Fundamental Limits of Secure Device-to-Device Coded Caching

Ahmed A. Zewail and Aylin Yener

Wireless Communications and Networking Laboratory (WCAN)
The School of Electrical Engineering and Computer Science
The Pennsylvania State University, University Park, PA 16802.
zewail@psu.edu yener@engr.psu.edu

***Abstract*—We consider a device-to-device coded caching system, where each user is guaranteed recover its requested file and is simultaneously prevented from recovering any file it did not request. We jointly optimize the cache placement and delivery policies such that a set of end users are able to satisfy their requests while preserving the confidentiality constraints. We develop an upper bound utilizing secret sharing schemes and one-time pad keying as well as a lower bound on the required transmission rate. Numerical results indicate that the gap between the bounds vanishes with increasing memory size.**

I. Introduction

Caching is an effective technique to alleviate congestion in communication networks during peak traffic times. Caching is implemented in two phases. First, functions of data contents are stored in cache memories of the users during the low traffic period, known as *the cache placement phase*. When the users actually request the files, there is then no need to download the contents that have already been placed in the user's cache. Additionally, the cache contents allow the system to create multicast opportunities, known as the global caching gain, during *the delivery phase*. References [1] and [2] have characterized the fundamental limits of caching establishing the trade-off between the cache size and the transmission rate needed to satisfy the users' requests, known as the rate-memory trade-off.

Device-to-device (D2D) communications is considered a promising paradigm for the evolving 5G architecture [3]. Instead of routing data via the network infrastructure, D2D communication invokes the radio technology that enables devices to communicate directly with each other. As the user demand for high data rate services, such as video streaming, continues to grow, D2D communications is expected to be deployed pervasively, improving spectral efficiency, throughput, energy efficiency, and delay performance of the network [4].

Recently, references [5] and [6] have studied a cache-aided network under confidentiality constraints. In particular, reference [5] has considered a network where a single server, with N files, each with size F bits, connected to K users, each equipped with a cache memory of size MF bits, via a noiseless link, with the requirement that each user recovers only its requested file and none of the remaining files. Additionally, references [7] and [8] have studied the fundamental limits of caching with the confidentiality requirement that any external eavesdropper that overhears the transmitted signals during the delivery phase should not gain any information about the data in the system, known as secure delivery.

In this work, we investigate the fundamental limits of secure coded caching in *device-to-device (D2D) networks*. Unlike previous work in [5], the server does not participate in the delivery phase, and users' requests must be satisfied via D2D communications only. Consequently, we reduce the load on the main server, which is a worthy objective as advocated in references [2]–[4]. The system operates over two successive phases. First, in the cache placement phase, the server stores functions of the N files in the users' cache. Second, in the delivery phase, each user requests one of the N files, and advertises its request to all network users. After obtaining the requests of all network users, each user utilizes its cache to transmit a signal that helps the other users satisfy their requests; the server remains silent over this phase. At the end of the delivery phase, each user is guaranteed to be able to reconstruct its requested file from the content of its cache and the received signals from the remaining $K-1$ users, and simultaneously must be prevented from obtaining any information about the remaining $N-1$ files. This model can capture a scenario, where the users have a subscription service where each of them can request limited content in a given time frame, e.g., per day.

For this D2D model, we derive lower and upper bounds on the rate-memory trade-off. To obtain the upper bound, we jointly optimize the cache placement and delivery phases. The server encodes each file using *secret sharing schemes*, and generates a set of random keys. The server places these file shares and keys in the cache memories of the users. For a file W with size F bits, an (m, n) secret sharing scheme generates n shares, $S_1, S_2, ..S_n$, such that accessing any m shares does not reveal any information about W, i.e.,

$$I(W; \mathcal{S}) = 0, \quad \forall \mathcal{S} \subseteq \{S_1, S_2, ..S_n\}, |\mathcal{S}| \leq m. \tag{1}$$

Furthermore, W can be reconstructed from n shares, i.e.,

$$H(W|S_1, S_2, .., S_n) = 0. \tag{2}$$

This work was supported in part by the National Science Foundation Grants CCF 13-19338 and CNS 13-14719.

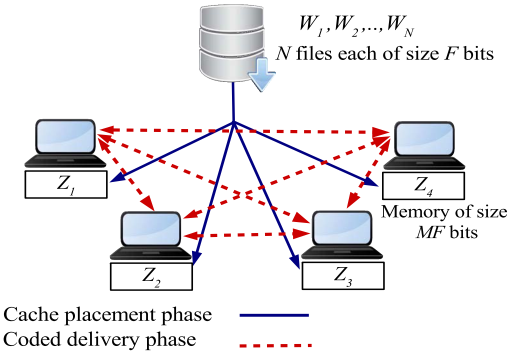

Fig. 1: Device-to-device coded caching system, with $K = 4$.

For large enough F, an (m, n) secret sharing scheme exists with shares of size equal to $\frac{F}{n-m}$ bits [9].

The lower bound is developed based on cut-set arguments. Our numerical results demonstrate that the gap between the lower and upper bounds vanishes as the memory size increases, and for systems with realistic parameters, the lower and upper bounds are observed to coincide.

Our 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. That is, the performance of the system under investigation and that of the one in [5], which must utilize the server, are very close to one another for realistic values of the system parameters.

The remainder of the paper is organized as follows. In Section II, we describe the system model. Section III states our main results. In Section IV, we detail the achievability technique. Section V contains the derivation of the lower bound. In Section VI, we provide numerical results to illustrate the system performance. Section VII summarizes our conclusions.

II. System Model

Consider a network where a central server, which stores a database of N files, $W_1, .., W_N$, each with size F bits, is connected to K users via a public noiseless link, similar to references [1], [5]–[8]. Each user is equipped with a cache memory with size MF bits, i.e., M denotes the normalized cache memory size. Let Z_k represent the content of the cache memory at user k. The system operates over two phases, cache placement and coded delivery, as depicted in Fig. 1.

A. Cache placement phase

In this phase, the server allocates functions of its database in the end users' cache. These possible allocations are designed to preserve the memory capacity constraint at each user without the knowledge of the file it will request in the future.

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

$$Z_k = \phi_k(W_1, W_2, .., W_N), \qquad k = 1, 2, .., K, \tag{3}$$

where $\phi_k : [2^F]^N \to [2^F]^M$, i.e., $H(Z_k) \leq MF$. ■

B. Delivery phase

During peak traffic, each user requests a randomly selected file [1]. We define d_k to denote the index of the file requested by user k, i.e., $d_k \in \{1, 2, .., N\}$, and $\boldsymbol{d} = (d_1, d_2.., d_K)$ to represent the demand vector of all network users at any request instance. Similar to [2] and [8], we assume that the users' requests must be satisfied via D2D communications only, i.e., the server does not participate in the delivery phase. With knowledge of the demand vector $\boldsymbol{d}$, user k maps the content of its cache memory, Z_k, into a signal that is transmitted to all network users over a noiseless interference-free link. From the $K-1$ received signals and Z_k, user k reconstructs W_{d_k}.

Definition 2. *(Coded Delivery): The delivery phase is defined by a set of encoding and decoding functions at each user. In particular, the mapping from the content of the cache memory of user k, and the demand vector* **d** *into the transmitted signal by user k is represented by the encoding function*

$$X_{k,\boldsymbol{d}} = \psi_k(Z_k, \boldsymbol{d}), \qquad k = 1, 2, .., K, \tag{4}$$

where $\psi_k : [2^F]^M \times \{1, ..N\}^K \to [2^F]^{R_k}$, and R_k is the normalized rate of the transmitted signal by user k. In addition, user k has a decoding function to recover its requested file

$$\hat{W}_{d_k} = \mu_k(Z_k, \boldsymbol{d}, X_{1,\boldsymbol{d}}, .., X_{k-1,\boldsymbol{d}}, X_{k+1,\boldsymbol{d}}, .., X_{K,\boldsymbol{d}}), \tag{5}$$

where $\mu_k : [2^F]^M \times \{1, ..N\}^K \times [2^F]^{\sum_{i\neq k} R_i} \to [2^F]$, and $k = 1, 2, .., K$. ■

We define $R_T^C = \sum_{i=1}^{K} R_i$ to denote the **normalized sum rate** of the transmitted signals by all network users at the request instance, noting that the primary goal of coded caching is to reduce this rate.

C. System Requirements

As illustrated above, the main server remains silent during the delivery phase, thus all users' requests must be satisfied via D2D communications. Furthermore, we impose confidentiality constraints on the database files. In particular, each user should be able to decode its requested file, however, it must not be able to gain any information about the content of the remaining $N-1$ files. We refer to these confidentiality requirements as ***secure caching***. Therefore, we have the following definition for a memory-rate pair to be *securely achievable*.

Definition 3. *The secure memory-rate pair (M, R_T^C) is said to be achievable if $\forall \epsilon, \delta > 0$ and $F \to \infty$, 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 the following constraints are satisfied*

$$\max_{\boldsymbol{d}, k \in \{1,..,K\}} Pr(\hat{W}_{d_k} \neq W_{d_k}) \leq \epsilon, \tag{6}$$

$$\max_{\boldsymbol{d},k\in\{1,..,K\}} I(\boldsymbol{W}_{-d_k}; X_{1,\boldsymbol{d}},..,X_{K,\boldsymbol{d}},Z_k) \le \delta, \tag{7}$$

where $\boldsymbol{W}_{-d_k} = \{W_1,..,W_N\}\backslash\{W_{d_k}\}$, i.e., set of all files except the one requested by user k. ■

The optimal secure memory-rate trade-off, i.e., the lower bound on the normalized sum rate, is defined as

$$R_T^{C*} = \inf\{R_T^C : (M, R_T^C) \text{ is securely achievable}\}. \tag{8}$$

III. Main Results

In this section, we present the main results of this paper.

Theorem 1. *For $M = \frac{Nt}{K-t} + \frac{1}{t} + 1$, and $t \in \{1,2,..,K-1\}$, the following secure rate is achievable*

$$R_T^C \le \frac{2K(N+M-1)}{1+(M-1)K+\sqrt{(1-(M-1)K)^2-4KN}}. \tag{9}$$

Moreover, the convex envelope of the above points, defined for each M, is also achievable. □

Theorem 2. *For $1 \le M \le N(K-1)$, the achievable secure rate is lower bounded by*

$$R_T^{C*} \ge \max_{s\in\{1,2,..,\min(K,N/2)\}} \frac{s\lfloor N/s\rfloor - 1 - (s-1)M}{\lfloor N/s\rfloor - 1}. \tag{10}$$

□

It is worth mentioning that under (7), each user should not be able to recover any file from the content of its cache memory. This implies that the transmission rate will be strictly positive even for large values of M. When secure delivery is the only requirement in the system [7], [8], it is easy to see that the transmission rate is equal to zero whenever $M \ge N$. On the other hand, it is evident from (10), by setting $s=1$, that the normalized sum rate is bounded below by 1. A similar conclusion was shown in [5], for multicast delivery phase by the server, where this bound is tight for large values of M, i.e., $M = N(K-1)$. The numerical results in Section VI indicate that the performance of the multicast secure coded caching in [5], and the performance of device-to-device secure coded caching, that we propose in this paper, are virtually the same for realistic values of the system parameters.

IV. Achievability

In this section, we detail the derivation of the upper bound in Theorem 1.

A. Cache placement phase

For $M = \frac{Nt}{K-t} + \frac{1}{t} + 1$, and $t \in \{1,2,..,K-1\}$, each file in the database is encoded using a secret sharing scheme [9]. In particular, a file, W_n, is encoded using $(t\binom{K-1}{t-1}, t\binom{K}{t})$ secret sharing scheme. We obtain $t\binom{K}{t}$ shares, each with size F_s bits, where

$$F_s = \frac{F}{t\binom{K}{t} - t\binom{K-1}{t-1}} = \frac{F}{(K-t)\binom{K-1}{t-1}}. \tag{11}$$

Each share is denoted by $S^j_{n,\mathcal{T}}$, where n is the file index i.e., $n \in \{1,..,N\}$, $j = 1,..,t$, and $\mathcal{T} \subseteq \{1,..,K\}, |\mathcal{T}| = t$. The server allocates the shares $S^j_{n,\mathcal{T}}$, $\forall j, n$ in the cache of user k whenever $k \in \mathcal{T}$.

Furthermore, the server generates $(t+1)\binom{K}{t+1}$ independent keys. Each key, $K^i_{\mathcal{T}_K}$, is uniformly distributed over $\{1,..,2^{F_s}\}$, where $i = 1,..,t+1$, and $\mathcal{T}_K \subseteq \{1,..,K\}, |\mathcal{T}_K| = t+1$. User k stores the keys $K^i_{\mathcal{T}_K}$, $\forall i$, if $k \in \mathcal{T}_K$.

Remark 1. In this placement scheme, each user stores $Nt\binom{K-1}{t-1}$ shares and $(t+1)\binom{K-1}{t}$ keys, thus the accumulated number of bits to be stored in each cache memory is given by

$$Nt\binom{K-1}{t-1}F_s + (t+1)\binom{K-1}{t}F_s = \frac{Nt}{K-t}F + (1+\frac{1}{t})F = MF. \tag{12}$$

Clearly, the proposed scheme satisfies the cache capacity constraint at each user. Also, from (12), we can get

$$t = \frac{1+(M-1)K+\sqrt{(1-(M-1)K)^2-4KN}}{2(N+M-1)}. \tag{13}$$

■

B. Coded Delivery phase

We focus our attention on the worst case scenario, where the K users request K distinct files. At each transmission instance, we consider a set $\mathcal{S} \subseteq \{1,..,K\}$, where $|\mathcal{S}| = t+1$. User k, where $k \in \mathcal{S}$, multicasts the following signal of length F_s bits

$$K^i_{\mathcal{S}} \oplus_{l\in\mathcal{S}\backslash\{k\}} S^j_{d_l,\mathcal{S}\backslash\{l\}}, \tag{14}$$

where the index i is chosen in way that guarantees the uniqueness of used keys at each transmission, and the index j is chosen to ensure that each transmission is formed by shares that had not been transmitted before. From the cache placement phase, we can 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. There are $\binom{K}{t+1}$ different choices of the set S, and for each choice $t+1$ signals of length F_s bits are transmitted, therefore the total number of the transmitted bits is given by

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

Consequently, we achieve the following normalized sum rate

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

Remark 2. At the end of the $(t+1)\binom{K}{t+1}$ transmissions, user k obtains $t\binom{K-1}{t}$ different shares from its requested file, in addition to the shares in its cache. Therefore, user k can reconstruct the requested file from its $t\binom{K}{t}$ shares.

On the other hand, user k does not obtain any new information about the shares of the other files. Note that, at any instance if user k belongs to the set $\mathcal{S}$, then the transmitted signals are formed from the shares of W_{d_k} and shares that have been already placed in its cache memory during the cache placement phase. For the other case, where user k does not belong to the set $\mathcal{S}$, all the transmitted signals are encrypted using one-time pads, which are unknown at user k, thus, user k cannot gain any information from these signals. Furthermore, 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 at each transmission, we can ensure the uniqueness use of each key. Therefore, the secrecy of the transmitted signals, from any external wiretapper that accesses the network links during the delivery phase, is also ensured, i.e., for $\delta > 0$, we have

$$\max_{\boldsymbol{d}} I(W_1, W_2, .., W_N; X_{1,\boldsymbol{d}}, .., X_{K,\boldsymbol{d}}) \leq \delta. \tag{17}$$

That is, we have secure delivery as well as secure caching. ■

Using memory sharing techniques as explained in [1] and [2], whose details we omit due to space constraints, the system can achieve the convex envelope of the points given by the values $M = \frac{Nt}{K-t} + \frac{1}{t} + 1$, and $t \in \{1, 2, .., K-1\}$.

C. Example

We demonstrate our scheme by the following example. Consider a system where $K = N = 4$ and $M = \frac{11}{2}$, i.e., $t = 2$. The server encodes each file using $(6, 12)$ secret sharing scheme. In particular, for W_n, the server generates 12 shares, denoted by $S^j_{n,\mathcal{T}}$, $j = 1, 2$, $|\mathcal{T}| = 2$, each of size $F/6$ bits. Moreover, the server generates the set of keys $K^i_{\mathcal{T}_K}$, uniformly distributed over $\{1, .., 2^{F/6}\}$, where $i = 1, 2, 3$, and $|\mathcal{T}_K| = 3$. User k stores the shares $S^j_{n,\mathcal{T}}$, and keys $K^i_{\mathcal{T}_K}$ whenever $k \in \mathcal{T}$ and $k \in \mathcal{T}_K$, respectively. Note that this allocation satisfies the cache capacity constraint. 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 will transmit the following signals

$$X_{1,\boldsymbol{d}} = \left\{\begin{matrix} S^1_{2,13} \oplus S^1_{3,12} \oplus K^1_{123}, S^1_{4,13} \oplus S^1_{3,14} \oplus K^1_{134} \\ S^1_{2,14} \oplus S^1_{4,12} \oplus K^1_{124} \end{matrix}\right\},$$

$$X_{2,\boldsymbol{d}} = \left\{\begin{matrix} S^2_{1,23} \oplus S^2_{3,12} \oplus K^2_{123}, S^2_{4,23} \oplus S^2_{3,24} \oplus K^2_{234} \\ S^2_{1,24} \oplus S^2_{4,12} \oplus K^2_{124} \end{matrix}\right\},$$

$$X_{3,\boldsymbol{d}} = \left\{\begin{matrix} S^1_{1,23} \oplus S^2_{2,13} \oplus K^3_{123}, S^2_{4,13} \oplus S^1_{1,34} \oplus K^2_{134} \\ S^1_{2,34} \oplus S^1_{4,23} \oplus K^3_{234} \end{matrix}\right\},$$

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

One can observe that with the help these signals, each user utilizes the content of its cache to recover its requested file, and it cannot obtain any information about the remaining 3 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 [8]. In the delivery phase, each user participates by 3 distinct transmissions each of size $F/6$ bits, thus $R^C_T = 2$. It worth mentioning that the system in [5] achieves a normalized secure rate $\simeq 1.3$, for this small system with these parameters. This difference is due to limited access of the shares at each user, unlike the case in [5] where the server has access to all shares during the delivery phase.

V. Lower Bound

The lower bound is based on cut-set arguments, similar to [2] and [5]. Assume that the first s users, $s \in \{1, 2, .., \min(N/2, K)\}$, request the files from 1 to s, such that user i requests W_i, $i \in \{1, 2, .., s\}$, while the remaining $K-s$ users do not request any file. Define $\boldsymbol{X}_1$ to represent the transmitted signals by the users to serve these requests, i.e., $\boldsymbol{X}_1 = \{X_{1,(1,..,s)}, .., X_{K,(1,..,s)}\}$. At the following 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 $\boldsymbol{X}_2 = \{X_{1,(s+1,..,2s)}, .., X_{K,(s+1,..,2s)}\}$. 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 $\boldsymbol{X}_q = \{X_{1,((q-1)s+1,..,qs)}, .., X_{K,((q-1)s+1,..,qs)}\}$, where $q \in \{1, .., \lfloor N/s \rfloor\}$. From the received signals over the request instances $1, 2, .., \lfloor N/s \rfloor$ and the information stored in its cache, i.e., Z_i, user i should be able to decode the files $i, i+s, .., i+(\lfloor N/s \rfloor - 1)s$. Now, consider the set of files that is given by $\mathcal{W} = \{W_1, .., W_{(q-1)s+k-1}, W_{(q-1)s+k+1}, .., W_{s\lfloor N/s \rfloor}\}$, i.e., set of all files except the one requested by user k at request instance q. Therefore, we have

$$\begin{aligned}
&(s\lfloor N/s \rfloor - 1)F = H(\mathcal{W}) \\
&\leq H(\mathcal{W}) - H(\mathcal{W}|\boldsymbol{X}_1, .., \boldsymbol{X}_{\lfloor N/s \rfloor}, Z_1, .., Z_s) + \epsilon && (18) \\
&= I(\mathcal{W}; \boldsymbol{X}_1, .., \boldsymbol{X}_{\lfloor N/s \rfloor}, Z_1, .., Z_s) + \epsilon && (19) \\
&= I(\mathcal{W}; \boldsymbol{X}_q, Z_k) + I(\mathcal{W}; \boldsymbol{X}_1, .., \boldsymbol{X}_{q-1}, \boldsymbol{X}_{q+1}, \\
&\qquad .., \boldsymbol{X}_{\lfloor N/s \rfloor}, Z_1, ..Z_{k-1}, Z_{k+1}, .., Z_s|\boldsymbol{X}_q, Z_k) + \epsilon. && (20)
\end{aligned}$$

Step (18) follows from condition (6). To simplify the notation, we define $\boldsymbol{\mathcal{X}} = \{\boldsymbol{X}_1, .., \boldsymbol{X}_{q-1}, \boldsymbol{X}_{q+1}, .., \boldsymbol{X}_{\lfloor N/s \rfloor}\}$ and $\boldsymbol{\mathcal{Z}} = \{Z_1, ..Z_{k-1}, Z_{k+1}, .., Z_s\}$. Now, (20) can be expressed as

$$\begin{aligned}
&I(\mathcal{W}; \boldsymbol{X}_q, Z_k) + I(\mathcal{W}; \boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Z}}|\boldsymbol{X}_q, Z_k) + \epsilon \\
&\leq I(\mathcal{W}; \boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Z}}|\boldsymbol{X}_q, Z_k) + \epsilon + \delta && (21) \\
&= H(\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Z}}|\boldsymbol{X}_q, Z_k) - H(\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Z}}|\mathcal{W}, \boldsymbol{X}_q, Z_k) + \epsilon + \delta && (22) \\
&\leq H(\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Z}}) + \epsilon + \delta && (23) \\
&= H(\boldsymbol{\mathcal{X}}) + H(\boldsymbol{\mathcal{Z}}|\boldsymbol{\mathcal{X}}) + \epsilon + \delta && (24) \\
&\leq H(\boldsymbol{\mathcal{X}}) + H(\boldsymbol{\mathcal{Z}}) + \epsilon + \delta && (25) \\
&\leq \sum_{j=1, j\neq q}^{\lfloor N/s \rfloor} H(\boldsymbol{X}_j) + \sum_{i=1, i\neq k}^{s} H(Z_i) + \epsilon + \delta && (26) \\
&\leq (\lfloor N/s \rfloor - 1)RF + (s-1)MF + \epsilon + \delta. && (27)
\end{aligned}$$

Note that step (21) is due to (7). Therefore, we get

$$R^C_T \geq \frac{(s\lfloor N/s \rfloor - 1) - (s-1)M}{\lfloor N/s \rfloor - 1}. \tag{28}$$

Taking into account all possible cuts, we obtain the lower bound stated in Theorem 2.

VI. Numerical Results

In this section, we present numerical results to demonstrate the proposed system's performance.

Fig. 2 shows the performance of D2D coded caching systems

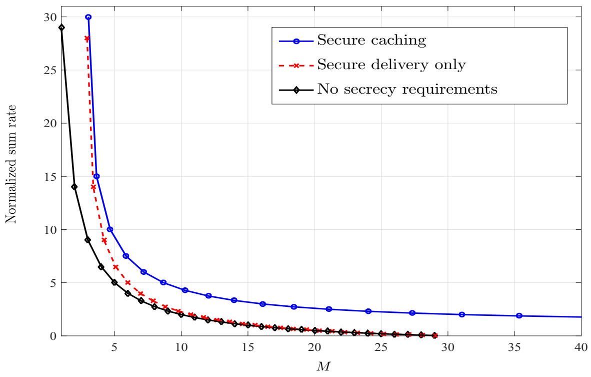

Fig. 2: Comparison between the required transmission rates under different system requirements for $N = K = 30$.

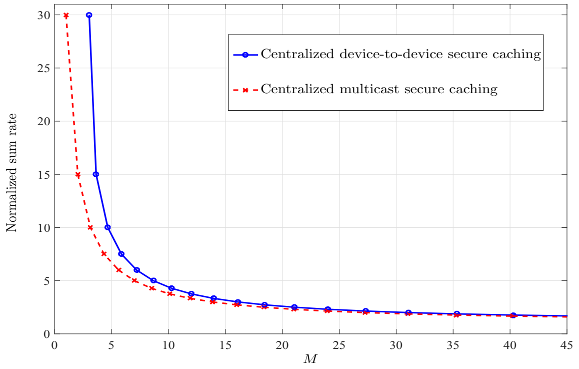

Fig. 3: The achievable secure rates for the single server and D2D coded caching for $N = K = 30$.

under different requirements. In particular, we compare our system with secure caching which also ensures secure delivery, the system with secure delivery only [8], and the system with no secrecy constraints [2]. For the latter two cases, the rate is equal to zero wherever $M \geq N$, as the whole database can be stored in the cache memory. However, as evident from Theorem 2, the rate under secure caching is bounded below by 1. In Fig. 3, we compare the performance of our system and the one considered in [5]. We can observe that the gap between the required transmission rate 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. Finally, Fig. 4 shows that the gap between the lower and upper bounds decreases as the memory size increases.

VII. Conclusions

In this work, we have characterized the fundamental limits of secure device-to-device caching. In particular, we have investigated a cache-aided network, where the users' requests should be served via device-to-device communications only. We have imposed confidentiality requirements over the cache placement and delivery phases. More specifically, each user must not obtain any information about any file that he had not requested. We have defined an achievable scheme for this network, where the server encodes each file using a proper secret sharing scheme and generates a set of random keys. The resulting shares and keys are placed in the users' cache during the cache placement phase. After the users' demands are announced to all by the server, each user transmits a signal to the remaining users encrypted with a one-time pad. As a byproduct of this achievability scheme, the system can ensure the secrecy of the files from any external eavesdropper that overhears the delivery phase as well as we have also developed a lower bound based on cut-set arguments. Our numerical results indicate that the gap between the lower and upper bounds decreases as the cache memory capacity increases.

Overall, the conclusion of this study is that D2D communications can take over the role of the server in the delivery phase with a negligible transmission overhead even in the presence of stringent confidentiality requirements. This adds further confidence to D2D communications potentially playing a significant role in upcoming communication systems [3].

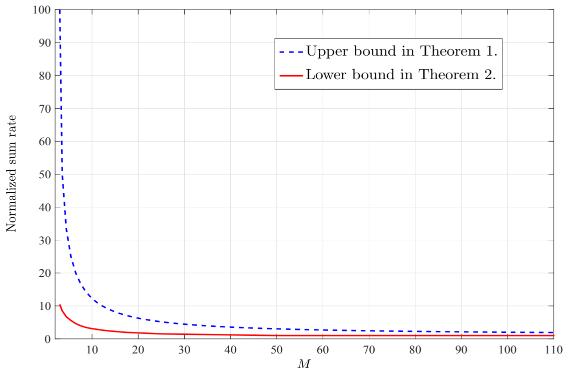

Fig. 4: The upper bound vs the lower bound for $N = K = 100$.

References

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

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

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

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

[5] N. K. V. Ravindrakumar, P. Panda and V. Prabhakaran, "Fundametal limits of secretive coded caching," in *IEEE International Symposium on Information Theory (ISIT)*, 2016.

[6] A. A. Zewail and A. Yener, "Coded caching for resolvable networks with security requirements," in *the 3rd Workshop on Physical-Layer Methods for Wireless Security, CNS'16*, 2016.

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

[8] Z. H. Awan and A. Sezgin, "Fundamental limits of caching in D2D networks with secure delivery," in *IEEE International Conference on Communication Workshop (ICCW)*, 2015.

[9] I. B. D. R. Cramer and J. B. Nielsen, *Secure Multiparty Computation and Secret Sharing*. Cambridge University Press, 2015.