Key Superposition Simultaneously Achieves Security and Privacy in Cache-Aided Linear Function Retrieval

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

***Abstract*—A coded caching scheme, referred to as *key superposition*, is proposed in the cache-aided content Secure and demand Private Linear Function Retrieval (SP-LFR) setup, where the following conditions are imposed: (a) each user is interested in retrieving an arbitrary linear combination of the files in the server's library; (b) the content of the library must be kept secure from a wiretapper who obtains the signal sent by the server; and (c) any subset of users together can not obtain any information about the demands of the remaining users. The scheme uses the superposition of *security keys* and *privacy keys* in both the placement and delivery phases to guarantee content security and demand privacy, respectively. The achieved load-memory tradeoff is optimal to within a constant multiplicative gap, except for the small memory regime when there are less file than users. The memory-load tradeoff does not increase compared to the best known schemes that only guarantee content security in all regimes or only demand privacy in some regime.**

I. Introduction

Coded caching reduces the communication load at peak times in networks with cache-enabled users. A coded caching system is composed of a server connected to K users through a shared link, where the server has access to a library of N files and each user can cache up to M files. The system operates in two phases. When the network is not congested, the server pushes some contents into the users' cache (placement phase). During peak times, each user demands one file from the file library, and the server responds by sending a signal over the shared link to enable every user to decode its demand files (delivery phase). Coded caching reduces the worst case communication load (or just *load* for short in the following) in the delivery phase by designing the cache contents in the placement phase so as to create multicasting opportunities in the delivery phase regardless of the demanded files.

Coded caching was first proposed by Maddah-Ali and Niesen (MAN) [1]. The MAN scheme was proved to achieve the information-theoretical 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 security of the files in the library against wiretappers and the privacy of the user demands are both important aspects in practical systems. In the coded caching literature, information-theoretic security and demand privacy were considered separately, to the best of our knowledge – a gap that this works aims to close in this work.

Content security was studied in [4] to protect the content of the library against a wiretapper who obtains the signal sent by the server in the delivery phase. The key idea in [4] is that the users cache, in addition to the content in the MAN scheme, some *security keys* for the MAN uncached part of the files; this is done in a structured way so that each user can retrive all the multicast signals it needs to decode.

Demand privacy was studied in [5]–[8], where a user should not gain any information about the index of the demanded file by another user from the transmitted signal. In our recent work [9], the demand privacy is enforced against colluding users, that is, any subset of users can not obtain any information about the demands of the other users, even if they exchange their cache contents. This problem was mentioned in the device-to-device setup [10]. Our proposed scheme in [9] uses *privacy keys*, and was inspired by a recent work on cache-aided linear function retrieval in [11] that showed that allowing users to retrieve arbitrary linear combinations of files does not worsen the achievable memory-load tradeoff compared to just retrieving single files. In the privacy key scheme, each user caches, in addition to the uncoded cached content as in MAN scheme, also some privacy keys formed as random linear combinations of the content that was not cached in [1]. Given the set of files demanded by the users, the server sends multicast signals so that each user can retrieve a specific linear combination of files (related to the stored privacy key). The decoded linear combination together with the privacy key allows each user to successfully retrieve the demanded file.

In this paper, we investigate content Secure and demand Private Linear Function Retrieval (SP-LFR) systems, where the security and privacy conditions are simultaneously enforced, and in addition, each user is interested in downloading a *linear* combination of the files at the server as in [11]. We propose to use a *superposition of security keys and privacy keys* to achieve content security and demand privacy simultaneously. In our approach, in the placement phase each user caches the superposition of security keys and privacy keys. In the delivery phase, both security and privacy keys are added to the multicast signals. In this way, the content is kept secure and the demands are kept private. Key superposition neither increases the memory size nor the communication load compared to schemes using either only security or only privacy keys. The achieved load-memory tradeoff is optimal to within a constant multiplicative gap except for the small memory regime when there are less file than users.

II. System Model

An (N, K) caching system consists of a server with N files $W_1, W_2, \ldots, W_N$ and K users $1, 2, \ldots, K$, where the server is connected to the users via an error-free shared link. The N files are identically and uniformly distributed over $\mathbb{F}_q^B$, for some prime power integer q, and some integer B denoting the file length. Each user $k \in [K]$ has a memory of size MB symbols. The system operates in two phases as follows.

Placement Phase: The server privately generates a random variable P from some probability space $\mathcal{P}$. Then it fills the cache of each user $k \in [K]$ with a *cache function* $\varphi_k : \mathcal{P} \times \mathbb{F}_q^{NB} \mapsto \mathbb{F}_q^{\lfloor MB \rfloor}$. The cache content of user k is

$$Z_k = \varphi_k(P, W_{[N]}), \quad \forall\, k \in [K].$$

The quantity M is the *memory size* at each user.

Delivery Phase: Each user $k \in [K]$ demands $\mathbf{d}_k = (d_{k,1}, \ldots, d_{k,N})^\top \in \mathbb{F}_q^N$, i.e., must retrieve the linear combination

$$\overline{W}_{\mathbf{d}_k} \triangleq d_{k,1} \cdot W_1 + \ldots + d_{k,N} \cdot W_N, \tag{1}$$

where the addition and multiplication are operated symbolwise on the finite field $\mathbb{F}_q$.

The file library $W_{[N]}$, the randomness P and the demands $\mathbf{d}_{[K]}$ are independent, that is

$$H(\mathbf{d}_{[K]}, P, W_{[N]}) = \sum_{k=1}^{K} H(\mathbf{d}_k) + H(P) + \sum_{n=1}^{N} H(W_n),$$

where the base of the logarithm is q.

Then the server creates a signal X by using the *encoding function* $\phi : \mathcal{P} \times \mathbb{F}_q^{KN} \times \mathbb{F}_q^{NB} \mapsto \mathbb{F}_q^{\lfloor RB \rfloor}$, for some $R \geq 0$. The transmitted signal is

$$X = \phi(P, \mathbf{d}_{[K]}, W_{[N]}).$$

The quantity R is referred to as the *load* of the system.

The following conditions must hold for an SP-LFR scheme:

$$\text{[Correctness]} \quad H(\overline{W}_{\mathbf{d}_k} \,|\, X, \mathbf{d}_k, Z_k) = 0, \ \forall\, k \in [K], \tag{2}$$

$$\text{[Security]} \quad I(W_{[N]}; X) = 0, \tag{3}$$

$$\text{[Privacy]} \quad I(\mathbf{d}_{[K]\setminus\mathcal{S}}; X, \mathbf{d}_{\mathcal{S}}, Z_{\mathcal{S}} \,|\, W_{[N]}) = 0, \quad \forall\, \mathcal{S} \subseteq [K], \mathcal{S} \neq \emptyset. \tag{4}$$

Objective: A memory-load pair $(M, R) \in [1, N] \times \mathbb{R}^+$ is said to be achievable if there exists a scheme such that all the conditions in (2)–(4) are satisfied. The optimal load-memory tradeoff of the system is defined as

$$R^*(M) = \inf_{B \in \mathbb{N}^+} \{R : (M, R) \text{ is achievable for } B\}.$$

Throughout the paper, we focus on the case $N \geq 2$ since for $N = 1$ demand privacy is impossible, i.e., there only one possible file to demand. The range of M is $[1, N]$ since in a coded caching scheme that guarantees content security one must have $M \geq 1$ as shown in [4].

Remark 1 (Possible notions of privacy). Different definitions of demand privacy for single file retrieval have been used in the literature, such as [5]–[8]. Here we adopt the definition in [9], which is the strongest among the definitions used in the literature and is motivated by the need to insure privacy regardless of the file distribution[1].

Remark 2 (Naming convention). We shall refer to a scheme satisfying (2) as Linear Function Retrieval (LFR) scheme. If in addition the scheme satisfies either (3) or (4), we shall refer to it as a Secure LFR (S-LFR) scheme or a Private LFR (P-LFR) scheme, respectively. If we impose the restriction that the demands $\mathbf{d}_1, \ldots, \mathbf{d}_K \in \{\mathbf{e}_1, \ldots, \mathbf{e}_N\}$, where $\mathbf{e}_n$ is the n-th standard vector of $\mathbb{F}_q^N$, then the problem formulation reduces to the case where each user is interested in one single file. Similarly to the LFR setup, in the File Retrieval (FR) setup, we refer a scheme that satisfies the correctness condition in (2) as a File Retrieval (FR) scheme. If in addition it satisfies (3) or (4), we refer it a Secure FR (S-FR) scheme or a Private FR (P-FR) scheme, respectively. If it satisfies both (3) and (4), we refer to it as SP-FR scheme. In addition to the security key scheme in [4] and privacy key scheme in [9], we will refer the FR scheme in [3] as YMA scheme, to the LFR scheme in [11] as WSJTC scheme, and to the scheme in [6] in P-FR setup as virtual users scheme.

III. Main Result

The following theorem is the main result of this paper, achieved by the *Key Superposition* scheme in Section V.

Theorem 1. *For the (N, K) SP-LFR system, the lower convex envelope of the following points is achievable*

$$(M_t, R_t) = \left(1 + \frac{t(N-1)}{K}, \frac{K-t}{t+1}\right), \ t \in [0 : K]. \tag{5}$$

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

Remark 3 (Subpacketization). In the full version of this paper [12], we describe a more general class of SP-LFR schemes by incorporating the idea of key superposition into the Placement Delivery Array (PDA) framework, known to give low subpacketization FR schemes with neither security or privacy guarantee [13]. The subpacketization of a coded caching scheme is the least number of packets each file needs to be split into. The advantage of the PDA framework is that, we can obtain low subpacketization SP-LFR schemes from various known PDAs. In particular, the key superposition scheme corresponds to the PDAs describing MAN schemes (MAN-PDAs). It turns out that the points in Theorem 1 are Pareto-optimal within the PDA framework, and the key superposition (MAN-PDA) scheme achieves these points with the lowest possible subpacketization.

[1]The assumption that files are independent and uniformly distributed is only used in the derivation of converse bounds in the full version of this paper [12].

IV. A Toy Example

We first give an example to illustrate the key idea in SP-FR setup, before describing the general SP-LFR scheme. Consider an $(N,K)=(3,2)$ system with $t=1$. Firstly, split each file into $\binom{K}{t}=2$ equal-size packets, denoted by $W_n=(W_{n,1},W_{n,2})$, for all $n\in[N], N=3$.

In the placement phase, the server first generates $\binom{K}{t+1}=1$ *security key*, and $K\binom{K-1}{t}=2$ *privacy keys* as follows. The security key, denoted by $V_{\{1,2\}}$, is uniformly chosen from $\mathbb{F}_2^{B/2}$. The privacy keys are

$$\begin{aligned}\overline{W}_{\mathbf{p}_1,2}&=p_{1,1}W_{1,2}\oplus p_{1,2}W_{2,2}\oplus p_{1,3}W_{3,2},\\ \overline{W}_{\mathbf{p}_2,1}&=p_{2,1}W_{1,1}\oplus p_{2,2}W_{2,1}\oplus p_{3,1}W_{3,1},\end{aligned}$$

where the two vectors $\mathbf{p}_1=(p_{1,1},p_{1,2},p_{1,3})^\top$ and $\mathbf{p}_2=(p_{2,1},p_{2,2},p_{2,3})^\top$ are independently and uniformly chosen from $\mathbb{F}_2^3$. The server populates the caches as

$$\begin{aligned}Z_1&=\{W_{1,1},W_{2,1},W_{3,1},\ V_{\{1,2\}}\oplus\overline{W}_{\mathbf{p}_1,2}\},\\ Z_2&=\{W_{1,2},W_{2,2},W_{3,2},\ V_{\{1,2\}}\oplus\overline{W}_{\mathbf{p}_2,1}\}.\end{aligned}$$

In the delivery phase, assume that users k demands file $W_k, k\in[K]$, i.e., their demands are $\mathbf{d}_1=(1,0,0)^\top,\mathbf{d}_2=(0,1,0)^\top$. Then the server creates two vectors $\mathbf{q}_1,\mathbf{q}_2$ as

$$[\mathbf{q}_1,\mathbf{q}_2]=[\mathbf{p}_1\oplus\mathbf{d}_1,\mathbf{p}_2\oplus\mathbf{d}_2]=\begin{bmatrix}p_{1,1}\oplus 1 & p_{2,1}\\ p_{1,2} & p_{2,2}\oplus 1\\ p_{1,3} & p_{2,3}\end{bmatrix}.$$

Accordingly, the server creates a signal $Y_{\{1,2\}}$ as

$$\begin{aligned}Y_{\{1,2\}}=&V_{\{1,2\}}\oplus(p_{1,1}\oplus 1)W_{1,2}\oplus p_{1,2}W_{2,2}\oplus p_{1,3}W_{3,2}\\ &\oplus p_{2,1}W_{1,1}\oplus(p_{2,1}\oplus 1)W_{2,1}\oplus p_{2,3}W_{3,1}.\end{aligned}$$

Then the server sends $X=(\mathbf{q}_1,\mathbf{q}_2,Y_{\{1,2\}})$ to the users. Notice that user k has stored $W_{k,k}, k\in[2]$. Thus,

- user 1 decodes $W_{1,2}$ by

$$\begin{aligned}W_{1,2}=&Y_{\{1,2\}}\oplus(V_{\{1,2\}}\oplus\overline{W}_{\mathbf{p}_1,2}) &(6a)\\ &\oplus p_{2,1}W_{1,1}\oplus(p_{2,1}\oplus 1)W_{2,1}\oplus p_{2,3}W_{3,1}, &(6b)\end{aligned}$$

- user 2 decodes $W_{2,1}$ by

$$\begin{aligned}W_{2,1}=&Y_{\{1,2\}}\oplus(V_{\{1,2\}}\oplus\overline{W}_{\mathbf{p}_2,1}) &(7a)\\ &\oplus(p_{1,1}\oplus 1)W_{1,2}\oplus p_{1,2}W_{2,2}\oplus p_{1,3}W_{3,2}, &(7b)\end{aligned}$$

where the signal among parenthesis in (6a) (resp. (7a)) is cached by user 1 (resp. 2), and the linear combination in (6b) (resp. (7b)) can be calculated by user 1 (resp. 2) by using the cache content and the coefficient $\mathbf{q}_2$ (resp. $\mathbf{q}_1$).

The privacy is guaranteed since from the users' viewpoint, the vectors $\mathbf{q}_1,\mathbf{q}_2$ are random vectors independently and uniformly distributed over $\mathbb{F}_2^3$. The security is guaranteed since signal $Y_{\{1,2\}}$ is accompanied by a unique security key $V_{\{1,2\}}$.

In this example, each packet is of size $\frac{B}{2}$ bits. Each user caches 4 packets, and the server sends 1 packets. The vectors $\mathbf{q}_1,\mathbf{q}_2$ can be sent in $H(\mathbf{q}_1,\mathbf{q}_2)=6$ symbols, which does not scale with B. So the scheme achieves $(M,R)=(2,\frac{1}{2})$.

V. Key Superposition Scheme

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

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

Notice that, the achievability of $(M_K,R_K)=(N,0)$ in (5) is trivial. In the following, we describe the scheme achieving (M_t,R_t) in (5) for a fixed parameter $t\in[0:K-1]$.

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

$$W_n=\{W_{n,\mathcal{V}}:\mathcal{V}\in\mathbf{\Omega}_t\},\quad\forall\, n\in[N]. \tag{8}$$

For notational simplicity, we shall use the following notation for each vector $\mathbf{a}=(a_1,\ldots,a_N)^\top\in\mathbb{F}_q^N$ and any $\mathcal{V}\in\mathbf{\Omega}_t$

$$\overline{W}_{\mathbf{a},\mathcal{V}}\triangleq\sum_{n=1}^{N}a_n\cdot W_{n,\mathcal{V}}. \tag{9}$$

Placement Phase: The server 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 $\{V_\mathcal{U}:\mathcal{U}\in\mathbf{\Omega}_{t+1}\}$, are independently and uniformly chosen from $\mathbb{F}_q^{B/\binom{K}{t}}$. The $K\binom{K-1}{t}$ privacy keys are constructed as

$$\{\overline{W}_{\mathbf{p}_j,\mathcal{V}}:j\in[K],\mathcal{V}\in\mathbf{\Omega}_t,j\notin\mathcal{V}\},$$

where the vectors $\mathbf{p}_1,\mathbf{p}_2,\ldots,\mathbf{p}_K$ are independently and uniformly chosen from $\mathbb{F}_q^N$. Then the server fills the caches as

$$\begin{aligned}Z_k=&\{W_{n,\mathcal{V}}:\mathcal{V}\in\mathbf{\Omega}_t,k\in\mathcal{V},n\in[N]\} &(10a)\\ &\bigcup\{V_{\{k\}\cup\mathcal{V}}+\overline{W}_{\mathbf{p}_k,\mathcal{V}}:\mathcal{V}\in\mathbf{\Omega}_t,k\notin\mathcal{V}\},\ k\in[K]. &(10b)\end{aligned}$$

The random variable P is given by

$$P=\{V_\mathcal{U}:\mathcal{U}\in\mathbf{\Omega}_{t+1}\}\cup\{\mathbf{p}_j:j\in[K]\}.$$

Delivery Phase: After receiving the user demands $\mathbf{d}_{[K]}$, the server generates K vectors

$$\mathbf{q}_k=\mathbf{p}_k+\mathbf{d}_k=(q_{k,1},q_{k,2},\ldots,q_{k,N})^\top,\ \forall\, k\in[K]. \tag{11}$$

Then, for each $\mathcal{U}\in\mathbf{\Omega}_{t+1}$, the server generates a signal

$$Y_\mathcal{U}\triangleq V_\mathcal{U}+\sum_{j\in\mathcal{U}}\overline{W}_{\mathbf{q}_j,\mathcal{U}\backslash\{j\}}, \tag{12}$$

and sends sends

$$X=\{\mathbf{q}_k:k\in[K]\}\cup\{Y_\mathcal{U}:\mathcal{U}\in\mathbf{\Omega}_{t+1}\}. \tag{13}$$

Correctness: Notice that with the definition in (9), we need to show that user k can obtain all the packets $\{\overline{W}_{\mathbf{d}_k,\mathcal{V}}:\mathcal{V}\in\mathbf{\Omega}_t\}$ by (1) and (8). In fact, for $\mathcal{V}\in\mathbf{\Omega}_t$ such that $k\in\mathcal{V}$, user k can compute $\overline{W}_{\mathbf{d}_k,\mathcal{V}}$ directly from the packets in its cache by (10a); for $\mathcal{V}\in\mathbf{\Omega}_t$ such that $k\notin\mathcal{V}$, by (12),

$$\begin{aligned}Y_{\{k\}\cup\mathcal{V}}&=V_{\{k\}\cup\mathcal{V}}+\overline{W}_{\mathbf{q}_k,\mathcal{V}}+\sum_{j\in\mathcal{V}}\overline{W}_{\mathbf{q}_j,\{k\}\cup\mathcal{V}\backslash\{j\}}\\ &=\overline{W}_{\mathbf{d}_k,\mathcal{V}}+(V_{\{k\}\cup\mathcal{V}}+\overline{W}_{\mathbf{p}_k,\mathcal{V}}) &(14a)\\ &\quad+\sum_{j\in\mathcal{V}}\overline{W}_{\mathbf{q}_j,\{k\}\cup\mathcal{V}\backslash\{j\}}, &(14b)\end{aligned}$$

where (14) follows from (9) and (11). Notice that the term among parenthesis in (14a) is stored by the user by (10b), and the term in (14b) can be computed by the cached contents in (10a) and the coefficients $\mathbf{q}_{\mathcal{V}}$, which is available at user k by (13). Thus, user k can decode $\overline{W}_{\mathbf{d}_k,\mathcal{V}}$.

Security: We prove a stronger condition, namely

$$\begin{aligned}
&I(\mathbf{d}_{[K]}, W_{[N]}; X)\\
&\overset{(a)}{=} I(\mathbf{d}_{[K]}, W_{[N]}; \mathbf{q}_{[K]}, \{Y_{\mathcal{U}}\}_{\mathcal{U}\in\mathbf{\Omega}_{t+1}})\\
&= I(\mathbf{d}_{[K]}, W_{[N]}; \mathbf{q}_{[K]}) + I(\mathbf{d}_{[K]}, W_{[N]}; \{Y_{\mathcal{U}}\}_{\mathcal{U}\in\mathbf{\Omega}_{t+1}} \,|\, \mathbf{q}_{[K]})\\
&\overset{(b)}{=} 0, \qquad (15)
\end{aligned}$$

where (a) follows from (13); and (b) follows since (i) $\mathbf{q}_{[K]} = \mathbf{d}_{[K]} + \mathbf{p}_{[K]}$ is independent of $(\mathbf{d}_{[K]}, W_{[N]})$ because $\mathbf{p}_{[K]}$ are independently and uniformly distributed over $\mathbb{F}_q^N$; and (ii) since $\{V_{\mathcal{U}} : \mathcal{U} \in \mathbf{\Omega}_{t+1}\}$ are independently and uniformly distributed over $\mathbb{F}_q^{B/\binom{K}{t}}$, by (12), $\{Y_{\mathcal{U}} : \mathcal{U} \in \mathbf{\Omega}_{t+1}\}$ are independent of $(\mathbf{d}_{[K]}, W_{[N]}, \mathbf{q}_{[K]})$.

Privacy: By (15) and $I(W_{[N]}, \mathbf{d}_{[K]}; X) = 0$ proved in (15) above, we only need to prove

$$\begin{aligned}
&I(\mathbf{d}_{[K]\setminus\mathcal{S}}; Z_{\mathcal{S}} \,|\, W_{[N]}, X, \mathbf{d}_{\mathcal{S}})\\
&\overset{(a)}{=} I(\mathbf{d}_{[K]\setminus\mathcal{S}}; Z_{\mathcal{S}} \,|\, W_{[N]}, \mathbf{q}_{[K]}, \mathbf{d}_{\mathcal{S}}, \{Y_{\mathcal{U}}\}_{\mathcal{U}\in\mathbf{\Omega}_{t+1}})\\
&\le I(\mathbf{d}_{[K]\setminus\mathcal{S}}; Z_{\mathcal{S}}, \{V_{\mathcal{U}}\}_{\mathcal{U}\in\mathbf{\Omega}_{t+1}} \,|\\
&\qquad W_{[N]}, \mathbf{q}_{[K]}, \mathbf{d}_{\mathcal{S}}, \{Y_{\mathcal{U}}\}_{\mathcal{U}\in\mathbf{\Omega}_{t+1}})\\
&\overset{(b)}{=} I(\mathbf{d}_{[K]\setminus\mathcal{S}}; Z_{\mathcal{S}}, \{V_{\mathcal{U}}\}_{\mathcal{U}\in\mathbf{\Omega}_{t+1}} \,|\\
&\qquad W_{[N]}, \mathbf{q}_{[K]\setminus\mathcal{S}}, \mathbf{d}_{\mathcal{S}}, \mathbf{p}_{\mathcal{S}}, \{Y_{\mathcal{U}}\}_{\mathcal{U}\in\mathbf{\Omega}_{t+1}})\\
&\overset{(c)}{=} I(\mathbf{d}_{[K]\setminus\mathcal{S}}; \{V_{\mathcal{U}}\}_{\mathcal{U}\in\mathbf{\Omega}_{t+1}} \,|\\
&\qquad W_{[N]}, \mathbf{q}_{[K]\setminus\mathcal{S}}, \mathbf{d}_{\mathcal{S}}, \mathbf{p}_{\mathcal{S}}, \{Y_{\mathcal{U}}\}_{\mathcal{U}\in\mathbf{\Omega}_{t+1}}) = 0,
\end{aligned}$$

where (a) follows from (13); (b) follows since given $\mathbf{d}_{\mathcal{S}}$, $\mathbf{q}_{\mathcal{S}}$ and $\mathbf{p}_{\mathcal{S}}$ determines each other by (11); and (c) follows since $Z_{\mathcal{S}}$ is determined by $\mathbf{p}_{\mathcal{S}}, \{V_{\mathcal{U}}\}_{\mathcal{U}\in\mathbf{\Omega}_{t+1}}$ and $W_{[N]}$ by (10).

Performance: By (8), each file is split into $\binom{K}{t}$ equal-size packets, each of length $B/\binom{K}{t}$, thus the subpacketization is $\binom{K}{t}$. For each user k, by the cached content in (10), for each $\mathcal{V} \in \mathbf{\Omega}_t$ such that $k \in \mathcal{V}$, there are N associated packets cached by the user, one from each file (see (10a)). For each $\mathcal{V} \in \mathbf{\Omega}_t$ such that $k \notin \mathcal{V}$, there is one associated coded packet cached at the user (see (10b)). Thus, the achieved memory is

$$M = \frac{1}{B} \cdot \frac{\left(\binom{K-1}{t-1} \cdot N + \binom{K-1}{t}\right) B}{\binom{K}{t}} = 1 + \frac{t(N-1)}{K}.$$

By (13), the server sends $\binom{K}{t+1}$ coded packets $\{Y_{\mathcal{U}} : \mathcal{U} \in \mathbf{\Omega}_{t+1}\}$, each of $B/\binom{K}{t}$ symbols, and the coefficient vectors $\mathbf{q}_{[K]}$ can be sent in KN symbols, thus the achieved load is

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

Gap: Due to space limitation, we only provide a proof sketch here. Details can be found in [12]. Let $R_{\text{MAN}}(M)$ be the lower convex envelope of $\{(M_t, R_t) : t \in [0 : K]\}$ in (5). We discuss separately the cases $N \ge K$ and $N < K, M \in [2, N)$.

For $N \ge K$, let $r(M)$ be the load-memory tradeoff of the MAN scheme in the FR setup, and $r^*_{\text{RF}}(M)$ be the optimal load-memory tradeoff in the same setup, then

$$\frac{R_{\text{MAN}}(M)}{R^*(M)} = \frac{R_{\text{MAN}}(M)}{r(M)} \cdot \frac{r(M)}{r^*_{\text{FR}}(M)} \cdot \frac{r^*_{\text{FR}}(M)}{R^*(M)},$$

where each term in the product above is upper bounded by some constant, and hence we obtain the desired constant gap.

For $N < K, M \in [2, N)$, we compare $R_{\text{MAN}}(M)$ to the following cut-set bound

$$R^*(M) \ge \max_{u\in[\min\{\lfloor\frac{N}{2}\rfloor, K\}]} \frac{uN - u^2M}{N-1}, \quad \forall M \in [1, N].$$

After some algebra we obtain the desired constant gap.

Remark 4 (Known setups subsumed by our construction). In our scheme we impose the superposition of privacy keys used in [9] and security keys used in [4]. In general, when users are demanding files we have the following. If the security keys $\{V_{\mathcal{U}} : \mathcal{U} \in \mathbf{\Omega}_{t+1}\}$ are removed (by setting them to be zero vector), then the scheme degrades to the privacy key scheme in [9][2]. If the privacy keys $\{W_{\mathbf{p}_i,\mathcal{V}} : \mathcal{V} \in \mathbf{\Omega}_t, j \in [K], j \notin \mathcal{V}\}$ are removed (by setting the vectors $\mathbf{p}_1, \ldots, \mathbf{p}_K$ to be zero vector), the scheme degrades to the security key scheme in [4]. If both the security and privacy keys are removed, then the scheme degrades to the MAN scheme [1]. We will compare the tradeoff of key superposition scheme to the various tradeoffs in less restrictive setups in Section VI.

VI. Performance Comparison

In Table I, we list the corner points of known achievable load-memory tradeoff in various setups in terms of a parameter t. The load-memory tradeoff of the schemes are given by the lower convex envelope of the corresponding points. In Fig. 1, we plot the memory-load tradeoff of the schemes in Table I for two typical regimes $N > K$ and $N < K$, where we choose parameters $(N, K) = (30, 10)$ and $(10, 30)$, respectively. For reference, we also plot the converse bound in [14], which works for all the above schemes. Comparing key superposition scheme with the other schemes, we make the following observations.

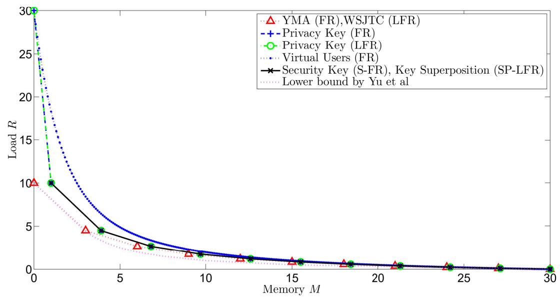
For the case $N > K$ (Fig. 1(a)), the key superposition scheme achieves the same performance as the security key scheme in S-FR setup and the privacy key schemes (in both P-FR and P-LFR setups) on the whole interval $[1, N]$. In fact, for the chosen parameter, the load-memory tradeoff of the security/privacy key schemes on the interval $M \in [1, N]$ are given by the lower convex envelope of the points $\{(M_t, R_t) : t \in [0 : K]\}$, which is exactly the same with that of key superposition scheme. The superposition of the privacy keys with the security keys neither increase the memory size nor the load. It was numerically verified in [9] that when $N > 2K+1$ and $0 < M < N-1-\frac{1}{K}$, the privacy key scheme outperforms

[2]In the case $N \le K$ and $t \le K - N$, some redundant signals are removed in the privacy key scheme [9] in the P-FR setup. Those signals can not be removed in SP-LFR or S-FR setups due to the use of security keys.

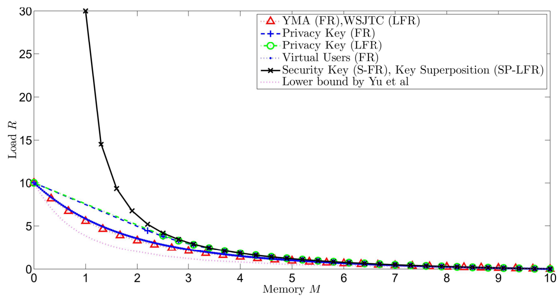
TABLE I: Performances of known achievable LFR and FR schemes[†].

Setup	Scheme	Memory M	Load R
FR	YMA [3]	$\frac{tN}{K}$	$\frac{\binom{K}{t+1}-\binom{K-\min\{N,K\}}{t+1}}{\binom{K}{t}}$
LFR	WSJTC [11]	$\frac{tN}{K}$	$\frac{\binom{K}{t+1}-\binom{K-\min\{N,K\}}{t+1}}{\binom{K}{t}}$
P-LFR	Privacy Key[‡] [9]	$1+\frac{t(N-1)}{K}$	$\frac{\binom{K}{t+1}-\binom{K-\min\{N,K\}}{t+1}}{\binom{K}{t}}$
P-FR	Privacy Key[‡] [9]	$1+\frac{t(N-1)}{K}$	$\frac{\binom{K}{t+1}-\binom{K-\min\{N-1,K\}}{t+1}}{\binom{K}{t}}$
	Virtual Users [6]	$\frac{t}{K}$	$\frac{\binom{KN}{t+1}-\binom{(K-1)N}{t+1}}{\binom{KN}{t}}$
S-FR	Security Key [4]	$1+\frac{t(N-1)}{K}$	$\frac{K-t}{t+1}$
SP-LFR	Key Superposition	$1+\frac{t(N-1)}{K}$	$\frac{K-t}{t+1}$

[†] For virtual users scheme, $t \in [0:NK]$, while for other schemes, $t \in [0:K]$.
[‡] In the privacy key scheme, the load-memory tradeoff curve was obtained by taking the lower convex envelope of the points in this row and a trivial point $(M,R)=(0,N)$.



(a) $N=30, K=10$



(b) $N=10, K=30$

Fig. 1: Load-memory tradeoff for secure/non-secure and private/non-private systems such that (a) $N>K$; (b) $N<K$.

the virtual users scheme in P-FR setup. Thus, in the regime $N>2K+1, 1\leq M\leq N-1-\frac{1}{K}$, the key superposition SP-LFR scheme also outperforms the virtual user P-FR scheme.

For the case $N \leq K$ (Fig. 1(b)), the key superposition scheme achieves the same performance as the security scheme in the whole interval $M\in[1,N]$ and the privacy key schemes when $M\geq 1+\frac{(K-N+1)(N-1)}{K}$. For small M, it is inferior to the privacy key scheme because in P-FR setup:

1) The trivial point $(M,R)=(0,N)$ can be achieved, and thus memory-sharing the other points with this point increases the performance;
2) For $M\in\{M_t : t\in[0:K-N]\}$, some redundant signals can be removed, similar to [3], [11].

In the SP-LFR systems, the above two points do not hold due to the use of security keys in the signals. Notice that, at the corner point, the additional load in the key superposition scheme due to the redundant signals compared to the privacy key scheme in P-FR setup is $\binom{K-\min\{N-1,K\}}{t+1}/\binom{K}{t}$, where $t=K\frac{M-1}{N-1}\in[0:K-N]$. This indicates that when $N<K$ and M is close to 1, the additional load is significant (e.g., at $M=1$, the additional load is $K-N+1$), which leads to the observation that the load of key superposition scheme diverges from that of the YMA [3], WSJTC [11], Privacy Key [9] and virtual users [6] schemes in the FR or LFR setup.

VII. Conclusions

The key superposition scheme proposed in this paper for the cache-aided content Security and demand Privacy linear function retrieval (SP-LFR) problem does not increase the load-memory tradeoff compared to the best known schemes guaranteeing only content security in all regimes or the best known schemes guaranteeing only privacy keys in some regimes. This is realized by the superposition of security keys and privacy keys in both cached contents and delivered signals.

Acknowledgement

This work was supported in part by NSF Award 1910309.

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] 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.
[5] K. Wan, and G. Caire, "On the coded caching with private demands," arXiv:1908.10821
[6] S. Kamath, "Demand private coded caching," arXiv:1909.03324.
[7] V. R. Aravind, P. Sarvepalli, A. Thangaraj, "Subpacketization in coded caching with demand privacy," arXiv: 1909.10471
[8] S. Kamath, J. Ravi, and B. K. Dey, "Demand-private coded caching and the exact tradeoff for $N=K=2$," arXiv:1911.06995
[9] Q. Yan, and D. Tuninetti, "Fundamental limits of caching for demand privacy against colluding users," arXiv:2008.03642.
[10] 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
[11] 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.
[12] Q. Yan, and D. Tuninetti, "Key superposition simultaneously achieves security and privacy in cache-aided linear function retrieval," arXiv:2009.06000.
[13] 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.
[14] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "Characterizing the rate-memory tradeoff in cache networks within a factor of 2," *IEEE Trans. Inf. Theory,* Vol. 65 , No. 1 , Jan. 2019.