Querying Twice to Achieve Information-Theoretic Verifiability in Private Information Retrieval

Stanislav Kruglik*, Son Hoang Dau†, Han Mao Kiah*, Huaxiong Wang*, Liang Feng Zhang‡

***Abstract*—Private Information Retrieval (PIR) protocols allow a client to retrieve any file of interest while keeping the files identity hidden from the database servers. While many existing PIR protocols assume servers to be honest but curious, we investigate the scenario of dishonest servers that provide incorrect answers to mislead clients into obtaining wrong results. We propose a unified framework for polynomial PIR protocols encompassing various existing protocols that optimize the download rate or total communication cost. We introduce a way to transform a polynomial PIR to a verifiable one without increasing the number of involved servers by doubling the queries. The security guarantees can be information-theoretic or computational, and the verification keys can be public or private. Moreover, in one of our protocols, the ratio between the additional download overhead associated with verification and the normal download cost approaches zero as the file size goes to infinity.**

***Index Terms*—Private Information Retrieval, Security, Verifiability, Privacy, Communication cost**

I. Introduction

Consider a database with n files represented by an n-tuple $\boldsymbol{x} = x_1 \cdots x_n$. Suppose that a client wants to retrieve a file x_i with $i \in [n]$. A *private information retrieval* (PIR) protocol allows him/her to retrieve x_i, while keeping its *identity* or *index* $i \in [n]$ private from the database servers [2]. The problem is motivated by the necessity to preserve the privacy of not only the downloaded content, but also the identity of the queried record [3]. Examples include the price of a specific stock or the details of a specific blockchain transaction. A trivial solution is simply to download the entire database and this clearly incurs tremendous communication costs. Unfortunately, in the case of a single server, Chor *et al.* [2] showed that this was the best information-theoretically secure solution. Nevertheless, in the same seminal paper, Chor *et al.* [2] showed that when the contents are replicated among several servers, the communication cost can be significantly reduced. Following [2], several authors have introduced PIR protocols that progressively reduced the communications cost [4], [5]. Formally, in this model, the client queries each of the k servers (each storing $\boldsymbol{x} = x_1 \cdots x_n$) once and retrieves x_i, while keeping the index i private from any subset of up to t honest-but-curious servers. In PIR literature, such a scheme is called *t-private k-server PIR scheme* and such a property is known as *t-privacy*. Motivated by the large size of stored files, the notion of PIR was revisited by the information theory community and the main goal was shifted to reducing the *download rate* – defined as the ratio of the retrieved file size to the amount of downloaded information [6]–[8]. In this case, we can define the *PIR capacity* as the maximum achievable download rate and this value was determined for a variety of PIR models [9].

Currently, many PIR schemes assume that servers are honest-but-curious and that they provide correct responses. This assumption, while theoretically interesting, is rather naive and irrelevant in practice, especially in a decentralized setting (e.g. in the blockchain environment) where servers are not supposed to be trusted. Given the frequencies of high-profile data breaches nowadays, even when the PIR servers are managed by trusted parties (e.g. government agencies) or outsourced to trustworthy service providers (e.g. AWS, Microsoft, and Google), we can't avoid the possibility that they could be attacked and controlled momentarily by malicious entities. This poses an interesting question: what can the client do if servers intentionally provide wrong responses? Here, we provide three different interpretations of this question and their formal definitions.

- *s-verifiablility (referred to as s-security in [10]).* The client can detect the presence of up to s malicious servers.
- *a-accountability*. The client can identify each of up to a malicious servers.
- *b-byzantine resistance/b-byzantine robustness*. The client can retrieve the correct result in the presence of up to b malicious servers.

It is clear that a-accountability implies a-verifiablility, while b-byzantine resistance implies both b-accountability and b-verifiablility. However, in certain low-latency applications, such as private media browsing [11], the requirement of byzantine resistance may be too stringent, as it may be sufficient to detect cheating and then refuse to pay. By simply requiring s-verifiability, we may obtain some savings in communication costs and number of servers involved. Recently, s-verifiable PIR schemes were studied in [10], [12], [13]. In comparison to them, our schemes generally provide better communication complexity and do not involve additional servers. Also, recently proposed committed PIR [14] and authenticated PIR [15] provide the same notion of verifiability. However, these schemes rely on computational assumptions (we discuss this in detail shortly). In comparison, we prove information-

*Stanislav Kruglik, Han Mao Kiah, and Huaxiong Wang are with the School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore (email: {stanislav.kruglik, hmkiah, hxwang}@ntu.edu.sg).

†Son Hoang Dau is with the School of Computing Technologies, STEM College, RMIT University, Melbourne, Australia (email: son-hoang.dau@rmit.edu.au).

‡Liang Feng Zhang is with the School of Information Science and Technology, ShanghaiTech University, Shanghai, China. (email: zhanglf@shanghaitech.edu.cn)

This paper was presented in part at the 2023 IEEE International Symposium on Information Theory [1] [DOI: 10.1109/ISIT54713.2023.10206529].

Corresponding author: *Stanislav Kruglik*.

theoretic security guarantees for certain protocols in this paper. The verifiable PIR protocols [16], [17] provide accountability - but rely on public-key cryptography and require a trusted setup. Another class of verifiable PIR schemes are b-byzantine-resistant PIR schemes [18]–[23]. Typically, such schemes rely on error-correcting techniques and require additional servers. In particular, these schemes cannot be deployed for two servers. Please refer to recent surveys [3], [24] and references therein for more details about recent advances in byzantine PIR.

In this paper, we continue the line of research started in [1], where we considered a two-server verifiable extension of Goldberg's PIR from [25], and for the first time, propose a unified framework that includes a number of well-known PIR protocols without restriction on the number of involved servers for the replicated database case, such as those in [2], [4], [25]. Notably, our framework can encompass both information-theoretic and computer science PIR settings, with a focus on download cost and total communication cost, respectively, under the same umbrella. We also demonstrate how to transform proposed schemes to publicly verifiable setup allowing third party to verify as well and thus settle the possible disputes between client and servers, and how to reduce extra communication incurred by verification. Then we demonstrate that, under certain conditions, these PIR protocols can be transformed into a verifiable PIR scheme with explicit security guarantees. Roughly speaking, our transformation involves the client sending two independent queries to each server. Hence, the client obtains two independent equations that he/she can use to verify the retrieval result. We note that our protocol does not increase the number of involved servers and is non-interactive in the sense that queries sent to a given server can be combined into a single query. The same holds for the answers to them. This eliminates the requirement of preserving an active connection to servers, typical for interactive schemes, for instance, schemes from [26]–[28]. Despite doubling the queries for each server, the fact that we can separate queries for retrieval and verification can be utilized to make the extra download incurred by verification negligibly small.

We noted at the time of writing that the idea of sending two queries to achieve verifiability was also independently developed in [15]. However, there are significant differences between our work and theirs, for example, the construction in [15] relies on functional secret sharing schemes that are closed under scalar multiplication, which, to the best of our knowledge, can only be computationally secure.

The rest of the paper is organized as follows. In Section II, we establish notations and definitions. In Section III, we introduce polynomial-based PIR protocol and show how several existing protocols can be encapsulated into it. Section IV describes the main result of this paper – transforming a polynomial PIR into verifiable PIR and its different generalizations. Comparison with related works is given in Section V. The paper is concluded in Section VI.

II. Preliminaries

For a positive integer $n > 0$, we denote the set of integers $\{1, \dots, n\}$ by $[n]$ and the set of non-negative integers by $\mathbb{Z}_{\geq 0}$.

Symbol	Description
k	Number of servers
t	Privacy and verifiability thresholds that coincide for schemes presented in this paper
n	Number of files in database
$\mathbf{q}$	Queries
$\mathbf{a}$	Answers
vk	Verification key
v	Secret parameter used in verification equations
ω	Generator of cyclic multiplicative group
$\mathbf{x}$	Database
$F_{\mathbf{x}}$	Multivatriate polynomial that represents the database
d	Degree of multivatiate polynomial
m	Number of variables in multivariate polynomial
o	Order of partial derivatives used in retrieval

TABLE I: Notation table.

For a prime number p, we denote the finite field of p elements by $\mathbb{F}_p$ and call it *the base field*. Let $\mathbb{F}_{p^e}$ be its field extension of degree e and we call it *the extended field*. To denote the vector space built over them, we add a corresponding dimension as a superscript. For a vector $\boldsymbol{V} \in \mathbb{F}^n$ and index $i \in [n]$, we denote by $\boldsymbol{V}_i$ the i-th component of $\boldsymbol{V}$. For a polynomial $f \in \mathbb{F}[u]$, we denote its o-th order derivative by $f^{(o)}(u)$. For the convenience of the reader, we introduce Table I, which contains the main notations used throughout the text.

A. Our model

Our model comprises a single client and k servers $S_1, \dots, S_k$. Each server has a copy of a database $\boldsymbol{x} = x_1 \cdots x_n \in \mathbb{F}_{p^e}^n$ with n files. In other words, x_i represents the file with index i and is defined over the extended field $\mathbb{F}_{p^e}$. The client wants to privately retrieve the value of x_i in presence of up to t colluding servers. This is the classical notion of PIR introduced by the seminal paper of Chor *et al.* [2] for $t = 1$. Let us formally define a PIR protocol.

Definition 1 (PIR)**.** A k-server *PIR protocol* comprises three algorithms that can be described as follows:

- $(\boldsymbol{q}_1, \dots, \boldsymbol{q}_k, \text{aux}_A, \text{aux}_R) \leftarrow \texttt{QueriesGen}(n, i)$ is a randomized *query-generation* algorithm for the client. As input, it takes the database size n and retrieval index i, and outputs queries $\boldsymbol{q}_1, \dots, \boldsymbol{q}_k$ and the auxiliary information aux_A used in answer-generation and aux_R used in retrieval. Query $\boldsymbol{q}_j$ with the auxiliary information aux_A is sent to server S_j while aux_R is kept private from servers and used in the *retrieval* algorithm.
- $\boldsymbol{a}_j \leftarrow \texttt{AnswerGen}(j, \boldsymbol{q}_j, \boldsymbol{x}, \text{aux}_A)$ is a deterministic *answer-generation* algorithm for server S_j. As inputs, it takes server number j, query $\boldsymbol{q}_j$, database $\boldsymbol{x} \in \mathbb{F}_{p^e}^n$ and auxiliary information aux_A, and outputs the answer $\boldsymbol{a}_j$.
- $x_i \leftarrow \texttt{Retrieve}(i, \boldsymbol{a}_1, \dots, \boldsymbol{a}_k, \text{aux}_R)$ is a deterministic *retrieval* algorithm for the client. As inputs, it takes the retrieval index i, servers answers $\boldsymbol{a}_1, \dots, \boldsymbol{a}_k$ and auxiliary information aux_R, and uses them to reconstruct x_i.

Any PIR protocol must satisfy the following correctness and privacy properties. As these properties are standard in the literature of PIR, we only give their informal definitions here. Note that while privacy can be defined in a computational

setting as well (see [29]), we focus on information-theoretic privacy in this work.

Definition 2 (Information-Theoretic Privacy)**.** A PIR protocol is *t-private* if any subset of t or less servers get no information about the file index from queries to them.

Definition 3 (Correctness)**.** A PIR protocol is *correct* if the client is always able to successfully retrieve the file from all k answers given that all servers correctly follow the protocol

In most prior work, one assumes that the PIR servers are honest-but-curious. That is, the servers provide correct answers but are interested in learning the index of the retrieved file. The task is to prevent the servers from learning the file index. However, assuming that all servers are honest and always follow the protocol is unrealistic nowadays, given the fact that servers can be hacked and even the most trusted parties can become malicious if under cyber attacks. Thus, in this paper, we consider the model where up to t servers are dishonest and collude to provide wrong answers. Nevertheless, the client can still *detect* that the retrieved file is incorrect. Formally, we define the k-server PIR protocol with result verification below.

Definition 4 (PIR with result verification)**.** A k-server *PIR protocol with result verification* comprises three algorithms.

- $(\mathsf{vk}, \boldsymbol{q}_1, \ldots, \boldsymbol{q}_k, \text{aux}_A, \text{aux}_R, \text{aux}_V) \leftarrow \texttt{QueriesGen}(n, i)$ is a randomized *query-generation* algorithm for the client. As before, the inputs are: database size n and index i, and the outputs are: verification key vk, queries $\boldsymbol{q}_1, \ldots, \boldsymbol{q}_j$ and auxiliary information aux_A and aux_V. As before, Query $\boldsymbol{q}_j$ and aux_A are sent to server S_j, while aux_R and aux_V are kept private from servers and used in *verification* algorithm for retrieval and verification correspondingly.
- $\boldsymbol{a}_j \leftarrow \texttt{AnswerGen}(j, \boldsymbol{q}_j, \boldsymbol{x}, \text{aux}_A)$ is a deterministic *answer-generation* algorithm for server S_j. Its inputs are server number j, query $\boldsymbol{q}_j$, database $\boldsymbol{x}$, and auxiliary information aux_A, and the output is the answer $\boldsymbol{a}_j$.
- $\{x_i, \perp\} \leftarrow \texttt{Verify}(i, \mathsf{vk}, \boldsymbol{a}_1, \ldots, \boldsymbol{a}_k, \text{aux}_R, \text{aux}_V)$ is a deterministic *verification* algorithm for the client. Its inputs are servers' answers $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_k$, verification key vk, and auxiliary information aux_R and aux_V. The client then determines whether it has obtained the correct value of x_i. Specifically, if it determines that the reconstructed value x_i is incorrect, it outputs the special symbol $\perp$. Otherwise, it outputs x_i.

The protocol is called *publicly verifiable* if vk is public. Otherwise, the protocol is *privately verifiable*. Public verification is preferred because such protocols allow the client to outsource the verification process. But publicly verifiable protocols usually relies on computational assumptions, while privately verifiable protocols can be information-theoretically secure. We first give an informal definition of verifiability in Definition 5 and then provide the formal ones in Definition 8 and Definition 9.

Definition 5 (Verifiability)**.** The PIR protocol is t-verifiable if no t servers[1] can persuade the client to output a wrong result.

We formally define verifiability guarantees through the notion of security experiment. Here, we assume the presence of an adversary who controls a set $\mathcal{T}$ of dishonest servers (here, $\mathcal{T} \subset [k]$ and $|\mathcal{T}| \leq t$) and knows the database $\mathbf{x} \in \mathbb{F}_{p^e}^n$, retrieval index i and also, the auxiliary information aux_A. Given $\mathbf{x}$, i and aux_A, after receiving the queries $\boldsymbol{q}_j,\ j \in \mathcal{T}$, the adversary then crafts answers $\hat{\boldsymbol{a}}_j$ for $j \in \mathcal{T}$. The objective of the adversary is to convince the client that the (incorrect) retrieved file is correct. When the protocol is publicly verifiable, the adversary also knows the value of the verification key vk.

Definition 6 (Security experiment)**.** An interactive security experiment $\text{EXP}^{\text{PriV}}_{\mathcal{A},\Pi}(n, \boldsymbol{x}, i, \mathcal{T})/\text{EXP}^{\text{PubV}}_{\mathcal{A},\Pi}(n, \boldsymbol{x}, i, \mathcal{T})$ for the PIR protocol Π for database $\boldsymbol{x} \in \mathbb{F}_{p^e}^n$ between adversary $\mathcal{A}$ that controls the set $\{j_1, \ldots, j_{|\mathcal{T}|}\} = \mathcal{T} \subset [k]$ of dishonest servers and challenger[2] in privately/publicly verifiable case can be described as follows:

- The challenger generates $(\mathsf{vk}, \boldsymbol{q}_1, \ldots, \boldsymbol{q}_k, \text{aux}_A, \text{aux}_R, \text{aux}_V) \leftarrow \texttt{QueriesGen}(n, i)$ and sends $\boldsymbol{q}_j$ for $j \in \mathcal{T}$ with aux_A to the adversary $\mathcal{A}$.
- The adversary $\mathcal{A}$ generates a modified answer $\mathbf{b}_j \leftarrow \mathcal{A}(j, \boldsymbol{q}_{j_1}, \ldots, \boldsymbol{q}_{j_{|\mathcal{T}|}}, \boldsymbol{x}, i, \text{aux}_A)/\mathbf{b}_j \leftarrow \mathcal{A}(j, \boldsymbol{q}_{j_1}, \ldots, \boldsymbol{q}_{j_{|\mathcal{T}|}}, \boldsymbol{x}, i, \mathsf{vk}, \text{aux}_A)$ for $j \in \mathcal{T}$ and sends them to the challenger.
- The challenger computes $\boldsymbol{a}_j \leftarrow \texttt{AnswerGen}(j, \boldsymbol{q}_j, \mathbf{x}, \text{aux}_A)$ for $j \in [k] \setminus \mathcal{T}$.
- The challenger runs the verification algorithm $\texttt{Verify}$ with inputs i, vk, $\mathbf{b}_j$ for $j \in \mathcal{T}$ and $\boldsymbol{a}_j$ for $j \in [k] \setminus \mathcal{T}$ and, if necessary, aux_R and aux_V, and computes an output y.
- If $y \notin \{x_i, \perp\}$, set the outcome $\text{EXP}^{\text{PriV}}_{\mathcal{A},\Pi}(n, \boldsymbol{x}, i, \mathcal{T})/\text{EXP}^{\text{PubV}}_{\mathcal{A},\Pi}(n, \boldsymbol{x}, i, \mathcal{T})$ of the experiment to be 1, otherwise set it to be 0.

In what follows, we define the two notions of verifiability, namely information-theoretic and computational, and the notion of a negligible function required for the latter.

Definition 7 (Negligible function)**.** A function from $\mathbb{N}$ to $\mathbb{R}^+$ is *negligible* and denoted as negl if for all $c > 0$ there exists a natural number λ_0 such that $\mathsf{negl}(\lambda) < \frac{1}{\lambda^c}$ for all $\lambda > \lambda_0$.

Definition 8 (Information-theoretic verifiability)**.** The protocol Π is (t, ϵ)-verifiable if for any adversary $\mathcal{A}$ controlling any set of $\{j_1, \ldots, j_{|\mathcal{T}|}\} = \mathcal{T} \subset [k]$, $|\mathcal{T}| \leq t$ servers $S_{j_1}, \ldots, S_{j_{|\mathcal{T}|}}$, n, $\boldsymbol{x} \in \mathbb{F}_{p^e}^n$ and any $i \in [n]$, we have $\Pr[\text{EXP}^{\text{PriV}}_{\mathcal{A},\Pi}(n, \boldsymbol{x}, i, \mathcal{T}) = 1] \leq \epsilon$. Here, the probability is taken over the randomness of $\mathcal{A}$ and the experiment.

Definition 9 (Computational verifiability)**.** The protocol Π is t-verifiable if for any probabilistic poly-time (PPT) adversary $\mathcal{A}$ controlling any set of $\{j_1, \ldots, j_{|\mathcal{T}|}\} = \mathcal{T} \subset$

[1]We use the same letter t to define the verifiability level as for the privacy level, as in our protocols, such values are the same.

[2]For the sake of simplicity in our explanations, in our derivations, the adversary acts as a set of dishonest servers, while the client acts as a challenger who also outsources the tasks to compute their part of the answers to servers not under the control of the adversary. However, we have decided to keep the definition of security experiments in accordance with the cryptography literature.

$[k]$, $|\mathcal{T}| \leq t$ servers $S_{j_1}, \ldots, S_{j_{|\mathcal{T}|}}$, n, $\boldsymbol{x} \in \mathbb{F}_{p^e}^n$ and any $i \in [n]$, $\Pr[\mathrm{EXP}^{\mathrm{PriV}}_{\mathcal{A},\Pi}(n, \boldsymbol{x}, i, \mathcal{T})/\mathrm{EXP}^{\mathrm{PubV}}_{\mathcal{A},\Pi}(n, \boldsymbol{x}, i, \mathcal{T}) = 1] \in \mathsf{negl}(\lambda)$, where the probability is taken over the randomness of $\mathcal{A}$ and the experiment.

The notion of computational verifiability relies on certain cryptographic assumptions. Protocols proposed in this paper rely on the following assumptions: the *discrete logarithm* (DL), the *t-polynomial Diffie-Hellman Inverse* (t-DHI), and the *t-strong Diffie-Hellman* (t-SDH) assumptions. Let us formally define them.

Definition 10 (DL assumption)**.** Let $\mathbb{G}$ be a cyclic multiplicative group of order $p > 2^\lambda$ with a generator ω. Let $\alpha \in \mathbb{F}_p \setminus \{0\}$ be unknown to a PPT adversary. Under the *discrete logarithm* (DL) assumption, the probability that the PPT adversary determines α from ω and ω^α is assumed to be $\mathsf{negl}(\lambda)$.

Definition 11 (t-DHI assumption)**.** Let $\mathbb{G}$ be a cyclic group of order $p > 2^\lambda$ with a generator ω. Let $\alpha \in \mathbb{F}_p \setminus \{0\}$ be unknown to a PPT adversary. Under the *t-polynomial Diffie-Hellman inverse* (t-DHI) assumption, the probability that the PPT adversary determines $\omega^{1/\alpha}$ from the values $\omega, \omega^\alpha, \ldots, \omega^{\alpha^t}$ is assumed to be $\mathsf{negl}(\lambda)$.

Definition 12 (t-SDH assumption)**.** Let $\mathbb{G}$ be a cyclic group of order $p > 2^\lambda$ with a generator ω. Let $\alpha \in \mathbb{F}_p \setminus \{0\}$ be unknown to a PPT adversary. Under the *t-strong Diffie-Hellman* (t-SDH) assumption, the probability that the PPT adversary determines $\omega^{\frac{1}{\alpha+c}}$ for any $c \in \mathbb{F}_p \setminus \{-\alpha\}$ from the values $\omega, \omega^\alpha, \ldots, \omega^{\alpha^t}$ is assumed to be $\mathsf{negl}(\lambda)$.

III. Polynomial-Based Private Information Retrieval

In this section, we formally describe a *polynomial-based PIR protocol*, where the file retrieval process can be represented as the evaluation of a low-degree multivariate polynomial F at a specific point $\boldsymbol{z}$. On one hand, the coefficients of F are unknown to the client; hence, the client requires the inputs of the servers. On the other hand, the evaluation point $\boldsymbol{z}$ is kept secret from the servers, so up to t colluding servers cannot gain knowledge about the index of the file being retrieved. At the end of the section, we show that with specific choices of F, we can recover well-known PIR protocols like those given in Chor *et al.* [2], Goldberg [25] and Woodruff-Yekhanin [4]. More importantly, in the rest of the paper, we demonstrate that certain polynomial-based PIR protocols can be endowed with verification capabilities and the verification guarantees can be explicitly determined.

Throughout this paper, we consider a database with n files and represent it as $\boldsymbol{x} = x_1 \cdots x_n \in \mathbb{F}_{p^e}^n$. Using the extension field $\mathbb{F}_{p^e}$ allows us to treat each file as a vector of e elements in $\mathbb{F}_p$, similar to the setting in [20], [25], [30]. To define a *polynomial-based PIR*, we require the following ingredients:

(P1) m *indeterminates*, $z_1, \ldots, z_m$ over $\mathbb{F}_p$;
(P2) an injective *index-encoding function* $\boldsymbol{E} : [n] \to \mathbb{F}_p^m$;
(P3) a *database multivariate polynomial* $F_{\boldsymbol{x}} \in \mathbb{F}_{p^e}[z_1, \ldots, z_m]$ such that $F_{\boldsymbol{x}}(\boldsymbol{E}(i)) = x_i$ for all i.

Given m, $\boldsymbol{E}$ and $F_{\boldsymbol{x}}$ that satisfy conditions (P1), (P2), (P3), we can then define the corresponding queries and answer generation procedures. Following the protocol introduced by Woodruff-Yekhanin, we also use the notion of derivatives of a polynomial over a finite field and their extension to *partial derivatives*. Recall that for $f(u) = \sum_{j=0}^{D} f_j u_j \in \mathbb{F}_{p^e}[u]$, the derivative is defined by $f'(u) = \sum_{j=1}^{D} j f_j u^{j-1}$. Higher-order derivatives and partial derivatives for multivariate polynomials over a finite field are defined in a similar way.

Let $F(\boldsymbol{z})$ be a multivariate polynomial with indeterminates $\boldsymbol{z} = (z_1, \ldots, z_m)$. For $\boldsymbol{r} = (r_1, \ldots, r_m) \in \mathbb{Z}_{\geq 0}^m$, we use $F^{(\boldsymbol{r})}(\boldsymbol{z})$ to represent the partial derivative $\frac{\partial^{\sum r_i}}{(\partial z_1)^{r_1} \cdots (\partial z_m)^{r_m}} F(\boldsymbol{z})$. Next, we set $o \geq 0$ and consider all partial derivatives up to order o. Specifically, we let $\mathcal{R}(o, m)$ to denote set of tuples $\mathcal{R}(o, m) \triangleq \{\boldsymbol{r} \in \mathbb{Z}_{\geq 0}^m : \sum_{i=1}^m r_i \leq o\}$. Clearly, the number of derivatives of order j is exactly the number of distributions of j balls over m bins, as the fact that ℓ balls belong to the i-th bin can be treated as having an ℓ-th order partial differential term for the variable i. Using the stars and bars approach, we can easily find that this value is equal to $\binom{j+m-1}{m-1}$. Utilizing [31, Equation 2.5.1], we can find that $|\mathcal{R}(o, m)| = \sum_{j=0}^{o} \binom{j+m-1}{m-1} = \binom{o+m}{o}$, and in our protocol, we require a server to compute all $\binom{o+m}{o}$ partial derivatives $F^{(\boldsymbol{r})}(\boldsymbol{z})$ for $\boldsymbol{r} \in \mathcal{R}(o, m)$ evaluated at a predetermined point.

Next, we consider m polynomials $c_1(u), \ldots, c_m(u) \in \mathbb{F}_p[u]$ and we write $\boldsymbol{c}(u) \triangleq (c_i(u))_{i=1}^m$ as a vector of polynomials. Substituting z_i by $c_i(u)$ for all $i \in [m]$, we obtain a univariate polynomial $f(u) \triangleq F(\boldsymbol{c}(u)) \in \mathbb{F}_{p^e}[u]$. A key observation in Woodruff-Yekhanin [4] is that we can employ the usual chain rule and product rule to determine derivatives of f using partial derivatives of F. We get a generalization of these results, formulated in Lemma 1, by sequentially applying chain and product rules.

Lemma 1 (Chain and Product rule)**.** Let $F(\boldsymbol{z})$ be a multivariate polynomial with indeterminates $\boldsymbol{z} = (z_1, \ldots, z_m)$ and $\boldsymbol{c}(u) \triangleq (c_i(u))_{i=1}^m$ be an m-tuple of polynomials in u. If $f(u) = F(\boldsymbol{c}(u))$, then for $o \geq 1$, we have that

$$f^{(o)}(u) = \sum_{\boldsymbol{r} \in \mathcal{R}(o,m) \setminus \{\boldsymbol{0}\}} F^{(\boldsymbol{r})}(\boldsymbol{c}(u)) \Gamma(\boldsymbol{c}(u), \boldsymbol{r}) \,. \tag{1}$$

Here, for brevity purposes, we use $\Gamma(\boldsymbol{c}(u), \boldsymbol{r})$ to denote a polynomial function that takes the derivatives of $c_i(u)$'s as inputs. Specifically, the set of inputs for $\Gamma(\boldsymbol{c}(u), \boldsymbol{r})$ is $\{c_i^{(\ell)}(u) : i \in [m], 0 \leq \ell \leq r_i\}$. Note that these inputs are independent of the polynomial $F(\boldsymbol{z})$ and the total degree from $c_i^{(\ell)}(u)$ of function Γ in equation (1) is upper bounded by o.

Example 1. To better clarify equation (1), let us consider the following example. Let $f(u) = F(\mathbf{c}_1(u), \mathbf{c}_2(u))$. Applying the chain rule, we get:

$$f^{(1)}(u) = F^{(1,0)}(\mathbf{c}(u)) \cdot \frac{d\mathbf{c}_1}{du} + F^{(0,1)}(\mathbf{c}(u)) \cdot \frac{d\mathbf{c}_2}{du}.$$

To obtain $f^{(2)}(u)$, we need to employ the chain and product rules as per Lemma 1. As a result, we can find that

$$\frac{d}{du}\left(F^{(1,0)}(\mathbf{c}(u)) \cdot \frac{d\mathbf{c}_1}{du}\right) = F^{(1,0)}(\mathbf{c}(u)) \cdot \frac{d\mathbf{c}_1}{du^2} +$$

$$F^{(2,0)}(\mathbf{c}(u)) \cdot \left(\frac{d\mathbf{c}_1}{du^2}\right)^2 + F^{(1,1)}(\mathbf{c}(u)) \cdot \frac{d\mathbf{c}_1}{du}\frac{d\mathbf{c}_2}{du}$$

and

$$\frac{d}{du}\left(F^{(0,1)}(\mathbf{c}(u)) \cdot \frac{d\mathbf{c}_1}{du}\right) = F^{(0,1)}(\mathbf{c}(u)) \cdot \frac{d\mathbf{c}_2}{du^2} + F^{(0,2)}(\mathbf{c}(u)) \cdot \left(\frac{d\mathbf{c}_2}{du^2}\right)^2 + F^{(1,1)}(\mathbf{c}(u)) \cdot \frac{d\mathbf{c}_1}{du}\frac{d\mathbf{c}_2}{du}.$$

By summing these terms together, we can find that

$$f^{(2)}(u) = F^{(1,0)}(\mathbf{c}(u)) \cdot \frac{d\mathbf{c}_1}{du^2} + F^{(0,1)}(\mathbf{c}(u)) \cdot \frac{d\mathbf{c}_2}{du^2} + F^{(2,0)}(\mathbf{c}(u)) \cdot \left(\frac{d\mathbf{c}_1}{du^2}\right)^2 + F^{(0,2)}(\mathbf{c}(u)) \cdot \left(\frac{d\mathbf{c}_2}{du^2}\right)^2 + F^{(1,1)}(\mathbf{c}(u)) \cdot \left(\frac{d\mathbf{c}_1}{du}\frac{d\mathbf{c}_2}{du} + \frac{d\mathbf{c}_2}{du}\frac{d\mathbf{c}_1}{du}\right).$$

The next lemma generalizes a result from Woodruff-Yekhanin [4, Lemma 2], which states that if we have sufficient evaluations of f and its partial derivatives, we will be able to uniquely determine f.

Lemma 2 (Interpolation of Derivatives)**.** Let $u_1, \ldots, u_k$ be k distinct points in $\mathbb{F}_p$. Let $v_j^{(i)}$ $(i \in \{0, 1, \ldots, o\}, j \in [k])$ be a set of (not necessarily distinct) values in $\mathbb{F}_{p^e}$. If $D + 1 \leq (o+1)k$, then there exists at most one polynomial f of degree at most D such that

$$f^{(i)}(u_j) = v_j^{(i)} \text{ for all } i \in \{0, 1, \ldots, o\},\ j \in [k]\,.$$

Furthermore, if the polynomial f exists, then

$$f(0) = \sum_{\substack{0 \leq i \leq o \\ j \in [k]}} \alpha_{i,j} v_j^{(i)} \tag{2}$$

where $\alpha_{i,j}$ are scalars in $\mathbb{F}_p$ dependent only on $u_1, \ldots, u_k$.

Proof. Suppose that there are two polynomials f and g with degree at most D such that $f^{(i)}(u_j) = g^{(i)}(u_j)$ for all $i \in \{0, 1, \ldots, o\}$ and $j \in [k]$. We consider $h = f - g$ and fix $j \in [k]$. Since $h^{(i)}(u_j) = 0$ for all $i \in \{0, 1, \ldots, o\}$, we have that $(u-u_j)^{o+1}$ divides $h(u)$. Therefore, we have $\prod_{j=1}^{k}(u-u_j)^{o+1}$ divides $h(u)$. Since $\deg h \leq D < (o+1)k = \deg(\prod_{j=1}^{k}(u - u_j)^{o+1})$, we have that $h \equiv 0$ and so, f and g are identical. As a result, the system to determine f from its derivatives has at most one solution.

Suppose the polynomial $f(u) = \sum_{j=0}^{D} f_j u^j$ exists. For $\ell \geq 0$, it is straightforward to verify that $f^{(\ell)}(u) = \sum_{j=\ell}^{D} \frac{j!}{(j-\ell)!} f_j u^{j-\ell}$. Thus, if we set $\boldsymbol{f} = (f_j)_{j=0}^{D}$, $\boldsymbol{v} = \left(v_j^{(i)}\right)_{j\in[k], i\in\{0,1,\ldots,o\}}$ and

$$\boldsymbol{M} = \begin{bmatrix} 1 & u_1 & u_1^2 & \cdots & u_1^o & \cdots & u_1^D \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 1 & u_k & u_k^2 & \cdots & u_k^o & \cdots & u_k^D \\ 0 & 1 & 2u_1 & \cdots & ou_1^{o-1} & \cdots & Du_1^{D-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 1 & 2u_k & \cdots & ou_1^{o-1} & \cdots & Du_k^{D-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & o! & \cdots & \frac{D!}{(D-o)!}u_1^{D-o} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & o! & \cdots & \frac{D!}{(D-o)!}u_k^{D-o} \end{bmatrix},$$

we can find $\boldsymbol{f}$ by solving the equations $\boldsymbol{M}\boldsymbol{f} = \boldsymbol{v}$ that has a unique solution, as proven previously. Furthermore, since $f(0) = f_0$ is the first entry of $\boldsymbol{f}$, the scalars $\alpha_{i,j}$ are given by the first row of inverse of $\boldsymbol{M}$. □

Finally, we are able to formally describe a polynomial-based PIR protocol. As before, we consider $t \leq k-1$ where k is the number of servers and t is the maximum number of colluding servers. Now, we also require $k < p$, choose k different nonzero elements $u_1, \ldots, u_k$ from $\mathbb{F}_p$ and make them publicly available. For instance, we can associate server S_j with $u_j = j$, as it was done in [10]. The main idea of the protocol is to represent the database as a low-degree multivariate polynomial, with each file corresponding to a value of the polynomial at a specific point. Our goal is to recover one of these values by sending a specific point to each server and requesting the value of the polynomial at that point, along with partial derivatives up to a certain order. Formally, the protocol is described below.

Polynomial-based k-server t-private PIR protocol Π_0

Queries generation for file x_i

- Client randomly generates $\boldsymbol{V}^{(j)} \in \mathbb{F}_p^m$ for $j \in [t]$ and these vectors kept secret from the servers.
- Set $\boldsymbol{c}(u) = \boldsymbol{E}(i) + \boldsymbol{V}^{(1)}u + \cdots + \boldsymbol{V}^{(t)}u^t$. Observe that $\boldsymbol{c}(u)$ is a curve of degree-t that belongs to $\mathbb{F}_p^m$. Critically, the curve $\boldsymbol{c}(u)$ passes through the point $(0, \boldsymbol{E}(i))$.
- For $j \in [k]$, send to server S_j the query $\boldsymbol{q}_j \triangleq \boldsymbol{c}(u_j)$.
- In this case, $\text{aux}_A = \emptyset$ and $\text{aux}_R = \{\boldsymbol{V}^{(1)}, \ldots, \boldsymbol{V}^{(t)}\}$.

Answer generation for server S_j with $j \in [k]$

- Server S_j computes $\boldsymbol{a}_j^{(\boldsymbol{r})} = F_{\boldsymbol{x}}^{(\boldsymbol{r})}(\boldsymbol{q}_j)$ for all $\boldsymbol{r} \in \mathcal{R}(o, m)$.
- Server j sends all $\binom{o+m}{o}$ evaluations to the client.

Retrieval algorithm

- Consider the polynomial $f(u)$ obtained from polynomial $F_{\boldsymbol{x}}(\boldsymbol{z})$ after parameterization by curve $\mathbf{c}(u)$. In other words, set $f(u) = F_{\boldsymbol{x}}(\boldsymbol{c}(u))$.
- Using (1), compute $f^{(\ell)}(u_j)$ for all $\ell \in \{0, 1, \ldots, o\}$ and $j \in [k]$. Note that the partial derivatives are provided by the servers, while the derivatives of $\mathbf{c}_i(u_j)$ can be determined using aux_R.
- Using Lemma 2, we recover the polynomial $f(u)$.
- Finally, x_i is given by the value $f(0)$.

We formally prove the correctness and state certain metrics of the protocol in the next theorem.

Theorem 1. Let d be the total degree of $F_{\boldsymbol{x}}$. If $dt+1 \leq (o+1)k$, then the protocol Π_0 is a k-server t-private PIR protocol with upload cost of m symbols in $\mathbb{F}_p$ and download cost of at most $\binom{o+m}{o}$ symbols in $\mathbb{F}_{p^e}$ per server.

Proof. For $\ell \in [m]$, the ℓ-th components of the queries $\boldsymbol{q}_j$'s form a $(t, t+1)$-Shamir's secret sharing scheme with the ℓ-th component of $\boldsymbol{E}(i)$ as the secret. This means that any subset of t or less shares does not reveal any information on this component. In other words, no information of $\boldsymbol{E}(i)$ is leaked from any t queries, and this demonstrates the t-privacy. The correctness follows from Lemma 2, while the upload and download cost values follow from the form of answers and queries. □

In what follows, we demonstrate that many well-known PIR protocols are in fact polynomial-based. To do so, we simply state the number of indeterminates m, the encoding function $\boldsymbol{E}$ and the multivariate polynomial $F_{\boldsymbol{x}}$ that satisfy (P1), (P2) and (P3).

(1) *Woodruff-Yekhanin [4] and Goldberg [25].* Let m and d be such that $\binom{m}{d} \geq n$. We choose E to be an encoding function that maps from $[n]$ to the set of $\binom{m}{d}$ binary vectors with exactly d ones. Here, we let $\boldsymbol{E}(i)_j$ denote the j-th coordinate of $\boldsymbol{E}(i)$. Then we set $F_{\boldsymbol{x}}$ to be the multivariate polynomial $\sum_{i\in[n]} x_i \prod_{j=1}^{m} z_j^{\boldsymbol{E}(i)_j}$. Here, since $\boldsymbol{E}(i)$ has weight d, the total degree of $F_{\boldsymbol{x}}$ is d. So, Theorem 1 requires that $dt+1 \leq (o+1)k$. Hence, for fixed k and t, by setting $d = \lfloor ((o+1)k-1)/t \rfloor$ and choosing m and d be such that $\binom{m}{d}$ to close to n, we have $m \sim (d!n)^{1/d}$.
   - When $d=1$, $m=n$, $o=0$, we recover the PIR protocol in Goldberg [25]. In this case, $\boldsymbol{E}$ maps each index $i \in [n]$ to the unit vector $\boldsymbol{e}_i$ with a 1 at the i-th coordinate and 0 elsewhere. Moreover, $F_{\boldsymbol{x}}(z_1, \ldots, z_n) = \sum_{i=1}^{n} x_i z_i$. Here, for k servers, the total upload and download costs are kn symbols in $\mathbb{F}_p$ and k symbols in $\mathbb{F}_{p^e}$, respectively. In this case, we also have $t = k-1$, and the download rate (measured by ratio of the retrieved file size to the total download costs) is given by $1/k$. This corresponds to the maximum asymptotic capacity $1-t/k$ (when $n \to \infty$) [9].[3]
   - For general d and $o=1$, we recover the PIR protocol in Woodruff-Yekhanin [4]. For each server, upload and download costs are m and $m+1$ symbols, respectively, and the total communication is $\sim 2m \sim 2(d!n)^{1/d}$ symbols. Note that Woodruff-Yekhanin protocol generalizes the interpolation-based PIR protocol in Chor *et al.* [2, Sec. 4.2] from $t=1$ and $d=k-1$ to general t and d.
   - For general d and $o>1$, it appears that our generalization is new. Here, upload and download costs are m and $\binom{o+m}{o}$ symbols, respectively, and the total communication is $\sim \binom{o+m}{o} \sim \frac{1}{o!}(d!n)^{o/d}$ symbols. In the special instance where $k=3$, $t=2$ and $o=2$, we have $d=4$. Then Π_0 incurs a total communication cost of $\sim \frac{1}{2!}(4!n)^{2/4} = \sqrt{6n}$ symbols. In comparison, Woodruff-Yekhanin's protocol with $o=1$ requires $d=2$ and incurs a total communication cost of $\sim 2(2!n)^{1/2} = \sqrt{8n}$ symbols. So, by setting $o=2$, we have 13.4% savings in communication. While we found other (k,t) pairs with the same savings, we cannot ascertain the existence of pairs with bigger savings.

(2) Chor *et al.* [2, Section 4.1][4] With $o=0$, $t=1$, and an $\boldsymbol{E}$ defined slightly different from the above, our scheme produces the PIR protocol in Chor *et al.* [2, Section 4.1]. More specifically, let m be such that $2^m \geq n$, and $\boldsymbol{E}$ be an encoding function that maps from $[n]$ to the set of length-m binary vectors, e.g., $\boldsymbol{E}(i)$ is the binary representation of i using m bits. Let $F_{\boldsymbol{x}}(z_1, \ldots, z_m) = \sum_{i=1}^{n} x_i \prod_{j=1}^{m} \left(1 - \boldsymbol{E}(i)_j + (-1)^{1-\boldsymbol{E}(i)_j} z_j\right)$. Equivalently, $F_{\boldsymbol{x}} = \sum_{i\in[n]} x_i \prod_{j=1}^{m} \phi(\boldsymbol{E}(i)_j, j)$, where the map ϕ satisfies $\phi(0,j) = (1-z_j)$ and $\phi(1,j) = z_j$. Here, all monomials have degree m and so, the total degree of $F_{\boldsymbol{x}}$ is m. The upload and download costs are $\log n$ symbols and one symbol, respectively. We note that in this regime, we require $k \geq m+1 = \Omega(\log n)$.

Example 2. We provide an instructive example to illustrate the polynomial-PIR framework. Consider Woodruff-Yekhanin's PIR scheme with $n=4$, $k=2$, and $t=1$. As described earlier, we can choose $o=1$ with $m=4$ and $d=3$. We can set $\boldsymbol{E}(1)=(1,1,1,0)$, $\boldsymbol{E}(2)=(1,1,0,1)$, $\boldsymbol{E}(3)=(1,0,1,1)$, $\boldsymbol{E}(4)=(0,1,1,1)$, and hence,

$$\begin{aligned} F_{\boldsymbol{x}}(\boldsymbol{z}) &= \sum_{i\in[4]} x_i \prod_{j=1}^{4} z_j^{\boldsymbol{E}(i)_j} \\ &= x_1 z_1 z_2 z_3 + x_2 z_1 z_2 z_4 + x_3 z_1 z_3 z_4 + x_4 z_2 z_3 z_4. \end{aligned}$$

Suppose that the database $\boldsymbol{x} = (1,0,0,0)$ and the client is interested in obtaining x_1. Then we have $F_{\boldsymbol{x}}(\boldsymbol{z}) = z_1 z_2 z_3$.

Let $p=11$ with $e=1$. We choose $u_1=1$ and $u_2=2$. As per Π_0, we randomly pick $\boldsymbol{V}^{(1)} = (1,2,3,4)^T$. Then

$$\boldsymbol{c}(u) = \boldsymbol{E}(1) + \boldsymbol{V}^{(1)} u = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix} \cdot u = \begin{bmatrix} 1+u \\ 1+2u \\ 1+3u \\ 4u \end{bmatrix},$$

and so,

$$\boldsymbol{c}(1) = \begin{bmatrix} 2 \\ 3 \\ 4 \\ 4 \end{bmatrix}, \ \boldsymbol{c}(2) = \begin{bmatrix} 3 \\ 5 \\ 7 \\ 8 \end{bmatrix}.$$

Then we have the following responses:

[3] We do note that the PIR protocol in Goldberg [25] is a non-universal version of the Bitar-El Rouayheb protocol [30]. By 'universality,' we mean the ability to perform recovery from any number of available servers within a certain range.

[4] This scheme was omitted from the following journal extension [32]; hence, in what follows, we refer to the initial FOCS paper [2].

Server i	$F_{\boldsymbol{x}}(\boldsymbol{c}(u_i))$	$\frac{\partial}{\partial z_1}F_{\boldsymbol{x}}$	$\frac{\partial}{\partial z_2}F_{\boldsymbol{x}}$	$\frac{\partial}{\partial z_3}F_{\boldsymbol{x}}$	$\frac{\partial}{\partial z_4}F_{\boldsymbol{x}}$
Server 1	2	1	8	6	0
Server 2	6	2	10	4	0

Since $f(u) = F_{\boldsymbol{x}}(\boldsymbol{c}(u))$, we have

$$f'(u) = \sum_{i=1}^{4} \frac{\partial}{\partial z_i} F_{\boldsymbol{x}}(\boldsymbol{c}(u)) \frac{\partial}{\partial u} c_i(u) = \sum_{i=1}^{4} \frac{\partial}{\partial z_i} F_{\boldsymbol{x}}(\boldsymbol{c}(u)) \boldsymbol{V}_i^{(1)}$$
$$= 1 \cdot \frac{\partial}{\partial z_1} F_{\boldsymbol{x}} + 2 \cdot \frac{\partial}{\partial z_2} F_{\boldsymbol{x}} + 3 \cdot \frac{\partial}{\partial z_3} F_{\boldsymbol{x}} + 4 \cdot \frac{\partial}{\partial z_4} F_{\boldsymbol{x}},$$

(ignoring $\boldsymbol{c}(u)$ to simplify the notation), and therefore,

$$f'(u_1) = 1 \cdot 1 + 2 \cdot 8 + 3 \cdot 6 + 4 \cdot 0 = 2,$$
$$f'(u_2) = 1 \cdot 2 + 2 \cdot 10 + 3 \cdot 4 + 4 \cdot 0 = 1.$$

Next, following the proof of Lemma 2, we know that $\boldsymbol{f}$, the vector of coefficients of f, satisfies $\boldsymbol{M}\boldsymbol{f} = \boldsymbol{v}$ with

$$\boldsymbol{M} = \begin{bmatrix} 1 & u_1 & u_1^2 & u_1^3 \\ 1 & u_2 & u_2^2 & u_2^3 \\ 0 & 1 & 2u_1 & 3u_1^2 \\ 0 & 1 & 2u_2 & 3u_2^2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 8 \\ 0 & 1 & 2 & 3 \\ 0 & 1 & 4 & 1 \end{bmatrix}, \boldsymbol{v} = \begin{bmatrix} f(u_1) \\ f(u_2) \\ f'(u_1) \\ f'(u_2) \end{bmatrix} = \begin{bmatrix} 2 \\ 6 \\ 2 \\ 1 \end{bmatrix}.$$

Therefore, the first entry of $\boldsymbol{f}$, which is precisely $f(0)$, can be calculated as $f(0) = (7, 5, 7, 9)\boldsymbol{v} = 1$, where $(7, 5, 7, 9)$ is the first row of $\boldsymbol{M}^{-1}$. The client then obtains $x_i = f(0) = 1$ as desired. In the next section, we modify this protocol so that the client is able to verify the correctness of the retrieved x_1.

IV. Verifiable polynomial PIR

In this section, we demonstrate how to modify a polynomial PIR with certain extra properties to make it information-theoretically privately verifiable. The fundamental idea is to provide an additional independent set of queries to the servers, enabling the user to compute the file of interest, denoted as x_i, along with a function involving x_i and a secret parameter v to verify that servers are truthful. By keeping the verification key secret and downloading the whole responses to the queries, we achieve an information-theoretically privately verifiable protocol. Later on, we extend this protocol to a publicly verifiable setup and reduce the communication cost by introducing linearly homomorphic commitment schemes. It should be noted that initially, such a framework was employed in a two-server PIR protocol with an optimal download rate [1].

A. Information-Theoretically Privately Verifiable Protocol

Let us modify the polynomial PIR protocol Π_0 to include verification capabilities. We do so by creating one more set of independent queries. As before, we have m, E and $F_{\boldsymbol{x}}$ satisfying conditions (P1), (P2), (P3) and we choose k distinct nonzero points $u_1, \ldots, u_k \in \mathbb{F}_p$. To add verifiability, for each $i \in [n]$, there must exist a publicly known curve $\boldsymbol{P}^{(i)}(\mathsf{v}) = (P_\ell^{(i)}(\mathsf{v}))_{\ell=1}^m \in (\mathbb{F}_p[\mathsf{v}])^m$ such that the following holds. It should be noted that these curves can be the same for all i, but to ensure generality, we assume that they are different.

(V1) For all $i \in [n]$ and $\ell \in [m]$, $P_\ell^{(i)}(\mathsf{v})$ is either a monomial v or a constant. In other words, $\deg P_\ell^{(i)}(\mathsf{v})$ is at most one.

(V2) For all $i \in [n]$,

$$F_{\boldsymbol{x}}(\boldsymbol{P}^{(i)}(\mathsf{v})) = R^{(i)}(\mathsf{v}) x_i \text{ for a polynomial } R^{(i)}(\mathsf{v}). \quad (3)$$

Note that if $R^{(i)}(\mathsf{v})$'s exist, they are uniquely determined by the choice of $\boldsymbol{P}^{(i)}$'s.

As a result, we can recover the values of $F_{\boldsymbol{x}}(\boldsymbol{E}(i))$ and $F_{\boldsymbol{x}}(\boldsymbol{P}^{(i)}(\mathsf{v}))$ in the same way as in Protocol Π_0, and check whether $\mathsf{v} \cdot F_{\boldsymbol{x}}(\boldsymbol{E}(i)) = F_{\boldsymbol{x}}(\boldsymbol{P}^{(i)}(\mathsf{v}))$. The resulting protocol is formally described below.

Information-theoretically verifiable PIR protocol Π_1

Queries generation to retrieve x_i with verification

- The client splits each query into two parts – one for file retrieval and one for verification.
- For file retrieval, we follow protocol Π_0. That is, the client randomly generates $\boldsymbol{V}^{(j)}$ for $j \in [t]$ and sets $\boldsymbol{c}(u) = \boldsymbol{E}(i) + \sum_{j=1}^t \boldsymbol{V}^{(j)} u^j$. Then the client computes $\boldsymbol{c}(u_j)$ for $j \in [k]$.
- For verification, the client randomly generates $\mathsf{v} \in \mathbb{F}_p \setminus \{0\}$ and randomly generates $\boldsymbol{U}^{(j)} \in \mathbb{F}_p^m$ for $j \in [t]$. Similar to before, the client sets $\boldsymbol{c}_{\mathsf{v}}(u) = \boldsymbol{P}^{(i)}(\mathsf{v}) + \sum_{j=1}^t \boldsymbol{U}^{(j)} u^j$ and computes $\boldsymbol{c}_{\mathsf{v}}(u_j)$ for $j \in [k]$.
- For $j \in [k]$, the client sends to server S_j the query $\boldsymbol{q}_j \triangleq (\boldsymbol{c}(u_j), \boldsymbol{c}_{\mathsf{v}}(u_j))$.
- In this case, $\mathsf{vk} = \mathsf{v}$, $\mathrm{aux}_A = \emptyset$, $\mathrm{aux}_R = \{\boldsymbol{V}^{(j)} : j \in [t]\}$, and $\mathrm{aux}_V = \{\boldsymbol{U}^{(j)} : j \in [t]\}$.

Answer generation for server S_j with $j \in [k]$

- Server S_j computes $\boldsymbol{a}_j^{(\boldsymbol{r})} = \left(F_{\boldsymbol{x}}^{(\boldsymbol{r})}(\boldsymbol{c}(u_j)), F_{\boldsymbol{x}}^{(\boldsymbol{r})}(\boldsymbol{c}_{\mathsf{v}}(u_j))\right)$ for all $\boldsymbol{r} \in \mathcal{R}(o, m)$.
- S_j sends all $\binom{o+m}{o}$ evaluations to the client. We note that each evaluation comprises two symbols in $\mathbb{F}_{p^e}$.

Retrieval with results verification

- Consider the polynomials $f(u)$ and $f_{\mathsf{v}}(u)$ obtained from $F_{\boldsymbol{x}}$ after parameterization by curve $\boldsymbol{c}(u)$ and $\boldsymbol{c}_{\mathsf{v}}(u)$, respectively. In other words, set $f(u) = F_{\boldsymbol{x}}(\boldsymbol{c}(u))$ and $f_{\mathsf{v}}(u) = F_{\boldsymbol{x}}(\boldsymbol{c}_{\mathsf{v}}(u))$.
- As with Π_0, we use the chain rule (1) to compute $f^{(\ell)}(u_j)$ for all $\ell \in \{0, 1, \ldots, o\}$ and $j \in [k]$. Then using Lemma 2, we recover the polynomial $f(u)$.
- Following similar steps, we recover the polynomial $f_{\mathsf{v}}(u)$ using the values $F_{\boldsymbol{x}}^{(\boldsymbol{r})}(\boldsymbol{c}_{\mathsf{v}}(u_j))$.
- Finally, x_i is given by the value $f(0)$.
- To verify the correctness of x_i, the client checks that $R^{(i)}(\mathsf{v}) f(0) = f_{\mathsf{v}}(0)$. If the equation holds, the algorithm outputs x_i. Otherwise, the output is $\perp$.

Remark 1. The exact form of the curves $\mathbf{P}^{(i)}(\mathbf{v})$ depends on the actual database multiplicative polynomial $F_{\boldsymbol{x}}$. For instance, in the Woodruff-Yekhanin protocol [4] and the Goldberg protocol [25], we can construct such $\boldsymbol{P}^{(i)}$ by replacing Δ arbitrary 1-entries of $\boldsymbol{E}(i)$ by v, where $d \geq \Delta \geq o + 1$. As $\boldsymbol{E}(i)$ contains exactly d 1-entries for every $i \in [n]$ and $F_{\boldsymbol{x}} = \sum_{i \in [n]} x_i \prod_{j=1}^m z_j^{\boldsymbol{E}(i)_j}$ in the protocols of Woodruff-Yekhanin and Goldberg, it is easy to see that $f_{\mathsf{v}}(0) = F_{\boldsymbol{x}}(\boldsymbol{P}^{(i)}(\mathsf{v})) = \mathsf{v}^\Delta F_{\boldsymbol{x}}(\boldsymbol{E}(i)) = \mathsf{v}^\Delta f(0)$. The requirement that $\Delta \geq o + 1$ is needed to guarantee the verifiablity of the

protocol (see Theorem 2). However, we do not know how to construct such curves (if exist) for the Chor *et al.* protocol [2, Section 4.1], and thus leave it as an open problem.

Example 3 (Example 2 continued)**.** We consider the setup as before. To equip the scheme with verification capabilities, we construct $\boldsymbol{P}^{(i)}$'s by replacing two "1" entries in $\boldsymbol{E}(i)$ by v's:

$$\boldsymbol{P}^{(1)}(\mathsf{v}) = \begin{bmatrix} \mathsf{v} \\ \mathsf{v} \\ 1 \\ 0 \end{bmatrix}, \ \boldsymbol{P}^{(2)}(\mathsf{v}) = \begin{bmatrix} \mathsf{v} \\ \mathsf{v} \\ 0 \\ 1 \end{bmatrix}, \ \boldsymbol{P}^{(3)}(\mathsf{v}) = \begin{bmatrix} \mathsf{v} \\ 0 \\ \mathsf{v} \\ 1 \end{bmatrix}, \ \boldsymbol{P}^{(4)}(\mathsf{v}) = \begin{bmatrix} 0 \\ \mathsf{v} \\ \mathsf{v} \\ 1 \end{bmatrix}.$$

It is straightforward to verify that the $\boldsymbol{P}^{(i)}$'s satisfy (V1) and (V2) with $R^{(i)}(\mathsf{v}) = \mathsf{v}^2$ for all $i \in [4]$.

To perform verification, we randomly pick $\boldsymbol{U}^{(1)} = (1,1,1,1)^T$ and $\mathsf{v} = 3$. Since

$$\boldsymbol{c}_{\mathsf{v}}(u) = \boldsymbol{P}^{(1)}(\mathsf{v}) + \boldsymbol{U}^{(1)}u = \begin{bmatrix} \mathsf{v} \\ \mathsf{v} \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \cdot u = \begin{bmatrix} \mathsf{v}+u \\ \mathsf{v}+u \\ 1+u \\ u \end{bmatrix},$$

we have

$$\boldsymbol{c}_{\mathsf{v}}(1) = \begin{bmatrix} 4 \\ 4 \\ 2 \\ 1 \end{bmatrix}, \ \boldsymbol{c}_{\mathsf{v}}(2) = \begin{bmatrix} 5 \\ 5 \\ 3 \\ 2 \end{bmatrix}.$$

Then we have the following responses:

Server i	$F_x(\boldsymbol{c}_{\mathsf{v}}(u_i))$	$\frac{\partial}{\partial z_1}F_x$	$\frac{\partial}{\partial z_2}F_x$	$\frac{\partial}{\partial z_3}F_x$	$\frac{\partial}{\partial z_4}F_x$
Server 1	10	8	8	5	0
Server 2	9	4	4	3	0

Since $f_{\mathsf{v}}(u) = F_{\boldsymbol{x}}(\boldsymbol{c}_{\mathsf{v}}(u))$, denoting $\boldsymbol{c}_{\mathsf{v}}(u) = \big(c_{\mathsf{v},1}(u), \ldots, c_{\mathsf{v},m}(u)\big)$, we have

$$\begin{aligned} f'_{\mathsf{v}}(u) =& \sum_{i=1}^{4} \frac{\partial}{\partial z_i} F_{\boldsymbol{x}}(\boldsymbol{c}_{\mathsf{v}}(u)) \frac{\partial}{\partial u} c_{\mathsf{v},i}(u) = \sum_{i=1}^{4} \frac{\partial}{\partial z_i} F_{\boldsymbol{x}}(\boldsymbol{c}_{\mathsf{v}}(u)) \boldsymbol{U}_i^{(1)} \\ =& 1 \cdot \frac{\partial}{\partial z_1} F_{\boldsymbol{x}} + 1 \cdot \frac{\partial}{\partial z_2} F_{\boldsymbol{x}} + 1 \cdot \frac{\partial}{\partial z_3} F_{\boldsymbol{x}} + 1 \cdot \frac{\partial}{\partial z_4} F_{\boldsymbol{x}}, \end{aligned}$$

(ignoring $\boldsymbol{c}_{\mathsf{v}}(u)$ to simplify the notation), and therefore,

$$\begin{aligned} f'_{\mathsf{v}}(u_1) &= 1 \cdot 8 + 1 \cdot 8 + 1 \cdot 5 + 1 \cdot 0 = 10, \\ f'_{\mathsf{v}}(u_2) &= 1 \cdot 4 + 1 \cdot 4 + 1 \cdot 3 + 1 \cdot 0 = 0. \end{aligned}$$

Similar to Example 2, following the notation in the proof of Lemma 2, we obtain

$$\begin{aligned} f_{\mathsf{v}}(0) &= \boldsymbol{M}_1^{-1} \big(f_{\mathsf{v}}(u_1), f_{\mathsf{v}}(u_2), f'_{\mathsf{v}}(u_1), f'_{\mathsf{v}}(u_2)\big)^{\mathrm{T}} \\ &= (7,5,7,9)(10,9,10,0)^{\mathrm{T}} = 9\,. \end{aligned}$$

To verify that the retrieved file $x_1 = 1$ is correct, we check that $R^{(1)}(\mathsf{v}) f(0) = \mathsf{v}^2 f(0)$ is indeed $f_{\mathsf{v}}(0)$, which is true in this case because $3^2 \cdot 1 = 9$.

Now, suppose that server S_1 is compromised by an adversary. Here, we assume that the adversary knows that the client require x_1, and wants the client to believe that x_1 is some value other than 1. In the next theorem, we bound the probability of the adversary succeeding from above. Specifically, since $\deg R^{(i)}(\mathsf{v}) = 2$ is always greater than one, this probability is shown to be at most $\frac{\delta}{p-1} = \frac{2}{10}$ where $\delta = \max_{i\in[n]} R^{(i)}(\mathsf{v})$.

Theorem 2. Let d be the total degree of $F_{\boldsymbol{x}}$. Furthermore, suppose that we have $\boldsymbol{P}^{(i)}(\mathsf{v})$'s satisfy conditions (V1) and (V2). Set $\delta = \max_{i\in[n]} \deg R^{(i)}(\mathsf{v})$. If $dt + 1 \leq (o+1)k$ and $\deg R^{(i)}(\mathsf{v}) > o$ for all $i \in [n]$, then the protocol Π_1 is a k-server t-private $\left(t, \frac{\delta}{p-1}\right)$-verifiable PIR protocol with upload cost of $2m$ symbols in $\mathbb{F}_p$ and download cost of at most $2\binom{o+m}{o}$ symbols in $\mathbb{F}_{p^e}$ per server.

Proof. The correctness and privacy of the PIR protocol follow from Theorem 1, while the upload and download cost values follow from the form of answers and queries. Hence, it remains to obtain the information-theoretic verifiability guarantees.

Without loss of generality, suppose that the client wants the file x_1. Also, we let the adversary control servers $S_1, \ldots, S_t$. Then following Definitions 6 and 8, we assume that the adversary knows the retrieved index, the database $\boldsymbol{x}$, and the queries $\boldsymbol{q}_j = (\boldsymbol{c}(u_j), \boldsymbol{c}_{\mathsf{v}}(u_j))$ for $j \in [t]$. To win the security experiment, the adversary has to provide *modified answers* $\boldsymbol{b}_j$ for $j \in [t]$ so that together with the correct answers $\boldsymbol{a}_j$ ($j \in \{t+1, \ldots, k\}$), the client is convinced that the file is $\tilde{x}$, where $\tilde{x} \neq x_1$.

Specifically, let $f(u)$ and $f_{\mathsf{v}}(u)$ be the polynomials obtained in the retrieval step with the correct responses $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_k$. On the hand, let $g(u)$ and $g_{\mathsf{v}}(u)$ be the polynomials obtained in the retrieval step with the adversarial responses $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_t, \boldsymbol{a}_{t+1}, \ldots, \boldsymbol{a}_k$. Then the adversary wins if

$$R^{(1)}(\mathsf{v}) g(0) = g_{\mathsf{v}}(0) \text{ and } f(0) \neq g(0)\,.$$

Since $R^{(1)}(\mathsf{v}) f(0) = f_{\mathsf{v}}(0)$, this is equivalent to $G(\mathsf{v}) = 0$, where

$$G(\mathsf{v}) \triangleq (f(0) - g(0)) R^{(1)}(\mathsf{v}) - (f_{\mathsf{v}}(0) - g_{\mathsf{v}}(0))\,^{5}. \quad (4)$$

Now, observe that since $f(0) - g(0) \neq 0$ and is independent of v, and that $\deg(R^{(1)}) > o$ due to our assumption, we have that $(f(0) - g(0)) R^{(1)}(\mathsf{v})$ as a polynomial in v has degree strictly greater than o. On the other hand, later in Claim 3, we show that $(f_{\mathsf{v}}(0) - g_{\mathsf{v}}(0))$ is a polynomial in v of degree at most o. Therefore, $G(\mathsf{v})$ is a nonzero polynomial with the same degree as $R^{(1)}(\mathsf{v})$. Next, by Schwartz-Zippel Lemma [33], [34], we have that regardless of choice of $G(\mathsf{v})$, the probability that $G(\mathsf{v}) = 0$ is at most $\frac{\deg R^{(i)}(\mathsf{v})}{p-1} \leq \frac{\delta}{p-1}$. This completes our proof.

The rest of the proof is now dedicated to demonstrating Claim 3 (stated later on). First, observe that the adversary has knowledge of the desired index one, $\boldsymbol{P}^{(1)}(\mathsf{v})$ and the queries $\boldsymbol{c}_{\mathsf{v}}(u_j)$ for $j \in [t]$. However, these are insufficient for the adversary to determine the random vectors $\boldsymbol{U}^{(s)}$'s and the value of v. Nevertheless, the adversary is able to write $\boldsymbol{U}^{(s)}$'s in terms of v, which we state in the following claim.

[5] In this proof, we consider $G(\mathsf{v})$ as a polynomial formed by the client from server responses. As a result, server responses composed of partial derivatives of the database multiplicative polynomial at certain points are uncontrollable by the client and are considered as constants. Meanwhile, the multipliers required to find the full derivatives, using partial ones as per Lemma 1, are controlled by the client and consideres as functions of v.

(Claim 1). Let $U_\ell^{(s)}$ be the ℓ-th component of $\boldsymbol{U}^{(s)}$. Then $U_\ell^{(s)}$ is a polynomial in v of degree at most one for all $s \in [t]$ and $\ell \in [m]$.

Fix $\ell \in [m]$. We write $\boldsymbol{c}_\mathsf{v}(u) = \big(c_{\mathsf{v},1}(u), \ldots, c_{\mathsf{v},m}(u)\big)$. Since $\boldsymbol{c}_\mathsf{v}(u) = \boldsymbol{P}^{(1)}(\mathsf{v}) + \sum_{s=1}^{t} \boldsymbol{U}^{(s)} u^s$, using the query vectors $\boldsymbol{c}_\mathsf{v}(u_j)$, $j \in [t]$, the adversary can form the following equations

$$\begin{bmatrix} u_1 & u_1^2 & \cdots & u_1^t \\ u_2 & u_2^2 & \cdots & u_2^t \\ \vdots & \vdots & \ddots & \vdots \\ u_t & u_t^2 & \cdots & u_t^t \end{bmatrix} \begin{bmatrix} U_\ell^{(1)} \\ U_\ell^{(2)} \\ \vdots \\ U_\ell^{(t)} \end{bmatrix} = \begin{bmatrix} c_{\mathsf{v},\ell}(u_1) - P_\ell^{(1)}(\mathsf{v}) \\ c_{\mathsf{v},\ell}(u_2) - P_\ell^{(1)}(\mathsf{v}) \\ \vdots \\ c_{\mathsf{v},\ell}(u_t) - P_\ell^{(1)}(\mathsf{v}) \end{bmatrix}.$$

Solving the system, $U_\ell^{(s)}$ is a linear combination of $P_\ell^{(1)}(\mathsf{v})$. Since $P_\ell^{(1)}(\mathsf{v})$ is a polynomial in v of degree at most one by (V1), we have $U_\ell^{(s)}$ is a polynomial in v of degree at most one, as required.

Next, we consider the function $h(u) \triangleq f_\mathsf{v}(u) - g_\mathsf{v}(u)$, where f_v and g_v are the polynomials obtained from the correct and modified responses. Recall that to retrieve these polynomials, the client uses Lemma 2 with the values $h^{(o')}(u_j)$ for $0 \le o' \le o$ and $j \in [k]$. These $h^{(o')}(u_j)$'s are in turn determined from the server responses using Lemma 1 or, in the case of $o' = 0$, are exactly the difference between correct and fabricated server responses[6].

(Claim 2). Let $0 \le o' \le o$. If $t+1 \le j \le k$, we have $h^{(o')}(u_j) = 0$. Otherwise, if $j \in [t]$, we have $h^{(o')}(u_j)$ is a polynomial in v of degree at most o'.

Let $0 \le o' \le o$. When $t+1 \le j \le k$, as the adversary cannot alter the responses from S_j, we have that $f_\mathsf{v}^{(o')}(u_j) - g_\mathsf{v}^{(o')}(u_j)$ and so, $h^{(o')}(u_j) = 0$.

Next, let $j \in [t]$. Again, as the adversary cannot alter $f_\mathsf{v}^{(o')}(u_j)$, we have that $f_\mathsf{v}^{(o')}(u_j)$ is constant with respect to v. On the other hand, for $o' > 0$, $g_\mathsf{v}^{(o')}(u_j)$ is computed using (1). While the partial derivatives/responses are crafted by the adversary, the value $\Gamma(\boldsymbol{c}_\mathsf{v}(u), \boldsymbol{r})$ depends on the derivatives of $\boldsymbol{c}_\mathsf{v}(u)$. Following Claim 1, we see that the coefficients of $c_{\mathsf{v},\ell(u)}$ are $U_\ell^{(s)}$'s, which are in turn polynomials in v of degree at most one. In general, for $\boldsymbol{r} \in \mathcal{R}(o', m)$, we have that each term of $\Gamma(\boldsymbol{c}_\mathsf{v}(u), \boldsymbol{r})$ is a product of at most o' such coefficients. Therefore, $\Gamma(\boldsymbol{c}_\mathsf{v}(u), \boldsymbol{r})$ in general is a polynomial in v of degree at most o'. As $h^{(o')}(u_j)$ is a linear combination of these polynomials or independent from them in case of $o' = 0$, Claim 2 follows.

Finally, we prove our objective.

(Claim 3). $h(0)$ is a polynomial of degree at most o.

Consider the system of equations defined in Lemma 2. Following Claim 2, each component on the right-hand side tuple $\boldsymbol{v}$ is a polynomial of degree at most o. On the other hand, Lemma 2 states that $h(0)$ is a linear combination of the components of $\boldsymbol{v}$. Therefore, $h(0)$ is a polynomial in v of degree at most o. This completes the proof of Claim 3 and the theorem. □

[6]We note that in the case of $o' = 0$, we do not employ Lemma 1, and $h^{(o')}$ is independent of v. However, to maintain consistency throughout the proof, we include it in the further derivations.

Example 4 (Example 3 continued)**.** Let us illustrate the key ideas in the proof of Theorem 2.

We assume the same setup as in Examples 2 and 3. Suppose that the adversary controls server S_1 and receives the query vector $\boldsymbol{c}_\mathsf{v}(1) = (4, 4, 2, 1)^T$ (see Example 3). Recall that $\boldsymbol{c}_\mathsf{v}(u) = \boldsymbol{P}^{(1)}(\mathsf{v}) + \boldsymbol{U}^{(1)} u$. Thus, using the (public) knowledge $\boldsymbol{P}^{(1)}(\mathsf{v}) = (\mathsf{v}, \mathsf{v}, 1, 0)^{\mathrm{T}}$, the adversary can derive that $\boldsymbol{U}^{(1)} = (4 - \mathsf{v}, 4 - \mathsf{v}, 1, 1)^T$.

As before, let $f_\mathsf{v}(u)$ and $g_\mathsf{v}(u)$ be the polynomials obtained from the correct and modified responses, respectively, and set $h(u) = f_\mathsf{v}(u) - g_\mathsf{v}(u)$. Then from Claim 2, as Server 2 (corresponding to $u_2 = 2$) is honest, we have that $h(2) = h'(2) = 0$. On the other hand, for $\boldsymbol{r} \in \mathcal{R}(1, 4) = \{0000, 1000, 0100, 0010, 0001\}$, we set $h_{\boldsymbol{r}}$ to be the difference between the correct and modified responses for $F_{\boldsymbol{x}}^{(\boldsymbol{r})}(1)(\boldsymbol{c}_\mathsf{v}(1))$ from Server 1 (malicious). For example, h_{1000} is the difference between $\frac{\partial}{\partial z_1} F_{\boldsymbol{x}}(\boldsymbol{c}_\mathsf{v}(1))$ and its fabricated value sent from Server 1. Using Lemma 1, noting that $\boldsymbol{U}^{(1)} = (4 - \mathsf{v}, 4 - \mathsf{v}, 1, 1)^T$ and that $\frac{\partial}{\partial u} c_{\mathsf{v},i}(u) = \boldsymbol{U}_i^{(1)}$, we have

$$\begin{aligned} h(1) &= h_{0000}, \\ h'(1) &= h_{1000}(4 - \mathsf{v}) + h_{0100}(4 - \mathsf{v}) + h_{0010} \cdot 1 + h_{0001} \cdot 1\,. \end{aligned}$$

Next, using the proof of Lemma 2, taking into account that $h(2) = h'(2) = 0$ and that the calculation is done over $\mathbb{F}_{11}$, we obtain

$$\begin{aligned} h(0) &= \boldsymbol{M}_1^{-1}\big(h(1), h(2), h'(1), h'(2)\big)^{\mathrm{T}} \\ &= 7h(1) + 5h(2) + 7h'(1) + 9h'(2) \\ &= 7h(1) + 7h'(1) \\ &= 7(h_{0000} + 4h_{1000} + 4h_{0100} + h_{0010} + h_{0001}) \\ &\quad + 4(h_{1000} + h_{0100})\mathsf{v}\,. \end{aligned}$$

Here, $h(0)$ is a polynomial in v of degree one and this corroborates with Claim 3. Hence, $G(\mathsf{v}) = (f(0) - g(0))R^{(1)}(\mathsf{v}) - h(0) = (f(0) - g(0))\mathsf{v}^2 - h(0)$ is always a polynomial of degree two, no matter how the adversary manipulates $g(0)$ and $h(0)$. Thus, Schwartz-Zippel lemma [33], [34] guarantees that the adversary succeeds with probability at most $\frac{2}{p-1}$.

We consider below a poor choice of $\boldsymbol{P}^{(1)}(\mathsf{v})$ and demonstrate a strategy for the adversary to fool the client with probability one.

We assume the same setting as in the previous examples. However, let's choose $\boldsymbol{P}^{(1)}(\mathsf{v}) = (\mathsf{v}, 1, 1, 0)^T$, which satisfies (V1) and (V2) with $R^{(1)}(\mathsf{v}) = \mathsf{v}$. Using the same $\boldsymbol{U}^{(1)} = (1, 1, 1, 1)^{\mathrm{T}}$ and $\mathsf{v} = 3$ as before, we have

$$\boldsymbol{c}_\mathsf{v}(u) = \begin{bmatrix} \mathsf{v} + u \\ 1 + u \\ 1 + u \\ u \end{bmatrix},\ \boldsymbol{c}_\mathsf{v}(1) = \begin{bmatrix} 4 \\ 2 \\ 2 \\ 1 \end{bmatrix},\ \boldsymbol{c}_\mathsf{v}(2) = \begin{bmatrix} 5 \\ 3 \\ 3 \\ 2 \end{bmatrix}.$$

Let the adversary control server S_1, which receives the query vector $\boldsymbol{c}_\mathsf{v}(1)$. As $\boldsymbol{c}_\mathsf{v}(1) = \boldsymbol{P}^{(1)}(\mathsf{v}) + \boldsymbol{U}^{(1)} \cdot 1$, the adversary can derive that $\boldsymbol{U}^{(1)} = \boldsymbol{c}_\mathsf{v}(1) - \boldsymbol{P}^{(1)}(\mathsf{v}) = (4 - \mathsf{v}, 1, 1, 1)^T$. Using the same notation and with some calculations, we have

$$h(0) = 7(h_{0000} + 4h_{1000} + h_{0100} + h_{0010} + h_{0001}) + 4h_{1000}\mathsf{v}\,.$$

In this case, $G(\mathsf{v}) = (f(0) - g(0))\mathsf{v} - h(0)$, and here, $G(\mathsf{v})$ can be made identically zero by a specific choice of answers from server S_1, regardless of the exact value of v. Specifically, the adversary can manipulate the responses such that

$$h_{0000} = 7, \quad h_{1000} = 1, \; h_{0100} = h_{0010} = h_{0001} = 0,$$
$$f(0) - g(0) = 4h_{1000} = 4\,.$$

Server i	$F_x(\boldsymbol{c}(u_i))$	$\frac{\partial}{\partial z_1}F_x$	$\frac{\partial}{\partial z_2}F_x$	$\frac{\partial}{\partial z_3}F_x$
Server 1 (correct)	2	1	8	6
Server 1 (modified)	12	1	8	6
Server 2	6	2	10	4
Server i	$F_x(\boldsymbol{c}_{\mathsf{v}}(u_i))$	$\frac{\partial}{\partial z_1}F_x$	$\frac{\partial}{\partial z_2}F_x$	$\frac{\partial}{\partial z_3}F_x$
Server 1 (correct)	5	4	8	8
Server 1 (modified)	1	5	8	8
Server 2	1	9	4	4

Indeed, using the modified responses, the client computes

$$g(0) = 5 \text{ and } g_{\mathsf{v}}(0) = 4,$$

and 'verifies' that $\mathsf{v}g(0) = g_{\mathsf{v}}(0)$ using $\mathsf{v} = 3$. Here, the adversary successfully wins the security game.

Remark 2. The exact characteristics of protocol Π_1 depend on parameters d, m and o. For instance, we have

- When $d = 1$, $m = n$, $o = 0$, we get the verifiable version of PIR protocol in Goldberg [25]. Here, for k servers, the total upload and download costs are $2kn$ symbols in $\mathbb{F}_p$ and $2k$ symbols in $\mathbb{F}_{p^e}$, respectively. In this case, we also have $t = k - 1$, and the download rate is given by $1/2k$. This is two times higher than the maximum asymptotic capacity $1 - t/k$ (when $n \to \infty$) [9].
- For general d and $o = 1$, we get the verifiable version of PIR protocol in Woodruff-Yekhanin [4]. For each server, upload and download costs are $2m$ and $2(m+1)$ symbols, respectively, and the total communication is $\sim 4m \sim 4(d!n)^{1/d}$ symbols. In other words, the total communication cost is sub-linear in n.
- For general d and $o > 1$, we get verifiable PIR with upload and download costs $2m$ and $2\binom{o+m}{o}$ symbols, respectively, and the total communication is $\sim 2\binom{o+m}{o} \sim 2\frac{1}{o!}(d!n)^{o/d}$ symbols. As before, the total communication cost is sub-linear in n.

B. Computational Publicly Verifiable Protocol

The protocol presented in the previous section is privately verifiable, and the verification key $\mathsf{vk} = \mathsf{v}$ is kept secret by the client. In contrast, in public verification, the client makes the verification key public, allowing a third party[7] to verify the correctness of the protocol execution by examining server responses. Hence, a third party is able to settle possible disputes between the client and servers. In this subsection, we convert Protocol Π_1 from Subsection IV-A into a publicly verifiable protocol, denoted as Π_2 by making verification keys public while securing value v in it that is used in verification under the discrete logarithm assumption in cyclic groups of large order. The security of the new protocol relies on the d-DHI assumption in the cyclic group $\mathbb{G}$ of prime order $p \geq 2^\lambda$ (see Definition 11).

[7]The third party must have all server responses together with auxiliary information aux_V to perform the verification. At the same time, it may leak some information about random vectors used for query generation and retrieval index.

As before, we have k distinct nonzero points $u_1, \ldots, u_k \in \mathbb{F}_p$; m, E and $F_{\boldsymbol{x}}$ satisfying conditions (P1), (P2), (P3); and $\boldsymbol{P}^{(i)}(\mathsf{v})$'s and $R^{(i)}(\mathsf{v})$'s satisfying (V1) and (V2). For this publicly verifiable protocol, we require in addition that $R^{(i)}(\mathsf{v}) = R(\mathsf{v})$ for some polynomial $R(\mathsf{v})$ for all $i \in [n]$.

In addition to $\mathbb{F}_p$, we choose a cyclic multiplicative group $\mathbb{G}$ of order $p \geq 2^\lambda$ with generator ω.

Publicly verifiable PIR protocol Π_2

Queries generation to retrieve x_i with verification

- We follow protocol Π_1 to generate the queries $\boldsymbol{q}_j$ for $j \in [k]$ and the client sends to server S_j. We refer to the reader to the previous subsection for details.
- Here, $\mathsf{vk} \triangleq \{\omega, \omega^{R(\mathsf{v})}\}$, which is made public. The value v is still kept private to the client.

Answer generation for server S_j with $j \in [k]$

- Following protocol Π_1, server S_j computes $\boldsymbol{a}_j^{(\boldsymbol{r})} \in \mathbb{F}_{p^e}^2$ for all $\boldsymbol{r} \in \mathcal{R}(o, m)$, which are then sent to the client.

Retrieval with public verification

- Again, we follow protocol Π_1 to recover $f(0)$ and $f_{\mathsf{v}}(0)$ using the evaluations $\boldsymbol{a}_j^{(\boldsymbol{r})}$'s.
- To verify the correctness of $f(0)$, the public verifier checks that $\left(\omega^{R(\mathsf{v})}\right)^{f(0)} = \omega^{f_{\mathsf{v}}(0)}$. It outputs $x_i = f(0)$ if the equation holds, and $\perp$ otherwise.

Theorem 3. Let d be the total degree of $F_{\boldsymbol{x}}$. Furthermore, suppose that we have $\boldsymbol{P}^{(i)}(\mathsf{v})$'s satisfy conditions (V1) and (V2) and $R^{(i)}(\mathsf{v}) = R(\mathsf{v})$ for $i \in [n]$. If $dt + 1 \leq (o+1)k$ and $\deg R(\mathsf{v}) = \delta > o$, then the protocol Π_2 is a k-server t-private computationally verifiable PIR protocol under δ-DHI assumption. Here, Π_2 has upload cost of $2m$ symbols in $\mathbb{F}_p$ and download cost of at most $2\binom{o+m}{o}$ symbols in $\mathbb{F}_{p^e}$ per server.

Proof. As before, the correctness and privacy of the PIR protocol follow from Theorem 1, while the upload and download cost values follow from the form of answers and queries. Therefore, it remains to demonstrate the computational verifiability guarantee.

Without loss of generality, suppose that the client wants the file x_1 and we let the adversary control servers $S_1, \ldots, S_t$. Following the proof of Theorem 2, we let $\{f(u), f_{\mathsf{v}}(u)\}$ and $\{g(u), g_{\mathsf{v}}(u)\}$ be the polynomial pairs obtained in the retrieval step with the correct responses and the adversarial responses, respectively. Then the adversary wins if and only if

$$(\omega^{R(\mathsf{v})})^{g(0)} = \omega^{g_{\mathsf{v}}(0)} \text{ and } f(0) \neq g(0)\,.$$

As before, this is equivalent to $\omega^{G(\mathsf{v})} = 1$, where $G(\mathsf{v})$ is defined in (4) and $G(0) \neq 0$. Following the proof of Theorem 2, we have that the degree of $G(\mathsf{v})$ is δ.

Now, the PPT adversary wins the security experiment if and only if $\omega^{G(\mathsf{v})} = 1$ and $f(0) \neq g(0)$. Let us assume a stronger

adversary that has access to the values $\omega, \omega^{\mathsf{v}}, \ldots, \omega^{\mathsf{v}^{\delta-1}}$ and let $G(\mathsf{v}) = \sum_{j=j_0}^{\delta} G_j \mathsf{v}^j$, where $G(\mathsf{v})$ is non-zero polynomial with coefficients independent from v. Here, j_0 is the smallest power with nonzero coefficient. Then we have that $1 = \omega^{G(\mathsf{v})} = (\omega^{\mathsf{v}^{j_0}})^{G_{j_0}} \prod_{j=j_0+1}^{\delta} \left(\omega^{\mathsf{v}^j}\right)^{G_j}$. Therefore,

$$\omega^{1/\mathsf{v}} = \prod_{j=j_0+1}^{\delta} \left(\omega^{\mathsf{v}^{j-j_0-1}}\right)^{-G_j/G_{j_0}}.$$

In other words, using the coefficients G_j's and values ω^{v^j}'s, the PPT adversary is able to determine $\omega^{1/\mathsf{v}}$. However, the δ-DHI assumption states that this event occurs with probability negligible in λ. □

Remark 3. The definition of verifiable PIR (see Definition 4) assumes that aux_R and aux_V are kept private from the servers, and in a publicly verifiable setup, they must be securely delivered to the entity performing verification. To overcome this limitation, we can introduce the notion of a *fully publicly verifiable* protocol, where both vk and aux_R with aux_V are public. To transform protocol Π_2 into a *fully publicly verifiable*, we use $\mathsf{vk} = \{g^{R(\mathsf{v})V_1^{(1)}}, \ldots, g^{R(\mathsf{v})V_e^{(1)}}, \ldots, g^{R(\mathsf{v})V_e^{(t)}}\}$, $\text{aux}_R = \{g^{V_1^{(1)}}, \ldots, g^{V_e^{(t)}}\}$ and $\text{aux}_V = \{g^{U_1^{(1)}}, \ldots, g^{U_e^{(t)}}\}$, where $V_1^{(1)}, \ldots, V_e^{(1)}$ and $U_1^{(1)}, \ldots, U_e^{(1)}$ are components of the element $\mathbf{V}^{(1)}$ and $\mathbf{U}^{(1)}$, correspondingly, in a pre-selected basis of $\mathbb{F}_{p^e}$ over $\mathbb{F}_p$. Now, each entity with access to server responses and knowledge of aux_R with aux_V and vk can recover $(g^{R(\mathsf{v})})^{f(0)}$ and $g^{f_{\mathsf{v}}(0)}$ and perform the verification. We note that the security of this scheme is based on the d-DHI assumption.

C. Computational privately verifiable PIR

The main drawback of previously mentioned protocols is that they double the download cost compared to the non-verifiable scheme Π_0. One possible solution to address this issue is to apply linearly homomorphic vector commitment to the second part of server responses. A deterministic vector commitment scheme allows to commit to given element $y \in \mathbb{F}_{p^e}$, treated as a vector of dimension e over $\mathbb{F}_p$, as $\text{Commit}(y)$ so that the commitments of two vectors can be compared to check if they are the same vector or not. The same property holds for individual entries of the vector. Ideally, the later check is short and computationally simple. For our purposes we require that vector commitment scheme are:

- **Binding.** Commitment cannot be opened to the values that are inconsistent with commitment vector. In other words, the probability that commitments for two different vectors coincides can be bounded by a negligible function. This property based on some cryptographic assumption.
- **Linearly homomorphic.** For any $y_1, y_2 \in \mathbb{F}_{p^e}$ and $\alpha_1, \alpha_2 \in \mathbb{F}_p$ it holds that

$$\text{Commit}(\alpha_1 y_1 + \alpha_2 y_2) = \big(\alpha_1 \odot \text{Commit}(y_1)\big) \bigoplus \big(\alpha_2 \odot \text{Commit}(y_2)\big), \quad (5)$$

  where $\bigoplus$ is group operation in corresponding group and $\odot$ means performing group operation several times.

For a detailed introduction to linearly homomorphic vector commitment schemes, we refer the reader to [35], [36], and the references therein. In what follows, we use two examples of linearly homomorphic vector commitment schemes. Note that they can be changed to any schemes with such a property. The first scheme is given by homomorphic hashes based on DL assumption (see Definition 10) in multiplicative group of finite field. Initially it was introduced for the verification of digital content distributed by rateless erasure codes in [37], and later applied to network coding in [38]. In this case we choose a prime number r so that $r-1$ is divisible by p and random non-zero elements $g_1, \ldots, g_e$ of $\mathbb{Z}_r$ of order p. We then choose a basis $\mathcal{B}$ of $\mathbb{F}_{p^e}$ over $\mathbb{F}_p$ and for any $y \in \mathbb{F}_{p^e}$ represented in basis $\mathcal{B}$ as $(y_1, \ldots, y_e)$ we define $\text{Commit}(y) = \prod_{l=1}^{e} g_l^{y_e} \mod r$. In this case, for $x, y \in \mathbb{F}_{p^e}$ and $\alpha, \beta \in \mathbb{F}_p$ we have $\text{Commit}(\alpha x + \beta y) = (\text{Commit}(x))^{\alpha} \cdot (\text{Commit}(y))^{\beta} \mod r$.

Another one is given by Kate-Zaverucha-Goldberg (KZG) polynomial commitment [36]. In this case, let us consider group of points of elliptic curve $\mathbb{G}$ with generator g and order $p \geq 2^{\lambda}$ and deploy a trusted setup $(g, g^r, \ldots, g^{r^{e-1}})$ for randomly chosen non-zero parameter r from $\mathbb{F}_p$ kept in secret. Next, we choose a basis $\mathcal{B}$ of $\mathbb{F}_{p^e}$ over $\mathbb{F}_p$. For any $y \in \mathbb{F}_{p^e}$ represented in basis $\mathcal{B}$ as $(y_1, \ldots, y_e)$ we define $\text{Commit}(y) = \prod_{l=1}^{e} (g^{r^{l-1}})^{y_l}$. In this case, for $x, y \in \mathbb{F}_{p^e}$ and $\alpha, \beta \in \mathbb{F}_p$ we have $\text{Commit}(\alpha x + \beta y) = (\text{Commit}(x))^{\alpha} \cdot (\text{Commit}(y))^{\beta}$.

Note that the first scheme requires extremely large field to operate, with sizes $r \geq 2^{1024}$ and $p \geq 2^{257}$ and its binding property is based on DL-assumption (see Definition 10) in multiplicative group of order p in finite field. In comparison, the second scheme operates over significantly smaller finite field, but requires a trusted setup and multiple elliptic curve operations. Its binding property is based on t-SDH assumption (see Definition 12).

The main idea of the modified verifiable PIR protocol is to apply linearly homomorphic vector commitment to responses used for verification in protocol Π_1, replace the original verification equation with its committed version, and check it using modified responses.

As before, we have k distinct nonzero points $u_1, \ldots, u_k \in \mathbb{F}_p$; m, E and $F_{\boldsymbol{x}}$ satisfying conditions (P1), (P2), (P3); and $\boldsymbol{P}^{(i)}(\mathsf{v})$'s and $R^{(i)}(\mathsf{v})$'s satisfying (V1) and (V2). We also use a linearly homomorphic vector commitment scheme described as above.

Computationally verifiable PIR protocol Π_3

Queries generation to retrieve x_i with verification

- We follow protocol Π_1 to generate the queries $\boldsymbol{q}_j$ for $j \in [k]$ and the client sends to server S_j. We refer to the reader to the previous subsection for details.
- Here, $\mathsf{vk} \triangleq \mathsf{v}$.
- We include the parameters required to deploy linearly homomorphic vector commitment in addition to aux_A and aux_V.

Answer generation for server S_j

- Following protocol Π_1, server S_j computes $F_{\boldsymbol{x}}^{(\boldsymbol{r})}(\boldsymbol{c}(u_j))$ for all $\boldsymbol{r} \in \mathcal{R}(o,m)$ for retrieval.
- For verification, server S_j computes $\text{Commit}(F_{\boldsymbol{x}}^{(\boldsymbol{r})}(\boldsymbol{c}_{\mathsf{v}}(u_j)))$ for all $\boldsymbol{r} \in \mathcal{R}(o,m)$.
- Hence, server S_j sends all $\binom{o+m}{o}$ evaluations to the client, where each evaluation comprises one symbol in $\mathbb{F}_{p^e}$ and one commitment of corresponding size.

Retrieval with results verification

- For file retrieval, we follow protocol Π_1 to recover $f(0)$. Set $C \triangleq \text{Commit}(R^{(i)}(\mathsf{v})f(0))$.
- For verification purposes, we aim to compute $\text{Commit}(f_{\mathsf{v}}(0))$. While we mimic the steps in Protocol Π_1, we note that we use the linearly homomorphic property (5) of the Commit to perform the computations. Specifically, we do the following:
  - For $\ell \in \{0,1,\ldots,o\}$ and $j \in [k]$, we use the chain rule (1) to compute $\text{Commit}(f_{\mathsf{v}}^{(\ell)}(u_j))$. In other words, we set $\text{Commit}(f_{\mathsf{v}}^{(\ell)}(u_j)) = \bigoplus_{\boldsymbol{r}\in\mathcal{R}(\ell,m)\setminus\{\boldsymbol{0}\}} \Gamma(\boldsymbol{c}_{\mathsf{v}}(u_j),\boldsymbol{r}) \odot \text{Commit}(F_{\boldsymbol{x}}^{(\boldsymbol{r})}(\boldsymbol{c}_{\mathsf{v}}(u_j)))$.
  - We use Lemma 2 to determine $\text{Commit}(f_{\mathsf{v}}(0))$. Specifically, we $C_{\mathsf{v}} \triangleq \bigoplus_{\substack{0\le\ell\le o\\ j\in[k]}} \alpha_{\ell,j} \odot \text{Commit}(f_{\mathsf{v}}^{(\ell)}(u_j))$, where $\alpha_{\ell,j} \in \mathbb{F}_p$ are scalars dependent only on u_j's.
- To verify the correctness of x_i, the client checks that $R^{(i)}(\mathsf{v}) \odot C = C_{\mathsf{v}}$. If the equation holds, the algorithm returns x_i. Otherwise, the output is $\perp$.

Theorem 4. Let d be the total degree of $F_{\boldsymbol{x}}$. Furthermore, suppose that we have $\boldsymbol{P}^{(i)}(\mathsf{v})$'s satisfy conditions (V1) and (V2). If $dt+1 \le (o+1)k$ and $\deg R^{(i)}(\mathsf{v}) > o$ for all $i \in [n]$, then the protocol Π_3 is a k-server t-private computationally verifiable PIR protocol under the same cryptographic assumption as binding property of utilized linearly homomorphic vector commitment scheme. Here, Π_3 has upload cost of $2m$ symbols in $\mathbb{F}_p$ and download cost of at most $\binom{o+m}{o}$ symbols in $\mathbb{F}_{p^e}$ and $\binom{o+m}{o}$ commitments per server.

Proof. As before, the correctness and privacy of the PIR protocol follow from Theorem 1, while the upload and download cost values follow from the form of answers and queries. Therefore, it remains to demonstrate the computational verifiability guarantee.

Without loss of generality, suppose that the client wants the file x_1 and we let the adversary control servers $S_1,\ldots,S_t$. Following the proof of Theorem 2, we let $\{f(u), f_{\mathsf{v}}(u)\}$ and $\{g(u), g_{\mathsf{v}}(u)\}$ be the polynomial pairs obtained in the retrieval step with the correct responses and the adversarial responses, respectively. Then the adversary wins if and only if

$$\text{Commit}(R^{(1)}(\mathsf{v})g(0)) = \text{Commit}(g_{\mathsf{v}}(0)) \text{ and } f(0) \neq g(0)\,.$$

Let us first define the following events.

$$\mathcal{E}_1 \triangleq \{R^{(1)}(\mathsf{v})g(0) = g_{\mathsf{v}}(0), f(0) \neq g(0)\},$$
$$\mathcal{E}_2 \triangleq \{\text{Commit}\big(R^{(1)}(\mathsf{v})g(0)\big) = \text{Commit}\big(g_{\mathsf{v}}(0)\big), R^{(1)}(\mathsf{v})g(0) \neq g_{\mathsf{v}}(0), f(0) \neq g(0)\},$$
$$\mathcal{E}_3 \triangleq \{\text{Commit}\big(R^{(1)}(\mathsf{v})g(0)\big) = \text{Commit}\big(g_{\mathsf{v}}(0)\big), f(0) \neq g(0)\},$$

Then $\mathcal{E}_1$ and $\mathcal{E}_3$ are the events that the adversary wins in Π_1 and Π_3, respectively. By Theorem 2, $\Pr[\mathcal{E}_1] \in \mathsf{negl}(\lambda)$. Moreover, $\mathcal{E}_2$ is a subset of the event that $R^{(1)}(\mathsf{v})g(0)$ and $g_{\mathsf{v}}(0)$ are different but have the same commitment, and hence, $\Pr[\mathcal{E}_2] \in \mathsf{negl}(\lambda)$ due to the binding property of the underlying commitment scheme. Lastly, it is clear that $\mathcal{E}_3 = \mathcal{E}_1 \cup \mathcal{E}_2$. Therefore, the probability that the adversary wins in Π_3 is

$$\Pr[\text{EXP}^{\text{PriV}}_{\mathcal{A},\Pi_3}(n,\boldsymbol{x},i,1,\ldots,t)=1] = \Pr[\mathcal{E}_3] = \Pr[\mathcal{E}_1\cup\mathcal{E}_2] \le \Pr[\mathcal{E}_1] + \Pr[\mathcal{E}_2] \in \mathsf{negl}(\lambda),$$

which is negligible in λ. □

We note that the additional communication overhead associated to verification in the protocol Π_3 becomes relatively small as the size e of each file x_i increases. Indeed, according to Theorem 4, the ratio between the additional download cost (the commitments) and the download cost without verification is equal to the size of each commitment divided by $e\log_2(p)$. As the commitment can have constant size, e.g., 384 bits in a typical setting of KZG when the curve BLS12-381 is used, this ratio actually tends to zero when e goes to infinity.

V. Comparisons to Related Works

In this section, we provide detailed comparisons with the most closely related works that either construct or can be used to construct verifiable PIR protocols where the client can detect the presence of adversarial servers.

Authenticated Private Information Retrieval [15]. Independent to our preliminary work [1], Colombo *et al.* [15, Construction 1] (USENIX Security'23) also employs the idea of doubling queries to verify the recovered file of interest. However, there are significant differences between our work and theirs regarding the security assumption and guarantee, which are explained below.

First, one of our verifiable PIR schemes (Π_1) is *information-theoretically* verifiable, while the construction in [15] is *computationally* verifiable. Indeed, a key ingredient in their protocol is a family of *function secret sharing (FSS) schemes*, introduced by Boyle *et al.* [39], [40]. In a nutshell, in a $\mathcal{F}$-FSS scheme, a *dealer* picks a function f from a family of functions $\mathcal{F}$ and outsources the computation of $f(x)$ to k *servers*, each of which receives a share of $f(\cdot)$. Upon receiving x from an *user*, each server uses their share of f to perform some computation on x and communicates the output to the user. After collecting sufficient responses from the servers, the user recovers $f(x)$. In the entire process, no information of f is revealed to any subset of $k-1$ servers. It is shown in [15] that if there is an $\mathcal{F}$-FSS scheme with $\mathcal{F}$ closed under scalar multiplication[8],

[8] $\mathcal{F}$ is closed under scalar multiplication if for all $f \in \mathcal{F}$ and $\mathsf{v} \in \mathbb{F}$, we have that $\mathsf{v}f$ belongs to $\mathcal{F}$ too.

one can construct a verifiable PIR. However, to the best of our knowledge, all known $\mathcal{F}$-FSS protocols with $\mathcal{F}$ closed under scalar multiplication are not information-theoretically secure[9]. Thus, the verifiability guarantee provided by the schemes in [15] relies on certain computational assumptions. In contrast, the general construction in our paper is self-contained (doesn't depend on any cryptographic primitive), which allows us to obtain an explicit information-theoretically verifiable PIR protocol.

Second, our PIR scheme can withstand a *significantly stronger adversary*. More specifically, in our security setting, the adversary not only controls the responses of t servers but also has the knowledge of the retrieval index. The information-theoretic security is still guaranteed for our construction under this more powerful adversary.

One distinctive advantage of the protocol in [15], however, is that it also allows the client to privately retrieve and verify any linear combination of elements in $\{x_i : i \in \mathcal{I}\}$ for $\mathcal{I} \subseteq [n]$, which is more general than ours.

Constructions of Verifiable Private Information Retrieval Based on Private Linear Computation. As suggested by a reviewer, one may also employ an information-theoretic *private computation* scheme to construct an information-theoretic verifiable private information retrieval scheme. The main idea is to privately query $f_i(\boldsymbol{x}) = x_i$ and $f_{i,\mathsf{v}}(\boldsymbol{x}) = \mathsf{v}x_i$, where $\mathsf{v} \in \mathbb{F}_p \setminus \{0\}$ is a secret value chosen by the client, in order to retrieve x_i and verify its correctness. We discuss this approach in detail below in comparison with ours. We note that a rigorous security proof for this construction is still missing.

The concept of information-theoretic private linear computation was independently introduced in [42], [43] and further extended in [44]–[50]. In its standard setting for uncoded storage (see, e.g. Karpuk [48]), there are k servers, each of which stores a copy of a database $\boldsymbol{x} = x_1 \cdots x_n$, and a client, who wants to privately evaluate h functions $f_1(\boldsymbol{x}), \ldots, f_h(\boldsymbol{x})$ on the database $\boldsymbol{x}$. The key performance metric for private computation is the *download rate*, which is defined as the ratio of the total size of the desired outputs $f_1(\boldsymbol{x}), \ldots, f_h(\boldsymbol{x})$ and the total size of the responses the client downloads from all servers. Note that in our work, we follow the standard notion of *communication cost* (see, e.g. [2], [4]), which is the sum of the *upload* and *download costs*. While the download rate metric is more relevant when the file size is large ($e = |x_i| \gg n$) and is the focus of the information theory community, the communication cost is more relevant for small file size ($e = |x_i| \ll n$) and is the focus of the computer science community.

We now compare our verifiable PIR schemes against schemes constructed from existing private linear computation schemes. In summary, our scheme achieves the same or worse download rates in the large file regime, and has lower communication cost in the small file regime.

Using the construction and the terminology in Karpuk [48, Section III] (note that the follow-up works [49], [50] essentially use the same construction but with more features, see [49, Section V] and [50, Remark 1]), the function space $\mathcal{Q}$ consists of all functions corresponding to $\mathbb{F}_p$-linear combinations of $x_1, \ldots, x_n$. This space $\mathcal{Q}$ is the smallest function space that is closed under function addition and $\mathbb{F}_p$-scalar multiplication and contains the functions $f_i(\boldsymbol{x}) = x_i$ and $f_{i,\mathsf{v}}(\boldsymbol{x}) = \mathsf{v}x_i$, $i \in [n]$, $\mathsf{v} \in \mathbb{F}_p \setminus \{0\}$. Note that the client needs to privately evaluate *two* functions $f_i(\boldsymbol{x})$ and $f_{i,\mathsf{v}}(\boldsymbol{x})$ in order to privately download x_i and verify its correctness. It is obvious that $\mathcal{Q}$ has $\mathbb{F}_p$-dimension n and is spanned by $\{f_i\}_{i\in[n]}$. According to [48, Theorem 1][10] and [50, Theorem 2], there exists private-computation-based PIR schemes with download rate[11] $(k-t)/2k$. In comparison, our scheme Π_1, with $d = 1, o = 0, m = n, t = k-1$, has the download cost $2k$ symbols over $\mathbb{F}_{p^e}$ (see Theorem 2), and hence, achieves the same download rate of $1/2k$ as in [48], [50]. For $t \leq k-2$, our download rate is smaller than that in [48], [50].

The scheme from [48], [50] has an upload cost of $k(k-t)n$ elements in $\mathbb{F}_p$ (see [48, Section III.C] and [50, Theorem 2]). In other words, the schemes based on [48], [50] require linear communication cost per server. In contrast, for smaller t, and small e, we can achieve *sub-linear* (in n) communication cost per server. Indeed, for $e = 1$, fixing $o = 1$ and $d \geq 2$, then for every $t \geq 1$ satisfying $dt + 1 \leq 2k$, or equivalently, $1 \leq t \leq (2k-1)/d$, our scheme has a communication cost approximately $4(d!n)^{1/d}k \in O(kn^{1/d})$ symbols in $\mathbb{F}_p$ (see Theorem 2 and Remark 2). This conclusion is still true for every $e \ll n$.

Alternatively, the client can privately query $f_i(\boldsymbol{x}) = x_i$ and $g_{i,\mathsf{v}} = x_i + \mathsf{v}$ (instead of $f_{x,\mathsf{v}} = \mathsf{v}x_i$), where $\mathsf{v} \in \mathbb{F}_p$. This scheme, while achieving a lower probability of adversary success ($1/p$ instead of $1/(p-1)$), incurs a higher upload cost due to the increase in the dimension of the function space ($n+1$ instead of n), which must be the space of all affine functions on $x_1, \ldots, x_n$.

Next, we discuss the non-colluding case where $t = 1$. To the best of our knowledge, this is the primary focus of most works on private linear computation schemes [42]–[47]. Specifically, in [42]–[44], private linear computation schemes with high download rates were proposed. In [42], [43], the authors demonstrated a corresponding converse bound, showing that their schemes were capacity-achieving[12]. To achieve adversary success rate $1/Q$, the file size of the schemes in [42]–[44] is exponential in Q and n (the number of files). In contrast, our file size is $\Theta(Q)$, which is significantly smaller.

Finally, we note that, our verifiable PIR schemes can withstand a significantly stronger adversary. Specifically, in our security setting, the adversary not only controls the responses

[9]In fact, the only known information-theoretically secure $\mathcal{F}$-FSS are given in [41] and they are not closed under scalar multiplication.

[10]Karpuk [48] uses N, T, and B for the number of servers, the number of colluding servers, and the number of functions to be evaluated, respectively.

[11]Note that in [48, Theorem 1], the stated download rate is $(k-t)/k$, not $(k-t)/2k$. The reason is that in the context of private computation, downloading $f_i(\boldsymbol{x})$ and $f_{i,\mathsf{v}}(\boldsymbol{x})$ would count as two symbols being retrieved. However, when applying to verifiable PIR setting, $f_{i,\mathsf{v}}(\boldsymbol{x})$ is counted as a redundant request used for verification purpose only.

[12]Capacity is defined as the maximum size of the message that can be privately retrieved (which is the size of one file) normalized by the number of downloaded information symbols.

of t colluding servers, but also has the knowledge of the retrieval index. In contrast, for private linear computation, t servers do not know both file indices and the function. Nevertheless, we demonstrate information-theoretic security against this stronger adversary.

Zhang and Wang [10] investigated non-interactive multi-server verifiable computations of low-degree polynomials, which is closely related to our approach. In fact, our polynomial-based k-server t-private PIR protocol can also be made verifiable utilizing schemes from [10] without requiring additional conditions (V1) and (V2). However, using Schemes 1 and 2 from [10] would require extra d or t servers, respectively, to perform verification. An application of their Scheme 3, while doesn't increase the number of required servers and provides the same results as ours, cannot be publicly verifiable as ours. Last but not least, in our paper, we also discuss an approach to reduce the extra communication cost incurred by verification by employing homomorphic commitment schemes.

Ke and Zhang [13] recently extended their two-server verifiable PIR scheme [12], which was based on Woodruff-Yekhanin PIR scheme [4], to a k-server t-private $(k-1)$-verifiable PIR protocol for a fixed number of malicious servers. In the case of k-server $(k-1)$-private $(k-1)$-verifiable PIR, our protocol achieves a slightly better communication complexity of $O\left(n^{\frac{1}{\lfloor \frac{2k-1}{k-1} \rfloor}} \log(p)\right)$, while the communication complexity of the protocol in [13] is $O\left(n^{\frac{1}{\lfloor \frac{2k-1}{k-1} \rfloor - 1}} \log(p)\right)$. Additionally, the authors of [13] provide a security proof only for the case when $k \leq 5$, whereas we provide a formal security proof for *all* values of k.

Verifiable PIR schemes with two servers. Although byzantine-resistant PIR [18]–[23] is a stronger notion than our verifiable PIR, such schemes must have at least *three* servers (so that error correction can work). Thus, we discuss here the setting with only two servers, which automatically eliminate byzantine-resistant schemes. In this scenario, encapsulating the PIR protocol described in Goldberg [25] within our framework yields the same result as described in [1]. To the best of our knowledge, there are no information-theoretically verifiable extensions of the PIR protocol in [25] for general $k > 2$ in the literature. On the other hand, encapsulating the Woodruff-Yekhanin protocol [4] offers better communication complexity compared to the information-theoretically verifiable protocol presented in [12]. Specifically, our protocol has a total communication cost of $8m + 4$ elements of field $\mathbb{F}_p$ when $\binom{m}{3} \geq n$, with a communication complexity of $O(n^{1/3} \log(p))$. In contrast, the protocol from [12] has a total communication of $4m + 2$ field elements when $\binom{m}{3}/m \geq n$, with a communication complexity of $O(n^{1/2} \log(p))$, which is higher than ours.

Let us summarize the advantages of our approach among the aforementioned competing ones in the Table II Since competing approaches on verifiable PIR focused on communication complexity optimization only (and not the download rate), in this table we consider k-server t-private PIR protocols deployed over the finite field $\mathbb{F}_p$ ($e = 1$) and a database of n files and measure the total communication cost. We do not include the protocol Π_3 in the comparison since the reduction of extra communication costs incurred by verification by linearly homomorphic commitment schemes makes sense only for PIR protocols over an extended field. We also assume that number of servers k is much smaller than the database size n.

VI. Conclusion

We explored the problem of verifiable private information retrieval where the client can detect the presence of malicious servers. We proposed a unified framework for polynomial-based PIR protocols encompassing various existing protocols that optimize the download rate or total communication cost. We also introduced a way to transform such a protocol to a verifiable one without increasing the number of involved servers by letting the client query twice. The security guarantees for verification can be information-theoretic or computational, and the verification keys can be public or private. In one of our protocols, which employs a homomorphic vector commitment scheme, the additional download overhead associated to verification can be made negligible to the download cost (without verification) as the file size increases. Developing new verification techniques that encompass more polynomial-based PIR protocols like Chor *et al.* and generalizing our approaches to coded PIR protocols are interesting open problems.

Acknowledgements.

This research / project is supported by the National Research Foundation, Singapore and Infocomm Media Development Authority under its Trust Tech Funding Initiative, Singapore Ministry of Education Academic Research Fund Tier 2 Grant T2EP20121-0007, Australia Research Council DECRA Grant DE180100768 and DP Grant DP200100731, National Natural Science Foundation of China Grant No. 62372299 and the Natural Science Foundation of Shanghai Grant No. 21ZR1443000. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore and Infocomm Media Development Authority.

The authors would like to thank the Prof. Qiben Yan and the anonymous reviewers for their valuable comments and suggestions, which greatly improve the paper.

References

[1] S. Kruglik, S. H. Dau, H. M. Kiah, and H. Wang, "Two-server private information retrieval with optimized download rate and result verification," in *2023 IEEE International Symposium on Information Theory (ISIT)*, 2023, pp. 1336–1341.

[2] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan, "Private information retrieval," in *IEEE 36th Annual Foundations of Computer Science (FOCS)*, 1995, pp. 41–50.

[3] S. Ulukus, S. Avestimehr, M. Gastpar, S. A. Jafar, R. Tandon, and C. Tian, "Private retrieval, computing, and learning: Recent progress and future challenges," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 3, pp. 729–748, 2022.

[4] D. Woodruff and S. Yekhanin, "A geometric approach to information-theoretic private information retrieval," *SIAM Journal on Computing*, vol. 37, no. 4, pp. 1046–1056, 2007.

[5] Z. Dvir and S. Gopi, "2-server PIR with subpolynomial communication," *Journal of the ACM*, vol. 63, no. 4, 2016.

Scheme	Colluding servers	Malicious servers	Verifiability	Trusted setup	Communication complexity
[15, Construction 1]	$t = k - 1$	$t = k - 1$	Computational private	NO	$O(\log(n)\log(p))$
[14]	$t \leq k$	$t \leq k$	Computational public	YES	$O(n^{\frac{1}{\lfloor (2k-1)/t \rfloor}} \log(p))$
[48]	$t = k - 1/t = k - 2$	$t = k - 1/t = k - 1$	Information-theoretically private	NO	$O(n \log(p))$
[13]	$t \leq k - 1$	$k - 1$	Information-theoretically private	NO	$O(\frac{k^2}{t}(\frac{nk}{t})^{1/(\lfloor (2k-1)/t \rfloor - 1)} \log(p))$
[10, Scheme 1/Scheme 4]	$t \leq k - 2$	$t \leq k - 2$	Information-theoretically private/Computational public	NO	$O(n^{\frac{1}{\lfloor \frac{2k-1}{t+1} \rfloor}} \log(p))$
[10, Scheme 2/Scheme 5]	$t \leq \lfloor \frac{k-1}{2} \rfloor$	$t \leq \lfloor \frac{k-1}{2} \rfloor$	Information-theoretically private/Computational public	NO	$O(n^{\frac{1}{\lfloor \frac{2k-1}{t} \rfloor - 1}} \log(p))$
[10, Scheme 3]	$t \leq k - 1$	$t \leq k - 1$	Information-theoretically private	NO	$O(n^{\frac{1}{\lfloor (2k-1)/t \rfloor}} \log(p))$
Protocol Π_1/Protocol Π_2	$t \leq k - 1$	$t \leq k - 1$	Information-theoretically private/Computational public	NO	$O(n^{\frac{1}{\lfloor (2k-1)/t \rfloor}} \log(p))$

TABLE II: Comparison of parameters of k-server t-private PIR protocols deployed over the finite field $\mathbb{F}_p$ and a database of n files

[6] N. B. Shah, K. V. Rashmi, and K. Ramchandran, "One extra bit of download ensures perfectly private information retrieval," in *2014 IEEE International Symposium on Information Theory (ISIT)*, 2014, pp. 856–860.

[7] T. H. Chan, S.-W. Ho, and H. Yamamoto, "Private information retrieval for coded storage," in *2015 IEEE International Symposium on Information Theory (ISIT)*, 2015, pp. 2842–2846.

[8] R. Tajeddine, O. W. Gnilke, and S. El Rouayheb, "Private information retrieval from MDS coded data in distributed storage systems," *IEEE Transactions on Information Theory*, vol. 64, no. 11, pp. 7081–7093, 2018.

[9] H. Sun and S. A. Jafar, "The capacity of robust private information retrieval with colluding databases," *IEEE Transactions on Information Theory*, vol. 64, no. 4, pp. 2361–2370, 2018.

[10] L. F. Zhang and H. Wang, "Multi-server verifiable computation of low-degree polynomials," in *2022 IEEE Symposium on Security and Privacy (SP)*, 2022, pp. 596–613.

[11] T. Gupta, N. Crooks, W. Mulhern, S. Setty, L. Alvisi, and M. Walfish, "Scalable and private media consumption with popcorn," in *13th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2016.

[12] P. Ke and L. F. Zhang, "Two-server private information retrieval with result verification," in *2022 IEEE International Symposium on Information Theory (ISIT)*, 2022, pp. 408–413.

[13] ——, "Private information retrieval with result verification for more servers," in *Applied Cryptography and Network Security: 21st International Conference (ACNS)*, 2023, pp. 197–216.

[14] Q. Cao, H. Y. Tran, S. H. Dau, X. Yi, E. Viterbo, C. Feng, Y.-C. Huang, J. Zhu, S. Kruglik, and H. M. Kiah, "Committed private information retrieval," in *2023 European Symposium on Research in Computer Security (ESORICS)*, 2023, pp. 393–413.

[15] S. Colombo, K. Nikitin, H. Corrigan-Gibbs, D. J. Wu, and B. Ford, "Authenticated private information retrieval," in *32nd USENIX Security Symposium (USENIX Security)*, 2023, pp. 3835–3851.

[16] L. F. Zhang and R. Safavi-Naini, "Verifiable multi-server private information retrieval," in *Applied Cryptography and Network Security: 12th International Conference (ACNS)*, 2014, pp. 62–79.

[17] L. Zhao, X. Wang, and X. Huang, "Verifiable single-server private information retrieval from LWE with binary errors," *Information Sciences*, vol. 546, pp. 897–923, 2021.

[18] K. Banawan and S. Ulukus, "The capacity of private information retrieval from byzantine and colluding databases," *IEEE Transactions on Information Theory*, vol. 65, no. 2, pp. 1206–1219, 2019.

[19] K. Kurosawa, "How to correct errors in multi-server PIR," in *Advances in Cryptology (ASIACRYPT)*, 2019, pp. 564–574.

[20] K. Devet, I. Goldberg, and N. Heninger, "Optimally robust private information retrieval," in *21st USENIX Security Symposium (USENIX Security)*, 2012, pp. 269–283.

[21] A. Beimel, *Robust Private Information Retrieval*. Springer Berlin Heidelberg, 2019, pp. 1–3.

[22] S. Kruglik, S. H. Dau, H. M. Kiah, and H. Wang, "k-server byzantine-resistant pir scheme with optimal download rate and optimal file size," in *2023 IEEE International Symposium on Information Theory (ISIT)*, 2023, pp. 1514–1519.

[23] R. Tajeddine, O. W. Gnilke, D. Karpuk, R. Freij-Hollanti, and C. Hollanti, "Private information retrieval from coded storage systems with colluding, byzantine, and unresponsive servers," *IEEE Transactions on Information Theory*, vol. 65, no. 6, pp. 3898–3906, 2019.

[24] S. Vithana, Z. Wang, and S. Ulukus, "Private information retrieval and its extensions: An introduction, open problems, future directions," *IEEE BITS the Information Theory Magazine*, 2023.

[25] I. Goldberg, "Improving the robustness of private information retrieval," in *2007 IEEE Symposium on Security and Privacy (SP)*, 2007, pp. 131–148.

[26] X. Yao, N. Liu, and W. Kang, "The capacity of multi-round private information retrieval from byzantine databases," in *2019 IEEE International Symposium on Information Theory (ISIT)*, 2019, pp. 2124–2128.

[27] G. Cormode, M. Mitzenmacher, and J. Thaler, "Practical verified computation with streaming interactive proofs," in *3rd Innovations in Theoretical Computer Science Conference (ITCS)*, 2012, pp. 90–112.

[28] J. Thaler, M. Roberts, M. Mitzenmacher, and H. Pfister, "Verifiable computation with massively parallel interactive proofs," in *4th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud)*, 2012.

[29] E. Kushilevitz and R. Ostrovsky, "Replication is not needed: Single database, computationally-private information retrieval," in *38th IEEE Symposium on Foundations of Computer Science (FOCS)*, 1997, pp. 364–373.

[30] R. Bitar and S. E. Rouayheb, "Staircase-PIR: Universally robust private information retrieval," in *2018 IEEE Information Theory Workshop (ITW)*, 2018, pp. 1–5.

[31] C.-C. Chen, K. M. Koh, and K. Khee-Meng, *Principles and techniques in combinatorics*. World Scientific, 1992.

[32] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan, "Private information retrieval," *Journal of the ACM*, vol. 45, no. 6, pp. 965–981, 1998.

[33] J. T. Schwartz, "Fast probabilistic algorithms for verification of polynomial identities," *Journal of the ACM*, vol. 27, no. 4, p. 701–717, 1980.

[34] R. Zippel, "Probabilistic algorithms for sparse polynomials," in *International Symposium on Symbolic and Algebraic Computation (ISSAC)*, 1979, pp. 216–226.

[35] K. Nazirkhanova, J. Neu, and D. Tse, "Information dispersal with provable retrievability for rollups," in *4th ACM Conference on Advances in Financial Technologies (AFT)*, 2022, pp. 180–197.

[36] A. Kate, G. M. Zaverucha, and I. Goldberg, "Constant-size commitments to polynomials and their applications," in *Advances in Cryptology (ASIACRYPT)*, 2010, pp. 177–194.

[37] M. Krohn, M. Freedman, and D. Mazieres, "On-the-fly verification of rateless erasure codes for efficient content distribution," in *2004 IEEE Symposium on Security and Privacy (SP)*, 2004, pp. 226–240.

[38] C. Gkantsidis and P. Rodriguez Rodriguez, "Cooperative security for network coding file distribution," in *25th IEEE International Conference on Computer Communications (INFOCOM)*, 2006, pp. 1–13.

[39] E. Boyle, N. Gilboa, and Y. Ishai, "Function secret sharing," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, 2015, pp. 337–367.

[40] E. Boyle, N. Chandran, N. Gilboa, D. Gupta, Y. Ishai, N. Kumar, and M. Rathee, "Function secret sharing for mixed-mode and fixed-point secure computation," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, 2021, pp. 871–900.

[41] W. M. Li and L. F. Zhang, "Towards efficient information-theoretic function secret sharing," *IEEE Access*, vol. 8, pp. 28 512–28 523, 2020.

[42] M. Mirmohseni and M. A. Maddah-Ali, "Private function retrieval," in *2018 Iran Workshop on Communication and Information Theory (IWCIT)*, 2018, pp. 1–6.

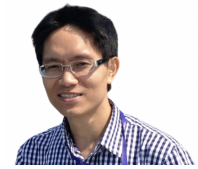
[43] H. Sun and S. A. Jafar, "The capacity of private computation," *IEEE Transactions on Information Theory*, vol. 65, no. 6, pp. 3880–3897, 2018.

[44] A. Gholami, K. Wan, H. Sun, M. Ji, and G. Caire, "On multi-message private computation," in *2024 IEEE International Symposium on Information Theory (ISIT)*, 2024, pp. 945–950.

[45] S. A. Obead, H.-Y. Lin, E. Rosnes, and J. Kliewer, "Capacity of private linear computation for coded databases," in *56th Annual Allerton*

Conference on Communication, Control, and Computing (Allerton), 2018, pp. 813–820.
[46] S. A. Obead and J. Kliewer, "Achievable rate of private function retrieval from mds coded databases," in *2018 IEEE International Symposium on Information Theory (ISIT)*, 2018, pp. 2117–2121.
[47] S. A. Obead, H.-Y. Lin, E. Rosnes, and J. Kliewer, "Private polynomial function computation for noncolluding coded databases," *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 1800–1813, 2022.
[48] D. Karpuk, "Private computation of systematically encoded data with colluding servers," in *2018 IEEE International Symposium on Information Theory (ISIT)*, 2018, pp. 2112–2116.
[49] N. Raviv and D. A. Karpuk, "Private polynomial computation from Lagrange encoding," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 553–563, 2020.
[50] J. Zhu, Q. Yan, X. Tang, and S. Li, "Symmetric private polynomial computation from Lagrange encoding," *IEEE Transactions on Information Theory*, vol. 68, no. 4, pp. 2704–2718, 2022.

Stanislav Kruglik (Senior Member, IEEE) received the bachelor's and master's degrees (Hons.) in applied mathematics and physics from the Moscow Institute of Physics and Technology (MIPT), Moscow, Russia, in 2015 and 2017, respectively, the master's degree (Hons.) in data science from the Skolkovo Institute of Science and Technology (Skoltech), Moscow, Russia, in 2017. He was awarded a Ph.D. degree in computer science from MIPT in February 2022. Since April 2022, he has been a Research Fellow at the School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore. His research interests include information theory and its applications, in particular to problems related to data storage and security. He was a recipient of the Simons Foundation Scholarship in 2015, the Russian President Scholarship in 2016, and the Potanin Foundation Scholarship in 2017.



Son Hoang Dau (Member, IEEE) received the B.S. degree in applied mathematics and informatics from Vietnam National University, Hanoi, Vietnam, in 2006, and the M.S. and Ph.D. degrees in mathematical sciences from Nanyang Technological University, Singapore, in 2009 and 2012, respectively. He worked as a research fellow at Singapore University of Technology and Design (8/2012-8/2015), University of Illinois at Urbana-Champaign (8/2015-8/2017), and Monash University (11/2017-1/2019), before joining RMIT University in 2019. He is currently a senior lecturer in Computer Science at School of Computing Technologies, STEM College, RMIT University. His research interests include coding theory, blockchain, and discrete mathematics.

Han Mao Kiah (Senior Member, IEEE) received the Ph.D. degree in mathematics from Nanyang Technological University (NTU), Singapore, in 2014. From 2014 to 2015, he was a Post-Doctoral Research Associate with the Coordinated Science Laboratory, University of Illinois at Urbana-Champaign. From 2015 to 2018, he was a Lecturer with the School of Physical and Mathematical Sciences (SPMS), NTU, where he is currently an Assistant Professor with SPMS. His research interests include DNA-based data storage, coding theory, enumerative combinatorics, and combinatorial design theory.

Huaxiong Wang received a Ph.D. in Mathematics from University of Haifa, Israel in 1996 and a Ph.D. in Computer Science from University of Wollongong, Australia in 2001. He has been with Nanyang Technological University in Singapore since 2006, where he is a Professor in the Division of Mathematical Sciences. Currently he is also the Co-Director of National Centre for Research in Digital Trust and the Deputy Director of Strategic Centre for Research in Privacy-Preserving Technologies and Systems at NTU. Prior to NTU, he held faculty positions at Macquarie University and University of Wollongong in Australia, and visiting positions at ENS de Lyon in France, City University of Hong Kong, National University of Singapore and Kobe University in Japan. His research interest is in cryptography and cybersecurity. He was the program co-chair of Asiacrypt 2020 and 2021.

Liang Feng Zhang received the Ph.D. degree in cryptography and information security from Nanyang Technological University, Singapore. Currently, he is an Associate Professor with the School of Information Science and Technology, ShanghaiTech University, Shanghai, China. His main research interests include cryptography and coding theory.