frac{K}{L} \rceil - 1\right) R_u. \quad (31)$$

Theorem 2: For any integer $L \leq K$, $M = \frac{Nt}{L-t} + \frac{2}{t} + 1$, and $t \in [L-1]$, the secure sum rate under decentralized coded caching is upper bounded by

$$R^*_T \leq R^D_T \leq \frac{2L(N+M-1)\left(\lceil \frac{K}{L} \rceil - 1\right)}{2+(M-1)L+\sqrt{(2-(M-1)L)^2-8LN}} + \frac{pt\langle\binom{L-p}{t}\rangle + \sum_{u=2}^{\min(p,t)}(t+1)\langle\binom{L-p}{t-u+1}\rangle}{(L-t)\binom{L-1}{t-1}}, \quad (32)$$

where $\langle\binom{h}{r}\rangle = \binom{h}{r}$ whenever $h \geq r$ and 0 otherwise, and $p = K - (\lceil \frac{K}{L} \rceil - 1)L$. In addition, the convex envelope of the above points, defined for each M, is also achievable. ■

Using memory sharing [6], we can achieve the convex envelope of the points given by the values $M = \frac{Nt}{L-t} + \frac{2}{t} + 1$, and $t \in [L-1]$.

C. Discussion

In the decentralized coded caching scheme proposed in [22], for server-based coded caching with secure delivery, key placement is done in a centralized manner after a decentralized caching of a fraction $\frac{M-1}{N-1}$ of each file, without the knowledge of the users' demands. For server-based systems with secure caching, a decentralized scheme was proposed in [26].

We note that developing decentralized schemes for D2D coded caching systems is more involved compared with decentralized schemes for server-based coded caching systems [6]. This due to the requirement that in D2D the server must disengage the delivery process, i.e., the end users collectively must possess pieces of the entire library. When there are no secrecy requirements, reference [15] has proposed a decentralized D2D coded caching scheme, which utilizes maximum distance separable (MDS) codes to encode the files at the server to satisfy the users' requests without the participation of the server during the delivery phase.

For our D2D secure coded caching, we utilize a grouping-based approach that allows disengaging the server from the delivery phase, and the key placement is done during the cache placement phase, without the knowledge of the users' demands. We choose this grouping-based approach instead of utilizing the MDS encoding in [15], as each user not only needs to store the keys used in encrypting its intended signals but also the keys that are used to encrypt its transmitted signals. Therefore, applying a decentralized cache placement based on MDS coding requires the users to dedicate a large fraction of their cache memories for the keys to be allocated by the server during the delivery phase after announcing the demand vector. By contrast, our proposed scheme ensures a practical self-sustainable system with reasonable fraction of each cache memory dedicated to encryption keys. Once a group of L users joins the system, sequentially, the server can place the keys in the memories of these users. At the end of the cache placement phase, before the beginning of the delivery phase, the server allocates the keys to be used in encrypting the signals intended to the last group in the caches of the users of the first and the last group. We remark that this grouping-based approach can be used to develop new decentralized schemes for multicast coded caching scenarios with secure delivery and secure caching that were considered in [22], [23], and [26] as well.

We observe, from (23), that the cache memory at each user is divided into two partitions, one for the shares of the files, and the other one for the keys. The partition assigned for the shares $\bar{Z}_l$ can be identical in multiple users. Thus, encrypting the signal with a one-time pad is necessary to satisfy the secure caching requirement. Note that each user from group 1 which participates in the last stage of the delivery phase, knows only the keys that it will use to encrypt its transmitted signal, thus it cannot gain any new information from the signals transmitted during this last stage. The secure delivery

requirement is satisfied as a byproduct as in the centralized scheme of Section III.

1) The Choice of L: A key element in designing the aforementioned semi-decentralized scheme, is the choice of the parameter L, which can be determined by observing the number of users in the system during the peak traffic hours over a sufficient amount of time. Then, we can choose L as the minimum value among the observed numbers of users in the system. We note that as long as L is close to K, the exact number of active users in the system, we can benefit from more multicast opportunities which helps in reducing the overall delivery load.

We note that a minor potential drawback of the provided scheme is that some users cache memories may be under-utilized by a small fraction. In particular, other than the users from group 1 who participate in serving the last group, a very small fraction of size, $\frac{1}{t}$, that is not scaled with the library size, is not utilized from each user memory. This fraction can be see as a cost for disengaging the server from the delivery phase. This fraction cannot be used to cache from data directly due to the secure caching requirement. A good estimate of L, i.e., choosing L close to K, will reduce the number of users that do not fully utilize their memory.

2) User Mobility During the Placement Phase: If a user, f, leaves the system during the cache placement phase, then its cached contents, Z_f, should be assigned by the server to populate the cache memory of the first user to join the system after this departure. If no user joins the system before the beginning of the delivery phase, then the server can update the contents of the last user that joined the system with Z_f.

V. Lower Bound

In this section, we derive a lower (converse) bound on the normalized sum of the required sum rate. The derivation is based on cut-set arguments [38], similar to [15], [39].

Assume that the first s users, where $s \in \{1, 2, \ldots, \min(N/2, K)\}$, request the files from 1 to s, such that user i requests W_i, $i \in \{1, 2, \ldots, s\}$. The remaining users are assumed to be given their requested files by a genie. We define X_1 to represent the transmitted signals by the users to respond to these requests, i.e., $X_1 = \{X_{1,(1,\ldots,s)}, \ldots, X_{K,(1,\ldots,s)}\}$. At the next request instance, the first s users request the files from $s+1$ to $2s$, such that user i requests W_{s+i}. These requests are served by transmitting the signals $X_2 = \{X_{1,(s+1,\ldots,2s)}, \ldots, X_{K,(s+1,\ldots,2s)}\}$. We proceed in the same manner, such that at the request instance q, the first s users request the files from $(q-1)s+1$ to qs, such that user i requests $W_{(q-1)s+i}$, and the users transmit the signals $X_q = \{X_{1,((q-1)s+1,\ldots,qs)}, \ldots, X_{K,((q-1)s+1,\ldots,qs)}\}$, where $q \in \{1, \ldots, \lfloor N/s \rfloor\}$. In addition, we define $\bar{X}_q = \{X_{s+1,((q-1)s+1,\ldots,qs)}, \ldots, X_{K,((q-1)s+1,\ldots,qs)}\}$ to denote the set of the transmitted signals by the users indexed by $s+1$ to K at request instance q.

From the received signals over the request instances $1, 2, \ldots, \lfloor N/s \rfloor$ and the information stored in its cache, i.e., Z_i, user i must be able to decode the files $W_i, W_{i+s}, \ldots, W_{i+(\lfloor N/s \rfloor - 1)s}$. Consider the set of files $\bar{\mathcal{W}} = \{W_1, \ldots, W_{(q-1)s+k-1}, W_{(q-1)s+k+1}, \ldots, W_{s\lfloor N/s \rfloor}\}$, i.e., the set of all requested files excluding the file, $W_{(q-1)s+k}$, which was requested by user k at the request instance q. Therefore, we have

$$
\begin{aligned}
&(s\lfloor N/s \rfloor - 1)F \\
&= H(\bar{\mathcal{W}}) \\
&\leq H(\bar{\mathcal{W}}) - H(\bar{\mathcal{W}}|X_1, \ldots, X_{\lfloor N/s \rfloor}, Z_1, \ldots, Z_s) + \epsilon && (33)\\
&= I(\bar{\mathcal{W}}; \bar{X}_1, \ldots, \bar{X}_{\lfloor N/s \rfloor}, Z_1, \ldots, Z_s) + \epsilon && (34)\\
&= I(\bar{\mathcal{W}}; \bar{X}_q, Z_k) + I(\bar{\mathcal{W}}; \bar{X}_1, \ldots, \bar{X}_{q-1}, \bar{X}_{q+1}, \ldots, \bar{X}_{\lfloor N/s \rfloor}, \\
&\quad Z_1, \ldots, Z_{k-1}, Z_{k+1}, \ldots, Z_s|\bar{X}_q, Z_k) + \epsilon. && (35)
\end{aligned}
$$

Step (33) follows from (4) as the users must be able to decode their requested files utilizing their cache's contents and received signals. To simplify the notation, we define

$$
\mathcal{X} = \{\bar{X}_1, \ldots, \bar{X}_{q-1}, \bar{X}_{q+1}, \ldots, \bar{X}_{\lfloor N/s \rfloor}\} \quad \text{and} \quad \mathcal{Z} = \{Z_1, \ldots, Z_{k-1}, Z_{k+1}, \ldots, Z_s\}.
$$

Now, (35) can be expressed as

$$
\begin{aligned}
&I(\bar{\mathcal{W}}; \bar{X}_q, Z_k) + I(\bar{\mathcal{W}}; \mathcal{X}, \mathcal{Z}|\bar{X}_q, Z_k) + \epsilon \\
&\leq I(\bar{\mathcal{W}}; \mathcal{X}, \mathcal{Z}|\bar{X}_q, Z_k) + \epsilon + \delta && (36)\\
&= H(\mathcal{X}, \mathcal{Z}|\bar{X}_q, Z_k) - H(\mathcal{X}, \mathcal{Z}|\bar{\mathcal{W}}, \bar{X}_q, Z_k) + \epsilon + \delta && (37)\\
&\leq H(\mathcal{X}, \mathcal{Z}|\bar{X}_q, Z_k) + \epsilon + \delta && (38)\\
&\leq H(\mathcal{X}, \mathcal{Z}) + \epsilon + \delta && (39)\\
&= H(\mathcal{X}) + H(\mathcal{Z}|\mathcal{X}) + \epsilon + \delta && (40)\\
&\leq H(\mathcal{X}) + H(\mathcal{Z}) + \epsilon + \delta && (41)\\
&\leq \sum_{j=1, j\neq q}^{\lfloor N/s \rfloor} H(\bar{X}_j) + \sum_{i=1, i\neq k}^{s} H(Z_i) + \epsilon + \delta && (42)\\
&\leq (\lfloor N/s \rfloor - 1)\frac{K-s}{K}RF + (s-1)MF + \epsilon + \delta. && (43)
\end{aligned}
$$

Note that step (36) is due to (5). Therefore, we can get

$$
R_T \geq \frac{K[(s\lfloor N/s \rfloor - 1) - (s-1)M]}{(\lfloor N/s \rfloor - 1)(K-s)}. \tag{44}
$$

Taking into account all possible cuts, we obtain the lower bound stated in the following theorem.

Theorem 3: The achievable secure rate is lower bounded by

$$
R_T^* \geq \max_{s \in \{1,2,\ldots,\min(K, N/2)\}} \frac{K[(s\lfloor N/s \rfloor - 1) - (s-1)M]}{(\lfloor N/s \rfloor - 1)(K-s)}. \tag{45}
$$

■

Remark 6: We note that $R_T^* \geq \frac{K}{K-1}$, which is obtained by setting $s = 1$ in Theorem 3, can be achieved whenever $M \geq N(K-1)$, using the proposed scheme in subsection III-F. ■

In addition, the multiplicative gap between the upper bound in Theorem 1 and the above lower bound is bounded by a constant as stated in the following theorem.

Theorem 4: For $M \geq 2 + \frac{N}{K-1}$, there exists a constant, c, independent of all the system parameters, such that

$$
1 \leq \frac{R_T^C}{R_T^*} \leq c. \tag{46}
$$

■

Proof: See the Appendix. □

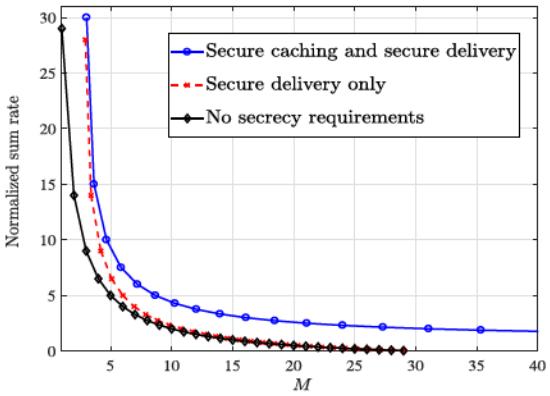

Fig. 2. Comparison between the required transmission rates under different system requirements for $N = K = 30$.

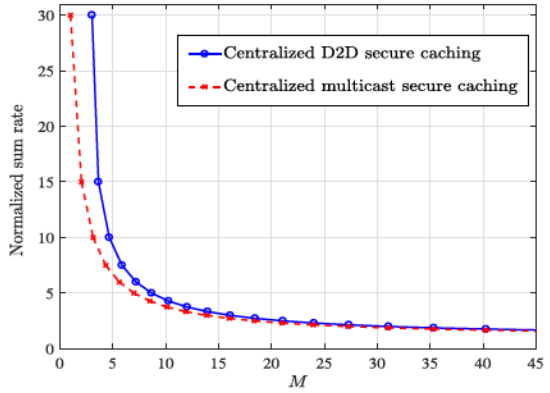

Fig. 3. The achievable secure rates for the single server and D2D coded caching for $N = K = 30$.

VI. Numerical Results

In this section, we demonstrate the performance of the proposed schemes numerically. Fig. 2 shows the performance of D2D coded caching systems under different requirements. In particular, we compare our system that provides both secure caching and secure delivery with the system with just secure delivery [23] and the one with no secrecy constraints [15]. For the latter two cases, the rate is equal to zero wherever $M \geq N$, as the entire database can be stored in each cache memory. However, by setting $s = 1$ in the lower bound stated in Theorem 3, we get that the sum rate under secure caching is bounded below by $\frac{K}{K-1}$.

In Fig. 3, we compare the performance of our system and the one considered in [26]. As expected, the system, in [26], where the server, with full access to the file shares, is responsible for the delivery phase, achieves lower transmission rate compared with the considered setup where the delivery phase has to be performed by users, each of which has limited access to the files shares. Interestingly, we observe that the gap between the required transmission rates vanishes as M increases, i.e., the loss due to accessing a limited number of shares at each user is negligible when M is sufficiently large.

Fig. 4 shows that the gap between the lower and upper bounds decreases as M increases. As mentioned before, Theorem 3 points out that the sum rate is bounded below by $\frac{K}{K-1}$, by setting $s = 1$. For large enough M, our proposed schemes achieves a sum rate equal to $\frac{K}{K-1}$, which matches the lower bound.

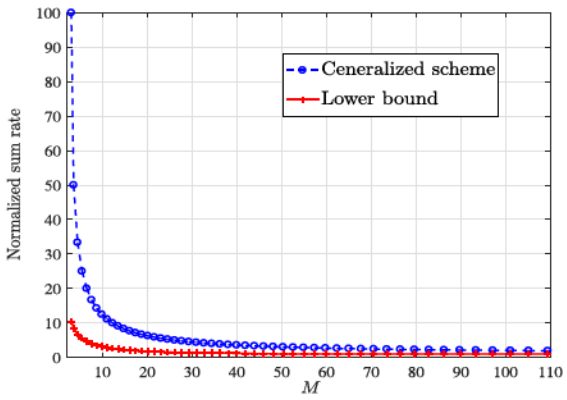

Fig. 4. The upper bound vs the lower bound for $N = K = 100$.

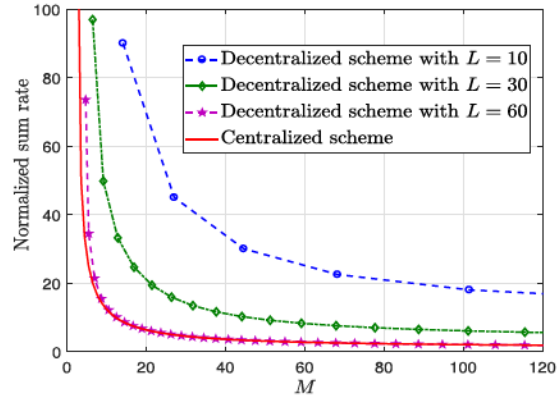

Fig. 5. Achievable rates via decentralized and centralized schemes $N = K = 100$.

In Fig. 5, we plot the achievable rates under different choices of L in a system with $K = 100$. It is worth noting that even with $L = 60$, which is much smaller than the number of users in the system (K), the gap between the achievable rate using the decentralized and centralized schemes is negligible for realistic values of M. In other words, even with a inaccurate lower bound of the number of users in the system K, the proposed decentralized scheme performs very close to the centralized one.

VII. Conclusion

In this work, we have characterized the fundamental limits of secure device-to-device coded caching systems. We have investigated a cache-aided network, where the users' requests must be served via D2D communications only. We have imposed secure caching constraint on all users, i.e., a user cannot obtain any information about any file that he had not requested. We have developed an achievable centralized coded caching scheme for this network, where the server encodes each file using a proper non-perfect secret sharing scheme and generates a set of random keys. The resulting shares and keys are carefully placed in the users' cache during the cache placement phase. After announcing the users' demands, each user transmits a one-time padded signal to the remaining users. In addition, we have provided a sequential decentralized scheme that does not require the knowledge of the number of the active users for cache placement. As a byproduct of the proposed achievability schemes, the system also keeps the files secret from any external eavesdropper that overhears the

delivery phase, i.e., the secure delivery is also guaranteed. We have derived a lower (converse) bound based on cut-set arguments. Furthermore, we have shown that our proposed scheme is order-optimal and optimal in the large memory region. Our numerical results indicate that the gap between the lower and upper bounds decreases as the cache memory capacity increases. Similarly, the performance of centralized and decentralized schemes are very close for large memories. Overall, we have shown that the D2D communications can replace the server in the delivery phase with a negligible transmission overhead. This offers an affirmation of D2D communications' significant role in upcoming communication systems.

APPENDIX A
PROOF OF THEOREM 4

First, we show that the multiplicative gap between the achievable rate in [26] for multicast coded caching and the achievable rate in Theorem 1 can be bounded by a constant. We recall the upper and lower bounds from [26].

$$R_{\text{Multicast}} \triangleq \frac{K(N+M-1)}{N+(K+1)(M-1)}, \tag{47}$$

$$R^*_{\text{Multicast}} \geq \max_{s\in\{1,2,\ldots,\min(K,N/2)\}} \frac{(s\lfloor N/s\rfloor-1)-(s-1)M}{(\lfloor N/s\rfloor-1)}. \tag{48}$$

Therefore, we have

$$\frac{R^C_T}{R_{\text{Multicast}}} = \frac{2(N+(K+1)(M-1))}{1+(M-1)K+\sqrt{(1-(M-1)K)^2-4KN}}. \tag{49}$$

To simplify the notation, let $U = M-1$ and $V = \sqrt{(1-KU)^2-4KN}$. Then, we have

$$\frac{R^C_T}{R_{\text{Multicast}}} = \frac{2KU}{1+KU+V}+\frac{2U}{1+KU+V}+2\frac{N}{1+KU+V}, \tag{50}$$

$$\leq 2+1+2\frac{N}{1+KU+V}. \tag{51}$$

Note that the minimum value of $M = \frac{N}{K-1}+2$, thus we have $U \geq \frac{N}{K-1}$ and

$$\frac{R^C_T}{R_{\text{Multicast}}} \leq 3+2\frac{N}{1+\frac{KN}{K-1}+V} \leq 5 = c'. \tag{52}$$

Now, consider

$$\frac{R^C_T}{R^*_T} = \frac{R^C_T}{R_{\text{Multicast}}}\times\frac{R_{\text{Multicast}}}{R^*_T} \leq \frac{R^C_T}{R_{\text{Multicast}}}\times\frac{R_{\text{Multicast}}}{R^*_{\text{Multicast}}}, \tag{53}$$

$$\leq c'\times c'' = c. \tag{54}$$

We observe that for any value for s, the RHS of (45) equals $\frac{K}{K-s}$ RHS of (48), thus we have $R^*_{\text{Multicast}} \leq R^*_T$ and we can get we get (53). The last step follows from [26, Theorem 3], i.e., $\frac{R_{\text{Multicast}}}{R^*_{\text{Multicast}}} \leq c''$, where c'' is a constant independent of the system parameters.

REFERENCES

[1] K. Doppler, M. Rinne, C. Wijting, C. B. Ribeiro, and K. Hugl, "Device-to-device communication as an underlay to LTE-advanced networks," *IEEE Commun. Mag.*, vol. 47, no. 12, pp. 42–49, Dec. 2009.

[2] M. N. Tehrani, M. Uysal, and H. Yanikomeroglu, "Device-to-device communication in 5G cellular networks: Challenges, solutions, and future directions," *IEEE Commun. Mag.*, vol. 52, no. 5, pp. 86–92, May 2014.

[3] A. Asadi, Q. Wang, and V. Mancuso, "A survey on device-to-device communication in cellular networks," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 4, pp. 1801–1819, Nov. 2014.

[4] L. W. Dowdy and D. V. Foster, "Comparative models of the file assignment problem," *ACM Comput. Surv.*, vol. 14, no. 2, pp. 287–313, 1982.

[5] K. C. Almeroth and M. H. Ammar, "The use of multicast delivery to provide a scalable and interactive video-on-demand service," *IEEE J. Sel. Areas Commun.*, vol. 14, no. 6, pp. 1110–1122, Aug. 1996.

[6] M. A. Maddah-Ali and U. Niesen, "Fundamental limits of caching," *IEEE Trans. Inf. Theory*, vol. 60, no. 5, pp. 2856–2867, May 2014.

[7] M. A. Maddah-Ali and U. Niesen, "Coding for caching: Fundamental limits and practical challenges," *IEEE Commun. Mag.*, vol. 54, no. 8, pp. 23–29, Aug. 2016.

[8] U. Niesen and M. A. Maddah-Ali, "Coded caching with nonuniform demands," *IEEE Trans. Inf. Theory*, vol. 63, no. 2, pp. 1146–1158, Feb. 2017.

[9] S. P. Shariatpanahi, S. A. Motahari, and B. H. Khalaj, "Multi-server coded caching," 2015, *arXiv:1503.00265*. [Online]. Available: https://arxiv.org/abs/1503.00265

[10] Q. Yang and D. Gündüz, "Coded caching and content delivery with heterogeneous distortion requirements," *IEEE Trans. Inf. Theory*, vol. 64, no. 6, pp. 4347–4364, Jun. 2018.

[11] S. S. Bidokhti, M. Wigger, and A. Yener, "Benefits of cache assignment on degraded broadcast channels," 2017, *arXiv:1702.08044*. [Online]. Available: https://arxiv.org/abs/1702.08044

[12] A. M. Ibrahim, A. A. Zewail, and A. Yener, "Centralized coded caching with heterogeneous cache sizes," in *Proc. IEEE WCNC*, Mar. 2017, pp. 1–6.

[13] A. M. Ibrahim, A. A. Zewail, and A. Yener, "Optimization of heterogeneous caching systems with rate limited links," in *Proc. IEEE ICC*, May 2017, pp. 1–6.

[14] A. A. Zewail and A. Yener, "Combination networks with or without secrecy constraints: The impact of caching relays," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 6, pp. 1140–1152, Jun. 2018.

[15] M. Ji, G. Caire, and A. F. Molisch, "Fundamental limits of caching in wireless D2D networks," *IEEE Trans. Inf. Theory*, vol. 62, no. 2, pp. 849–869, Feb. 2016.

[16] M. Ji, G. Caire, and A. F. Molisch, "Wireless device-to-device caching networks: Basic principles and system performance," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 1, pp. 176–189, Jan. 2016.

[17] M. Ji, R.-R. Chen, G. Caire, and A. F. Molisch, "Fundamental limits of distributed caching in multihop D2D wireless networks," in *Proc. IEEE ISIT*, Jun. 2017, pp. 2950–2954.

[18] A. Shabani, S. P. Shariatpanahi, V. Shah-Mansouri, and A. Khonsari, "Mobility increases throughput of wireless device-to-device networks with coded caching," in *Proc. IEEE ICC*, May 2016, pp. 1–6.

[19] A. Tebbi and C. W. Sung, "Coded caching in partially cooperative D2D communication networks," 2017, *arXiv:1709.06281*. [Online]. Available: https://arxiv.org/abs/1709.06281

[20] A. K. Chorppath, J. Hackel, and F. H. P. Fitzek, "Network coded caching and D2D cooperation in wireless networks," in *Proc. EWC*, May 2017, pp. 1–6.

[21] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Trans. Inf. Theory*, vol. 46, no. 2, pp. 388–404, Mar. 2000.

[22] A. Sengupta, R. Tandon, and T. C. Clancy, "Fundamental limits of caching with secure delivery," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 2, pp. 355–370, Feb. 2015.

[23] Z. H. Awan and A. Sezgin, "Fundamental limits of caching in D2D networks with secure delivery," in *Proc. IEEE ICC Workshops*, Jun. 2015. pp. 464–469.

[24] A. A. Zewail and A. Yener, "Coded caching for resolvable networks with security requirements," in *Proc. IEEE CNS Workshops*, Oct. 2016, pp. 621–625.

[25] Z. H. Awan and R. Mathar, "Bounds on caching D2D networks with secure delivery," in *Proc. IEEE ISWCS*, Aug. 2018, pp. 1–5.

[26] V. Ravindrakumar, P. Panda, N. Karamchandani, and V. M. Prabhakaran, "Private coded caching," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 3, pp. 685–694, Mar. 2018.

[27] M. Ito, A. Saito, and T. Nishizeki, "Secret sharing scheme realizing general access structure," *Electron. Commun. Jpn.*, vol. 72, no. 9, pp. 56–64, 2010.
[28] M. Ito, A. Saito, and T. Nishizeki, "Secret sharing scheme realizing general access structure," in *Proc. IEEE Globecom*, Nov. 1987, pp. 99–102.
[29] Q. Li, X. X. Li, X. J. Lai, and K. F. Chen, "Optimal assignment schemes for general access structures based on linear programming," *Designs, Codes Cryptogr.*, vol. 74, no. 3, pp. 623–644, Mar. 2015.
[30] G. R. Blakley and C. Meadows, "Security of ramp schemes," in *Proc. Workshop Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 1984, pp. 242–268.
[31] C. E. Shannon, "Communication theory of secrecy systems," *Bell Labs Tech. J.*, vol. 28, no. 4, pp. 656–715, Oct. 1949.
[32] S. Jin, Y. Cui, H. Liu, and G. Caire, "New order-optimal decentralized coded caching schemes with good performance in the finite file size regime," 2016, *arXiv:1604.07648*. [Online]. Available: https://arxiv.org/abs/1604.07648
[33] A. A. Zewail and A. Yener, "Fundamental limits of secure device-to-device coded caching," in *Proc. IEEE Asilomar*, Nov. 2016, pp. 1414–1418.
[34] H. Yamamoto, "Secret sharing system using (k, L, n) threshold scheme," *Electron. Commun. Jpn. (Part I: Commun.)*, vol. 69, no. 9, pp. 46–54, 1986.
[35] R. Cramer, I. B. Damgard, and J. B. Nielsen, *Secure Multiparty Computation and Secret Sharing*. Cambridge, U.K.: Cambridge Univ. Press, 2015.
[36] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979.
[37] M. A. Maddah-Ali and U. Niesen, "Decentralized coded caching attains order-optimal memory-rate tradeoff," *IEEE/ACM Trans. Netw.*, vol. 23, no. 4, pp. 1029–1040, Aug. 2015.
[38] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. Hoboken, NJ, USA: Wiley, 2006.
[39] A. Sengupta and R. Tandon, "Beyond cut-set bounds—The approximate capacity of D2D networks," in *Proc. IEEE ITA*, Feb. 2015, pp. 78–83.

Ahmed A. Zewail (S'07–M'19) received the B.Sc. degree (Hons.) in electrical engineering from Alexandria University, Alexandria, Egypt, in 2011, the M.Sc. degree in wireless communications from Nile University, Giza, Egypt, in 2013, and the Ph.D. degree from Pennsylvania State University, in 2019. His graduation project was about developing warehouse management systems using RFID and received the first place in INDAC-Siemens 2011 Competition. His M.Sc. thesis focused on the capacity and degrees of freedom of relay networks. His Ph.D. dissertation focused on secrecy guarantees in emerging networks, e.g., untrusted relay networks and cache-aided networks. From 2011 to 2013, he was a Research Assistant at the Wireless Intelligence Networks Center (WINC), Nile University. From 2013 to 2019, he was a Research Assistant at the Wireless Communications and Networking (WCAN) Laboratory, Pennsylvania State University. He is currently with Qualcomm, Inc., San Diego, CA, USA. His research interests include network information theory, cache-aided networks, wireless communications, and physical-layer security.

Aylin Yener (S'91–M'01–SM'14–F'15) received the B.Sc. degree in electrical and electronics engineering and the B.Sc. degree in physics from Bogazici University, Istanbul, Turkey, and the M.S. and Ph.D. degrees in electrical and computer engineering from the Wireless Information Network Laboratory (WINLAB), Rutgers University, New Brunswick, NJ, USA.

She is currently a Distinguished Professor of electrical engineering at The Pennsylvania State University, University Park, PA, USA, where she joined the faculty as an Assistant Professor in 2002. Since 2017, she has been the Dean's Fellow of the College of Engineering, The Pennsylvania State University. She was a Visiting Professor of electrical engineering at Stanford University from 2016 to 2018, where she was a Visiting Associate Professor from 2008 to 2009. Her current research interests include information security, green communications, caching systems, and more generally in the fields of information theory, communication theory, and networked systems. She was a recipient of the NSF CAREER Award in 2003, the Best Paper Award in Communication Theory from the IEEE International Conference on Communications in 2010, the Penn State Engineering Alumni Society (PSEAS) Outstanding Research Award in 2010, the IEEE Marconi Prize Paper Award in 2014, the PSEAS Premier Research Award in 2014, the Leonard A. Doggett Award for Outstanding Writing in Electrical Engineering at Penn State in 2014, the IEEE Women in Communications Engineering Outstanding Achievement Award in 2018, and the IEEE Communications Society Best Tutorial Paper Award in 2019. She is currently a Distinguished Lecturer for the IEEE Information Theory Society for the period of 2019–2020, the IEEE Communications Society for the period of 2018–2020, and the IEEE Vehicular Technology Society for the period of 2017–2021. She is serving as the Vice President for the IEEE Information Theory Society in 2019. Previously, she was the Second Vice President in 2018, a member of the Board of Governors from 2015 to 2018, and the Treasurer of the IEEE Information Theory Society from 2012 to 2014. She has served as the Student Committee Chair for the IEEE Information Theory Society from 2007 to 2011, and was the Co-Founder of the Annual School of Information Theory in North America in 2008. She was the Technical (Co)-Chair for various symposia/tracks at the IEEE ICC, PIMRC, VTC, WCNC, and Asilomar, in 2005, 2008–2014, and 2018, respectively. Previously, she has served as an Editor for the IEEE TRANSACTIONS ON COMMUNICATIONS from 2009 to 2012, an Editor for the IEEE TRANSACTIONS ON MOBILE COMPUTING from 2017 to 2018, and an Editor and an Editorial Advisory Board Member for the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS from 2001 to 2012. She has also served a Guest Editor for the IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY in 2011, and the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS in 2015. She currently serves as a Senior Editor for the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, and is on the inaugural Senior Editorial Board for the IEEE JOURNAL ON SELECTED AREAS IN INFORMATION THEORY.