SIAM J. Appl. Algebra Geometry
Vol. 1, pp. 647–664

Private Information Retrieval from Coded Databases with Colluding Servers[*]

Ragnar Freij-Hollanti[†], Oliver W. Gnilke[†], Camilla Hollanti[†], and David A. Karpuk[‡]

Abstract. We present a general framework for private information retrieval (PIR) from arbitrary coded databases that allows one to adjust the rate of the scheme to the suspected number of colluding servers. If the storage code is a generalized Reed–Solomon code of length n and dimension k, we design PIR schemes that achieve a PIR rate of $\frac{n-(k+t-1)}{n}$ while protecting against any t colluding servers, for any $1 \leq t \leq n-k$. This interpolates between the previously studied cases of $t=1$ and $k=1$ and achieves PIR capacity in both of these cases asymptotically as the number of files in the database grows.

Key words. private information retrieval, distributed storage systems, generalized Reed–Solomon codes

AMS subject classifications. 68P20, 68P30, 94B27, 14G50

DOI. 10.1137/16M1102562

1. Introduction.

Private information retrieval (PIR) addresses the question of how to retrieve data items from a database without disclosing information about the identity of the data items retrieved, and was introduced by Chor et al. in [4, 5]. The classic PIR model of [5] views the database as an m-bit binary string $x = [x^1 \cdots x^m] \in \{0,1\}^m$ and assumes that the user wants to retrieve a single bit x^i without revealing any information about the index i. We consider a natural extension of this model, wherein the database is a string $x = [x^1 \cdots x^m]$ of files x^i, which are themselves bit strings, and the user wants to download one of the files x^i without revealing its index.

The rate of a PIR scheme in this model is measured as the ratio of the gained information over the downloaded information, while upload costs of the requests are usually ignored. The trivial solution is to download the entire database. This, however, incurs a significant communication overhead whenever the database is large and is therefore not useful in practice. While the trivial solution is the only way to guarantee *information-theoretic privacy* in the case of a single server [5], this problem can be remedied by replicating the database onto n servers that do not communicate.

The study of PIR recently received renewed attention when Shah et al. introduced a model of coded private information retrieval (cPIR) [10, 11]. Here, all files are distributed over the

[*]Received by the editors November 7, 2016; accepted for publication (in revised form) August 2, 2017; published electronically November 14, 2017.
http://www.siam.org/journals/siaga/1/M110256.html
Funding: The fourth author's work was supported by Academy of Finland grant 268364. The third author's work was supported by Academy of Finland grants 276031, 282938, and 303819.

[†]Department of Mathematics and Systems Analysis, Aalto University, P.O. Box 11100, FI-00076 AALTO, Espoo, Finland (ragnar.freij@aalto.fi, oliver.gnilke@aalto.fi, camilla.hollanti@aalto.fi).
[‡]Department of Mathematics and Systems Analysis, Aalto University, P.O. Box 11100, FI-00076 AALTO, Espoo, Finland. Current address: Departamento de Matemáticas, Carrera 1 No. 18a 10, Edificio H, Primer Piso, 111711 Bogotá, Colombia (da.karpuk@uniandes.edu.co).

servers according to a storage code, so there is no assumption that the contents of all servers are identical. It is shown in [10] that for a suitably constructed storage code, privacy can be guaranteed by downloading a single bit more than the size of the desired file. However, this requires exponentially many servers in terms of the number of files. Blackburn, Etzion, and Paterson achieved the same low download complexity with a linear number of servers [3]. This is a vast improvement upon [10], but still far from applicable storage systems where the number of files tends to dwarf the number of servers.

While [10] effectively answered the question of how low the communication cost of a PIR scheme can be, it highlighted another cost parameter that should not be neglected in the era of big data, namely the *storage overhead*. We define the storage overhead as the ratio of the total number of coded bits stored on all servers to the total number of uncoded bits of data. Fazeli, Vardy, and Yaakobi showed in [6] that it is possible to reduce the storage overhead significantly. However, this requires subpacketizing the file and distributing it over a number of servers that grows to infinity as the desired storage overhead decreases.

In contrast to the schemes in [10], whose strengths appear as the number of servers tends to infinity, we are considering the setting where we are given a storage system with a fixed number of servers. While this in no way optimizes the storage overhead, it does keep the overhead fixed. Moreover, we allow some subsets of servers to be *colluding*, by which we mean that they may inform each other of their interaction with the user. This is very natural in a distributed storage system where communication between servers is required to recover data in the case of node failures. PIR over fixed maximum distance separable (MDS) storage systems was considered in [16]. There, two PIR schemes were presented for arbitrary $[n, k]$ MDS codes, one of which had rate $\frac{1}{n}$ and protected against $t = n-k$ colluding servers, and the other of which had rate $\frac{n-k}{n}$ and $t = 1$. In section 4 we present these as special instances of a scheme that can handle any number $1 \leq t \leq n-k$ of colluding servers. Curiously, neither the performance of our scheme nor the underlying field size depends on the number of files stored. The rate of our scheme depends on the minimum distance of a certain star product, and in the case where the storage code is a generalized Reed–Solomon (GRS) code (Theorem 10), we can achieve a rate of $\frac{n-(k+t-1)}{n}$.

The capacity (i.e., maximum possible rate) of a PIR scheme for a replicated storage system was derived in [14] (without collusion) and [13] (with colluding servers). The corresponding capacity of a coded storage system was given in [1] in the case of no colluding servers. A previous version of this paper conjectured a formula for the capacity of coded PIR with colluding servers that gives the capacity bounds of [1, 13] as special cases. However, this conjecture was recently disproven by Sun and Jafar in [15], who gave an explicit example of a scheme for coded PIR with colluding servers, the rate of which exceeded our conjectured upper bound. The work of [15] also characterizes the capacity of PIR for MDS coded data with colluding servers for other parameters, in particular for $m = 2$ files of length $k = n-1$. In general, however, the capacity of coded PIR with colluding servers remains open. Recently, another PIR scheme for coded databases with colluding servers was presented in [18] for MDS coded data. However, the rates achieved in the present work outperform the rates presented in [18] even for a moderate number of files, and the scheme in [18] requires the underlying field to be large. The case of a linear storage code that is not necessarily MDS, without collusion, has been studied in [7]. They show that for certain codes a rate of $\frac{n-k}{n}$ can still be achieved, outperforming the general result in Theorem 7.

2. Coding-theoretic preliminaries.

In this section we briefly collect some standard coding-theoretic definitions and results that we will need in the following sections.

2.1. Basic definitions.

We will use $\mathbb{F}_q$ to denote the field with q elements, where q is a prime power, and $[n]$ for the set $\{1, 2, \ldots, n\}$. For any two vectors $v, w \in \mathbb{F}_q^n$, we denote their standard inner product by $\langle v, w\rangle$. If $V \subseteq \mathbb{F}_q^n$, then we denote its orthogonal complement by

$$V^\perp = \{w \in \mathbb{F}_q^n \mid \langle v, w\rangle = 0 \text{ for all } v \in V\}, \tag{1}$$

and we write $V \perp W$ if $W \subseteq V^\perp$.

For a code $C \subseteq \mathbb{F}_q^n$ or a vector $v \in \mathbb{F}_q^n$ we use C_I and v_I to denote their respective projections onto the coordinates in $I \subseteq [n]$. The support of a codeword $c \in \mathbb{F}_q^n$ is $\operatorname{supp}(c) = \{i \in [n] : c_i \neq 0\}$, and the support of a code $C \subseteq \mathbb{F}_q^n$ is $\operatorname{supp}(C) = \cup_{c\in C} \operatorname{supp}(c)$. The minimum distance of $C \subseteq \mathbb{F}_q^n$ is

$$d = d_C = \min\{|I| : |C_{[n]\setminus I}| < |C|\}. \tag{2}$$

For linear codes, this can alternatively be written as

$$d = \min\{|\operatorname{supp}(c)| : c \in C\}. \tag{3}$$

A linear code $C \subseteq \mathbb{F}_q^n$ of dimension k and with minimum distance d is called an $[n, k, d]$-code, or an $[n, k, d]_q$-code if we wish to emphasize the field of definition. By an elementary result that is usually attributed to Singleton [12], if C is an $[n, k, d]$-code, then

$$d \leq n - k + 1. \tag{4}$$

A code that satisfies (4) with equality is called a *maximum distance separable* (MDS) code. An $[n, k, d]$ MDS code will be more concisely denoted as an $[n, k]$ MDS code, with $d = n-k+1$ being implied.

Given a linear $[n, k, d]$-code C, a subset $K \subseteq [n]$ of size $|K| = k$ is an *information set* of C if the natural projection $C \to C_K$ is a bijection. Equivalently, the columns of any generator matrix of C corresponding to the indices in K are linearly independent. If C is an MDS code, then every $K \subseteq [n]$ of size k is an information set.

The repetition code $\operatorname{Rep}(n)_q \subseteq \mathbb{F}_q^n$ is the one-dimensional code generated by the all-ones vector. It is an $[n, 1]$ MDS code.

2.2. Generalized Reed–Solomon codes.

Our proposed PIR scheme will be most interesting when the code defining the storage system is a GRS code. As such, we recall the basic properties of such codes here.

Definition 1 (GRS codes). *Let $\alpha = [\alpha_1 \cdots \alpha_n] \in \mathbb{F}_q^n$ satisfy $\alpha_i \neq \alpha_j$ for $i \neq j$, and let $v = [v_1 \cdots v_n] \in \mathbb{F}_q^{\times n}$. We define the generalized Reed–Solomon (GRS) code of dimension k associated to these n-tuples to be*

$$GRS_k(\alpha, v) = \{(v_i f(\alpha_i))_{1\leq i\leq n} \mid f \in \mathbb{F}_q[x],\ \deg(f) < k\}. \tag{5}$$

The canonical generator matrix for this code is given by

$$G(\alpha, v) := \begin{bmatrix} 1 & \cdots & 1 \\ \alpha_1 & \cdots & \alpha_n \\ \alpha_1^2 & \cdots & \alpha_n^2 \\ \vdots & \ddots & \vdots \\ \alpha_1^{k-1} & \cdots & \alpha_n^{k-1} \end{bmatrix} \cdot \mathrm{diag}(v), \tag{6}$$

where $\mathrm{diag}(v)$ is the diagonal matrix with the values v_i on the diagonal. In data storage applications, it is often desirable to have an explicit encoding matrix that is systematic, i.e., having an identity submatrix in the first k columns. For this purpose, define

$$\tilde{G}(\alpha, v) := \begin{bmatrix} f_1(\alpha_1) & \cdots & f_1(\alpha_n) \\ \vdots & \ddots & \vdots \\ f_k(\alpha_1) & \cdots & f_k(\alpha_n) \end{bmatrix} \cdot \mathrm{diag}(v), \tag{7}$$

where

$$f_i(x) = v_i^{-1} \prod_{j \in [k] \setminus \{i\}} \frac{x - \alpha_j}{\alpha_i - \alpha_j}$$

for $i = 1, \dots, k$. Note that $f_i(\alpha_i) = v_i^{-1}$ and $f_i(\alpha_j) = 0$ for $j \in [k] \setminus \{i\}$; hence this is a systematic generator matrix for $GRS_k(\alpha, v)$.

The code $GRS_k(\alpha, v)$ is an $[n, k]$ MDS code. From the Lagrange interpolation formula, it follows that the dual of $GRS_k(\alpha, v)$ is given by $GRS_{n-k}(\alpha, u)$, where

$$u_i = \left(v_i \prod_{j \neq i} (\alpha_i - \alpha_j) \right)^{-1}. \tag{8}$$

2.3. Star products.

The star product of two codes will play an integral role in our PIR scheme, essentially determining its rate.

Definition 2. *Let V, W be subvector spaces of $\mathbb{F}_q^n$. We define the star (or Schur) product $V \star W$ to be the subspace of $\mathbb{F}_q^n$ generated by the Hadamard products $v \star w = [v_1 w_1 \cdots v_n w_n]$ for all pairs $v \in V, w \in W$.*

The following proposition collects some basic properties of the star product that will prove useful in the coming sections.

Proposition 3. *The star product satisfies the following properties:*

(i) *If C is any linear code in $\mathbb{F}_q^n$ and $\mathrm{Rep}(n)_q \subseteq \mathbb{F}_q^n$ is the repetition code of length n over $\mathbb{F}_q$, then $C \star \mathrm{Rep}(n)_q = C$.*

(ii) *If C and D are any linear codes in $\mathbb{F}_q^n$ with $\mathrm{supp}(C) = \mathrm{supp}(D) = [n]$, and $(C \star D)^\perp = H$, then $d_H \geq d_{C^\perp} + d_{D^\perp} - 2$.*

(iii) *If $C \subseteq \mathbb{F}_q^n$ is any MDS code, then $(C \star C^\perp)^\perp = \mathrm{Rep}(n)_q$.*

(iv) *The star product of two GRS codes in $\mathbb{F}_q^n$ with the same parameter α is again a GRS code with parameter α. More specifically, $GRS_k(\alpha, v) \star GRS_\ell(\alpha, w) = GRS_{\min\{k+\ell-1, n\}}(\alpha, v \star w)$ for any parameters v, w.*

Proof. Property (i) follows immediately from the definition of the star product. Property (ii) is Theorem 5 of [17]. To see that property (iii) holds, let $H = (C \star C^\perp)^\perp$. The containment $\mathrm{Rep}(n)_q \subseteq H$ is obvious for any code C. If C is an $[n, k]$ MDS code, then $C^\perp$ is an $[n, n-k]$ MDS code, so property (ii) implies that $d_H \geq d_C + d_{C^\perp} - 2 = n$. Hence by the Singleton bound the dimension of H is 1, and therefore $H = \mathrm{Rep}(n)_q$.

To see that property (iv) holds, consider some arbitrary codewords $(v_i f(\alpha_i)) \in GRS_k(\alpha, v)$ and $(w_i g(\alpha_i)) \in GRS_\ell(\alpha, w)$. We clearly have

$$(v_i f(\alpha_i)) \star (w_i g(\alpha_i)) = (v_i w_i (fg)(\alpha_i)) \in GRS_{\min\{k+\ell-1,n\}}(\alpha, v \star w); \tag{9}$$

hence the containment $GRS_k(\alpha, v) \star GRS_\ell(\alpha, w) \subseteq GRS_{\min\{k+\ell-1,n\}}(\alpha, v \star w)$ holds. To see the reverse containment, note that $GRS_{\min\{k+\ell-1,n\}}(\alpha, v \star w)$ is generated as an $\mathbb{F}_q$-vector space by codewords of the form $(v_i w_i f_m(\alpha_i))$, where $f_m(x) = x^m$ is a monomial of degree $m < k + \ell - 1$. We can clearly decompose such a codeword as

$$(v_i w_i f_m(\alpha_i)) = (v_i f_a(\alpha_i)) \star (w_i f_b(\alpha_i)), \tag{10}$$

where $f_a(x) = x^a$ and $f_b(x) = x^b$ for any a, b such that $a < k$, $b < \ell$, and $a + b = m$. This shows the reverse inclusion and completes the proof. ■

3. Coded storage and private information retrieval.

Let us describe the distributed storage systems we consider; this setup follows that of [1, 16]. To provide clear and concise notation, we have consistently used superscripts to refer to files, subscripts to refer to servers, and parenthetical indices for entries of a vector. So, for example, the query q^i_j is sent to the jth server when downloading the ith file, and $y^i_j(a)$ is the ath entry of the vector y^i_j stored on server j.

Suppose we have files $x^1, \ldots, x^m \in \mathbb{F}_q^{b \times k}$. Data storage proceeds by arranging the files into a $bm \times k$ matrix

$$X = \begin{bmatrix} x^1 \\ \vdots \\ x^m \end{bmatrix}. \tag{11}$$

Each file x^i is encoded using a linear $[n, k, d]$-code C with generator matrix G_C into an encoded file $y^i = x^i G_C$. In matrix form, we encode the matrix X into a matrix Y by right-multiplying by G_C:

$$Y = XG_C = \begin{bmatrix} y^1 \\ \vdots \\ y^m \end{bmatrix} = \begin{bmatrix} y_1 & \cdots & y_n \end{bmatrix} = \begin{bmatrix} y^1_1 & \cdots & y^1_n \\ \vdots & \ddots & \vdots \\ y^m_1 & \cdots & y^m_n \end{bmatrix}. \tag{12}$$

The jth column $y_j \in \mathbb{F}_q^{bm \times 1}$ of the matrix Y is stored by the jth server. Here the vector $y^i_j \in \mathbb{F}_q^{b \times 1}$ represents the part of the ith file stored on the jth server.

Such a storage system allows any $d_C - 1$ servers to fail while still allowing users to successfully access any of the files x^i. In particular, if C is an MDS code, the resulting distributed storage system is maximally robust against server failures.

Private information retrieval (PIR) is the process of downloading a file from a database without revealing to the database which file is being downloaded [5]. Here by "database" we mean a collection of servers, e.g., all of the n servers used in the distributed storage system described earlier.

Definition 4. *Suppose we have a distributed storage system as described above. A* linear PIR scheme over $\mathbb{F}_q$ *for such a storage system consists of the following:*

1. *For each index $i \in [m]$, a probability space $(\mathcal{Q}^i, \mu^i)$ of* queries. *When the user wishes to download $x^i \in \mathbb{F}_q^{b\times k}$, a query $q^i \in \mathcal{Q}^i$ is selected randomly according to the probability measure μ_i. Each q^i is a set $q^i = \{q_1^i, \dots, q_n^i\}$, where q_j^i is sent to the jth server, and furthermore q_j^i is itself a row vector of the form*

$$q_j^i = [q_j^{i1} \cdots q_j^{im}], \quad \textit{where} \quad q_j^{i\ell} \in \mathbb{F}_q^{1\times b} \textit{ for all } \ell \in [m]. \tag{13}$$

2. Responses *$r_j^i = \langle q_j^i, y_j \rangle \in \mathbb{F}_q$ which the servers compute and transmit to the user. We set $r^i = [r_1^i \cdots r_n^i]$ to be the total response vector.*
3. *An* iteration process, *which repeats steps 1–2 a total of s times until the desired file x^i can be reconstructed from the s responses r^i.*
4. *A* reconstruction function *which takes as input the various r^i over all of the s iterations and returns the file x^i.*

Here we view b and s as secondary parameters, which we are free to adjust to enable the user to download exactly one whole file. If one restricts to the case of $b = 1$ row per file, the size of the file k may be too small, in which case it is not clear how to take advantage of a high rate scheme which inherently downloads more symbols per iteration than there are in a file. On the other hand, restricting to schemes with $s = 1$ iteration may fail to download an entire file. The freedom to adjust b and s allows one to avoid such complications.

The *rate* of a PIR scheme measures its efficiency by comparing the size of a file with how much information we downloaded in total.

Definition 5. *The* rate *of a linear PIR scheme is defined to be $\frac{bk}{ns}$.*

Note that Definition 5 ignores the cost to the user of uploading the queries to the servers. This can be justified by considering $x^i \in V^{b\times k}$, where V is some finite-dimensional vector space over $\mathbb{F}_q$, and encoding and data retrieval proceeds in an obvious way. In this setting the size of the queries is easily seen to be minimal in comparison to download costs when $\dim V \gg 1$.

It would be more precise to define the rate as $\frac{bk}{E[w(q^i)]s}$, where $E(\cdot)$ denotes expectation and $w(q^i)$ is the number of queries q_j^i which are not the zero vector, since such queries can be ignored. For comparison with earlier work on PIR, we use Definition 5 for the remainder of this paper.

Definition 6. *A PIR scheme* protects against t colluding servers *if for every set $T = \{j_1, \dots, j_t\} \subseteq [n]$ of size t, we have*

$$I(Q_T^i; i) = 0, \tag{14}$$

where Q_T^i denotes the joint distribution of all tuples $\{q_{j_1}^i, \dots, q_{j_t}^i\}$ of queries sent to the servers *in T as we range over all s iterations of the PIR scheme, and $I(\cdot\,;\cdot)$ denotes the mutual information of two random variables.*

In other words, for every set T of servers of size t, there exists a probability distribution $(\mathcal{Q}_T, \mu_T)$ such that for all $i \in [m]$, the projection of $(\mathcal{Q}^i, \mu^i)$ to the coordinates in T is $(\mathcal{Q}_T, \mu_T)$. Hence, no subset of servers of size t will learn anything about the index i of the file that is being requested. If a PIR scheme protects against t colluding servers, it also clearly protects against t' colluding servers for all $t' \leq t$.

4. A general PIR scheme for coded storage with colluding servers.

Our goal is to find high-rate PIR schemes which protect against many colluding servers. To that end, the following construction provides a general PIR scheme for coded databases which protects against a flexible number of colluding servers.

4.1. Scheme construction.

Let C be a linear $[n,k,d]_q$ code with generator matrix G_C, and consider the distributed storage system $Y = XG_C$ as in section 3. We choose another linear code $D \subseteq \mathbb{F}_q^n$, the *retrieval code*. As we will see, the retrieval code essentially determines the privacy properties of the scheme.

Throughout this section, i will denote the index of the file we wish to retrieve. We begin by simplifying notation, defining

$$c := d_{C\star D} - 1. \tag{15}$$

The queries are constructed so that the total response vector during one iteration is of the form

$$r^i = (\text{codeword of } C \star D) + \tilde{y}^i, \tag{16}$$

where $\tilde{y}^i$ is a vector containing c distinct symbols of y^i in known locations, and zeros elsewhere. Multiplying r^i by a generator matrix of $(C\star D)^\perp$ then allows us to recover these c symbols.

To allow the user to download exactly one file over s iterations, we force the file size bk to be an integer multiple of c by setting

$$b = \frac{\operatorname{lcm}(c,k)}{k} \quad \text{and} \quad s = \frac{\operatorname{lcm}(c,k)}{c} \tag{17}$$

so that $bk = sc$. During each iteration of the scheme, we download

$$g := \frac{k}{s} = \frac{c}{b} \tag{18}$$

symbols from every row of y^i. After s iterations, the scheme will have downloaded $sg = k$ symbols of the ath row $y^{i,a}$ of y^i for all $a \in [b]$.

We also fix a subset $J \subseteq [n]$ of servers of size

$$|J| = \max\{c, k\} \tag{19}$$

which stays constant throughout the scheme. By reindexing the servers if necessary, we may assume without loss of generality that

$$J = \{1, \ldots, |J|\}. \tag{20}$$

The set J will be the set of all servers from which we obtain encoded symbols. We will also make use of sets $J_u^a \subseteq J$ where $u \in [s]$ and $a \in [b]$, which are defined so that during the uth iteration we obtain the symbol $y_j^i(a)$ from every server $j \in J_u^a$.

For clarity of presentation, we will describe steps 1–2 in detail for the first iteration of the scheme, which will help elucidate the structure of the queries and responses of subsequent iterations.

1. *Query construction.* We select mb codewords $d^{\ell,a} = [d^{\ell,a}(1)\cdots d^{\ell,a}(n)]$ uniformly at random from D for $\ell \in [m]$ and $a \in [b]$. For $\ell \in [m]$ and $j \in [n]$, define

$$d_j^\ell = \left[d^{\ell,1}(j)\cdots d^{\ell,b}(j)\right] \in \mathbb{F}_q^{1\times b} \quad \text{and} \quad d_j = \left[d_j^1\cdots d_j^m\right] \in \mathbb{F}_q^{1\times mb}. \tag{21}$$

We partition $J_1 := [c] \subseteq J$ into b subsets as follows:

$$J_1^1 = \{1,\dots,g\},\ J_1^2 = \{g+1,\dots,2g\},\ \dots,\ J_1^b = \{g(b-1),\dots,gb=c\}; \tag{22}$$

and we define the queries q_j^i by

$$q_j^i = \begin{cases} d_j + e_{b(i-1)+a} & \text{if } j \in J_1^a, \\ d_j & \text{if } j \notin J_1, \end{cases} \tag{23}$$

where $e_{b(i-1)+a} \in \mathbb{F}_q^{1\times mb}$ denotes the $(b(i-1)+a)$th standard basis vector. Thus for $j \in J_1^a$, the query q_j^i is simply d_j but with the entry $d^{i,a}(j)$ replaced with $d^{i,a}(j)+1$.

2. *Responses.* To understand the response vector r^i, we first calculate r_j^i for $j \notin J_1$. We have

$$r_j^i = \langle q_j^i, y_j\rangle = \langle d_j, y_j\rangle = \sum_{\ell=1}^m \langle d_j^\ell, y_j^\ell\rangle = \sum_{\ell=1}^m\sum_{a=1}^b d^{\ell,a}(j) y_j^\ell(a). \tag{24}$$

For $j \in J_1^{a_0}$ for some $a_0 \in [b]$, the same calculation reveals that

$$r_j^i = \sum_{\ell=1}^m\sum_{a=1}^b d^{\ell,a}(j) y_j^\ell(a) + y_j^i(a_0). \tag{25}$$

We see that the value of the total response vector during the first iteration is

$$r^i = \sum_{\ell=1}^m\sum_{a=1}^b \begin{bmatrix} d^{\ell,a}(1)y_1^\ell(a) \\ \vdots \\ d^{\ell,a}(n)y_n^\ell(a)\end{bmatrix} + \begin{bmatrix} y_1^i(1) \\ \vdots \\ y_g^i(1) \\ \vdots \\ y_{g(b-1)}^i(b) \\ \vdots \\ y_c^i(b) \\ 0_{(n-c)\times 1}\end{bmatrix} = \underbrace{\sum_{\ell=1}^m\sum_{a=1}^b d^{\ell,a}\star y^{\ell,a}}_{\in C\star D} + \begin{bmatrix} y_1^i(1) \\ \vdots \\ y_g^i(1) \\ \vdots \\ y_{g(b-1)}^i(b) \\ \vdots \\ y_c^i(b) \\ 0_{(n-c)\times 1}\end{bmatrix}, \tag{26}$$

where $y^{\ell,a} \in C$ is the ath row of y^ℓ.

3. *Iteration.* During the uth iteration for $u = 2, \ldots, s$, we repeat steps 1–2 but recursively define the subset $J_u^a \subseteq J$ to be the cyclic shift of J_{u-1}^a within J to the right by g indices. Thus if $J_{u-1}^a = \{j_1, \ldots, j_g\}$, then

$$J_u^a = \{j_1 + g, j_2 + g, \ldots, j_g + g\}, \tag{27}$$

where if $j \in J_u^a$ satisfies $j > |J|$, it is replaced with $(j-1) \pmod{|J|} + 1$. We let $J_u = J_u^1 \cup \cdots \cup J_u^b$ and define the queries during the uth iteration by

$$q_j^i = \begin{cases} d_j + e_{b(i-1)+a} & \text{if } j \in J_u^a, \\ d_j & \text{if } j \notin J_u. \end{cases} \tag{28}$$

The response vector r^i during the uth iteration is of the form

$$r^i = (\text{codeword of } C \star D) + y_{J_u}^i, \tag{29}$$

where $y_{J_u}^i$ is a vector with entries $y_j^i(a)$ in some known positions for all $j \in J_u^a$ and all $a \in [b]$, and zeros elsewhere.

4. *Data reconstruction.* Let S be a generator matrix for $(C \star D)^\perp$. Since $c = d_{C\star D} - 1$, every c columns of S are linearly independent. To reconstruct the file x^i, we begin by considering the response vector r^i from the first iteration and computing

$$Sr^i = S(\text{codeword of } C \star D) + S \begin{bmatrix} y_1^i(1) \\ \vdots \\ y_c^i(b) \\ 0_{(n-c)\times 1} \end{bmatrix} = S \begin{bmatrix} y_1^i(1) \\ \vdots \\ y_c^i(b) \\ 0_{(n-c)\times 1} \end{bmatrix}. \tag{30}$$

From Sr^i we can obtain the values of $y_1^i(1), \ldots, y_c^i(b)$, since the first c columns of S are linearly independent. If r^i is instead the response during the uth iteration of the scheme, we similarly obtain all entries of the form $y_j^i(a)$ for $j \in J_u^a$ and $a \in [b]$ from the product Sr^i. For a fixed row a, the sets $J_1^a, \ldots, J_s^a$ are disjoint and consist of $sg = k$ servers in total; hence we retrieve k distinct symbols of $y^{i,a}$ for every row a.

One can visualize the entire PIR scheme as in Figure 1, wherein we show what portions of the encoded file y^i we are downloading during each iteration, for parameters $k = 6$, $c = 4$, and $n = 10$ (in the left-hand figure). Here each file consists of $b = 2$ rows, and the scheme requires $s = 3$ iterations. We have $J = \{1, \ldots, k\}$, and the sets J_u^a are given by

$$\begin{aligned} J_1^1 &= \{1,2\}, \quad & J_1^2 &= \{3,4\}, \\ J_2^1 &= \{3,4\}, \quad & J_2^2 &= \{5,6\}, \\ J_3^1 &= \{5,6\}, \quad & J_3^2 &= \{1,2\}. \end{aligned} \tag{31}$$

In Figure 1 we denote the encoded symbols downloaded during the first iteration in red, those downloaded during the second iteration in blue, and those downloaded during the third iteration in green.

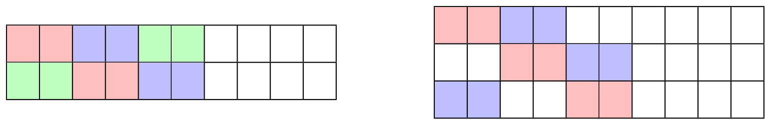

Figure 1. *Visualizing the PIR scheme of section 4.1. On the left, downloading from a system with parameters $(k, c, n) = (6, 4, 10)$. Since $k > c$, we have $J = \{1, \dots, k\}$, and the scheme requires $b = 2$ rows and $s = 3$ iterations. On the right, a system with parameters $(k, c, n) = (4, 6, 10)$. Here $c > k$, so $J = \{1, \dots, c\}$ and the scheme requires $b = 3$ rows and $s = 2$ iterations. Depicted is the encoded file $y^i \in \mathbb{F}_q^{b\times n}$, along with the encoded symbols downloaded in the first (red), second (blue), and third (green) iterations. The columns which contain colored blocks are those in J.*

In the right-hand side of Figure 1 we repeat this exercise for parameters $k = 4$, $c = 6$, and $n = 10$. In this case we have $b = 3$ rows per file and require $s = 2$ iterations. The sets J_u^a are given by

$$\begin{aligned} J_1^1 &= \{1, 2\}, \quad J_1^2 = \{3, 4\}, \quad J_1^3 = \{5, 6\}, \\ J_2^1 &= \{3, 4\}, \quad J_2^2 = \{5, 6\}, \quad J_2^3 = \{1, 2\}, \end{aligned} \tag{32}$$

which are depicted in Figure 1.

4.2. Proofs of correctness and privacy.

In this section we provide proofs that the PIR scheme described in the previous subsection is correct (retrieves the desired file) and preserves privacy (does not reveal the index i of the desired file to any group of t colluding servers).

Theorem 7. *Let C be an $[n, k, d]$-code, and suppose we have a retrieval code D such that either* (i) *$d_{C\star D} - 1 \le k$, or* (ii) *there exists $J \subseteq [n]$ of size $\max\{d_{C\star D} - 1, k\}$ such that every subset of J of size k is an information set of C. Then the PIR scheme of section 4.1 is correct, that is, retrieves the desired file with rate $(d_{C\star D} - 1)/n$.*

Proof. If condition (i) is satisfied, we choose $J \subseteq [n]$ of size k to be any information set of C. In the data reconstruction phase of the PIR scheme, we retrieve k symbols from each row $y^{i,a}$ of y^i, corresponding to the columns belonging to J. Since every $K \subseteq J$ of size k is an information set, this suffices to recover every $y^{i,a}$ and therefore all of x^i. The rate of the scheme is easily seen to be

$$\frac{bk}{ns} = \frac{k \cdot \frac{\operatorname{lcm}(c,k)}{k}}{n \cdot \frac{\operatorname{lcm}(c,k)}{c}} = \frac{d_{C\star D} - 1}{n}, \tag{33}$$

which completes the proof. ■

Theorem 8. *Then the PIR scheme described in section 4.1 protects against $d_{D^\perp} - 1$ colluding servers.*

Proof. Let $T = \{j_1, \dots, j_t\} \subseteq [n]$ be a set of servers of size $t \le d_{D^\perp} - 1$. We begin by showing that during a single iteration, we have $I(q^i_{j_1}, \dots, q^i_{j_t}; i) = 0$. From $t \le d_{D^\perp} - 1$ it follows immediately that every t columns of the generator matrix of D are linearly independent. Therefore, the code D_T is the entire space $\mathbb{F}_q^t$.

First consider the distribution of one of the vectors

$$d_j = [d^{1,1}(j) \cdots d^{1,b}(j) \cdots d^{m,1}(j) \cdots d^{m,b}(j)] \in \mathbb{F}_q^{1\times bm} \tag{34}$$

for a single $j \in T$. As $D_{\{j\}}$ is distributed uniformly on $\mathbb{F}_q$, and the codewords $d^{\ell,a}$ are selected uniformly at random from D, it follows that d_j is uniform on $\mathbb{F}_q^{1\times bm}$.

Similarly, as D_T is all of $\mathbb{F}_q^t$, we see that the joint distribution $\{d_j \mid j \in T\}$ is uniform over $\left(\mathbb{F}_q^{1\times bm}\right)^t$. If $f(i,j)$ denotes the index of the standard basis vector as in (23) and (28), we see that

$$\{q^i_{j1},\dots,q^i_{jt}\} = \{d_j + e_{f(i,j)} \mid j \in T \cap J\} \cup \{d_j \mid j \in T \setminus J\} \tag{35}$$

is uniformly distributed for all i, as translating the uniform distribution by any vector results again in the uniform distribution. The distribution $\{q^i_{j1},\dots,q^i_{jt}\}$ of the queries is therefore independent of the index i of the desired file; hence $I(q^i_{j1},\dots,q^i_{jt};i) = 0$ is satisfied for a single iteration.

Now consider the joint distribution Q^i_T of all queries to all servers in T, as we range over all iterations of the scheme. For each iteration, the vectors $d^{\ell,a}$ are chosen independently of all other iterations, from which arguments identical to the above show that Q^i_T is uniform on $\left(\mathbb{F}_q^{1\times bm}\right)^{ts}$. Thus $I(Q^i_T;i) = 0$ as desired. ■

4.3. Examples.

In this subsection we show how some previously constructed PIR schemes fit into the general framework of our scheme. Throughout this section, we assume that i is the index of the file we wish to retrieve. In case either $b = 1$ or $s = 1$, we will suppress all indices relating to rows or iterations, respectively.

Example 1. Let C be any systematic $[n,k,d]$ storage code, and set $D = \mathrm{Rep}(n)$ so that $C \star D = C$. The above-outlined scheme has rate $(d_C - 1)/n$ and as $d_{D^\perp} - 1 = 1$, it only provides privacy against noncolluding servers ($t = 1$). For simplicity we assume $d_C - 1|k$ and therefore require only $b = 1$ row per file but $s = k/(d_C - 1)$ iterations. We set $J = [k]$.

As $D = \mathrm{Rep}(n)$, sampling from D uniformly at random amounts to sampling uniformly at random from $\mathbb{F}_q$ itself. Thus the query construction in this example amounts to selecting a single vector $d_0 = [d^1 \cdots d^m] \in \mathbb{F}_q^m$ uniformly at random, and setting $d_1 = \cdots = d_n = d_0$. Now set $J_1 = [d_C - 1]$, and define the queries by

$$q^i_j = \begin{cases} d_0 + e_i & \text{if } j \in J_1, \\ d_0 & \text{if } j \notin J_1. \end{cases} \tag{36}$$

The total response vector is then easily seen to be

$$r^i = \sum_{\ell=1}^{m} d^\ell y^\ell + \begin{bmatrix} y^i(1) \\ \vdots \\ y^i(d_C - 1) \\ 0_{(n-(d_C-1))\times 1} \end{bmatrix}. \tag{37}$$

Let S be a generator matrix of $C^\perp$ which is in systematic form. Then

$$Sr^i = \left[y^i(1)\cdots y^i(d_C-1)\right] = \left[x^i(1)\cdots x^i(d_C-1)\right]. \tag{38}$$

In the second iteration, we obtain J_2 by shifting J_1 to the next set of $d_C - 1$ servers and repeat the above in the obvious way. After $k/(d_C - 1)$ iterations we recover $[y^i(1)\cdots y^i(k)] = x^i$.

Privacy against any single server is clear, since for a fixed j the queries q_j^i are independent samples of the uniform distribution on $\mathbb{F}_q^m$.

This is essentially a paraphrasing of the scheme of [16, Theorem 1] in the language of our construction. However, in [16], the scheme is only presented for MDS codes.

Example 2. Let C be any $[n,k]$ MDS code, and set $D = C^\perp$. We have $(C \star D)^\perp = \mathrm{Rep}(n)$ by Proposition 3(iv). Since $d_{C\star D} - 1 = 1$ and $d_{D^\perp} - 1 = n-k$, the above-outlined scheme has rate $1/n$ and provides privacy against any $n-k$ colluding servers.

We have $b = 1$ row per file and require $s = k$ iterations of the scheme. We set $J = [k]$ and $J_1 = \{1\}$, so that the queries in the first iteration are given by

$$q_j^i = \begin{cases} d_j + e_i & \text{if } j = 1, \\ d_j & \text{if } j > 1, \end{cases} \tag{39}$$

where the vectors $d_j \in \mathbb{F}_q^n$ are as in the construction. The response vector is

$$r^i = \sum_{\ell=1}^m d^\ell \star y^\ell + \begin{bmatrix} y^i(1) \\ 0_{(n-1)\times 1} \end{bmatrix}, \tag{40}$$

where $d^\ell \in D$ and $y^\ell \in C$. As $(C \star D)^\perp = \mathrm{Rep}(n)$, the reconstruction function takes a particularly simple form. In particular, we can take $S = [1 \cdots 1]$ and see immediately that $Sr = y^i(1)$. Iterating this procedure k times while setting $J_u = \{u\}$ for $u \in [k]$ yields $[y^i(1) \cdots y^i(k)]$, which suffices to reconstruct x^i by the MDS property.

This is the scheme of [16, Theorem 2], again rephrased in the context of our scheme.

Example 3. Let $C = \mathrm{Rep}(n)$, and let D be any $[n,t]$ MDS code. We have $d_{C\star D} - 1 = d_D - 1 = n - t$ and $d_{D^\perp} - 1 = t$. The above-outlined scheme thus has rate $(n-t)/n$ and provides privacy against any t colluding servers. We have $b = n-t$ rows per file but require only $s = 1$ iteration of the scheme. We set $J = [n-t]$.

With $d_j \in \mathbb{F}_q^{1\times m(n-t)}$ as in the scheme construction, the queries q_j^i are of the form

$$q_j^i = \begin{cases} d_j + e_{(n-t)(i-1)+j} & \text{if } j \in J, \\ d_j & \text{if } j \notin J, \end{cases} \tag{41}$$

which yields a response vector of the form

$$r^i = \sum_{\ell=1}^m \sum_{a=1}^{n-t} x^\ell(a) d^{\ell,a} + \begin{bmatrix} x^i(1) \\ \vdots \\ x^i(n-t) \\ 0_{t\times 1} \end{bmatrix}. \tag{42}$$

If S is a generator matrix of $D^\perp$ in systematic form, then $Sr^i = x^i$, which completes the retrieval scheme.

4.4. Remarks on scheme construction. Theorem 7 implies that for *any* storage code C at all, provided that we choose D such that $d_{C\star D} - 1 \leq k$, we can choose J to be any information set of C and achieve rate $(d_{C\star D} - 1)/n$ and protection against $d_{D^\perp} - 1$ colluding servers. In particular, if C is any MDS code, then we can choose any subset J of servers of size $|J| = \max\{d_{C\star D} - 1, k\}$. Thus the scheme achieves the stated rate and protection against collusion for any MDS storage code C and any retrieval code D.

It is likely that condition (ii) in Theorem 7 is somewhat conservative. We do not really need every subset of J of size k to be an information set of C—only subsets of the form $J_1^a \cup \cdots \cup J_s^a$ for $a \in [b]$, which index the servers retrieving symbols from the ath row of y^i. However, we prefer to state condition (ii) as it is for the sake of simplicity.

Theorem 8 implies that we must have $\operatorname{supp}(D) = [n]$ to achieve any nontrivial privacy with our scheme. For if $\operatorname{supp}(D) \neq [n]$, then we would have some standard basis vector in $D^\perp$, implying $d_{D^\perp} - 1 = 0$. This can be interpreted by saying that every server has to see some amount of randomness.

It may be the case that the user does not have the freedom to adjust the number of rows in a file. For example, each file might be stored as a single row of X, in which case it is not obvious how to take advantage of a scheme which downloads more symbols per iteration than there are in a file. One way to remedy this is to simply have the user download multiple files. Thus one could rephrase Example 3 so that the user downloads $n - t$ files from the database, instead of one file which consists of $n-t$ symbols. However, according to recent results [2], the capacity of such multimessage PIR is higher than that of single-message PIR. Thus to make a more valid comparison with known rate and capacity results, we have chosen to describe our schemes as only retrieving one file.

While our interest in this paper is in download cost, we observe that the user in each iteration of our scheme uploads bnm symbols from $\mathbb{F}_q$, for a total of $bnms$ uploaded symbols. In particular, while the download cost does not depend on the number of files stored, the upload cost grows linearly in m. While the upload cost also depends linearly on b and s, the size of the file does as well, so while the upload cost grows with these parameters, so does the total amount of privately downloaded information.

5. Private information retrieval from GRS codes.

In [16], two PIR schemes were presented for arbitrary $[n, k]$ MDS codes, one of which had rate $\frac{1}{n}$ and protected against $t = n-k$ colluding servers, and the other of which had rate $\frac{n-k}{n}$ and $t = 1$. These schemes are essentially variations of Examples 1 and 2 in this paper. The authors asked if one can adapt their schemes in the "intermediate regime" where $1 < t < n - k$ for arbitrary n and k. In this section, we show how to do this via the construction in section 4 for some suitably chosen $[n, k]$ MDS storage codes, namely for GRS codes.

By Proposition 3 we know that the class of all GRS codes associated to a fixed n-tuple $\alpha \in \mathbb{F}_q^n$ is closed under taking star products and duals. Moreover, while the dimension of a star product $C\star D$ of two generic codes can be as high as $\dim(C)\cdot\dim(D)$, in the case of GRS codes it is only $\dim(C) + \dim(D) - 1$, which is useful when we want to maximize minimum distances. Indeed, the following theorem from [9], which can be seen as a multiplicative version of the Singleton bound, shows that among all storage codes, our PIR schemes give the best privacy-rate tradeoff precisely for GRS storage codes.

Theorem 9 (immediate corollary of [9, Theorem 2] and [8, Theorem 14]). *Let C_1 and C_2 be linear codes of dimension k_1 and k_2 and support $[n]$. Then*

$$d_{C_1\star C_2} - 1 \leq \max\{0, n-(k_1+k_2-1)\}. \tag{43}$$

Conversely, if neither C_1, C_2, nor $(C_1\star C_2)^\perp$ is the length n repetition code, then the above bound is an equality exactly when both C_1 and C_2 are GRS codes.

By Proposition 3(iv), GRS codes satisfy Theorem 9 with equality. The following theorem instantiates our scheme in the case where the storage code C and retrieval code D are GRS.

Theorem 10. *Let $C = GRS_k(\alpha, v)$, and consider the distributed storage system $Y = XG_C$ as in section 3. Then for all t such that $1 \leq t \leq n-k$, there exists a retrieval code D such that the PIR scheme constructed in section 4 has rate $\frac{n-(k+t-1)}{n}$ and protects against any t colluding servers.*

Proof. We will give a linear code D satisfying $d_{D^\perp} - 1 = t$ and $d_{C\star D} - 1 = n-(k+t-1)$; the theorem then follows immediately from Theorems 7 and 8. Let $D = GRS_t(\alpha, u)$ for an arbitrary vector $u \in \mathbb{F}_q^{\times n}$; since its dual is also MDS we see that $d_{D^\perp} = t+1$. Then $(C\star D)^\perp = GRS_{n-k-t+1}(\alpha, w)$ with w given by (8). It follows that $d_{C\star D} - 1 = n-(k+t-1)$ as desired. ■

When $k+t > n$, our PIR scheme has rate zero, regardless of the storage code chosen. This can be seen readily from Theorem 9, as the retrieval code D must have rank at least t, so $d_{C\star D} - 1 \leq \max\{0, n-(k+t-1)\} = 0$, and thus the number of retrieved symbols per iteration is $d_{C\star D} - 1 = 0$.

When $k = 1$, that is, the data is stored via a replication system, our scheme provides a rate of $\frac{n-t}{n}$. It is known by [13] that the capacity for PIR in this case is $\frac{1-t/n}{1-(t/n)^m}$. Thus our scheme is asymptotically capacity-achieving in that the resulting rates approach capacity as the number of files $m \to \infty$.

Similarly, when $t = 1$, that is, without server collusion, our scheme provides a rate of $\frac{n-k}{n}$. By [1] the capacity for PIR in this case is $\frac{1-k/n}{1-(k/n)^m}$; thus our schemes are again asymptotically capacity-achieving.

The capacity of coded PIR with colluding servers is known when $m = 2$ and $k = n-1$ by [15], but no general result is known for $k > 1$ and $t > 1$. A previous version of this paper conjectured the following.

Conjecture 1 (disproven; see [15]). *Let C be an $[n,k,d]$ code that stores m files via the distributed storage system $Y = XG_C$, and fix $1 \leq t \leq n-k$. Any PIR scheme for Y that protects against any t colluding servers has rate at most $\frac{1-\frac{k+t-1}{n}}{1-(\frac{k+t-1}{n})^m}$.*

However, this conjecture was disproven in [15], where the authors exhibited an explicit PIR scheme for $m = 2$ files distributed over $n = 4$ servers using a rate $1/2$ storage code C, which protects against $t = 2$ collusion. The exhibited scheme has rate $3/5$, while our conjectured capacity was $4/7$.

We will refrain from stating any further conjectures on the capacity of coded PIR with server collusion. However, the question remains open as to whether our schemes are asymptotically capacity-achieving as $m \to \infty$ for general k and t. This is consistent with the

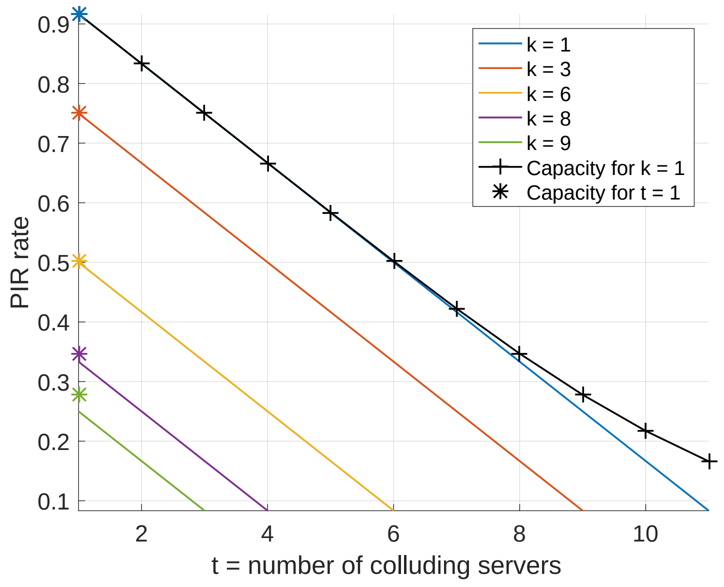

Figure 2. *Achievable PIR rates for $n = 12$ servers and $m = 8$ files, as a function of $t =$ the number of colluding servers, for various storage code rates. The black curve represents the PIR capacity for the case of $k = 1$ (data stored via a replication system) as computed from* [13]*. The asterisks show the capacity for the noncolluding case $t = 1$ as given in* [1]*. The PIR capacity is unknown when $t \geq 2$ and $k \geq 2$.*

results in [15], where it is also proven that, although positive retrieval rates are possible when $k+t > n$, the rates decrease to 0 as $m \to \infty$. We further remark that the rates of our schemes do not depend on the number of files stored, and the field size required to achieve the rates of Theorem 10 is only $q \geq n$, which is needed to guarantee the existence of GRS codes. This is in contrast with the capacity-achieving schemes of [13, 14, 15], wherein the field size grows as $q = O(n^m)$. Similarly, for the scheme of [18] for MDS coded data with colluding servers, which outperforms our scheme when the number of files m is small, the field size is required to satisfy $q \geq O\left(\binom{n}{k}\right)$.

In Figure 2 we plot for $n = 12$ servers the achievable PIR rates as a function of the number of colluding servers t, for various code rates k/n. The black curve represents the capacity for the case of $k = 1$, that is, when the data is stored using a replication system [13], while the asterisks represent the capacity obtained in [1] for the noncolluding case $t = 1$ at different code rates. One can see that even for a relatively small number of files and a relatively large amount of collusion, our scheme is quite close to capacity.

6. An example in the intermediate regime.

We will illustrate our scheme with an explicit example in the case when $t = 2$ and $k = 2$. This is the first case not covered by our Examples 1–3. The storage code C is MDS, with $[n, k, d] = [5, 2, 4]$. We will have $c = k = 2$, and hence we require only $b = 1$ row per file and $s = 1$ iteration; thus we are free to ignore these parameters in what follows.

Example 4. Let $\alpha = [0, 1, 2, 3, 4] \in \mathbb{F}_5^5$, and let $\mathbf{1} \in \mathbb{F}_5^5$ be the all-ones vector. Consider the storage code $C = GRS_2(\alpha, \mathbf{1})$ over $\mathbb{F}_5$, encoded by its systematic generator matrix

$$G_C := \begin{bmatrix} 1 & 0 & 4 & 3 & 2 \\ 0 & 1 & 2 & 3 & 4 \end{bmatrix} \tag{44}$$

as in (7). So each file x^i is divided into two blocks, $x^i(1)$ and $x^i(2)$, and distributed onto the five servers as follows:

$$\begin{cases} x^i(1) & \text{on server } 1, \\ x^i(2) & \text{on server } 2, \\ 4x^i(1) + 2x^i(2) & \text{on server } 3, \\ 3x^i(1) + 3x^i(2) & \text{on server } 4, \\ 2x^i(1) + 4x^i(2) & \text{on server } 5. \end{cases} \tag{45}$$

The random codewords $d^1, \ldots, d^m$ used to query the servers will be drawn from $D = GRS_2(\alpha, \mathbf{1})$, for which we choose the canonical generator matrix

$$G_D := \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 & 4 \end{bmatrix}. \tag{46}$$

Note that $D^\perp$ is a $[5, 3]$ MDS code, so our scheme protects against $t = d_{D^\perp} - 1 = 2$ colluding servers. The reason we choose different generator matrices for C and D is practical, as the systematic generator matrix is better for decoding, while the canonical generator matrix is preferable for computations.

Observe that $C \star D = GRS_3(\alpha, \mathbf{1})$. We compute its dual $(C \star D)^\perp = GRS_2(\alpha, u)$, where $u_i = (\prod_{j \neq i}(\alpha_i - \alpha_j))^{-1}$ for $i = 1, \ldots, 5$. Since α_i runs over the entire field $\mathbb{F}_5$, these products are unusually easy to evaluate; indeed, we have $u_i = -1$ for all $i = 1, \ldots, 5$. Thus $(C \star D)^\perp = GRS_2(\alpha, -\mathbf{1}) = GRS_2(\alpha, \mathbf{1})$, so $(C \star D)^\perp$ is identical to C, and we use the same generator matrix $G_H = G_C$.

For each file index $\ell \in [m]$, we sample uniformly at random from D by multiplying G_D on the left by a uniform random vector $z^\ell = [z^\ell(1), z^\ell(2)] \in \mathbb{F}_5^2$, so that $d^\ell = z^\ell G_D$ and $d_j = [d^1(j) \cdots d^m(j)]$ for $j \in [m]$. We let $z_1 = [z^\ell(1) \cdots z^m(1)]$ and $z_2 = [z^1(2) \cdots z^m(2)]$, which are independent and uniformly distributed over $\mathbb{F}_5^m$.

Suppose we want to retrieve the file x^i for some $i \in [m]$. We select $d_{C \star D} - 1 = 2$ servers from which to download blocks from x^i, and for simplicity here we choose the systematic nodes. The queries q^i_j sent to the servers will now be the following vectors in $\mathbb{F}_5^m$:

$$\begin{aligned} q^i_1 &= d_1 + e_i &&= z_1 + e_i, \\ q^i_2 &= d_2 + e_i &&= z_1 + z_2 + e_i, \\ q^i_3 &= d_3 &&= z_1 + 2z_2, \\ q^i_4 &= d_4 &&= z_1 + 3z_2, \\ q^i_5 &= d_5 &&= z_1 + 4z_2, \end{aligned} \tag{47}$$

where e_i is the ith standard basis vector. Observe that for each pair of servers, the corresponding joint distribution of queries is the uniform distribution over $(\mathbb{F}_5^m)^2$.

The servers now respond by projecting their stored data onto the query vector, whence we obtain a response vector

$$r^i = \begin{bmatrix} \sum_{\ell=1}^m d^\ell(1)x^\ell(1) & +x^i(1) \\ \sum_{\ell=1}^m (d^\ell(1)+d^\ell(2))x^\ell(2) & +x^i(2) \\ \sum_{\ell=1}^m (d^\ell(1)+2d^\ell(2))(4x^\ell(1)+2x^\ell(2)) & \\ \sum_{\ell=1}^m (d^\ell(1)+3d^\ell(2))(3x^\ell(1)+2x^\ell(2)) & \\ \sum_{\ell=1}^m (d^\ell(1)+4d^\ell(2))(2x^\ell(1)+4x^\ell(2)) & \end{bmatrix} \in C\star D + \begin{bmatrix} x^i(1) \\ x^i(2) \\ 0 \\ 0 \\ 0 \end{bmatrix}. \tag{48}$$

To finally decode the desired symbols, we now compute the matrix product $G_{(C\star D)^\perp}\cdot r^i$. One calculates that indeed

$$G_{(C\star D)^\perp}\cdot r^i = G_{(C\star D)^\perp}\cdot \begin{bmatrix} x^i(1) \\ x^i(2) \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} x^i(1) \\ x^i(2) \end{bmatrix}. \tag{49}$$

We have therefore extracted the two desired data blocks using five queries, while maintaining privacy against $t = 2$ colluding servers.

REFERENCES

[1] K. Banawan and S. Ulukus, *The Capacity of Private Information Retrieval from Coded Databases*, preprint, https://arxiv.org/abs/1609.08138, 2016.

[2] K. Banawan and S. Ulukus, *Multi-Message Private Information Retrieval: Capacity Results and Near-Optimal Schemes*, preprint, https://arxiv.org/abs/1702.01739, 2017.

[3] S. Blackburn, T. Etzion, and M. Paterson, *PIR Schemes with Small Download Complexity and Low Storage Requirements*, preprint, https://arxiv.org/abs/1609.07027, 2016.

[4] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan, *Private information retrieval*, in Proceedings of the IEEE Annual Symposium on Foundations of Computer Science, IEEE, Washington, DC, 1995, pp. 41–50.

[5] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan, *Private information retrieval*, J. ACM, 45 (1998), pp. 965–981.

[6] A. Fazeli, A. Vardy, and E. Yaakobi, *PIR with Low Storage Overhead: Coding Instead of Replication*, preprint, https://arxiv.org/abs/1505.06241, 2015.

[7] S. Kumar, E. Rosnes, and A. G. i Amat, *Private Information Retrieval in Distributed Storage Systems using an Arbitrary Linear Code*, preprint, https://arxiv.org/abs/1612.07084, 2016.

[8] D. Mirandola and G. Zémor, *Critical pairs for the product Singleton bound*, IEEE Trans. Inform. Theory, 61 (2015), pp. 4928–4937, https://doi.org/10.1109/TIT.2015.2450207.

[9] H. Randriambololona, *An upper bound of Singleton type for componentwise products of linear codes*, IEEE Trans. Inform. Theory, 59 (2013), pp. 7936–7939.

[10] N. B. Shah, K. V. Rashmi, and K. Ramchandran, *One extra bit of download ensures perfectly private information retrieval*, in Proceedings of the 2014 IEEE International Symposium on Information Theory, IEEE, Washington, DC, 2014, pp. 856–890.

[11] N. B. Shah, K. V. Rashmi, K. Ramchandran, and P. V. Kumar, *Privacy-Preserving and Secure Distributed Storage Codes*, http://people.eecs.berkeley.edu/~nihar/publications/privacy_security.pdf, 2013.

[12] R. C. Singleton, *Maximum distance q-nary codes*, IEEE Trans. Inform. Theory, 10 (1964), pp. 116–118.

[13] H. Sun and S. Jafar, *The Capacity of Robust Private Information Retrieval with Colluding Databases*, preprint, https://arxiv.org/abs/1605.00635, 2016.

[14] H. Sun and S. Jafar, *The capacity of private information retrieval*, IEEE Trans. Inform. Theory, 63 (2017), pp. 4075–4088, https://doi.org/10.1109/TIT.2017.2689028.

[15] H. Sun and S. Jafar, *Private Information Retrieval from MDS Coded Data with Colluding Servers: Settling a Conjecture by Freij-Hollanti et al.*, preprint, https://arxiv.org/abs/1701.07807, 2017.

[16] R. Tajeddine and S. El Rouayheb, *Private information retrieval from MDS coded data in distributed storage systems*, in Proceedings of the 2016 IEEE International Symposium on Information Theory (ISIT), IEEE, Washington, DC, 2016, pp. 1411–1415; see http://www.ece.iit.edu/~salim/ for an extended version.

[17] J. H. van Lint and R. M. Wilson, *On the minimum distance of cyclic codes*, IEEE Trans. Inform. Theory, 32 (1986), pp. 23–40.

[18] Y. Zhang and G. Ge, *A General Private Information Retrieval Scheme for MDS Coded Databases with Colluding Servers*, preprint, https://arxiv.org/abs/1704.06785, 2017.