Server- and User-Private Linear Function Retrieval in Multi-Server Systems with Cache-Aided Users

Qifa Yan and Daniela Tuninetti
University of Illinois Chicago, Chicago, IL 60607, USA, Email:{qifa, danielat}@uic.edu

***Abstract*—This paper investigates the ultimate performance limits of distributed multi-server systems with cache-aided users, where the users aim to retrieve a linear function of the files of a library that are replicated at multiple non-colluding servers. In addition to correct decoding, the following conditions are imposed: (a) the content of the library must be kept secure from a wiretapper who obtains all the signals sent by the servers; (b) any subset of users together can not obtain any information about the demands of the remaining users; and (c) the users' demands must be kept private against any individual server. A Distributed Key Superposition (DKS) scheme is proposed, which uses the idea of superposition of security and privacy keys to guarantee conditions (a) and (b) simultaneously, as in the single server setup. Condition (b) is guaranteed by charging each server to deliver a fraction of each file, and insuring that the privacy keys used by a server are generated and push to the user caches by another server. Interestingly, the achievable load-memory tradeoff with the additional constrain (c) is the same as the single server case if there are at least two servers.**

I. Introduction

Caches distributed across users in a network can be used to reduce the communication load. Coded caching reduces the communication load by means of creating multicast opportunities through leveraging the overlap in cache contents. Coded caching was proposed by Maddah-Ali and Niesen (MAN) for networks with a single server and multiple cache-aided users [1]. The system consists of a placement phase, in which the server pushes contents into the caches of the users without knowing their future demands, and a delivery phase, when users demand one file and the server satisfy the demands by broadcasting coded packets over a shared link. The MAN scheme achieves the optimal memory-load tradeoff among all uncoded placement schemes when $N \geq K$ [2], and after removing some redundant transmissions also for $N < K$ [3]. The scheme in [3] was generalized to the setup that allows each user to retrieve an arbitrary linear combination of the files in [4], and surprisingly showed that linear function retrieval does not increase the load compared to single file retrieval.

Content security and demand privacy are both important aspects in practical systems. Content security was studied in [5], where the content of the library must be secured against a wiretapper who obtains the signal sent by the server in the delivery phase. The key idea in [5] is that the users cache, in addition to the content as in the MAN scheme [1], also some *security keys* for the MAN uncached part of the files; this is done in a structured way so that each user can retrieve all the multicast messages needed for correct decoding. We studied demand privacy against colluding in [6], where any subset of users must not obtain any information about the demands of other users, even if they exchange the content in their caches. The key idea in [6] is that users cache, in addition to the content as in the MAN scheme [1], also some *privacy keys*. These keys are random linear combinations of the uncached parts of the files. In the delivery phase, each user decodes a linear function of the files as in [4], where the linear function is determined by both the demands and the privacy keys. In [8] we investigated the Secure and demand Private Linear Function Retrieval (SP-LFR) problem, by including in [4] the constraints in [5], [6]. We proposed to superpose (i.e., sum together) the security and privacy keys, and showed that the load-memory tradeoff is the same as in the setup with only content security guarantees.

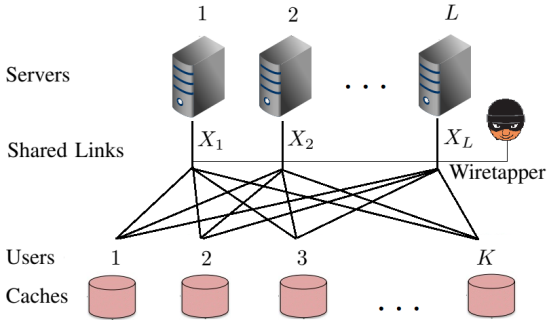


Fig. 1: System model

In this paper, we add to the setup in [8] the constraint that privacy must be guaranteed also against multiple non-colluding servers, akin to the private function retrieval setup (which only includes one user without a cache) [**?**]. The problem is illustrated in Fig. 1, where each of the $L \geq 2$ servers is connected to the users through a dedicated shared-link. We propose a Distributed Key Superposition (DKS) scheme, which borrows the key superposition idea from the single server network [8]. In particular, each file is split into L subfiles, and each server is responsible for transmitting one subfile. To guarantee privacy against servers, the privacy keys used by a server are generated and push to the user caches by another server. The overall design is organized in a structured way such that each server only needs to cache 2 subfiles out of L subfiles for each file. Interestingly, we show that the overall achieved memory-load for DKS is not increased compared to the single server case [8], where the demand privacy against the server was not guaranteed.

II. System Model

An (N, K, L) system, illustrated in Fig. 1, consists of L non-colluding servers (denoted by $1, \dots, L$) each having access to a library of N files (denotes as $W_1, W_2, \dots, W_N$). The N files are independent and uniformly distributed over $\mathbb{F}_q^B$, where $B \in \mathbb{N}^+$ is the file length and q a prima power if the finite field size. Each server is connected to K users (denoted as $1, 2, \dots, K$) via a dedicated error-free shared link.

The system has two phases.

Placement Phase: Each server $h \in [L]$ generates some random variable P_h from some finite alphabet space $\mathcal{P}_h$. Then it provides user $k \in [K]$ with some content $Z_{k,h}$, determined by P_h and the file library $W_{[N]}$, i.e.,

$$H(Z_{k,h} \,|\, P_h, W_{[N]}) = 0, \quad \forall\, k \in [K]. \tag{1}$$

User $k \in [L]$ combines the contents from all servers with some function φ_k, i.e.,

$$Z_k = \varphi_k(Z_{k,1}, Z_{k,2}, \dots, Z_{k,L}) \in \mathbb{F}_q^{MB}. \tag{2}$$

for some $M \in [0, N]$. The quantity M is referred to as *memory size*. Each server $h \in [L]$ knows the contents it provided to all the users, i.e., $Z_{[K],h}$, but does not know the contents provided by the other servers $Z_{[K],[L]\setminus\{h\}}$.

Delivery Phase: Each user $k \in [K]$ generates a demand $\mathbf{d}_k = (d_{k,1}, d_{k,2}, \dots, d_{k,N})^\top \in \mathbb{F}_q^N$, meaning it is interested in retrieving the linear combination of the files

$$\overline{W}_{\mathbf{d}_k} = \sum_{n=1}^{N} d_{k,n} \cdot W_n. \tag{3}$$

All random variables are independent, i.e.,

$$H(\mathbf{d}_{[K]}, P_{[L]}, W_{[N]}) = \sum_{k=1}^{K} H(\mathbf{d}_k) + \sum_{h=1}^{L} H(P_h) + \sum_{n=1}^{N} H(W_n). \tag{4}$$

Then user $k \in [K]$ generates L queries $Q_{k,1}, \dots, Q_{k,L}$ as functions of Z_k and $\mathbf{d}_k$, i.e.,

$$H(Q_{k,[L]} \,|\, Z_k, \mathbf{d}_k) = 0. \tag{5}$$

If any randomness is needed in the queries, it has to be stored in the cache. Then user $k \in [K]$ sends query $Q_{k,h}$ to server $h \in [L]$. Upon receiving the queries from all the users, server $h \in [L]$ creates a signal X_h by using the randomness P_h, the queries $Q_{[K],h}$ and the file library $W_{[N]}$, i.e.,

$$H(X_h \,|\, P_h, Q_{[K],h}, W_{[N]}) = 0, \quad \forall\, h \in [L], \tag{6}$$

and sends it to the users through its dedicated shared link. Every users receives

$$X = (X_1, X_2, \dots, X_L) \in \mathbb{F}_2^{RB}, \tag{7}$$

for some $R \in [0, N]$. The quantity R is referred to as *load*.

A valid scheme must satisfy the following conditions:

$$\text{[Correctness]} \qquad H(\overline{W}_{\mathbf{d}_k} \,|\, X, \mathbf{d}_k, Z_k) = 0, \ \forall k \in [K]; \tag{8}$$

$$\text{[Security]} \qquad I(W_{[N]}; X) = 0; \tag{9}$$

$$\text{[Server Privacy]} \qquad I(\mathbf{d}_{[K]}; P_h, Q_{[K],h} \,|\, W_{[N]}) = 0, \ \forall h \in [L]; \tag{10}$$

$$\text{[User Privacy]} \qquad I(\mathbf{d}_{[K]\setminus\mathcal{S}}; X, \mathbf{d}_{\mathcal{S}}, Z_{\mathcal{S}} \,|\, W_{[N]}) = 0, \quad \forall \mathcal{S} \subseteq [K], \mathcal{S} \neq \emptyset. \tag{11}$$

Objective: A memory-load pair $(M, R) \in [0, N]^2$ is said to be achievable if, for any $\epsilon > 0$ and for sufficiently large B, there exists a scheme satisfying all the conditions above, such that the memory size and the load is smaller than $M + \epsilon$ and $R + \epsilon$, respectively. The objective is to characterize the optimal memory-load tradeoff of the system, defined as

$$R^*(M) \triangleq \liminf_{B \to \infty} \big\{(M, R) : (M, R) \text{ is achievable}\big\}. \tag{12}$$

In addition to characterizing $R^*(M)$, in this paper we are also interested in the subpacketization level, defined as the minimum number B needed to realize the scheme.

Throughout this paper, we consider the case $N \geq 2$ and $L \geq 2$. In fact, privacy is impossible for $N = 1$ (i.e., there is only one possible file to be demanded). For $L = 1$, it must hold that $M \geq N$ as showed in the Appendix, but the point $(M, R) = (N, 0)$ can be trivially achieved by storing all the files at the caches of the users.

Remark 1. The constrains in (8)–(11) imply:

1) The correctness condition in (8) guarantees that each user can correctly decodes its requested lienar function.
2) The security condition in (9) guarantees that a wiretapper, who is not a user in the system and observes the delivery signal, can not obtain any information about the contents of the files in the library.
3) The server privacy condition in (10) guarantees that any individual server can not obtain any information about the demands of the users.
4) The user privacy condition in (11) guarantees that a subset of users, who exchange their cache contents and demands, cannot jointly learn any information about the demands of the other users. This notion of user privacy is the strongest among the definitions used in the literature and is motivated by the need to insure privacy regardless of the file distribution, thus the conditioning on $W_{[N]}$.

Moreover, with X in (7), the conditions in (8), (9) and (11) are exactly the same as in [**?**]. It was proved in [**?**, Appendix A] that the conditions in (9) and (11) together imply

$$I(W_{[N]}, \mathbf{d}_{[K]}; X) = 0, \tag{13}$$

that is, the wiretapper having access to X in fact can not obtain any information on both the contents of the files and the demands of the users. In other words, the random variable $W_1, \dots, W_N, \mathbf{d}_1, \dots, \mathbf{d}_K, X$ are mutually independent, where the crucial resource to insure independence in (13) is the availability of the randomness $P_{[L]}$ at the server.

Remark 2 (Minimum memory size). It was proved in [5] that, in order to guarantee the correctness condition in (8) and the security condition in (9) simultaneously, the memory size M has to be no less than one. Thus the load-memory tradeoff is thus defined for $M \in [1, N]$.

III. Main Result

The following theorem is our main result, achieved by the *Distributed Key Superposition (DKS)* scheme. The detailed description of the scheme can be found in Section IV.

Theorem 1. *For an (N, K, L) system, with $N \geq 2$ and $L \geq 2$, the lower convex envelope of the following points is achievable*

$$(M_t, R_t) = \left(1 + \frac{t(N-1)}{K}, \frac{K-t}{t+1}\right), \ t \in [0:K]. \quad (14)$$

Moreover, this load-memory tradeoff is optimal to within a multiplicative gap of at most 8 in all regimes except for small memory $M \in [1, 2)$ with $N < K$.

Remark 3 (Comparison with the Single Server Case). We proved in Appendix that server privacy and security conditions simultaneously imply $M \geq N$ if $L = 1$. In or previous work [8], we investigated the case $L = 1$ without the server privacy condition in (10). The construction therein was based on Placement Delivery Array (PDA) framework, known to give low subpacketization coded caching schemes for single file retrieval with neither security or privacy guarantees [9]. It can be observed that, the memory-load tradeoff in Theorem 1 is exactly the same as the tradeoff achieved by the PDA describing the MAN schemes in the single server case with constrains (8), (9), and (11) (see [8, Corollary 1]), which is also the same as in the single server case with only correctness and security constrains [5]. This indicates that imposing server privacy in a distributed system with $L \geq 2$ servers does not incur any additional cost in terms of load. The relationship of the DKS scheme in Theorem 1 and the key superposition scheme in [8] will be further elaborated upon in Remark 4.

We first give an example to illustrate the key idea in the DKS scheme, before describing the general scheme.

An Example: Consider an $(N, K, L) = (3, 2, 2)$ system. Let the files be $W_1, W_2, W_3 \in \mathbb{F}_2^B$, and $t = 1$. Firstly, split each file W_n into $L\binom{K}{t} = 4$ equal-size packets, denoted by

$$W_n = (W_{n,1,1}, W_{n,1,2}, W_{n,2,1}, W_{n,2,2}), \quad \forall\, n \in [3]. \quad (15)$$

For any vector $\mathbf{a} = (a_1, a_2, a_3)^\top \in \mathbb{F}_2^3$, we denote

$$\overline{W}_{\mathbf{a},h,i} = a_1 W_{1,h,i} \oplus a_2 W_{2,h,i} \oplus a_3 W_{3,h,i}. \quad (16)$$

In the placement phase, each server first generates $\binom{K}{t+1} = 1$ *security key*, and $K\binom{K-1}{t} = 2$ *privacy keys* as follows.

- Tthe security keys generated by server 1 and 2, denoted by V_1 and V_2, respectively, are uniformly and independently chosen from $\mathbb{F}_2^{B/2}$.
- The privacy keys generated as follows. Server 1 generates two vectors $\mathbf{p}_{1,2}, \mathbf{p}_{2,2}$ uniformly and independent from $\mathbb{F}_2^3$, and accordingly generates its privacy keys $\overline{W}_{\mathbf{p}_{1,2},h,2}$ and $\overline{W}_{\mathbf{p}_{2,2},h,1}$. Server 2 uniformly and independently generates 2 vectors $\mathbf{p}_{1,1}, \mathbf{p}_{2,1}$, and form its privacy keys $\overline{W}_{\mathbf{p}_{1,1},1,2}$ and $\overline{W}_{\mathbf{p}_{2,1},1,1}$.

The contents provided by the servers and cached by the users are listed in Table I. It is easy to observe that Z_k can be computed from $(Z_{k,1}, Z_{k,2})$ for $k \in [2]$.

TABLE I: The contents provided by the servers and cached by the users.

k	1	2
$Z_{k,1}$	$W_{1,1,1}, W_{2,1,1}, W_{3,1,1}$ $V_1, \overline{W}_{\mathbf{p}_{1,2},2,2}, \mathbf{p}_{1,2}$	$W_{1,1,2}, W_{2,1,2}, W_{3,1,2}$ $V_1, \overline{W}_{\mathbf{p}_{2,2},2,1}, \mathbf{p}_{2,2}$
$Z_{k,2}$	$W_{1,2,1}, W_{2,2,1}, W_{3,2,1}$ $V_2, \overline{W}_{\mathbf{p}_{1,1},1,2}, \mathbf{p}_{1,1}$	$W_{1,2,2}, W_{2,2,2}, W_{3,2,2}$ $V_2, \overline{W}_{\mathbf{p}_{2,1},1,1}, \mathbf{p}_{2,1}$
Z_k	$W_{1,1,1}, W_{2,1,1}, W_{3,1,1}$ $V_1 \oplus \overline{W}_{\mathbf{p}_{1,1},1,2}, \mathbf{p}_{1,1}$ $W_{1,2,1}, W_{2,2,1}, W_{3,2,1}$ $V_2 \oplus \overline{W}_{\mathbf{p}_{1,2},2,2}, \mathbf{p}_{1,2}$	$W_{1,1,2}, W_{2,1,2}, W_{3,1,2}$ $V_1 \oplus \overline{W}_{\mathbf{p}_{2,1},1,1}, \mathbf{p}_{2,1}$ $W_{1,2,2}, W_{2,2,2}, W_{3,2,2}$ $V_2 \oplus \overline{W}_{\mathbf{p}_{2,2},2,1}, \mathbf{p}_{2,1}$

In the delivery phase, assume that the demand vector of user $k \in [2]$ is $\mathbf{d}_k$. User $k \in [2]$ creates two vectors $[\mathbf{q}_{k,1}, \mathbf{q}_{k,2}] = [\mathbf{p}_{k,1} \oplus \mathbf{d}_k, \mathbf{p}_{k,2} \oplus \mathbf{d}_k]$, and sends $\mathbf{q}_{k,1}$ and $\mathbf{q}_{k,2}$ to servers 1 and 2, respectively. Then the server $h \in [2]$ creates a signal

$$Y_h = V_h \oplus \overline{W}_{\mathbf{q}_{1,h},h,2} \oplus \overline{W}_{\mathbf{q}_{2,h},h,1}, \quad (17)$$

and sends $X_h = (\mathbf{q}_{1,h}, \mathbf{q}_{2,h}, Y_h)$.

Notice that user $k \in [2]$ has stored $W_{[3],[2],k}$, so it can compute $\overline{W}_{\mathbf{d}_k,h,k}$ for $h \in [2]$. Thus, user $k \in [2]$ only needs to decode the packets $W_{\mathbf{d}_k,h,k'}$ for $h \in [2]$ and $k' \in [2]\backslash\{k\}$. In fact, for each $h \in [2]$, we have

- user 1 decodes $\overline{W}_{\mathbf{d}_1,h,2}$ by

$$\overline{W}_{\mathbf{d}_1,h,2} = Y_h \oplus (V_h \oplus \overline{W}_{\mathbf{p}_{1,h},h,2}) \oplus \overline{W}_{\mathbf{q}_{2,h},h,1}, \quad (18)$$

- user 2 decodes $\overline{W}_{\mathbf{d}_2,2,1}$ by

$$\overline{W}_{\mathbf{d}_2,h,1} = Y_h \oplus (V_h \oplus \overline{W}_{\mathbf{p}_{2,h},h,1}) \oplus \overline{W}_{\mathbf{q}_{1,h},h,2}. \quad (19)$$

The signals in parenthesis of (18) (resp. (19)) is cached by user 1 (resp. 2), and the linear combination $\overline{W}_{\mathbf{q}_{2,h},h,1}$ (resp. $\overline{W}_{\mathbf{q}_{1,h},h,2}$) can be calculated by user 1 (resp. 2) by using the cache content and the coefficient $\mathbf{q}_{2,h}$ (resp. $\mathbf{q}_{1,h}$).

Server privacy is guaranteed since, from the view point of server $h \in [2]$, the query vectors $\mathbf{q}_{1,h}, \mathbf{q}_{2,h}$ it receives are independent and uniformly distributed over $\mathbb{F}_2^3$, and independent of $\mathbf{p}_{1,\bar{h}}, \mathbf{p}_{2,\bar{h}}$ generated by it, where $\bar{h} \in [2]\backslash\{h\}$. User privacy is guaranteed since, from the view point of user $k \in [2]$, the vectors $\mathbf{q}_{k,1}, \mathbf{q}_{k,2}$ are random vectors independently and uniformly distributed over $\mathbb{F}_2^3$. Finally, security is guaranteed since the transmitted signals Y_1 and Y_2 are accompanied by a unique security key V_1 and V_2, respectively.

In this example, each packet is of size $\frac{B}{4}$ bits. Each user caches 8 packets and each server sends 1 packet, where the size of the various keys does not scale with B. Thus the scheme achieves the point $(M, R) = (2, 1/2)$. To complete the tradeoff, the other points are $(M, R) = (1, ?)$ and $(M, R) = (3, 0)$

IV. Distributed Key Superposition Scheme

We now generalize the example from the previous section. For $t \in [0:K]$, let

$$\mathbf{\Omega}_t \triangleq \{\mathcal{V} \subseteq [K] : |\mathcal{V}| = t\}. \quad (20)$$

Notice that, the achievablity of $(M_K, R_K) = (N, 0)$ in (14) is trivial. In the following, we describe the scheme achieving

(M_t, R_t) in (14) for a fixed parameter $t \in [0:K-1]$. The other points can be achieved by memory sharing.

Firstly, the server partitions each file W_n into $L\binom{K}{t}$ equal-size packets, denoted by

$$W_n = \{W_{n,h,\mathcal{V}} : h \in [L], \mathcal{V} \in \mathbf{\Omega}_t\}, \quad \forall\, n \in [N]. \tag{21}$$

We shall use the following notation for each vector $\mathbf{a} = (a_1, \ldots, a_N)^\top \in \mathbb{F}_q^N$ and any $h \in [L], \mathcal{V} \in \mathbf{\Omega}_t$

$$\overline{W}_{\mathbf{a},h,\mathcal{V}} \triangleq \sum_{n\in[N]} a_n \cdot W_{n,h,\mathcal{V}}. \tag{22}$$

Placement Phase: Each server $h \in [L]$ first generates $\binom{K}{t+1}$ *security keys* and $K\binom{K-1}{t}$ *privacy keys* as follows.

- The $\binom{K}{t+1}$ security keys, denoted by $\{V_{h,\mathcal{U}} : \mathcal{U} \in \mathbf{\Omega}_{t+1}\}$, are independently and uniformly chosen from $\mathbb{F}_q^{B/\binom{K}{t}}$.
- Denote

$$\overline{h} = \begin{cases} h+1, & \text{if } h \in [L-1] \\ 1, & \text{if } h = L \end{cases}. \tag{23}$$

The $K\binom{K-1}{t}$ privacy keys are are constructed as

$$\left\{\overline{W}_{\mathbf{p}_{j,\overline{h}},\overline{h},\mathcal{V}} : j \in [K], \mathcal{V} \in \mathbf{\Omega}_t, j \notin \mathcal{V}\right\}, \tag{24}$$

where the vectors $\mathbf{p}_{1,\overline{h}}, \mathbf{p}_{2,\overline{h}}, \ldots, \mathbf{p}_{K,\overline{h}}$ are independently and uniformly chosen from $\mathbb{F}_q^N$.

The random variable P_h is given by

$$P_h = \{V_{h,\mathcal{U}} : \mathcal{U} \in \mathbf{\Omega}_{t+1}\} \cup \{\mathbf{p}_{j,\overline{h}} : j \in [K]\}. \tag{25}$$

Then the server $h \in [L]$ provides user $k \in [K]$ with contents

$$\begin{aligned} Z_{k,h} = &\left\{W_{n,h,\mathcal{V}} : \mathcal{V} \in \mathbf{\Omega}_t, k \in \mathcal{V}, n \in [N]\right\} && (26a)\\ &\bigcup \left\{V_{h,\{k\}\cup\mathcal{V}} \,:\, \mathcal{V} \in \mathbf{\Omega}_t, k \notin \mathcal{V}\right\} && (26b)\\ &\bigcup \left\{\overline{W}_{\mathbf{p}_{k,\overline{h}},\overline{h},\mathcal{V}} : \mathcal{V} \in \mathbf{\Omega}_t, k \notin \mathcal{V}\right\} && (26c)\\ &\bigcup \{\mathbf{p}_{k,\overline{h}}\}. && (26d) \end{aligned}$$

The contents stored by user k is

$$\begin{aligned} &Z_k = \{\mathbf{p}_{k,\overline{h}} : h \in [L]\} && (27a)\\ &\bigcup \left\{W_{n,h,\mathcal{V}} : \mathcal{V} \in \mathbf{\Omega}_t, k \in \mathcal{V}, h \in [L], n \in [N]\right\} && (27b)\\ &\bigcup \left\{V_{h,\{k\}\cup\mathcal{V}} + \overline{W}_{\mathbf{p}_{k,h},h,\mathcal{V}} \,:\, \mathcal{V} \in \mathbf{\Omega}_t, k \notin \mathcal{V}, h \in [L]\right\} && (27c) \end{aligned}$$

It can be easily verified that k can obtain Z_k from $(Z_{k,1}, \ldots, Z_{k,L})$.

Delivery Phase: Assume the demand vector of user $k \in [K]$ is $\mathbf{d}_k$, $\forall\, k \in [K]$. User $k \in [K]$ generates L query vectors $\mathbf{q}_{k,1}, \ldots, \mathbf{q}_{k,L} \in \mathbb{F}_q^N$, and sends

$$Q_{k,h} \triangleq \mathbf{q}_{k,h} \triangleq \mathbf{p}_{k,h} + \mathbf{d}_k = (q_{k,h,1}, \ldots, q_{k,h,N})^\top \tag{28}$$

to the server $h \in [L]$. Then for each $\mathcal{U} \in \mathbf{\Omega}_{t+1}$, the server $h \in [L]$ generates a signal

$$Y_{h,\mathcal{U}} \triangleq V_{h,\mathcal{U}} + \sum_{j\in\mathcal{U}} \overline{W}_{\mathbf{q}_{j,h},h,\mathcal{U}\setminus\{j\}}, \tag{29}$$

and sends

$$X_h = \{\mathbf{q}_{k,h} : k \in [K]\} \cup \{Y_{h,\mathcal{U}} : \mathcal{U} \in \mathbf{\Omega}_{t+1}\} \tag{30}$$

to the users via its dedicated shared link.

Correctness: With the definition in (22), we need to show that user $k \in [K]$ can obtain all the packets $\{\overline{W}_{\mathbf{d}_k,h,\mathcal{V}} : h \in [L], \mathcal{V} \in \mathbf{\Omega}_t\}$ by (3) and (21). In fact, for $\mathcal{V} \in \mathbf{\Omega}_t$ such that $k \in \mathcal{V}$, user k can compute $\overline{W}_{\mathbf{d}_k,h,\mathcal{V}}$ directly from the packets in its cache by (27b) for all $h \in [L]$; for $\mathcal{V} \in \mathbf{\Omega}_t$ such that $k \notin \mathcal{V}$, by (29), for each $h \in [L]$,

$$\begin{aligned} &Y_{h,\{k\}\cup\mathcal{V}} && (31a)\\ &= V_{h,\{k\}\cup\mathcal{V}} + \overline{W}_{\mathbf{q}_{k,h},h,\mathcal{V}} + \sum_{j\in\mathcal{V}} \overline{W}_{\mathbf{q}_{j,h},h,\{k\}\cup\mathcal{V}\setminus\{j\}} && (31b)\\ &= \overline{W}_{\mathbf{d}_k,h,\mathcal{V}} && (31c)\\ &\quad + (V_{h,\{k\}\cup\mathcal{V}} + \overline{W}_{\mathbf{p}_{k,h},h,\mathcal{V}}) && (31d)\\ &\quad + \sum_{j\in\mathcal{V}} \overline{W}_{\mathbf{q}_{j,h},h,\{k\}\cup\mathcal{V}\setminus\{j\}}, && (31e) \end{aligned}$$

where (31) follows from (22) and (28). Notice that the term among parenthesis in (31d) is stored by the user by (27c), and the term in (31e) can be computed by the contents in (27b) and the coefficients $\mathbf{q}_{\mathcal{V}}$, which is available by user k by (7). Thus, user k can decode $\overline{W}_{\mathbf{d}_k,h,\mathcal{V}}$.

Security: We prove the stronger condition (13), i.e.,

$$\begin{aligned} &I(\mathbf{d}_{[K]}, W_{[N]}; X) && (32a)\\ \overset{(7),(30)}{=}\ &I(\mathbf{d}_{[K]}, W_{[N]}; \mathbf{q}_{[K],[L]}, \{Y_{h,\mathcal{U}}\}_{h\in[L],\mathcal{U}\in\mathbf{\Omega}_{t+1}}) && (32b)\\ =\ &I(\mathbf{d}_{[K]}, W_{[N]}; \mathbf{q}_{[K],[L]}) \\ &+ I(\mathbf{d}_{[K]}, W_{[N]}; \{Y_{h,\mathcal{U}}\}_{h\in[L],\mathcal{U}\in\mathbf{\Omega}_{t+1}} \,|\, \mathbf{q}_{[K],[L]}) && (32c)\\ =\ &0, && (32d) \end{aligned}$$

where (32d) follows since (a) the vectors $\mathbf{q}_{k,h} = \mathbf{d}_k + \mathbf{p}_{k,h}$, $k \in [K], h \in [L]$ are independent of $(\mathbf{d}_{[K]}, W_{[N]})$ because $\mathbf{p}_{[K],[L]}$ are independently and uniformly distributed over $\mathbb{F}_q^N$; and (b) the signals $\{Y_{h,\mathcal{U}} : h \in [L], \mathcal{U} \in \mathbf{\Omega}_{t+1}\}$ are independent of $(\mathbf{d}_{[K]}, W_{[N]}, \mathbf{q}_{[K],[L]})$ by (29), since $\{V_{h,\mathcal{U}} : h \in [L], \mathcal{U} \in \mathbf{\Omega}_{t+1}\}$ are independently and uniformly distributed over $\mathbb{F}_q^{B/\binom{K}{t}}$.

Server Privacy: We need to prove (10). For any $h \in [L]$,

$$\begin{aligned} &I(\mathbf{d}_{[K]}; P_h, Q_{[K],h} \,|\, W_{[N]}) && (33a)\\ \overset{(25),(28)}{=}\ &I(\mathbf{d}_{[K]}; \{V_{h,\mathcal{U}}\}_{\mathcal{U}\in\mathbf{\Omega}_{t+1}}, \mathbf{p}_{[K],\overline{h}}, \mathbf{q}_{[K],h} \,|\, W_{[N]}) && (33b)\\ =\ &I(\mathbf{d}_{[K]}; \mathbf{q}_{[K],h} \,|\, W_{[N]}) \\ &+ I(\mathbf{d}_{[K]}; \{V_{h,\mathcal{U}}\}_{\mathcal{U}\in\mathbf{\Omega}_{t+1}}, \mathbf{p}_{[K],\overline{h}} \,|\, \mathbf{q}_{[K],h} W_{[N]}) && (33c)\\ =\ &0, && (33d) \end{aligned}$$

where (33d) holds because (a) $\mathbf{q}_{[K],h} = \mathbf{p}_{[K],h} + \mathbf{d}_{[K]}$ by (28) are independent of $(\mathbf{d}_{[K]}, W_{[N]})$ since the vectors $\mathbf{p}_{[K],h}$ are independent random variables uniformly distributed over $\mathbb{F}_q^N$; (b) $\{V_{h,\mathcal{U}}\}_{\mathcal{U}\in\mathbf{\Omega}}, \mathbf{p}_{[K],\overline{h}}$ are random variables independent of $(\mathbf{d}_{[K]}, \mathbf{q}_{[K],h}, W_{[N]})$ since $\overline{h} \neq h$.

User Privacy: By (32d) and since $I(W_{[N]}, \mathbf{d}_{[K]}; X) = 0$ holds, thus we only need to prove

$$\begin{aligned}
& I(\mathbf{d}_{[K]\setminus\mathcal{S}}; Z_{\mathcal{S}} \,|\, W_{[N]}, X, \mathbf{d}_{\mathcal{S}}) && \text{(34a)}\\
\overset{(7),(30)}{=}\ & I(\mathbf{d}_{[K]\setminus\mathcal{S}}; Z_{\mathcal{S}} \,| && \text{(34b)}\\
& W_{[N]}, \mathbf{q}_{[K],[L]}, \mathbf{d}_{\mathcal{S}}, \{Y_{h,\mathcal{U}}\}_{h\in[L],\mathcal{U}\in\mathbf{\Omega}_{t+1}}) && \text{(34c)}\\
\leq\ & I(\mathbf{d}_{[K]\setminus\mathcal{S}}; Z_{\mathcal{S}}, \{V_{h,\mathcal{U}}\}_{h\in[L],\mathcal{U}\in\mathbf{\Omega}_{t+1}} \,| \\
& W_{[N]}, \mathbf{q}_{[K],[L]}, \mathbf{d}_{\mathcal{S}}, \{Y_{h,\mathcal{U}}\}_{h\in[L],\mathcal{U}\in\mathbf{\Omega}_{t+1}}) && \text{(34d)}\\
\overset{(28)}{=}\ & I(\mathbf{d}_{[K]\setminus\mathcal{S}}; Z_{\mathcal{S}}, \{V_{h,\mathcal{U}}\}_{h\in[L],\mathcal{U}\in\mathbf{\Omega}_{t+1}} \,| \\
& W_{[N]}, \mathbf{q}_{[K]\setminus\mathcal{S},[L]}, \mathbf{d}_{\mathcal{S}}, \mathbf{p}_{\mathcal{S},[L]}, \{Y_{h,\mathcal{U}}\}_{h\in[L],\mathcal{U}\in\mathbf{\Omega}_{t+1}}) && \text{(34e)}\\
=\ & I(\mathbf{d}_{[K]\setminus\mathcal{S}}; \{V_{h,\mathcal{U}}\}_{h\in[L],\mathcal{U}\in\mathbf{\Omega}_{t+1}} \,| \\
& W_{[N]}, \mathbf{q}_{[K]\setminus\mathcal{S},[L]}, \mathbf{d}_{\mathcal{S}}, \mathbf{p}_{\mathcal{S},[L]}, \{Y_{h,\mathcal{U}}\}_{h\in[L],\mathcal{U}\in\mathbf{\Omega}_{t+1}}) && \text{(34f)}\\
=\ & 0, && \text{(34g)}
\end{aligned}$$

where (34f) holds since $Z_{\mathcal{S}}$ is determined by $\mathbf{p}_{\mathcal{S},[L]}$, $\{V_{h,\mathcal{U}}\}_{h\in[L],\mathcal{U}\in\mathbf{\Omega}_{t+1}}$ and $W_{[N]}$ by construction.

Performance: By (21), each file is split into equal-size packets of length $B/\big(L\binom{K}{t}\big)$ symbols. For each user k, by the cached content in (27), for each $\mathcal{V}\in\mathbf{\Omega}_t$ such that $k\in\mathcal{V}$, there are NL associated packets cached by the user, L from each file (see (27b)). For each $\mathcal{V}\in\mathbf{\Omega}_t$ such that $k\notin\mathcal{V}$, there is L associated coded packets cached at the user (see (27c)). In addition, the vectors $\mathbf{p}_{k,[L]}$ can be sent in NL symbols. Thus, the achieved memory size is

$$\begin{aligned}
M &= \liminf_{B\to\infty} \frac{1}{B}\cdot\left(\frac{\big(\binom{K-1}{t-1}\cdot NL + \binom{K-1}{t}L\big)B}{L\binom{K}{t}} + NL\right)\\
&= 1+\frac{t(N-1)}{K}. \qquad (35)
\end{aligned}$$

By (7) and (30), the servers sends $L\binom{K}{t+1}$ coded packets of $B/\big(L\binom{K}{t}\big)$ symbols, and the coefficient vectors (which can be sent in KNL symbols), thus the achieved load is

$$R = \liminf_{B\to\infty}\frac{1}{B}\left(\binom{K}{t+1}\frac{B}{L\binom{K}{t}} + KNL\right) = \frac{K-t}{t+1}. \qquad (36)$$

Optimality: The approximate optimality result directly follows from [8], for the single server case, but without server privacy. Obviously, the tradeoff in the setup in [8] works in our setup as well. Moreover, the achieved memory-load tradeoff $R^*(M)$ is exactly the same with that in [8]. Therefore, the gap directly follows from [8, Theorems 1 and 3].

Remark 4 (Relation to the key superposition scheme in [8]). Our proposed scheme originates from the key superposition scheme in [8] for single server systems, when the server privacy condition is not required. We explain the relationship as follows.

The partition of the files in (21) can be viewed as a two layer partition: each file W_n is first partitioned into L subfiles, and each subfile is further partitioned as in the MAN scheme [1]

$$W_n = \{W_{n,h} : h\in[L]\}, \qquad (37)$$

where $W_{n,h} = \{W_{n,h,\mathcal{V}} : \mathcal{V}\in\mathbf{\Omega}_t\}$ for each $h\in[L]$. Each server $h\in[L]$ is responsible for satisfying the demands of the users regarding to the h-th subfiles. Thus, each server $h\in[L]$ intends to apply the key superposition scheme in [8] to the subfiles $\{W_{n,h}\}_{n=1}^N$. In the key superposition scheme, the server generates a set of privacy keys for each user, and a security key for each signal, which are stored at the users' caches in superposed form. In our multi-server setup, in order to guarantee server privacy, the privacy keys used by server h must not generated by itself. Instead, they are generated by another server. As a result, the privacy keys in (24) are generated by server h, but used for retrieving information from server $\overline{h}$, where $\overline{h}\neq h$.

Notice that, due to the fact that each server $h\in[L]$ applies the key superposition scheme to the subfiles $\{W_{n,h}\}_{n=1}^N$, where each subfile is $\frac{1}{L}$ of the total file size, the total system achieves the same memory-load pairs as the key superposition scheme, which is independent of L, as long $L\geq 2$.

V. Performance Comparison

In this section, we compare the performance of our proposed scheme against that of the multi-user Private Information Retrieval (PIR) scheme in [10]. In [10], each user is interested in retrieving a single file, the cache contents of the users are transparent to all the servers, but only the correctness and server privacy aconditions are imposed; the "product scheme" achieves the lower convex envelop of the point $(0,N)$ and the following points

$$M_t' = \frac{tN}{K},\ R_t' = \frac{K-t}{t+1}\cdot\frac{1-\frac{1}{L^N}}{1-\frac{1}{L}},\ t\in[0:K], \qquad (38)$$

where the factor $\frac{1-\frac{1}{L^N}}{1-\frac{1}{L}}\leq 1$ is optimal PIR load, which is also optimal for private linear function retrieval [**?**], and $\frac{K-t}{t+1}$ is the MAN load [1] as for our scheme in (36).

For reference, we also plot the curve of load of scheme in [4], which does not satisfy any security or privacy conditions[1] and achieves the lower convex envelope of

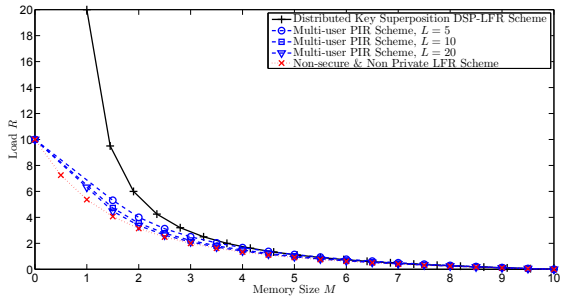
$$M_t'' = \frac{tN}{K},\ R_t'' = \frac{\binom{K}{t+1} - \binom{K-\min\{N,K\}}{t+1}}{\binom{K}{t}},\ t\in[0:K], \qquad (39)$$

where the load is the optimal load under the constraint of uncoded cache placement for single file retrieval [3].

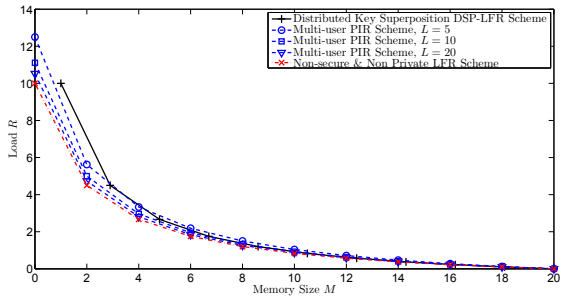
Fig. 2 shows the tradeoff curves for the case $N<K$ (i.e., $N=10, K=20$ in Fig. 2(a)) and the case $N>K$ (i.e., $N=20, K=10$ in Fig. 2(b)) for $L\in\{5,10,20\}$. Note that only the tradeoff in (38) depends on L.

We observe that our DKS scheme is inferior to both (38) (with only server privacy) and (39) (with neither privacy nor security) in the small memory regime. The gap appears significant for $N<K$, where the leftmost point achieved by (38) is $(0,\min\{N,K\big(\frac{1-1/L^N}{1-1/L}\big)\})$ and by (39) is $(0,\min\{N,K\})$, while for our scheme is $(1,K)$. This is due to the following:

[1]Notice that, the scheme in [4] can be similarly adapted to system with L servers, by first partition each file into L subfiles, and each server is responds for one subfile, as discussed in Remark 4, and it will achieve the same load-memory tradeoff.



(a) $N = 10, K = 20$



(b) $N = 20, K = 10$

Fig. 2: Load-memory tradeoff of different schemes (a) $N < K$; (b) $N > K$.

1) in (38), the point $(M, R) = (0, N)$ can be trivially achieved by sending all the files to the users, and
2) in (39) for $t < K - N$, some redundant signals can be removed from the MAN scheme,

neither of which is possible in our setup since (a) $M \geq 1$ (as pointed out in Remark 2) due to the security condition in (9), and (b) the use of random security keys destroys the linear dependencies among multicast messages thus there are no redundant transmissions. We conjecture that our scheme is actually order optimal and part of ongoing work is to develop a converse bound that truly incorporates all the in (9)-(11).

For other regimes, the load of tour scheme approaches that single file retrieval without any further constraints, which is very pleasant from a practical point of view given all the extra conditions in (9)-(11).

VI. Conclusions

The distributed key superposition scheme proposed in this paper for retrieving linear functions from multiple servers simultaneously guarantees content security against a wiretapper having access to the delivery signals and demand privacy against both servers and colluding users. The memory-load tradeoff does not increase compared to a previously known scheme that only guarantees content security and demand privacy against colluding users in the single server case. The proposed scheme is over optimal except in the small memory regime with less files than uses, for which we conjecture a new converse bound ought to be derived that truly incorporates the security constraint.

Acknowledgement:

This work was supported in part by NSF Award 1910309.

Appendix

When there is only one server, the cache contents and the delivery signal come from the server. Intuitively, for server privacy, each user must decode all the files, thus

$$H(W_{[N]} \,|\, X, Z_k) = 0. \tag{40}$$

In fact, for $L = 1$ and any $k \in [K]$, by (1)-(2), the cached content is determined by the randomness[2] P and the files $W_{[N]}$, i.e., $H(Z_k \,|\, P, W_{[N]}) = 0$. Hence, by (7) and (10),

$$I(\mathbf{d}_{[K]}; X, P, Q_{[K]}, W_{[N]}, Z_k) = 0. \tag{41}$$

Thus, the demands $\mathbf{d}_{[K]}$ are independent of X, Z_k and $W_{[N]}$. Moreover, for any $n \in [N]$,

$$H(W_n \,|\, X, Z_k) \overset{(41)}{=} H(W_n \,|\, X, \mathbf{d}_k = \mathbf{e}_n, Z_k) \overset{(8)}{=} 0. \tag{42}$$

Therefore, (40) is proved. Furthermore, for any $k \in [K]$,

$$\begin{aligned} NB &= H(W_{[N]}) && (43)\\ &= I(W_{[N]}; X, Z_k) + H(W_{[N]} \,|\, X, Z_k) && (44)\\ &\overset{(40)}{=} I(W_{[N]}; X) + I(W_{[N]}; Z_k \,|\, X) && (45)\\ &\overset{(9)}{=} I(W_{[N]}; Z_k \,|\, X) && (46)\\ &\leq H(Z_k) && (47)\\ &\leq MB. && (48) \end{aligned}$$

Therefore, $M \geq N$ must hold for the case $L = 1$.

References

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

[2] K. Wan, D Tuninetti, and P Piantanida, "On the optimality of uncoded cache placement," in *Proc. IEEE Inf. Theory Workshop (ITW),* Cambridge, UK, pp. 161–165, Sep. 2016.

[3] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr,"The exact rate-memory tradeoff for caching with uncoded prefetching," *IEEE Trans. Inf. Theory,* vol. 64, pp. 1281–1296, Feb. 2018.

[4] K. Wan, H. Sun, M. Ji, D. Tuninetti, and G. Gaire, "On the optimal load-memory tradoeff of cache-aided scaler linear function retrieval," arXiv:2001.03577v1.

[5] 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.

[6] Q. Yan, and D. Tuninetti, "Fundamental limits of caching for demand privacy against colluding users," arXiv:2008.03642.

[7] K. Wan, H. Sun, M. Ji, D. Tunietti, and G. Gaire, "Fundametnal limits of device-to-device private caching with trusted server," arXiv:1912.09985

[8] Q. Yan, and D. Tuninetti, "Key superposition simultaneously achieves security and privacy in cache-aided linear function retrieval," arXiv:2009.06000.

[9] Q. Yan, M. Cheng, X. Tang, and Q. Chen, "On the placement delivery array design for centralized coded caching scheme," *IEEE Trans. Inf. Theory,* vol. 63, no. 9, pp. 5821–5833, Sep. 2017.

[10] X. Zhang, K. Wan, H. Sun, M. Ji, and G. Caire, "On the fundamental liimits of cache-aided multi-user private information retrieval," arXiv:2010.06492.

[2]Since we are discussing the case $L = 1$, for notational clearity, we drop the index h from the notations P_h, $Q_{k,h}$ and X_h in this subsection.