Verifiable Information-Theoretic Function Secret Sharing

Stanislav Kruglik[1], Son Hoang Dau[2], Han Mao Kiah[1], Huaxiong Wang[1], and Liang Feng Zhang[3]

[1]School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore (email: stanislav.kruglik, hmkiah, hxwang@ntu.edu.sg)
[2]School of Computing Technologies, STEM College, RMIT University, Melbourne, Australia (email: sonhoang.dau@rmit.edu.au)
[3]School of Information Science and Technology, ShanghaiTech University, Shanghai, China. (email: zhanglf@shanghaitech.edu.cn)

Abstract

A *function secret sharing* (FSS) (Boyle et al., Eurocrypt 2015) is a cryptographic primitive that enables additive secret sharing of functions from a given function family $\mathcal{F}$. FSS supports a wide range of cryptographic applications, including private information retrieval (PIR), anonymous messaging systems, private set intersection and more. Formally, given positive integers $r \geq 2$ and $t < r$, and a class $\mathcal{F}$ of functions $f : [n] \to \mathbb{G}$ for an Abelian group $\mathbb{G}$, an r-party t-private FSS scheme for $\mathcal{F}$ allows a dealer to split each $f \in \mathcal{F}$ into r function shares $f_1, \ldots, f_r$ among r servers. Shares have the property that $f = f_1 + \cdots + f_r$ and functions are indistinguishable for any coalition of up to t servers. FSS for point function $f_{\alpha_1,\ldots,\alpha_l,\beta_1,\ldots,\beta_l}$ for different α and $l < t$ that evaluates to β_i on input α_i for all $i \in [l]$ and to zero on all other inputs for $l = 1$ are known under the name of *distributed point functions* (DPF). FSS for special interval functions $f^{<}_{\alpha,\beta}$ that evaluate to β on inputs lesser than α and to zero on all other inputs are known under the name of distributed comparison functions (DCF).

Most existing FSS schemes are based on the existence of one-way functions or pseudo-random generators, and as a result, hiding of function f holds only against computationally bounded adversaries. Protocols employing them as building blocks are computationally secure. Several exceptions mostly focus on DPF for four, eight or $d(t+1)$ servers for positive integer d, and none of them provide verifiability.

In this paper, we propose DPF for $d(t+l-1)+1$ servers, where d is a positive integer, offering a better key size compared to the previously proposed DPF for $d(t+1)$ servers and DCF for $dt+1$ servers, also for positive integer d. We introduce their *verifiable* extension in which any set of servers holding t keys cannot persuade us to accept the wrong value of the function. This verifiability notion differs from existing verifiable FSS schemes in the sense that we verify not only the belonging of the function to class $\mathcal{F}$ but also the correctness of computation results. Our schemes provide a secret key size $O(n^{1/d} \cdot s\log(p))$ for DPF and $O(n^{1/d} \cdot s\log(p))$ for DCF, where p^s is the size of group $\mathbb{G}$.

1 Introduction

A secret-sharing scheme enables a dealer to divide a secret into multiple shares, ensuring that only authorized subsets of shares can reconstruct the secret. Unauthorized subsets, on the other hand, contain absolutely no information about the secret, thereby achieving information-theoretic security. The concept of secret sharing was independently introduced over 40 years ago in [Sha79, Bla79] and subsequently expanded upon by [ISN89, BJK97]. Since then, secret-sharing schemes have evolved into fundamental components in various distributed applications, including private information retrieval [CKGS98], secure multi-party computation [GBOW88], and Lagrange coded computing [YLR+19].

Motivated by distributed applications involving private access to large databases under communication constraints, such as multi-server private information retrieval (PIR), secure keyword search, and incremental secret sharing, the ordinary notion of secret sharing was extended to function secret sharing [BGI15]. In the latter, the function f is split into r succinctly described function shares $f_1, \dots, f_r$, such that any subset of t shares hides the function f and scheme supports the additive reconstruction of values of f at point x over some fixed Abelian group as $f(x) = \sum_i f_i(x)$. More concretely, each function share f_i is described by a short key k_i in such a way that, for the appropriate evaluation algorithm Eval_i, it holds that $\mathrm{Eval}_i(k_i, x) = f_i(x)$. The efficiency of function secret-sharing schemes can be measured by the total size of keys.

The majority of existing function secret sharing constructions are tailored for distributed point functions. Formally, for any $\alpha \in [n]$ and $\beta \in \mathbb{G}$, where $\mathbb{G}$ is some Abelian group, a *point function* $f_{\alpha,\beta}$ is defined as a function such that $f_{\alpha,\beta}(\alpha) = \beta$ and $f_{\alpha,\beta}(\alpha') = 0$ for $\alpha' \neq \alpha$. Its multi-point extension $f_{\alpha_1,\dots,\alpha_l,\beta_1,\dots,\beta_l}$ is a function such that $f_{\alpha_1,\dots,\alpha_l,\beta_1,\dots,\beta_l}(\alpha_i) = \beta_i$ and $f_{\alpha_1,\dots,\alpha_l,\beta_1,\dots,\beta_l}(\alpha') = 0$ for $\alpha' \neq \alpha_i$ for all $i \in [l]$. A computationally-secure function secret-sharing scheme based on the existence of pseudo-random generators with seed length λ for point functions with $l = 1$ was constructed in [BGI15]. For $t = 1$ and $r = 2$, the scheme has a key size $O(\lambda \log(n))$, while for $r \geq 3$ and $t = r - 1$, the key size is $O(\lambda \cdot 2^{r/2} \cdot \sqrt{n})$. Prior to this paper, such schemes under the name of *distributed point functions* were considered in [GI14]. The schemes were based on pseudo-random generators with seed length λ and had a key size $O(\lambda \cdot \log(n)^{\log_2 3})$. Later on, multiple improvements and extensions of $l = 1$ case, including function secret sharing based on one-way functions with reduced key size, were proposed in [BGI16]. Such schemes have a wide range of cryptographic applications, including PIR [CKGS98], anonymous messaging systems [CGBM15], private set intersection [DIL+22], pseudorandom correlation generators [BCGI18], and many others. Note that in most cases, extensions to $l > 1$ were obtained by concatenating schemes for $l = 1$.

Function secret sharing schemes can also be deployed for special interval functions. In this paper, we focus on the *comparison function* denoted as $f^{<}_{\alpha,\beta}$ for any $\alpha \in [n]$ and $\beta \in \mathbb{G}$, where $\mathbb{G}$ is some Abelian group. For any $\alpha' < \alpha$, $f^{<}_{\alpha,\beta}(\alpha') = \beta$, and $f^{<}_{\alpha,\beta}(\alpha') = 0$ otherwise. The function secret sharing schemes for these functions are known under the name of *distributed comparison function* (DCF) [BGI16]. The computationally secure construction, based on the existence of pseudo-random generators with seed length λ for $t = 1$ and $r = 2$ from [BGI16], has a key size $O(\log(n) \cdot (4\lambda + \log(n))$. Later on in [BCG+21], the key size was reduced to $O(\log(n) \cdot (\lambda + \log(n))$. Recently, non-asymptotic improvement of key size was attained in [GYW+23]. Such schemes serve as the main building blocks for arithmetic and logical shift gates in secure multi-party computation within the preprocessing model presented in [BGI19]. These gates play a crucial role in various applications associated with fixed-point arithmetic and machine learning. Function secret-sharing schemes for other types of functions, including hard-core predicates of one-way functions, were considered in [Kos18, OKK18].

To deploy such schemes in a real-world environment, it is essential to be secure against malicious adversaries. The existing line of research on verifiable function secret sharing mostly focuses on ensuring correctness, meaning that function being shared belong to a given class $\mathcal{F}$ [BGI16, BBCG+21, dCP22]. Some of these schemes leak information during the verification procedure [BGI16] while others are limited to point functions with $\beta = 1$ [BBCG$^+$21]. At the same time, all the above schemes focus on two-server schemes. However, in a multi-server environment, it is more natural for malicious servers to try to fabricate function shares that belong to the same function class but represent a different function than the one initially shared. If the resulting function still belongs to the same function class as before, such behavior can go undetected by the previously mentioned techniques.

As with many cryptographic notions, the privacy property of function secret sharing can be based either on computational hardness assumptions and be computationally private, or not rely on any assumptions and be information-theoretically private. Most of the previously known function secret sharing schemes are computationally private and, as mentioned before, focus on the two-server case. However, in modern distributed applications, a multi-server setup is more preferred [LA$^+$20]. Also, computationally private schemes may have limitations in specific applications, such as distributed key generation [DS17]. At the same time, information-theoretically private schemes may be desired or even required for other applications. These facts have created an interest in information-theoretically private function secret-sharing schemes.

The first information-theoretic function secret-sharing schemes were proposed in [LZLL20, LZ20]. Luo *et al.* in [LZLL20] introduced three explicit constructions of a function secret-sharing scheme for the point function $f_{\alpha,\beta}$. Two of them are based on non-linear reconstruction, while the remaining one is linear but non-additive. The paper [LZ20] focused on point functions $f_{\alpha,\beta}$ with $\beta = 1$ and their multi-point extensions by concatenating keys for single-point cases. The reconstruction algorithm of the proposed schemes is based on the underlying information-theoretically secure PIR scheme and, as a result, is non-additive.

Driven by the fact that the reconstruction process is non-additive and, consequently, may entail increased complexity and require larger server responses [FIKW22], the efficiency of the schemes in [LZLL20, LZ20] was evaluated based on communication complexity. Essentially, communication complexity can be defined as the total size of r secret keys and server responses to compute $f(\alpha)$ in the worst case. As a result, the scheme from [LZLL20] with linear reconstruction has a communication complexity of $O(s\log(p)\cdot n)$, while its non-linear improvement can reach an optimal communication complexity of $O(s\log(p)\cdot\log(n))$. For both schemes, $\mathbb{G} = \mathbb{Z}_{p^s}$. In comparison, schemes from [LZ20] for general $t \geq 1$ and $r > t$ can achieve a communication complexity of $O(n^{1/\lfloor(2r-1/t\rfloor})$ for $\mathbb{G} = \mathbb{Z}_2$.

Recently, Boyle et al. considered the information-theoretically private function secret-sharing schemes for general point functions $f_{\alpha,\beta}$ and additive reconstruction for the case of non-colluding servers in [BGIK22]. The authors proposed a four-server information-theoretically private scheme for $\mathbb{G} = \mathbb{Z}_{p^s}$ with key size $O(s\log(p)\cdot 2^{2p\sqrt{\log(n)\log\log(n)}})$, where $p \geq 3$ is a prime number and $s \geq 1$. The scheme is based on query conversion in two-server PIR with sub-polynomial communication and a non-linear retrieval algorithm from [DG16]. Later on, in [LKZ23], these results were extended to an eight-server scheme for $\mathbb{G} = \mathbb{Z}_p$ with key size $O(2^{10\sqrt{\log(n)\log\log(n)}} + \log(p))$ for any prime p and a four-server scheme for $\mathbb{G} = \mathbb{Z}_{p^s}$ with key size $O(s\log(p)\cdot 2^{c(p)\sqrt{\log(n)\log\log(n)}})$ in the non-colluding case for any prime p and $s \geq 1$ with $c(p) = 6$ for $p = 2$ and $c(p) = 2p$ for $p \geq 3$. The extension is based on query conversion in sub-polynomial PIR schemes [DG16, Efr09]. Applying their transformation to Woodruff-Yekhanin PIR from [WY05], the authors of [LKZ23] obtained $d(t+1)$-server t-private information-theoretically secure function secret-sharing schemes for general point functions $f_{\alpha,\beta}$ with $\mathbb{G} = \mathbb{Z}_p$ and key size $O(\log(p)\cdot n^{1/\lfloor(2d-1)/t\rfloor})$ for any prime number p and integer $d \geq 1$. However, the schemes proposed in [BGIK22, LKZ23] do not offer multi-point extension nor verifiability, while most of them can be implemented for *four* or *eight* servers only. At the same time, to the best of our knowledge, there are no information-theoretically secure distributed comparison functions.

1.1 Our contribution

We continue the line of research on information-theoretically secure function secret sharing for point functions, which was initiated in [BGIK22, LKZ23], and consider the general case of $l \geq 1$. Note that previous schemes can offer an extension to the general $l \geq 1$ only through concatenation of solutions for $l = 1$, which results in l times increase in key sizes.

Motivated by the distributed notion of function-secret sharing, we introduce a method to verify the correctness of the reconstructed function value even when servers respond with incorrect function evaluations. In other words, we ensure that the result we obtain is correct with a negligible probability of accepting an incorrect result under the discrete logarithm assumption in cyclic groups. In contrast, existing verifiable function secret sharing schemes only ensure that the result we obtain is a value of the function from the initial function class. These results can be formulated as the following two theorems.

Theorem 1 (**$(d(t+l-1)+1)$-server t-private FSS scheme for $f_{\alpha_1,\ldots,\alpha_l,\beta_1,\ldots,\beta_l}$**)**.** Let $p \geq 2$ be a prime number and $p^s > d(t+l-1)+l$ for integers $s \geq 1$ and $d \geq 1$. There exists an $d(t+l-1)+1$-server information-theoretically t-private function secret sharing scheme with additive reconstruction for point function $f_{\alpha_1,\ldots,\alpha_l,\beta_1,\ldots,\beta_l}$ with input $[n]$, output $\mathbb{Z}_{p^s}$ and communication cost $O(s\log(p) \cdot n^{1/d})$.

Theorem 2 (**$(d(t+l-1)+1)$-server t-private t-secure FSS scheme for $f_{\alpha_1,\ldots,\alpha_l,\beta_1,\ldots,\beta_l}$**)**.** Let $p \geq 2$ be a prime number and $p^s > d(t+l-1)+l$ for integers $s \geq 1$ and $d \geq 1$. There exists an $d(t+l-1)+1$-server information-theoretically t-private function secret sharing scheme with additive reconstruction for point function $f_{\alpha_1,\ldots,\alpha_l,\beta_1,\ldots,\beta_l}$. Moreover, the reconstructed value that passes the verification performed by the client is guaranteed to be correct with all but negligible probability under the Discrete Logarithm assumption, given the collusion of at most t malicious servers who may send incorrect values. This scheme has input $[n]$, output $\mathbb{Z}_{p^s}$ and communication cost $O(s\log(p) \cdot n^{1/d})$.

Function secret sharing schemes for point functions can provide a solution for private computation independently introduced in [MMA18, SJ18]. In a nutshell, in a private computation setup, there exist r distributed and non-colluding servers, each of which stores the same set of n independent datasets, and the client wants to compute a linear function of the datasets privately. By privacy here, we mean that any single server does not get any information about the identities of datasets employed and coefficients of the linear function. Later on, these results were generalized for the case of colluding servers in [ZYTL22, RK19]. However, the main focus of the abovementioned papers is download cost optimization, which makes sense only in the case of large databases. In this case, upload cost can be disregarded since it is negligible compared to download cost. In contrast, the scheme from Theorem 1 provides a solution independent of this assumption, while the scheme from Theorem 2 can also ensure the verification of obtained results.

For the convenience of the reader, let us provide a comparison table of proposed solutions for function secret sharing of $f_{\alpha_1,\ldots,\alpha_l,\beta_1,\ldots,\beta_l}$ with those that have already existed and are theoretically secure. Note that if authors have proposed a construction for $f_{\alpha,\beta}$ only, we transform it into a solution for $f_{\alpha_1,\ldots,\alpha_l,\beta_1,\ldots,\beta_l}$ by concatenating keys for each individual f_{α_i,β_i}. For schemes with non-additive reconstructions, we measure the total communication cost instead of key size, as discussed before.

Motivated by the applications of distributed comparison functions in machine learning and scientific computing, we have introduced their information-theoretic extension. Despite having been overlooked until now, this approach holds potential for scenarios where information-theoretic security is desired or required, while also potentially accelerating implementation through simpler constructions. In the same way as for function secret sharing schemes for point functions, we introduce a method to verify the

Scheme	r	t	**Key size / Communication cost**	$\mathbb{G}$	**Additive reconstruction**	**Verifiability**
[LZ20]	$> t$	≥ 1	$O(l \cdot n^{1/\lfloor(2r-1/t\rfloor})$	$\mathbb{Z}_2$	NO	NO
[LZLL20, Theorem 1]	$> t$	> 1	$O(l \cdot s\log(p) \cdot n)$	$\mathbb{Z}_{p^s}$	YES	NO
[LZLL20, Theorem 3]	$> t$	> 1	$O(l \cdot s\log(p)\log(n))$	$\mathbb{Z}_{p^s}$	NO	NO
[BGIK22]	4	1	$O(l \cdot s\log(p) \cdot 2^{2p\sqrt{\log(n)\log\log(n)}})$	$\mathbb{Z}_{p^s}$ $p \geq 3$	YES	NO
[LKZ23, Theorem 6]	4	1	$O(l \cdot s\log(p) \cdot 2^{c(p)\sqrt{\log(n)\log\log(n)}})$ $c(2) = 6$, $c(p) = 2p$ for $p \geq 3$	$\mathbb{Z}_{p^s}$	YES	NO
[LKZ23, Theorem 7]	$d(t+1)$ $d \geq 1$	≥ 1	$O(l \cdot \log(p) \cdot n^{1/\lfloor(2d-1)/t\rfloor})$	$\mathbb{Z}_p$	YES	NO
[LKZ23, Theorem 8]	8	1	$O(l \cdot (2^{10\sqrt{\log(n)\log\log(n)}} + \log(p)))$	$\mathbb{Z}_p$	YES	NO
Theorem 1	$d(t+l-1)+1$ $d \geq 1$	≥ 1	$O(s\log(p) \cdot n^{1/d})$	$\mathbb{Z}_{p^s}$	**YES**	NO
Theorem 2	$d(t+l-1)+1$ $d \geq 1$	≥ 1	$O(s\log(p) \cdot n^{1/d})$	$\mathbb{Z}_{p^s}$	**YES**	**YES**

Table 1: Comparison of existing information-theoretic function secret-sharing schemes for $f_{\alpha_1,\ldots,\alpha_l,\beta_1,\ldots,\beta_l}$. *As can be noticed, the solution from Theorem 1 can provide additive reconstruction together with a flexible number of servers. In comparison to the scheme with such properties from [LKZ23, Theorem 7], it gives a better key size at the price of a slight increase in the number of servers for the case of $l > 1$ and $t > 1$. At the same time, for $l = 1$, the number of servers is smaller. In comparison to other schemes with additive reconstruction, the key size in Theorem 1 is better for large l at the cost of an increase in the number of servers. Theorem 2 provides the only solution to check the correctness of the computation result. Importantly, the key size in it does not exponentially depend on the output group size, the large value of which is essential to ensure a low probability of incorrect function computation.*

correctness of the reconstructed value in the presence of a limited number of servers providing incorrect function evaluations under the Discrete Logarithm assumption. These results can be formulated as the following two theorems.

Theorem 3 ($(dt{+}1)$-server t-private FSS scheme for $f^{<}_{\alpha,\beta}$)**.** Let $p \geq 2$ be a prime number and $p^s > dt{+}1$ for integers $s \geq 1$ and $d \geq 1$. There exists an $dt + 1$-server information-theoretically t-private function secret sharing scheme with additive reconstruction for comparison function $f^{<}_{\alpha,\beta}$ with input $[n]$, output $\mathbb{Z}_{p^s}$ and communication cost $O(s\log(p) \cdot n^{1/d})$.

Theorem 4 (r-server $(dt+1)$-private t-secure FSS scheme for $f^{<}_{\alpha,\beta}$)**.** Let $p \geq 2$ be a prime number and $p^s > dt{+}1$ for integers $s \geq 1$ and $d \geq 1$. There exists an $dt{+}1$-server information-theoretically t-private function secret sharing scheme with additive reconstruction for comparison function $f^{<}_{\alpha,\beta}$. Moreover, the reconstructed value that passes the verification performed by the client is guaranteed to be correct with all but negligible probability under the Discrete Logarithm assumption, given the collusion of at most t malicious servers who may send incorrect values. This scheme has input $[n]$, output $\mathbb{Z}_{p^s}$ and communication cost $O(s\log(p) \cdot n^{1/d})$.

1.2 Overview of techniques

Our schemes are based on a related cryptographic primitive – information-theoretic PIR [CKGS98]. The latter allows the client to retrieve a single bit from an n-bit database by communicating with $r \geq 2$ servers, such that any coalition of up to t servers cannot get any information about the retrieval index. In fact, multi-server private information retrieval schemes served as the initial motivation for the construction of function secret sharing schemes for distributed point functions, as such schemes for point functions directly give an m-server PIR protocol. In this paper, following the approaches from [LZ20, BGIK22, LKZ23], we use this connection in inverse direction and build FSS schemes from PIR schemes.

The main idea behind the generic transformation from PIR to FSS for point function, as discussed in [LZ20], is to form keys for $f_{\alpha,1}$ as queries to retrieve the file with index α by the corresponding PIR scheme. In this case, to find $f_{\alpha,1}(\alpha')$, the client sends servers the vector of length n with a '1' in position α' and '0' in the remaining positions. Each server treats this vector as a database and performs an underlined PIR protocol with the key as a query. All results are sent back to the client, who performs the retrieval algorithm, the result of which corresponds to the required value of $f(\alpha')$. As a result, we obtain an FSS for the point function $f_{\alpha,1}$ with the same communication costs as the initial PIR. An important point to note is that the reconstruction algorithm of the PIR utilized in the proposed transformation must not rely on the secret parameter since the client and dealer are no longer the same entity. This can be overcome by a new transformation involving independent additive sharing of the secret parameter at the price of strict restrictions on the number of participating servers [BGIK22, LKZ23].

At the same time, schemes based on PIR without modifications thereof have limitations on the size of the output group. Additionally, the reconstruction algorithm can be linear but non-additive. To overcome this, we utilized a general polynomial-based PIR from [KDK+23] and modified queries within it to obtain $\beta \cdot x_\alpha$ in place of x_α in an ordinary PIR scheme by summing query responses. Here, β belongs to the additive group of the field over which the scheme is deployed. As a result, we obtained an information-theoretic function secret sharing scheme with additive reconstruction for $f_{\alpha,\beta}$. To transform it into a scheme for $f_{\alpha_1,\dots,\alpha_l,\beta_1,\dots,\beta_l}$ for $l \geq 1$, we form queries to get $\beta_1 \cdot x_{\alpha_1}, \dots, \beta_l \cdot x_{\alpha_l}$ simultaniously. This modification involves transitioning to ramp security, introduced for the first time in [BM85], and sharing queries using a ramp secret sharing scheme instead of an ordinary Shamir scheme. To ensure additive reconstruction, we multiply queries by multipliers determined by the inverse Vandermonde matrix, so that summing query responses yields the sum of queried values. The resulting scheme is formulated in Theorem 1.

To enable verifiability, we create a second set of queries to retrieve $\mathrm{v} \cdot \beta_1 \cdot x_{\alpha_1}, \dots, \mathrm{v} \cdot \beta_l \cdot x_{\alpha_l}$ for a random non-zero parameter v from the base field. The results of these queries are interlinked through v, serving as a means to verify the correctness of the protocol. However, the parameter v must be kept secret and this appears to contradict the requirement of FSS. To address this issue, we transition to computational guarantees of variability and utilize ω^{v}, where ω is the generator of some cyclic multiplicative group, as a publicly available verification key. Consequently, the correctness of the obtained results is verified through the equations $(\omega^{\mathrm{v}})^{(\beta_1 \cdot x_{\alpha_1} + \dots + \beta_l \cdot x_{\alpha_l})_\ell} = \omega^{\mathrm{v} \cdot (\beta_1 \cdot x_{\alpha_1} + \dots + \beta_l \cdot x_{\alpha_l})_\ell}$ for all individual components in corresponding s-dimensional vectors over the base field. The correctness of this verification proof and the the secrecy of the parameter v is rooted in the Discrete Logarithm problem. The corresponding scheme is formulated in Theorem 2.

To construct an FSS scheme for the comparison function, we also employ the polynomial-based PIR scheme from [KDK+23]. However, for now, each server stores the database $x = x_1 \cdots x_n$, where $x_i = \sum_{\ell=1}^{i} (\mathbf{e}_{\alpha'+1})_\ell$, with $\mathbf{e}_{\alpha'+1}$ being the vector of length n with '1' in position $\alpha'+1$ and '0' elsewhere. The subscript ℓ denotes the ℓ-th coordinate of the vector. Clearly, $x_\alpha = 1$ if $\alpha' < \alpha$ and $x_\alpha = 0$ otherwise. Afterward, we modify the queries of the initial PIR to obtain $\beta \cdot x_\alpha$ in place of x_α by summing query responses. As a result, we obtain an r-server information-theoretically t-private FSS scheme for the comparison function formulated in Theorem 3. To ensure verifiability, we check the value $\beta \cdot x_\alpha$ in the same way as we checked the value of $\beta_1 \cdot x_{\alpha_1} + \cdots + \beta_l \cdot x_{\alpha_l}$. The corresponding scheme is formulated in Theorem 4.

2 Preliminaries

Notation. We denote by $\mathbb{F}_{p^s}$ the finite field with p^s elements and by $\mathbb{Z}_{p^s}$ the corresponding Abelian group, where p is a prime number and $s \geq 1$ is the extension degree. By $\mathbb{F}^*_{p^s}$ we denote all non-zero elements of finite field $\mathbb{F}_{p^s}$. For a natural number n, we use the notation $[n] = \{1, \dots, n\}$.

Point function. Given a domain size n and an Abelian group $\mathbb{G}$, the *point function* $f_{\alpha_1,\dots,\alpha_l,\beta_1,\dots,\beta_l} : [n] \to \mathbb{G}$ defined by l distinct integers $\alpha_1, \dots, \alpha_l \in [n]$ and l group elements $\beta_1, \dots, \beta_l \in \mathbb{G}$ satisfies

$$f_{\alpha_1,\dots,\alpha_l,\beta_1,\dots,\beta_l}(x) = \begin{cases} \beta_i, & x = \alpha_i \ (\forall i \in [l]); \\ 0, & \text{otherwise.} \end{cases}$$

Comparison function. Given a domain size n and an Abelian group $\mathbb{G}$, the *comparison function* $f^<_{\alpha,\beta} : [n] \to \mathbb{G}$ defined by $\alpha \in [n]$ and $\beta \in \mathbb{G}$ satisfies

$$f^<_{\alpha,\beta}(x) = \begin{cases} \beta, & x < \alpha; \\ 0, & \text{otherwise.} \end{cases}$$

Vandermonde matrix For distinct real numbers $u_1, \dots, u_r$ the *Vandermonde matrix* is defined by

$$V = \begin{bmatrix} 1 & u_1 & \dots & u_1^{r-1} \\ 1 & u_2 & \dots & u_2^{r-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & u_r & \dots & u_r^{r-1} \end{bmatrix}.$$

The inverse of matrix V is defined by

$$V^{-1} = \begin{bmatrix} C_{1,1} & C_{1,2} & \dots & C_{1,r} \\ C_{2,1} & C_{2,2} & \dots & C_{2,r} \\ \vdots & \vdots & \ddots & \vdots \\ C_{r,1} & C_{r,2} & \dots & C_{r,r} \end{bmatrix},$$

where elements of V^{-1} are defined by (see, for example, [Raw19])

$$C_{i,j} = (-1)^{i+j} \frac{U_{r-i,j}}{\prod_{l<j}^{r}(u_j - u_l)} \quad \text{for} \quad U_{r-i,j} = U_{r-i}(u_1, \dots, u_{j-1}, u_{j+1}, \dots, u_r)$$
$$\text{and} \quad U_{r-i}(u_1, \dots, u_r) = \sum_{1 \leq l_1 < \dots < l_{r-i} \leq r} c_{l_1} c_{l_2} \dots c_{l_{r-i}}. \qquad (1)$$

2.1 Function secret sharing

In this section, we provide a formal description of information-theoretically secure function secret sharing (itFSS).

Definition 1 (**(r,t)-itFSS).** Let $\mathcal{F}$ be a family of efficiently computable functions and $r > t \geq 1$. A *t-private r-server information-theoretic FSS scheme* ((r,t)-itFSS) for $\mathcal{F}$ is a tuple $\Pi = (\mathtt{Gen}, \mathtt{Eval}_1, \dots, \mathtt{Eval}_r, \mathtt{Rec})$ of algorithms with the following syntax:

- $(\mathbf{k}^{(1)}, \dots, \mathbf{k}^{(r)}) \leftarrow \mathtt{Gen}(f, r, t)$: a randomized *key generation* algorithm executed by the dealer that takes a point function $f \in \mathcal{F}$ as input and outputs r secret keys $\mathbf{k}^{(1)}, \ldots, \mathbf{k}^{(r)}$.

- $\mathbf{S}^{(j)} \leftarrow \texttt{Eval}_j(\mathbf{k}^{(j)}, x)$: a deterministic *evaluation* algorithm for server j that takes the secret key $\mathbf{k}^{(j)}$ and a point $x \in [n]$ as input and outputs a share $\mathbf{S}^{(j)}$ of $f(x)$.
- $f(x) \leftarrow \texttt{Rec}(\mathbf{S}^{(1)}, \ldots, \mathbf{S}^{(r)})$: a deterministic *reconstruction* algorithm[1] executed by the client that takes $\mathbf{S}^{(1)}, \ldots, \mathbf{S}^{(r)}$ as input and outputs $f(x)$.

We require Π to satisfy the following requirements:

- **Correctness** For any $f \in \mathcal{F}$, any $x \in \text{Dom}(f)$, and any $(\mathbf{k}^{(1)}, \ldots, \mathbf{k}^{(r)}) \leftarrow \texttt{Gen}(f)$,

$$\Pr\left[\texttt{Rec}(\texttt{Eval}_1(\mathbf{k}^{(1)}, x), \ldots, \texttt{Eval}_r(\mathbf{k}^{(r)}, x)) = f(x)\right] = 1.$$

- t**-Privacy** For any subset $\mathcal{T} \subseteq [r]$ such that $|\mathcal{T}| \leq t$ and any adversary $\mathcal{A}$ that plays the roles of the honest-but-curious servers $\{\mathcal{S}_j : j \in \mathcal{T}\}$, consider the following standard indistinguishably experiment:
  - The adversary $\mathcal{A}$ chooses two functions $f_0, f_1 \in \mathcal{F}$ and gives them to the challenger;
  - The challenger generates a random bit b and r secret keys for the function f_b as $\texttt{Gen}(f_b) \rightarrow (\mathbf{k}^{(1)}, \ldots, \mathbf{k}^{(r)})$. It then gives the keys $\{\mathbf{k}^{(j)} : j \in \mathcal{T}\}$ to $\mathcal{A}$;
  - Given the keys $\{\mathbf{k}^{(j)} : j \in \mathcal{T}\}$, $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$ on the value of b.

  Taking the probabilities over the randomness of the challenger and adversary, $\Pr[b' = b] = 1/2$.

Remark 1. We note that in practical scenarios of function secret sharing schemes, the key generation algorithm $\texttt{Gen}$ is executed by the *dealer*, the evaluation algorithm $\texttt{Eval}_i$ is executed by *server* i, while the reconstruction algorithm $\texttt{Rec}$ is performed by the *client*. Moreover, there is no direct private communication link between the dealer and the client, allowing the dealer to upload f and its shares to the cloud servers and disappear, with no extra private communication channels to be established. Thus, $\texttt{Rec}$ cannot use any auxiliary information created by the dealer. In fact, in a typical setting of FSS, $\texttt{Rec}$ simly sums up the output shares to recover $f(x)$.

Remark 2. Definition 1 introduces an information-theoretically private function secret sharing scheme, and the values of the adversary's guess on b in the indistinguishability experiment are equiprobable. In other words, $|\Pr[b' \neq b] - 1/2| = 0$. However, most existing function secret sharing schemes are computationally private, and the equiprobability condition is replaced with $|\Pr[b' \neq b] - 1/2| \leq \texttt{negl}(\lambda)$ for a security parameter λ under some cryptographic assumption for a probabilistic polynomial-time adversary. In most cases, the assumptions involve the existence of one-way functions or pseudorandom generators.

In this paper, for itFSS schemes with additive reconstruction, we measure efficiency as the maximal length of the secret keys $\sum_{j=1}^{r} |\mathbf{k}^{(j)}|$, following their computationally secure counterparts.

2.2 Private information retrieval

Our FSS protocols rely on a t-private r-server information-theoretic PIR scheme, where a database with n elements is replicated among r servers.

[1]In case of additive reconstruction, that is the main focus of this paper, this is just a summation of input shares.

Definition 2 (**(r,t)-PIR**)**.** Let $r > t \geq 1$. A *t-private r-server information-theoretic PIR scheme* ((r,t)-PIR) is a tuple $P = (\texttt{QueriesGen}, \texttt{AnswerGen}_1, \ldots, \texttt{AnswerGen}_r, \texttt{Retrieve})$ of algorithms with the following syntax:

- $(\mathbf{q}_1, \ldots, \mathbf{q}_r, \text{aux}_R) \leftarrow \texttt{QueriesGen}(r, n, i, t)$: a randomized *query generation* algorithm executed by the client that takes the number r of servers, the size n of the database, and the index $i \in [n]$ of interest, and outputs r queries $\mathbf{q}_1, \ldots, \mathbf{q}_r$ and auxiliary information aux_R used in the retrieval.
- $\mathbf{a}_j \leftarrow \texttt{AnswerGen}_j(\mathbf{q}_j, \mathbf{x})$: a deterministic answer generation algorithm for server j that takes the query $\mathbf{q}_j$ and the database $\mathbf{x}$, and outputs the answer $\mathbf{a}_j$.
- $x_i \leftarrow \texttt{Retrieve}(r, n, \mathbf{a}_1, \ldots, \mathbf{a}_r, \text{aux}_R)$: is a deterministic retrieval algorithm executed by the client that takes the number of servers r, the number of elements in the database n, the answers to queries $\mathbf{a}_1, \ldots, \mathbf{a}_r$ and the auxiliary information aux_R, and outputs the expected database item x_i.

We require P to satisfy the following requirements:

- **Correctness** For any integers r, t, n, any database x, any index i, any queries $(\mathbf{q}_1, \ldots, \mathbf{q}_r, \text{aux}_R) \leftarrow \texttt{QueriesGen}(r, n, i)$, it holds that

$$\Pr\left[\texttt{Retrieve}(r, n, \texttt{AnswerGen}_1(\mathbf{q}_1, \mathbf{x}), \ldots, \texttt{AnswerGen}_r(\mathbf{q}_r, \mathbf{x})) = x_i\right] = 1.$$

- **t-Privacy** For any subset $\mathcal{T} \subseteq [r]$ such that $|\mathcal{T}| \leq t$ and any adversary $\mathcal{A}$ that plays the role of honest-but-curious servers $\{\mathcal{S}_j : j \in \mathcal{T}\}$, consider the following standard indistinguishably experiment:
  - The adversary $\mathcal{A}$ chooses two indices $i_0, i_1 \in [n]$ and gives them to the challenger;
  - The challenger generates a random bit b and generates r queries $\mathbf{q}_1, \ldots, \mathbf{q}_r$ by executing $\texttt{QueriesGen}(r, n, i_b)$. After that, it gives queries $\{\mathbf{q}_j : j \in \mathcal{T}\}$ to $\mathcal{A}$;
  - Given the queries $\{\mathbf{q}_j : j \in \mathcal{T}\}$, $\mathcal{A}$ outputs a guess bit b' on the value of b.

  Taking the probabilities over the randomness of the challenger and adversary, $\Pr[b' = b] = 1/2$.

Remark 3. We note that in practical scenarios of PIR, the queries generation algorithm `QueriesGen` and the reconstruction algorithm `Retrieve` are executed by the client, while the answer generation algorithm $\texttt{AnswerGen}_j$ is executed by server j.

Our techniques rely on the notion of polynomial PIR introduced for the first time in [CKGS98] and later generalized for general polynomials and reconstruction algorithms in [KDK+23]. More precisely, we represent the file retrieval process as the calculation of a low-degree multivariate polynomial F at a specific point z. On one hand, polynomial coefficients are unknown to the client. On the other hand, the evaluation point z is kept secret from any coalition of up to t servers. In fact, for given database $\mathbf{x} = x_1 \cdots x_n$ corresponding polynomial $F_{\mathbf{x}}$ of total degree d is formed such that $F_{\mathbf{x}}(\mathbf{E}(i)) = x_i$ for all i for specific injective index encoding function $\mathbf{E} : [n] \to \mathbb{F}_p^m$. Note that with specific choices of $F_{\mathbf{x}}$ and $\mathbf{E}$, we can recover the well-known PIR protocols from [CKGS98, Gol07, WY05]. However, since we transform PIR schemes to FSS schemes, generally, we do not have a direct communication link between the client and the dealer. As a result, the retrieval algorithm cannot employ auxiliary information aux_R, and consequently, cannot use partial derivatives, as done by Woodruff-Yekhanin [WY05], without additional transformations that result in strict restrictions on the number of participating servers. Taking these restrictions into account, we can formulate the polynomial-based r-server t-private PIR protocol, formulated as follows (see protocol Π_0). In this protocol, the client generates queries based on a query generation algorithm and sends them to servers. Each server, upon receiving queries, computes answers to them according to an answer generation algorithm and sends the results back to the client. The latter performs file retrieval based on received answers from servers according to a retrieval algorithm.

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

Public parameters

- r (number of servers), n (database size), t (max number of colluding servers), $m \in [n]$, $d \geq 1$.
- $\mathbb{F}_{p^s}$ (a finite field with p^s elements).
- $\mathbf{E} : [n] \to \mathbb{F}_{p^s}^m$ is an injective function satisfying $F_{\mathbf{x}}(\mathbf{E}(i)) = x_i$ for every $\mathbf{x} \in \mathbb{F}_{p^s}^n$ and every $i \in [n]$, where $F_{\mathbf{x}}(z_1, \ldots, z_m) \in \mathbb{F}_{p^s}[z_1, \ldots, z_m]$ is a multivariate polynomial of total degree d corresponding to $\mathbf{x}$. Note that while the client knows the construction of $F_{\mathbf{x}}$ from $\mathbf{x}$, the polynomial itself is only known to the servers, who store $\boldsymbol{x}$.

Queries generation for file x_i

- Client publicly chooses a set of r different non-zero elements $u_1, u_2, \ldots, u_r \in \mathbb{F}_{p^s}$ and assigns them to servers $1, 2, \ldots, r$, respectively.
- Client randomly generates $\mathbf{V}^{(j)} \in \mathbb{F}_{p^s}^m$ for $j \in [t]$ and these vectors kept secret from the servers.
- Set $\mathbf{c}(u) = \mathbf{E}(i) + \mathbf{V}^{(1)}u + \cdots + \mathbf{V}^{(t)}u^t \in (\mathbb{F}_{p^s}[x])^m$. Observe that $\mathbf{c}(u)$ is a curve of degree-t that belongs to $\mathbb{F}_{p^s}^m$. More importantly, the curve $\mathbf{c}(u)$ passes through the point $(0, \mathbf{E}(i))$.
- For $j \in [r]$, the client sends to server j the query $\mathbf{q}_j \triangleq \mathbf{c}(u_j)$.

Answer generation for server $j \in [r]$

- Server j computes $\mathbf{a}_j = F_{\mathbf{x}}(\mathbf{q}_j)$ and sends it back to the client

Retrieval algorithm

- Consider the polynomial $\phi(u)$ obtained from polynomial $F_{\mathbf{x}}$ after parameterization by curve $\mathbf{c}(u)$. In other words, set $\phi(u) = F_{\mathbf{x}}(\mathbf{c}(u))$.
- Interpolate $\phi(u)$ from the values $f(\mathbf{c}(u_j)) = \mathbf{a}_j = F_{\mathbf{x}}(\mathbf{q}_j)$, $j \in [r]$. The file of interest x_i is given by the value $\phi(0)$.

The efficiency of PIR protocols is mainly measured by the total communication cost, defined as the maximum size of queries and server answers required to retrieve a single database element, given by

$\sum_{j=1}^{r}(|\mathbf{q}_j| + |\mathbf{a}_j|)$. For the protocol Π_0 with $dt + 1 \leq r$, we obtain enough points for interpolation. Simultaneously, the upload cost is m symbols in $\mathbb{F}_{p^s}$ per server, while the download cost is 1 symbol in $\mathbb{F}_{p^s}$ per server, where the parameter m is defined by the specific scheme instance. Consequently, the total communication cost is $(m + 1)r$ symbols of $\mathbb{F}_{p^s}$.

2.3 Results verification

Schemes defined in previous subsections assume honest-but-curious servers. However, these assumptions cannot be fulfilled in cloud environments, and servers may provide incorrect answers, persuading us to accept incorrect results. This poses an interesting question: what can we do if servers provide wrong responses? This topic was heavily investigated within the PIR framework, and we can find several interpretations of it; see, e.g. the introduction section of [KDK+23] for an overview. Here, we focus on the simplest case when we can detect the presence of t-servers that try to persuade us to accept the incorrect result for FSS case. As mentioned before, we detect the fact that the final result is incorrect, which distinguishes us from the existing line of research on verifiable function secret sharing schemes where we check that the function we compute belongs to a certain class that mostly corresponds to a malicious dealer.

Following the existing line of research on multi-server verifiable schemes [ZW22], we define an r-server verifiable function-secret sharing scheme consisting of key generation, evaluation, and verification algorithms as follows. Since the dealer and the client do not have a direct communication link, the verification key generated by the dealer must be publicly available. As a result, we can ensure only computational verification based on certain cryptographic assumptions. Corresponding security property is defined through the notion of a probabilistic security experiment between a probabilistic polynomial-time (PPT) adversary that can control a certain number of servers and its challenger. We note that such a security experiment is a generalization of the single-server experiment from [PRV12] to a multi-server setup. Before we proceed further, let us denote the notion of a negligible function and the cryptographic assumption that we will further employ to ensure the verifiability of our schemes.

Definition 3 (Negligible function)**.** A function from $\mathbb{N}$ to $\mathbb{R}^+$ is negligible and denoted as $\texttt{negl}$ if for all $c > 0$ there exists a natural number λ_0 such that $\texttt{negl}(\lambda) < \frac{1}{\lambda^c}$ for all $\lambda > \lambda_0$.

Definition 4 (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 $\texttt{negl}(\lambda)$.

Definition 5. Let $\mathcal{F}$ be a family of efficiently computable functions and $r > t \geq 1$. A r-server verifiable t-information-theoretically private t-computationally secure function secret sharing scheme, or (r, t, t)-itFSS for short, with respect to the function family $\mathcal{F}$ is a tuple of algorithms $\Pi = (\texttt{Gen}, \texttt{Eval}_1, \ldots, \texttt{Eval}_r, \texttt{Ver})$ with the following syntax:

- $(\text{vk}, \mathbf{k}^{(1)}, \ldots, \mathbf{k}^{(r)}) \leftarrow \texttt{Gen}(f, r, t)$: is a randomized key generation algorithm executed by the *dealer* that takes point function $f \in \mathcal{F}$ and outputs r secret keys $\mathbf{k}^{(1)}, \ldots, \mathbf{k}^{(r)}$ alongside with a verification key vk.
- $\mathbf{S}^{(j)} \leftarrow \texttt{Eval}_j(\mathbf{k}^{(j)}, x)$: is a deterministic evaluation algorithm for *server* j that takes secret key $\mathbf{k}^{(j)}$ and point $x \in [n]$ and output $S^{(j)}$, an output share of $f(x)$.
- $\{f(x), \perp\} \leftarrow \texttt{Ver}(\text{vk}, \mathbf{S}^{(1)}, \ldots, \mathbf{S}^{(r)})$: is a deterministic verification algorithm performed by the client. More specifically, the algorithm takes as input the verification key vk and r shares

$\mathbf{S}^{(1)}, \ldots, \mathbf{S}^{(r)}$ and reconstructs $f(x)$ if possible or outputs the special symbol $\perp$ indicating that at least one of the answers is incorrect.

Moreover, Π must satisfy the following requirements:

- **Correctness** For any $f \in \mathcal{F}$ and any $x \in [n]$, if $(\mathbf{k}^{(1)}, \ldots, \mathbf{k}^{(r)}) \leftarrow \texttt{Gen}(f)$, then

$$\Pr\left[\texttt{Ver}(\texttt{Eval}_1(\mathbf{k}^{(1)}, x), \ldots, \texttt{Eval}_r(\mathbf{k}^{(r)}, x)) = f(x)\right] = 1.$$

- t**-Privacy** For any subset $\mathcal{T} \subseteq [r]$ such that $|\mathcal{T}| \leq t$ and any adversary $\mathcal{A}$ that plays the roles of honest-but-curious servers $\{\mathcal{S}_j : j \in \mathcal{T}\}$, the standard indistinguishably experiment can be defined as follows
  - The adversary $\mathcal{A}$ chooses two functions $f_0, f_1 \in \mathcal{F}$ and gives them to the challenger;
  - The challenger generates random bit b and r secret keys for function f_b as $\texttt{Gen}(f_b) \to (\mathbf{k}^{(1)}, \ldots, \mathbf{k}^{(r)})$. After, it gives keys $\{\mathbf{k}^{(j)}\}$ for $j \in \mathcal{T}$ to the adversary;
  - The adversary outputs his guess on the value of b based on keys available to him.

  Taking the probabilities over the randomness of the challenger and adversary, the probabilities of the adversary's guesses on the values of b must be equal to $1/2$.

- t**-Security** For any subset $\mathcal{T} = \{j_1, \ldots, j_{|\mathcal{T}|}\} \subseteq [r]$ such that $|\mathcal{T}| \leq t$ and any adversary $\mathcal{A}$ that controls dishonest servers $\{\mathcal{S}_j : j \in \mathcal{T}\}$, the interactive security experiment can be defined as follows
  - Challenger generates $(\text{vk}, \mathbf{k}^{(1)}, \ldots, \mathbf{k}^{(r)}) \leftarrow \texttt{Gen}(f)$ and sends k_j for $j \in \mathcal{T}$ together with x to the adversary $\mathcal{A}$.
  - The adversary generates crafted shares of $f(x)$ as $\hat{\mathbf{S}}^{(j_1)}, \ldots, \hat{\mathbf{S}}^{(j_{|\mathcal{T}|})} \leftarrow \mathcal{A}(\text{vk}, \mathbf{k}^{(j_1)}, \ldots, \mathbf{k}^{(j_{|\mathcal{T}|})}, x)$
  - Challenger computes $\mathbf{S}^{(j)} \leftarrow \texttt{Eval}_j(\mathbf{k}^{(j)}, x)$ for $j \in [r] \setminus \mathcal{T}$
  - Challenger runs the verification algorithm $\texttt{Ver}$ with inputs vk, $\mathbf{S}^{(j)}$ for $j \in [r] \setminus \mathcal{T}$ and $\hat{\mathbf{S}}^{(j)}$ for $j \in \mathcal{T}$ and outputs y

  The probability that $y \neq \{f(x), \perp\}$ over the randomness of the challenger and adversary must be upper bounded by negligible function from security parameter λ for any probabilistic poly-time adversary.

3 Constructions of Distributed Point Functions

In this section, we present a construction of information-theoretic function secret sharing for point functions for a general number of points l and generalize it to the verifiable case, in which we can verify the correctness of the recovered value of the shared function.

Let us start with the *non-verifiable* scheme formulated in Theorem 1. First, we formulate the scheme Π_1 below and formally prove the theorem later on. In this scheme, the dealer generates keys according to a key generation algorithm and sends them to servers. The client, who wants to evaluate the value of a secretly shared function at a specific point, sends it to the servers. With the point of interest from the client and the key from the dealer, the servers evaluate the function share according to an evaluation algorithm and send them back to the client. The latter computes the value of the function at the interested point utilizing a reconstruction algorithm.

$(d(t+l-1)+1)$*-server t-private FSS scheme* Π_1 *for* $f = f_{\alpha_1,\dots,\alpha_l,\beta_1,\dots,\beta_l}$

Public parameters

- n (database size), t (max number of colluding servers), $d \geq 1$, $l \geq 1$, $m \leq n$ so that $\binom{m}{d} \geq n$.
- $r = d(t+l-1)+1$ (number of servers).
- $\mathbb{F}_{p^s}$ is a finite field with p^s elements, and $p \geq 2^\lambda$ is a prime number.
- $u_1, \dots, u_r$ are r distinct elements from $\mathbb{F}_{p^s}^*$ chosen by the dealer and assigned to servers $1, \dots, r$, respectively.
- $\{\xi_1, \dots, \xi_l\}$ are l distinct elements from $\mathbb{F}_{p^s}$ that do not intersect with $\{u_1, \dots, u_r\}$, also chosen by the dealer.
- $\mathbf{E} : [n] \times \mathbb{F}_{p^s}^* \to \mathbb{F}_{p^s}^m$ is an injective function and $F_{\mathbf{x}}(z_1, \dots, z_m) \in \mathbb{F}_{p^s}[z_1, \dots, z_m]$ is a multivariate polynomial of total degree d corresponding to $\mathbf{x}$ satisfying $F_{\mathbf{x}}(\mathbf{E}(\alpha, \beta)) = \beta x_\alpha$ for every $\mathbf{x} \in \mathbb{F}_{p^s}^n$, $\alpha \in [n]$, and $\beta \in \mathbb{F}_{p^s}^*$. An example of $\mathbf{E}$ and $F_{\mathbf{x}}$ is given in Lemma 5.

Key generation for function $f_{\alpha_1,\dots,\alpha_l,\beta_1,\dots,\beta_l}$

- For reconstruction, dealer privately creates a random m-dimensional curve $\mathbf{c}(u) = \left(c_\ell(u)\right)_{\ell\in[m]}$, where $c_\ell(u) \in \mathbb{F}_{p^s}[u]$ and $\deg(c_\ell) = t+l-1$ for every $\ell \in [m]$, that passes through the points $(\xi_1, \mathbf{E}(\alpha_1, \beta_1)), \dots, (\xi_l, \mathbf{E}(\alpha_l, \beta_l))$.
- Dealer sets $\mathbf{k}^{(j)} \triangleq \mathbf{c}(u_j) \in \mathbb{F}_{p^s}^m$ and shares the key with server j, for $j \in [r]$.

Evaluation algorithm for server $j \in [r]$

- Upon receiving $\alpha' \in [n]$ from the client, server j converts α' to an indicator vector $\mathbf{e}_{\alpha'}$ of length n with a '1' at position α' and zeros elsewhere, and then computes its polynomial representation $F_{\mathbf{e}_{\alpha'}}$ by treating the vector $\mathbf{e}_{\alpha'}$ as a database.
- Server j computes the output share $\mathbf{S}^{(j)}$ of $f(\alpha')$ as

$$\mathbf{S}^{(j)} \triangleq \left(\sum_{i_1=1}^{r} \sum_{i_2=1}^{l} C_{i_1,j} \cdot \xi_{i_2}^{i_1-1}\right) \cdot F_{\mathbf{e}_{\alpha'}}(\mathbf{k}^{(j)}) \in \mathbb{F}_{p^s},$$

  where $C = (C_{i_1,j_1})_{i_1,j_1\in[r]}$ is the inverse of the Vandermonde matrix $V = (u_{i_1}^{j_1-1})_{i_1,j_1\in[r]}$, as given by (1).

Reconstruction algorithm

- The client sums r shares $\mathbf{S}^{(1)}, \dots, \mathbf{S}^{(r)}$ to obtain $f(\alpha')$.

Lemma 5. Let $\mathbf{E}(\alpha,\beta)\colon [n]\times\mathbb{F}_{p^s}\to\mathbb{F}_{p^s}^m$ be defined as follows. First, choose an arbitrary set B of n vectors in $\mathbb{F}_{p^s}^m$ that have exactly d ones and $m-d$ zeros (this is possible if $\binom{m}{d}\geq n$), and let $\tau\colon [n]\to B$ be a fixed bijection. Then $\mathbf{E}(\alpha,\beta)$ can be obtained by replacing a one in $\tau(\alpha)$ (e.g. the one with the largest index) by β. Finally, for $\boldsymbol{x}\in\mathbb{F}_{p^s}^n$, let $F_{\mathbf{x}}(z_1,\dots,z_m)=\sum_{i\in[n]}x_i\prod_{\ell=1}^m z_\ell^{\mathbf{E}(i,1)_\ell}$, where $\mathbf{E}(i,1)_\ell$ is the restriction of vector $\mathbf{E}(i,1)$ to the ℓ-th coordinate. Then, $F_{\mathbf{x}}$ has total degree d and moreover, for every $\alpha\in[n]$ and $\beta\in\mathbb{F}_{p^s}$, it holds that

$$F_{\mathbf{x}}(\mathbf{E}(\alpha,\beta))=\beta x_\alpha.$$

As a consequence, when $\boldsymbol{x}=\mathbf{e}_{\alpha'}$, we have that $F_{\mathbf{e}_{\alpha'}}=\prod_{\ell=1}^m z_\ell^{\mathbf{E}(\alpha',1)_\ell}$, which satisfies

$$F_{\mathbf{e}_{\alpha'}}(\mathbf{E}(\alpha,\beta))=\beta(\mathbf{e}_{\alpha'})_\alpha=\begin{cases}\beta, & \text{if } \alpha=\alpha',\\ 0, & \text{otherwise.}\end{cases}$$

Proof. As $\mathbf{E}(i,1)$ has exactly d ones and $m-d$ zeros for every $i\in[n]$, we obtain that $\deg(F_{\mathbf{x}})=d$. For $i\neq\alpha$, since $\tau(i)\neq\tau(\alpha)$, there is an $\ell'\in[m]$ satisfying $(\mathbf{E}(i,1))_{\ell'}\neq 0$ but $(\mathbf{E}(\alpha,\beta))_{\ell'}=0$, which implies that $\prod_{\ell=1}^m(\mathbf{E}(\alpha,\beta))_\ell^{\mathbf{E}(i,1)_\ell}=0$. On the other hand, when $i=\alpha$, let $\ell_\beta\in[m]$ be the largest index of a one in $\tau(i)=\tau(\alpha)$, we have $(\mathbf{E}(i,1))_\ell=(\mathbf{E}(\alpha,\beta))_\ell\in\{0,1\}$ for all $\ell\in[m]\setminus\{\ell_\beta\}$, whereas $(\mathbf{E}(i,1))_{\ell_\beta}=1$ and $(\mathbf{E}(\alpha,\beta))_{\ell_\beta}=\beta$. In this case, $\prod_{\ell=1}^m(\mathbf{E}(\alpha,\beta))_\ell^{\mathbf{E}(i,1)_\ell}=\beta$. Thus, $F_{\mathbf{x}}(\mathbf{E}(\alpha,\beta))=\beta x_\alpha$. The case when $\boldsymbol{x}=\mathbf{e}_{\alpha'}$ follows in a straightforward manner. $\square$

Example 1. This example illustrates a construction of the mapping $\mathbf{E}$ and the polynomial representation $F_{\mathbf{e}_{\alpha'}}$ used in Π_1 and Π_2. For a more general $F_{\mathbf{x}}$, please refer to Example 2. Let $n=6$, $m=4$, and $d=2$. We define the mapping $\mathbf{E}$ as follows: $\mathbf{E}(1,\beta)=(1,\beta,0,0)$, $\mathbf{E}(2,\beta)=(1,0,\beta,0)$, $\mathbf{E}(3,\beta)=(1,0,0,\beta)$, $\mathbf{E}(4,\beta)=(0,1,\beta,0)$, $\mathbf{E}(5,\beta)=(0,0,1,\beta)$, and $\mathbf{E}(6,\beta)=(0,1,0,\beta)$. For $\alpha'=1$, the polynomial representation $F_{\mathbf{e}_{\alpha'}}=F_{\mathbf{e}_1}$ of the indicator vector $\mathbf{e}_1$ of length 6 would be $F_{\mathbf{e}_1}(z_1,z_2,z_3,z_4)=z_1\cdot z_2$. Clearly, $F_{\mathbf{e}_1}(\mathbf{E}(1,\beta))=1\cdot\beta=\beta$ whereas $F_{\mathbf{e}_1}(\mathbf{E}(\alpha,\beta))=0$ for $\alpha\neq 1=\alpha'$.

Now we are ready to formally prove Theorem 1.

Proof. First, the key size of of Π_1 is m symbols of $\mathbb{F}_{p^s}$. Using $\mathbf{E}$ and F as in Lemma 5, we can get $m\sim(d!n)^{1/d}$. As a result, treating $d\in O(1)$ as a fixed constant, the key size is in $O(s\log(p)\cdot n^{1/d})$. We now proceed to prove the correctness and t-privacy properties as per Definition 1.

Proof of correctness. Let us denote

$$\phi(u)\triangleq F_{\mathbf{e}_{\alpha'}}(\boldsymbol{c}(u))=\prod_{\ell=1}^m\left(c_\ell(u)\right)^{\mathbf{E}(\alpha',1)_\ell}\in\mathbb{F}_{p^s}[u].$$

By their definitions, $\mathbf{E}(\alpha',1)$ has exactly d ones and $m-d$ zeros, and $\deg(c_\ell)=t+l-1$. Hence, $\deg(\phi)=d(t+l-1)=r-1$. Thus, we can write $\phi(u)=\sum_{i_1\in[r]}\phi_{i_1-1}u^{i_1-1}$. Moreover, $\phi(u_j)=F_{\mathbf{e}_{\alpha'}}(\boldsymbol{c}(u_j))=F_{\mathbf{e}_{\alpha'}}(\mathbf{k}^{(j)})$, $j\in[r]$. As a result, we can form the following system of equations:

$$\begin{bmatrix}1 & u_1 & \dots & u_1^{r-1}\\ 1 & u_2 & \dots & u_2^{r-1}\\ \vdots & \vdots & \ddots & \vdots\\ 1 & u_r & \dots & u_r^{r-1}\end{bmatrix}\cdot\begin{bmatrix}\phi_0\\ \phi_1\\ \vdots\\ \phi_{r-1}\end{bmatrix}=\begin{bmatrix}F_{\mathbf{e}'_\alpha}(\mathbf{k}^{(1)})\\ F_{\mathbf{e}'_\alpha}(\mathbf{k}^{(2)})\\ \vdots\\ F_{\mathbf{e}'_\alpha}(\mathbf{k}^{(r)})\end{bmatrix}, \tag{2}$$

which gives us the following formula for ϕ_{i_1-1}, $i_1 \in [r]$,

$$\phi_{i_1-1} = \sum_{j\in[r]} C_{i_1,j} F_{\mathbf{e}_{\alpha'}}(\mathbf{k}^{(j)}) \tag{3}$$

where $C = (C_{i_1,j_1})_{i_1,j_1\in[r]}$ is the inverse of the Vandermonde matrix $V = (u_{i_1}^{j_1-1})_{i_1,j_1\in[r]}$, as given by (1). Note that by the definition of $\boldsymbol{c}(\cdot)$ and Lemma 5, for $i_2 \in [l]$,

$$\phi(\xi_{i_2}) = F_{\mathbf{e}_{\alpha'}}(\boldsymbol{c}(\xi_{i_2})) = F_{\mathbf{e}_{\alpha'}}(\mathbf{E}(\alpha_{i_2}, \beta_{i_2})) = \beta_{i_2} \cdot (\mathbf{e}_{\alpha'})_{\alpha_{i_2}} = \begin{cases} \beta_{i_2}, \text{ if } \alpha' = \alpha_{i_2}, \\ 0, \text{ otherwise} . \end{cases}$$

By its definition, $f(\alpha') = f_{\alpha_1,\ldots,\alpha_l,\beta_1,\ldots,\beta_l}(\alpha') = \beta_{i_2}$ if $\alpha' = \alpha_{i_2}$, $i_2 \in [l]$, and 0 otherwise. Combining this with (3), we obtain

$$\begin{aligned} f(\alpha') = f_{\alpha_1,\ldots,\alpha_l,\beta_1,\ldots,\beta_l}(\alpha') &= \sum_{i_2\in[l]} \phi(\xi_{i_2}) = \sum_{i_2\in[l]} \sum_{i_1\in[r]} \phi_{i_1-1}\xi_{i_2}^{i_1-1} \\ &= \sum_{i_2\in[l]} \sum_{i_1\in[r]} \left(\sum_{j\in[r]} C_{i_1,j} F_{\mathbf{e}_{\alpha'}}(\mathbf{k}^{(j)}) \right) \xi_{i_2}^{i_1-1} \\ &= \sum_{j\in[r]} \left(\sum_{i_2\in[l]} \sum_{i_1\in[r]} C_{i_1,j}\xi_{i_2}^{i_1-1} \right) F_{\mathbf{e}_{\alpha'}}(\mathbf{k}^{(j)}) \\ &= \sum_{j\in[r]} \boldsymbol{S}^{(j)}, \end{aligned}$$

which means that the client can recover $f(\alpha')$ as the sum of the output shares $\boldsymbol{S}^{(j)}$, $j \in [r]$, as claimed.

Proof of privacy. Without loss of generality, let us assume that the adversary obtains $\mathbf{k}^{(1)}, \ldots, \mathbf{k}^{(t)}$, which are evaluations of a random m-dimensional curve of degree $t + l - 1$ at points $u_1, \ldots, u_t$. For an arbitrary set of values of $\mathbf{c}(\xi_1) = \mathbf{E}(\alpha_1, \beta_1)$, …, $\mathbf{c}(\xi_l) = \mathbf{E}(\alpha_l, \beta_l)$, which corresponds to a unique $f_{\alpha_1,\ldots,\alpha_l,\beta_1,\ldots,\beta_l}$, we have a set of $t+l$ evaluation points $\{u_j\}_{j\in[r]} \cup \{\xi_{i_2}\}_{i_2\in[l]}$ for $\boldsymbol{c}(\cdot)$. Consequently, there exists a unique curve $\mathbf{c}$ such that $\mathbf{c}(u_1) = \mathbf{k}^{(1)}$, …, $\mathbf{c}(u_t) = \mathbf{k}^{(t)}$ and $\mathbf{c}(\xi_1) = \mathbf{E}(\alpha_1, \beta_1)$, …, $\mathbf{c}(\xi_l) = \mathbf{E}(\alpha_l, \beta_l)$ (via Lagrange interpolation). As a result, in the standard indistinguishability experiment, for any pair of functions f_0 and f_1 in the form of $f_{\alpha_1,\ldots,\alpha_l,\beta_1,\ldots,\beta_l}$, we can obtain a unique m-dimensional curve to which queries available to the adversary belong. This fact leads to an equal probability of each function and concludes the proof. □

Let us construct the *verifiable* FSS scheme Π_2 stated in Theorem 2. In this scheme, dealer generates keys according to key generation algorithm and send them to servers. Client, who wants to evaluate the value of secretly shared function in specific point sends it to the servers. Having point of interest from client and function key from dealer, server evaluate function share according to evaluation algorithm and sends them back to the client. The latter computes the value of function in interested point and verify its correctness utilizing verification algorithm.

$(d(t + l - 1) + 1)$*-server t-private t-**secure** FSS scheme* Π_2 *for* $f_{\alpha_1,\ldots,\alpha_l,\beta_1,\ldots,\beta_l}$

Public parameters

- n (database size), t (max number of colluding servers), $d \geq 1$, $l \geq 1$, $m \leq n$ so that $\binom{m}{d} \geq n$.
- $r = d(t+l-1)+1$ (number of servers).
- $\mathbb{F}_{p^s}$ is a finite field with p^s elements, and $p \geq 2^\lambda$ is a prime number, and $\mathbb{G}$ is a cyclic multiplicative group of order p with a generator ω.
- $u_1, \ldots, u_r$ are r distinct elements from $\mathbb{F}_{p^s}^*$ chosen by the dealer and assigned to servers $1, \ldots, r$, respectively.
- $\{\xi_1, \ldots, \xi_l\}$ are l distinct elements from $\mathbb{F}_{p^s}$ that do not intersect with $\{u_1, \ldots, u_r\}$, also chosen by the dealer.
- $\mathbf{E} : [n] \times \mathbb{F}_{p^s}^* \to \mathbb{F}_{p^s}^m$ is an injective function and $F_{\mathbf{x}}(z_1, \ldots, z_m) \in \mathbb{F}_{p^s}[z_1, \ldots, z_m]$ is a multivariate polynomial of total degree d corresponding to $\mathbf{x}$ satisfying $F_{\mathbf{x}}(\mathbf{E}(\alpha, \beta)) = \beta x_\alpha$ for every $\mathbf{x} \in \mathbb{F}_{p^s}^n$, $\alpha \in [n]$, and $\beta \in \mathbb{F}_{p^s}^*$. An example of $\mathbf{E}$ and $F_{\mathbf{x}}$ is given in Lemma 5.

Key generation for function $f_{\alpha_1,\ldots,\alpha_l,\beta_1,\ldots,\beta_l}$

- For reconstruction, dealer privately creates a random m-dimensional curve $\mathbf{c}(u) = \big(c_\ell(u)\big)_{\ell \in [m]}$, where $c_\ell(u) \in \mathbb{F}_{p^s}[u]$ and $\deg(c_\ell) = t+l-1$ for every $\ell \in [m]$, that passes through the points $(\xi_1, \mathbf{E}(\alpha_1, \beta_1)), \ldots, (\xi_l, \mathbf{E}(\alpha_l, \beta_l))$.
- For verification, dealer randomly generates a *secret* $\mathrm{v} \in \mathbb{F}_p^*$ and privately creates a random m-dimensional curve $\mathbf{c}_{\mathrm{v}}(u) = \big(c_{\mathrm{v},\ell}(u)\big)_{\ell \in [m]}$, where $c_{\mathrm{v},\ell}(u) \in \mathbb{F}_{p^s}[u]$ and $\deg(c_{\mathrm{v},\ell}(u)) = t+l-1$ for all $\ell \in [m]$, that passes through the points $(\xi_1, \mathbf{E}(\alpha_1, \mathrm{v} \cdot \beta_1)), \ldots, (\xi_l, \mathbf{E}(\alpha_l, \mathrm{v} \cdot \beta_l))$.
- Dealer splits each key into two parts – one for reconstruction and one for verification: $\mathbf{k}^{(j)} = \big(\mathbf{k}_1^{(j)}, \mathbf{k}_2^{(j)}\big) \triangleq (\mathbf{c}(u_j), \mathbf{c}_{\mathrm{v}}(u_j))$ and shares the key with server j, for $j \in [r]$, while the verification key $\mathrm{vk} \triangleq \{\omega, \omega^{\mathrm{v}}\}$ is made public.

Evaluation algorithm for server $j \in [r]$

- Upon receiving $\alpha' \in [n]$ from the client, Server j converts α' to an indicator vector $\mathbf{e}_{\alpha'}$ of length n with a '1' at position α' and zeros elsewhere, and then computes its polynomial representation $F_{\mathbf{e}'_\alpha}$ by treating the vector $\mathbf{e}'_\alpha$ as a database (see Lemma 5).
- Compute the output share $\mathbf{S}^{(j)}$ of $f(\alpha')$ as

$$\mathbf{S}^{(j)} \triangleq \left(\left(\sum_{i_1=1}^{r} \sum_{i_2=1}^{l} C_{i_1,j} \cdot \xi_{i_2}^{i_1-1} \right) \cdot F_{\mathbf{e}_{\alpha'}}(\mathbf{k}_1^{(j)}); \left(\sum_{i_1=1}^{r} \sum_{i_2=1}^{l} C_{i_1,j} \cdot \xi_{i_2}^{i_1-1} \right) \cdot F_{\mathbf{e}_{\alpha'}}(\mathbf{k}_2^{(j)}) \right),$$

  where $C = (C_{i_1,j_1})_{i_1,j_1 \in [r]}$ is the inverse of the Vandermonde matrix $V = (u_{i_1}^{j_1-1})_{i_1,j_1 \in [r]}$, as given by (1).

Verification algorithm

- The client sums r shares $\mathbf{S}^{(1)}, \ldots, \mathbf{S}^{(r)}$ element-wise to get the values of $f(\alpha')$ and $f_{\mathrm{v}}(\alpha') = \mathrm{v} \cdot f(\alpha')$.

- To verify the correctness of $f(\alpha')$, the client represents the values of $f(\alpha')$ and $f_{\mathrm{v}}(\alpha')$ as vectors $(f(\alpha')_1, \ldots, f(\alpha')_s)$ and $(f_{\mathrm{v}}(\alpha')_1, \ldots, f_{\mathrm{v}}(\alpha')_s)$ over $\mathbb{F}_p$. For all $\ell \in [s]$, it uses the public verification key $(\omega, \omega^{\mathrm{v}})$ and verifies that $(\omega^{\mathrm{v}})^{f(\alpha')_\ell} = \omega^{f_{\mathrm{v}}(\alpha')_\ell}$. The client accepts $f(\alpha')$ if all s equations hold, and outputs $\perp$ otherwise.

Now we are ready to formally prove Theorem 2.

Proof. The key size of Π_2 is clearly equal to $2m$ symbols of $\mathbb{F}_{p^s}$. Using $\mathbf{E}$ and $F_{\mathbf{e}_{\alpha'}}$ as in Lemma 5, we can get $m \sim (d!n)^{1/d}$. As a result, treating $d \in O(1)$ as a fixed constant, the key size is in $O(s \log(p) \cdot n^{1/d})$. The proofs of correctness and t-privacy properties almost coincide with the proof of Theorem 1 and are omitted here. Let us formally prove the t-security property.

Without loss of generality, let us assume that the adversary controls the first t servers. Let $\mathbf{S}^{(j)} = (\mathbf{a}^{(j)}, \mathbf{b}^{(j)})$ for $j \in [r]$ be the shares obtained by correctly executing the key evaluation algorithm by each server. Let $\hat{\mathbf{S}}^{(j)} = (\hat{\mathbf{a}}^{(j)}, \hat{\mathbf{b}}^{(j)})$ be the values of shares chosen by $\mathcal{A}$ for servers $j \in [t]$. Note that by the definition of $\boldsymbol{c}_{\mathrm{v}}(\cdot)$ and Lemma 5, for $i_1 \in [l]$,

$$\phi_{\mathrm{v}}(\xi_{i_1}) = F_{\mathbf{e}_{\alpha'}}(\boldsymbol{c}_{\mathrm{v}}(\xi_{i_2})) = F_{\mathbf{e}_{\alpha'}}(\mathbf{E}(\alpha_{i_1}, \mathrm{v} \cdot \beta_{i_1}) = \mathrm{v} \cdot \beta_{i_1} \cdot (\mathbf{e}_{\alpha'})_{\alpha_{i_1}} = \begin{cases} \mathrm{v} \cdot \beta_{i_1}, \text{ if } \alpha' = \alpha_{i_1}, \\ 0, \text{ otherwise} . \end{cases}$$

By its definition, $f_{\mathrm{v}}(\alpha') = \mathrm{v} \cdot f_{\alpha_1,\ldots,\alpha_l,\beta_1,\ldots,\beta_l}(\alpha')) = \mathrm{v} \cdot \beta_{i_1}$ if $\alpha' = \alpha_{i_1}$, $i_1 \in [l]$, and 0 otherwise. By the same logic as proof of Theorem 1, we get

$$f(\alpha') = \sum_{j=1}^{r} \mathbf{a}^{(j)} = A \tag{4}$$

$$\mathrm{v} \cdot f(\alpha') = \sum_{j=1}^{r} \mathbf{b}^{(j)} = B \tag{5}$$

while

$$\hat{f}(\alpha') = \sum_{j=1}^{t} \hat{\mathbf{a}}^{(j)} + \sum_{j=t+1}^{r} \mathbf{a}^{(j)} = \hat{A}$$

and

$$\hat{\mathrm{v}} \cdot \hat{f}(\alpha') = \sum_{j=1}^{t} \hat{\mathbf{b}}^{(j)} + \sum_{j=t+1}^{r} \mathbf{b}^{(j)} = \hat{B}$$

The adversary $\mathcal{A}$ wins the security experiment if $\hat{f}(\alpha') \neq f(\alpha')$ and $\omega^{\hat{B}_\ell} = \omega^{\mathrm{v} \cdot \hat{A}_\ell}$ for all $\ell \in [s]$. Let us denote by $E_\ell^{(a)}$ the event that $\hat{f}(\alpha')_\ell \neq f(\alpha')_\ell$ and by $E_\ell^{(b)}$ the event that $\omega^{\hat{B}_\ell} = \omega^{\mathrm{v} \cdot \hat{A}_\ell}$. Clearly the event E that $\mathcal{A}$ wins the security experiment can be written as

$$E = \left(\cap_{\ell \in [s]} E_\ell^{(b)}\right) \cup \left(\cup_{\ell \in [s]} E_\ell^{(a)}\right)$$

and for the probability of E we have the following estimate

$$\begin{aligned}\Pr[E] &= \Pr\left[\left(\cap_{\ell\in[s]} E_\ell^{(b)}\right) \cup \left(\cup_{\ell\in[s]} E_\ell^{(a)}\right)\right] = \Pr\left[\cup_{\ell\in[s]}\left(\left(\cap_{\ell\in[s]} E_\ell^{(b)}\right) \cap E_\ell^{(a)}\right)\right] \\ &\le \sum_{\ell\in[s]} \Pr\left[\left(\cap_{\ell\in[s]} E_\ell^{(b)}\right) \cap E_\ell^{(a)}\right] \le \sum_{\ell\in[s]} \Pr[E_\ell^{(b)} \cap E_\ell^{(a)}]. \end{aligned} \tag{6}$$

From equations (4) and (5), and the fact that servers $j \in [r] \setminus [t]$ are honest, it is clear that the following equations hold:

$$(\hat{A} - A)_\ell = \sum_{j=1}^{t} (\hat{\mathbf{a}}^{(j)} - \mathbf{a}^{(j)})_\ell = \sum_{j=1}^{t} \Delta_\ell^{(j)} \neq 0$$

$$(\hat{B} - B)_\ell = \sum_{j=1}^{t} (\hat{\mathbf{b}}^{(j)} - \mathbf{b}^{(j)})_\ell = \sum_{j=1}^{t} \Xi_\ell^{(j)}$$

As $\omega^{B_\ell} = (\omega^{\mathrm{v}})^{A_\ell}$, for all $\ell \in [s]$ the event $E_\ell^{(b)} \cap E_\ell^{(a)}$ occurs if and only if $(\omega^{\mathrm{v}})^{\sum_{j=1}^t \Delta_\ell^{(j)}} = \omega^{\sum_{j=1}^t \Xi_\ell^{(j)}}$. From the description of the security experiment it follows that all $\Delta_\ell^{(j)}$ and $\Xi_\ell^{(j)}$ are known to $\mathcal{A}$ and independent from ω and ω^{v}. Hence, $\Pr(E_\ell^{(b)} \cap E_\ell^{(a)})$ can be bounded from above by the probability of learning the value $\mathrm{v} = \frac{\sum_{j=1}^t \Xi_\ell^{(j)}}{\sum_{j=1}^t \Delta_\ell^{(j)}}$, where $\sum_{j=1}^t \Delta_\ell^{(j)} \neq 0$, from discrete logarithm relationship in the group $\mathbb{G}$. Note that learning the value v from ω^{v} is equivalent to solving the same discrete logarithm problem. As a result, employing the equation (6), we get that the probability of adversary success within the security experiment is upper-bounded by a negligible function, and the theorem statement follows. □

4 Constructions of Distributed Comparison Functions

In this section, we give a construction of information-theoretic function secret sharing for *comparison functions* (also known as *distributed comparison functions*) and develop its verifiable version, in which the client can check the correctness of the recovered value of the shared function.

Let us start with the *non-verifiable* scheme Π_3 formulated in Theorem 3. First, we formulate the schemes below and formally prove the theorem later on. In this scheme, the dealer generates keys according to a key generation algorithm and sends them to servers. The client, who wants to evaluate the value of a secretly shared function at a specific point, sends it to the servers. With the point of interest from the client and the function key from the dealer, the servers evaluate the function share according to an evaluation algorithm and send them back to the client. The latter computes the value of the function at the interested point utilizing a reconstruction algorithm.

$(dt+1)$-server t-private FSS scheme Π_3 for $f^{<}_{\alpha,\beta}$

Public parameters

- n (database size), t (max number of colluding servers), $d \ge 1$, $m \le n$ so that $\binom{m}{d} \ge n$.

- $r = dt + 1$ (number of servers).
- $\mathbb{F}_{p^s}$ is a finite field with p^s elements, and $p \geq 2^\lambda$ is a prime number.
- $u_1, \ldots, u_r$ are r distinct elements from $\mathbb{F}_{p^s}^*$ chosen by the dealer and assigned to servers $1, \ldots, r$, respectively.
- $\mathbf{E} : [n] \times \mathbb{F}_{p^s}^* \to \mathbb{F}_{p^s}^m$ is an injective function and $F_{\mathbf{x}}(z_1, \ldots, z_m) \in \mathbb{F}_{p^s}[z_1, \ldots, z_m]$ is a multivariate polynomial of total degree d corresponding to $\mathbf{x}$ satisfying $F_{\mathbf{x}}(\mathbf{E}(\alpha, \beta)) = \beta x_\alpha$ for every $\mathbf{x} \in \mathbb{F}_{p^s}^n$, $\alpha \in [n]$, and $\beta \in \mathbb{F}_{p^s}^*$. An example of $\mathbf{E}$ and $F_{\mathbf{x}}$ is given in Lemma 5.

Key generation for function $f_{\alpha,\beta}^{<}$

- For reconstruction, dealer privately creates a random m-dimensional curve $\mathbf{c}(u) = \big(c_\ell(u)\big)_{\ell \in [m]}$, where $c_\ell(u) \in \mathbb{F}_{p^s}[u]$ and $\deg(c_\ell) = t$ for every $\ell \in [m]$, that passes through the point $(0, \mathbf{E}(\alpha, \beta))$.
- Dealer sets $\mathbf{k}^{(j)} \triangleq \mathbf{c}(u_j) \in \mathbb{F}_{p^s}^m$ and shares the key with server j, for $j \in [r]$.

Evaluation algorithm for server $j \in [r]$

- Upon receiving $\alpha' \in [n]$ from the client, Server j converts α' to an indicator vector $\mathbf{e}_{\alpha'+1}$ of length n with a '1' at position $\alpha'+1$ and zeros elsewhere. It then computes the polynomial representation $F_{\mathbf{x}}$ of database $\mathbf{x} = x_1 \cdots x_n$ with $x_i \triangleq \sum_{\ell=1}^{i} (\mathbf{e}_{\alpha'+1})_\ell$ for all $i \in [n]$. It is clear that $x_\alpha = 1$ if $\alpha' < \alpha$ and 0 otherwise.
- Compute the output share $\mathbf{S}^{(j)}$ of $f(\alpha')$ as $\mathbf{S}^{(j)} \triangleq C_{1,j} F_{\mathbf{x}}(\mathbf{k}^{(j)})$, where $C = (C_{i,j})_{i,j \in [r]}$ is the inverse of the Vandermonde matrix $V = (u_i^{j-1})_{i,j \in [r]}$, as given by (1).

Reconstruction algorithm

- The client sums r shares $\mathbf{S}^{(1)}, \ldots, \mathbf{S}^{(r)}$ to get the value of $f(\alpha')$.

Example 2. This example illustrates a construction of the mapping $\mathbf{E}$ and the polynomial representation $F_{\mathbf{x}}$ used in Π_3 and Π_4 (Section 4). We choose the same $n = 6$, $m = 4$, and $d = 2$, as well as the same mapping $\mathbf{E} : [n] \times \mathbb{F}_{p^s}^* \to \mathbb{F}_{p^s}^m$ as in Example 1. Let $\alpha' = 2$, then $\mathbf{e}_{\alpha'+1} = \mathbf{e}_3 = (0, 0, 1, 0, 0, 0)$ and $\mathbf{x} = (0, 0, 1, 1, 1, 1)$. In this case, the polynomial representation $F_{\mathbf{x}}$ takes the following form: $F_{\mathbf{x}}(z_1, z_2, z_3, z_4) = z_1 z_4 + z_2 z_3 + z_3 z_4 + z_2 z_4$.

Now we are ready to formally prove Theorem 3.

Proof. The statement on the key size can be explained in exactly the same way as in the previous theorems. Let us formally prove correctness and t-privacy properties as per Definition 1. Let

$$\phi(u) \triangleq F_{\mathbf{x}}(\boldsymbol{c}(u)) = \sum_{i \in [n]} x_i \prod_{\ell=1}^{m} \big(c_\ell(u)\big)^{\mathbf{E}(i,1)_\ell} \in \mathbb{F}_{p^s}[u].$$

By their definitions, $\mathbf{E}(i,1)$ has exactly d ones and $m-d$ zeros, and $\deg(c_\ell)=t$. Hence, $\deg(\phi)=dt=r-1$. Thus, we can write $\phi(u)=\sum_{i_1\in[r]}\phi_{i_1-1}u^{i_1-1}$. Moreover, $\phi(u_j)=F_{\mathbf{x}}(\boldsymbol{c}(u_j))=F_{\mathbf{x}}(\mathbf{k}^{(j)})$, $j\in[r]$. As a result, we can form the following system of equations:

$$\begin{bmatrix} 1 & u_1 & \dots & u_1^{r-1} \\ 1 & u_2 & \dots & u_2^{r-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & u_r & \dots & u_r^{r-1} \end{bmatrix} \cdot \begin{bmatrix} \phi_0 \\ \phi_1 \\ \vdots \\ \phi_{r-1} \end{bmatrix} = \begin{bmatrix} F_{\mathbf{x}}(\mathbf{k}^{(1)}) \\ F_{\mathbf{x}}(\mathbf{k}^{(2)}) \\ \vdots \\ F_{\mathbf{x}}(\mathbf{k}^{(r)}) \end{bmatrix}, \tag{7}$$

which gives us $\phi_0=\sum_{j\in[r]}C_{1,j}F_{\mathbf{x}}(\mathbf{k}^{(j)})$, where $C=(C_{i,j})_{i,j\in[r]}$ is the inverse of the Vandermonde matrix $V=(u_i^{j-1})_{i,j\in[r]}$, as given by (1). Note that due to its definition, $x_\alpha=1$ if $\alpha'<\alpha$ and 0 otherwise. Therefore, as $\boldsymbol{c}(\cdot)$ passes through $(0,\mathbf{E}(\alpha,\beta))$, using Lemma 5, we have

$$\phi_0=\phi(0)=F_{\mathbf{x}}(\boldsymbol{c}(0))=F_{\mathbf{x}}(\mathbf{E}(\alpha,\beta))=\beta x_\alpha=\begin{cases}\beta, & \text{if } \alpha'<\alpha,\\ 0, & \text{otherwise.}\end{cases}$$

Thus,

$$f(\alpha')=f^{<}_{\alpha,\beta}(\alpha')=\phi_0=\sum_{j\in[r]}C_{1,j}F_{\mathbf{x}}(\mathbf{k}^{(j)})=\mathbf{S}^{(1)}+\cdots+\mathbf{S}^{(r)},$$

which means that $f(\alpha')=f^{<}_{\alpha,\beta}(\alpha')$ can be reconstructed by summing up the output shares $\mathbf{S}^{(r)}$, $j\in[r]$, as claimed. □

We now proceed to construct a *verifiable* FSS Π_4 for comparison functions as formulated earlier in Theorem 4. In this scheme, the dealer generates keys according to a key generation algorithm and sends them to servers. The client, who wants to evaluate the value of a secretly shared function at a specific point, sends it to the servers. With the point of interest from the client and the function key from the dealer, the servers evaluate the function share according to an evaluation algorithm and send them back to the client. The latter computes the value of the function at the interested point and verify its correctness utilizing a verification algorithm.

$(dt+1)$-server t-private t-secure FSS scheme Π_4 for $f^{<}_{\alpha,\beta}$

Public parameters

- n (database size), t (max number of colluding servers), $d\geq 1$, $m\leq n$ so that $\binom{m}{d}\geq n$.
- $r=dt+1$ (number of servers).
- $\mathbb{F}_{p^s}$ is a finite field with p^s elements, where $p\geq 2^\lambda$ is a prime number. A cyclic multiplicative group $\mathbb{G}$ of order $p\geq 2^\lambda$ with generator ω is also chosen and made public by the dealer.
- $u_1,\dots,u_r$ are r distinct elements from $\mathbb{F}^*_{p^s}$ chosen by the dealer and assigned to servers $1,\dots,r$, respectively.
- $\mathbf{E}:[n]\times\mathbb{F}^*_{p^s}\to\mathbb{F}^m_{p^s}$ is an injective function and $F_{\mathbf{x}}(z_1,\dots,z_m)\in\mathbb{F}_{p^s}[z_1,\dots,z_m]$ is a multivariate polynomial of total degree d corresponding to $\mathbf{x}$ satisfying $F_{\mathbf{x}}(\mathbf{E}(\alpha,\beta))=\beta x_\alpha$ for every $\mathbf{x}\in\mathbb{F}^n_{p^s}$, $\alpha\in[n]$, and $\beta\in\mathbb{F}^*_{p^s}$. An example of $\mathbf{E}$ and $F_{\mathbf{x}}$ is given in Lemma 5.

Key generation for function $f^{<}_{\alpha,\beta}$

- For reconstruction, we follow scheme Π_3. That is, the dealer generates a random m-dimensional curve $\mathbf{c}(u)$ of degree t that passes through the point $(0, \mathbf{E}(i, \beta))$.
- For verification, the dealer randomly generates $\mathrm{v} \in \mathbb{F}_p^*$ and another m-dimensional curve $\mathbf{c}_\mathrm{v}(u)$ of degree t that passes through the point $(0, \mathbf{E}(i, \mathrm{v} \cdot \beta))$.
- Dealer splits each key into two parts – one for reconstruction and one for verification: $\mathbf{k}^{(j)} = (\mathbf{k}_1^{(j)}, \mathbf{k}_2^{(j)}) \triangleq (\mathbf{c}(u_j), \mathbf{c}_\mathrm{v}(u_j))$ for $j \in [r]$. Moreover, $\mathrm{vk} = \{\omega, \omega^\mathrm{v}\}$ is made public.

Evaluation algorithm for server $j \in [r]$

- Upon receiving $\alpha' \in [n]$ from the client, Server j converts α' to an indicator vector $\mathbf{e}_{\alpha'+1}$ of length n with a '1' at position $\alpha'+1$ and zeros elsewhere. It then computes the polynomial representation $F_\mathbf{x}$ of database $\mathbf{x} = x_1 \cdots x_n$ with $x_i \triangleq \sum_{\ell=1}^{i} (\mathbf{e}_{\alpha'+1})_\ell$ for all $i \in [n]$.
- Compute the output share $\mathbf{S}^{(j)}$ of $f(\alpha')$ as

$$\mathbf{S}^{(j)} = (C_{1,j} F_\mathbf{x}(\mathbf{k}_1^{(j)}), C_{1,j} F_\mathbf{x}(\mathbf{k}_2^{(j)})),$$

  where $C = (C_{i,j})_{i,j \in [r]}$ is the inverse of the Vandermonde matrix $V = (u_i^{j-1})_{i,j \in [r]}$, as given by (1).

Verification algorithm

- The client sums r shares $\mathbf{S}^{(1)}, \ldots, \mathbf{S}^{(r)}$ element-wise to get $f(\alpha')$ and $f_\mathrm{v}(\alpha') = \mathrm{v} \cdot f(\alpha')$.
- To verify the correctness of $f(\alpha')$, the client represents the values of $f(\alpha')$ and $f_\mathrm{v}(\alpha')$ as vectors $(f(\alpha')_1, \ldots, f(\alpha')_s)$ and $(f_\mathrm{v}(\alpha')_1, \ldots, f_\mathrm{v}(\alpha')_s)$ over $\mathbb{F}_p$. For all $\ell \in [s]$, it checks that $(\omega^\mathrm{v})^{f(\alpha')_\ell} = \omega^{f_\mathrm{v}(\alpha')_\ell}$ and accepts $f(\alpha')$ if all s equations hold and outputs $\perp$ otherwise.

Proof. The statement on the key size can be argued in exactly the same way as the previous theorems. The proofs of correctness and t-privacy properties almost coincide with the proof of Theorem 3 and are omitted here. The proof of t-security coincides with that of t-security within Theorem 4 with $l = 1$. □

5 Conclusion

We initiated the study of information-theoretic function secret sharing for general point functions and comparison functions as well as propose their verifiable extensions. We leave several open problems for extending our results:

- Is it possible to build our verification schemes on top of distributed point function schemes with key size $n^{o(1)}$?
- Is it possible to extend distributed point function schemes with key size $n^{o(1)}$ to multi-point case?
- Is it possible to extend our constructions to other functions, for instance hard-core predicates of one-way functions?

References

[BBCG$^+$21] Dan Boneh, Elette Boyle, Henry Corrigan-Gibbs, Niv Gilboa, and Yuval Ishai. Lightweight techniques for private heavy hitters. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 762–776. IEEE, 2021.

[BCG$^+$21] Elette Boyle, Nishanth Chandran, Niv Gilboa, Divya Gupta, Yuval Ishai, Nishant Kumar, and Mayank Rathee. Function secret sharing for mixed-mode and fixed-point secure computation. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 871–900. Springer, 2021.

[BCGI18] Elette Boyle, Geoffroy Couteau, Niv Gilboa, and Yuval Ishai. Compressing vector ole. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 896–912, 2018.

[BGI15] Elette Boyle, Niv Gilboa, and Yuval Ishai. Function secret sharing. In *Annual international conference on the theory and applications of cryptographic techniques*, pages 337–367. Springer, 2015.

[BGI16] Elette Boyle, Niv Gilboa, and Yuval Ishai. Function secret sharing: Improvements and extensions. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1292–1303, 2016.

[BGI19] Elette Boyle, Niv Gilboa, and Yuval Ishai. Secure computation with preprocessing via function secret sharing. In *Theory of Cryptography: 17th International Conference, TCC 2019, Nuremberg, Germany, December 1–5, 2019, Proceedings, Part I 17*, pages 341–371. Springer, 2019.

[BGIK22] Elette Boyle, Niv Gilboa, Yuval Ishai, and Victor I Kolobov. Information-theoretic distributed point functions. In *3rd Conference on Information-Theoretic Cryptography (ITC 2022)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2022.

[BJK97] George Robert Blakley Jr and Grigorii Anatol'evich Kabatiansky. Generalized ideal secret-sharing schemes and matroids. *Problemy Peredachi Informatsii*, 33(3):102–110, 1997.

[Bla79] George Robert Blakley. Safeguarding cryptographic keys. In *Managing Requirements Knowledge, International Workshop on*, pages 313–313. IEEE Computer Society, 1979.

[BM85] George Robert Blakley and Catherine Meadows. Security of ramp schemes. In *Advances in Cryptology: Proceedings of CRYPTO 84 4*, pages 242–268. Springer, 1985.

[CGBM15] Henry Corrigan-Gibbs, Dan Boneh, and David Mazières. Riposte: An anonymous messaging system handling millions of users. In *2015 IEEE Symposium on Security and Privacy*, pages 321–338. IEEE, 2015.

[CKGS98] Benny Chor, Eyal Kushilevitz, Oded Goldreich, and Madhu Sudan. Private information retrieval. *Journal of the ACM (JACM)*, 45(6):965–981, 1998.

[dCP22] Leo de Castro and Anitgoni Polychroniadou. Lightweight, maliciously secure verifiable function secret sharing. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 150–179. Springer, 2022.

[DG16] Zeev Dvir and Sivakanth Gopi. 2-server pir with subpolynomial communication. *Journal of the ACM (JACM)*, 63(4):1–15, 2016.

[DIL$^+$22] Samuel Dittmer, Yuval Ishai, Steve Lu, Rafail Ostrovsky, Mohamed Elsabagh, Nikolaos Kiourtis, Brian Schulte, and Angelos Stavrou. Streaming and unbalanced psi from function secret sharing. In *International Conference on Security and Cryptography for Networks*, pages 564–587. Springer, 2022.

[DS17] Jack Doerner and Abhi Shelat. Scaling oram for secure computation. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 523–535, 2017.

[Efr09] Klim Efremenko. 3-query locally decodable codes of subexponential length. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 39–44, 2009.

[FIKW22] Ingerid Fosli, Yuval Ishai, Victor I Kolobov, and Mary Wootters. On the download rate of homomorphic secret sharing. In *13th Innovations in Theoretical Computer Science Conference (ITCS 2022)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2022.

[GBOW88] S Goldwasser, M Ben-Or, and A Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computing. In *Proc. of the 20th STOC*, pages 1–10, 1988.

[GI14] Niv Gilboa and Yuval Ishai. Distributed point functions and their applications. In *Advances in Cryptology–EUROCRYPT 2014: 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings 33*, pages 640–658. Springer, 2014.

[Gol07] Ian Goldberg. Improving the robustness of private information retrieval. In *2007 IEEE Symposium on Security and Privacy (SP'07)*, pages 131–148. IEEE, 2007.

[GYW$^+$23] Xiaojie Guo, Kang Yang, Xiao Wang, Wenhao Zhang, Xiang Xie, Jiang Zhang, and Zheli Liu. Half-tree: Halving the cost of tree expansion in cot and dpf. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 330–362. Springer, 2023.

[ISN89] Mitsuru Ito, Akira Saito, and Takao Nishizeki. Secret sharing scheme realizing general access structure. *Electronics and Communications in Japan (Part III: Fundamental Electronic Science)*, 72(9):56–64, 1989.

[KDK$^+$23] Stanislav Kruglik, Son Hoang Dau, Han Mao Kiah, Huaxiong Wang, and Liang Feng Zhang. Querying twice to achieve information-theoretic verifiability in private information retrieval, 2023.

[Kos18] Takeshi Koshiba. Fourier-based function secret sharing with general access structure. In *International Conference on Mathematics and Computing*, pages 417–428. Springer, 2018.

[LA$^+$20] Songze Li, Salman Avestimehr, et al. Coded computing: Mitigating fundamental bottlenecks in large-scale distributed computing and machine learning. *Foundations and Trends® in Communications and Information Theory*, 17(1):1–148, 2020.

[LKZ23] Junru Li, Pengzhen Ke, and Liang Feng Zhang. Efficient information-theoretic distributed point function with general output groups. *Cryptology ePrint Archive*, 2023.

[LZ20] Wen Ming Li and Liang Feng Zhang. Towards efficient information-theoretic function secret sharing. *IEEE Access*, 8:28512–28523, 2020.

[LZLL20] Jinglong Luo, Liang Feng Zhang, Fuchun Lin, and Changlu Lin. Efficient threshold function secret sharing with information-theoretic security. *IEEE Access*, 8:6523–6532, 2020.

[MMA18] Mahtab Mirmohseni and Mohammad Ali Maddah-Ali. Private function retrieval. In *2018 Iran Workshop on Communication and Information Theory (IWCIT)*, pages 1–6. IEEE, 2018.

[OKK18] Takuya Ohsawa, Naruhiro Kurokawa, and Takeshi Koshiba. Function secret sharing using fourier basis. In *Advances in Network-Based Information Systems: The 20th International Conference on Network-Based Information Systems (NBiS-2017)*, pages 865–875. Springer, 2018.

[PRV12] Bryan Parno, Mariana Raykova, and Vinod Vaikuntanathan. How to delegate and verify in public: Verifiable computation from attribute-based encryption. In *Theory of Cryptography: 9th Theory of Cryptography Conference, TCC 2012, Taormina, Sicily, Italy, March 19-21, 2012. Proceedings 9*, pages 422–439. Springer, 2012.

[Raw19] EA Rawashdeh. A simple method for finding the inverse matrix of vandermonde matrix. *Mat. Vesn*, 71:207–213, 2019.

[RK19] Netanel Raviv and David A Karpuk. Private polynomial computation from lagrange encoding. *IEEE Transactions on Information Forensics and Security*, 15:553–563, 2019.

[Sha79] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

[SJ18] Hua Sun and Syed Ali Jafar. The capacity of private computation. *IEEE Transactions on Information Theory*, 65(6):3880–3897, 2018.

[WY05] David Woodruff and Sergey Yekhanin. A geometric approach to information-theoretic private information retrieval. In *20th Annual IEEE Conference on Computational Complexity (CCC'05)*, pages 275–284. IEEE, 2005.

[YLR$^+$19] Qian Yu, Songze Li, Netanel Raviv, Seyed Mohammadreza Mousavi Kalan, Mahdi Soltanolkotabi, and Salman A Avestimehr. Lagrange coded computing: Optimal design for resiliency, security, and privacy. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1215–1225. PMLR, 2019.

[ZW22] Liang Feng Zhang and Huaxiong Wang. Multi-server verifiable computation of low-degree polynomials. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 596–613. IEEE, 2022.

[ZYTL22] Jinbao Zhu, Qifa Yan, Xiaohu Tang, and Songze Li. Symmetric private polynomial computation from lagrange encoding. *IEEE Transactions on Information Theory*, 68(4):2704–2718, 2022.