SN Computer Science (2022) 3:174
https://doi.org/10.1007/s42979-022-01073-5

ORIGINAL RESEARCH

Communicationless Evaluation of Quadratic Functions over Secret Shared Dynamic Database

Daniel Berend[1,2] · Dor Bitan[1] · Shlomi Dolev[2]

Received: 20 January 2021 / Accepted: 10 February 2022 / Published online: 25 February 2022

Abstract
One of the most active fields of research in cryptography is finding efficient homomorphic encryption schemes, particularly information-theoretically secure schemes which are not based on unproven computational hardness assumptions. We suggest here an information-theoretically secure secret sharing scheme based on Shamir's secret sharing scheme. While Shamir's scheme supports no homomorphic multiplications of secrets, our scheme efficiently supports one homomorphic multiplication of secrets in addition to homomorphic additions of, practically, any number of such multiplied secrets. We focus on the single-client–multi-server setting. Therefore, our scheme enables a single user to share a database of m records (secrets) among N semi-honest servers with $O(m^2)$ ciphertext, using a novel variant of Shamir's secret sharing scheme and polynomials of degree $N-1$. Then, our scheme enables homomorphic evaluation of quadratic functions and 2-CNF circuits over the database with no communication between the servers. Our scheme is perfectly secure against attacks of a single server and information-theoretically statistically secure against attacks of coalitions of less than $N-1$ servers. One of the main advantages of our scheme over known schemes is enabling the evaluation of quadratic functions and 2-CNF secrets over a *dynamic* database of secrets. A dynamic database of secrets is a database of secrets that can grow in the future with no need for storing and re-sharing existing secrets by the user. To the best of our knowledge, the challenging support for the dynamic property was not obtained in this setting elsewhere before.

Keywords Dynamic secret sharing · Information-theoretic security · Outsourcing of computation

Introduction

Background

The Secure Outsourcing Problem

Consider the following scenario. A user is holding some highly confidential data (hereafter referred to as 'the secrets') and wishes to outsource the storage of this data to an untrusted server while enabling the server to perform computations over the data obliviously. A vast amount of papers were written on this problem in the past 4 decades ever since it was brought up by Rivest, Adelman, and Dertouzos in [22]. Solutions differ in their overall approach, in their security and efficiency level, and various attributes.

Two main approaches for solving the secure outsourcing problem are discussed in the literature. Some of the known solutions are base on the centralized approach, in which a single server is employed [2, 9, 16–18, 25–27]. Other solutions take the distributed approach, in which the user distributes the information between several servers [1, 4, 5, 10, 12, 13, 15, 21].

In the distributed approach, the user employs a secret sharing scheme to distributes secret shares of the data among the servers. Secret sharing is a fundamental cryptographic primitive, first introduced by Shamir [24] and Blakley [7] (independently) in 1979. Therefore, shares of a secret value are distributed to a set of parties in such a way that only authorized sets of parties can reconstruct the secret, while unauthorized sets cannot gain information about it. In the centralized approach, the user typically encrypts the information using a homomorphic encryption scheme and uploads an encrypted version of the data to the server. The

✉ Dor Bitan
dorbi@post.bgu.ac.il

1 Department of Mathematics, Ben-Gurion University of the Negev, Beer Sheva, Israel

2 Department of Computer Science, Ben-Gurion University of the Negev, Beer Sheva, Israel

centralized approach typically compels two security issues. First, it creates a single point of failure (SPOF) by putting all the information on a single server. Second, it requires storing and managing encryption keys. This work is based on the distributed approach, which avoids these issues.

Security of Cryptographic Schemes

The security of cryptographic schemes may be either *information-theoretic* or *computational*. In information-theoretically (IT) secure schemes, the security of the system is derived purely from information theory and depends neither on the computing power of the adversary nor on any computational hardness assumptions. It is possible in such schemes that some information about the plaintext will be revealed to an adversary by the ciphertext, but that leakage of information can be quantified by statistical tools and may be controlled by an appropriate choice of parameters. IT-secure schemes, in which there is negligible leakage of information, are often called *statistically information-theoretic secure*, or simply *statistically secure*. IT-secure systems, in which there is absolutely no leakage of information, are *perfectly secure* schemes. The second type of security is *computational security*. It refers to cryptographic schemes that are based on computational hardness assumptions. The security of these schemes is based on unproven assumptions regarding the existence of algorithms for solving specific mathematical problems and the computing power of the possible adversary. While the centralized approach can achieve no more than computational security, the distributed approach often achieves IT-security. In this work, we suggest an IT-secure solution to the outsourcing problem.

Dynamic Outsourcing

As mentioned, in this work, we look for solutions for the secure outsourcing problem. We assume that a single user holds a highly confidential database and wishes to distribute the database among several servers to minimize the risk of a security leak. An essential requirement that arises in this setting is adding new records to the database over time. A secure outsourcing scheme is *dynamic* if it enables the user to add (or remove) new records to the database with no need for storing and re-sharing existing secrets by the dealer [6].In many practical applications, dynamic outsourcing schemes have significant benefits over non-dynamic schemes. When outsourcing a database to semi-trusted clouds, part of the records may not be known and determined in the future. A user who employs a non-dynamic scheme must store a copy of the entire database. In this work, we suggest a dynamic solution for the secure outsourcing problem.

Homomorphic Properties of Cryptographic Schemes

Another essential requirement that arises when outsourcing a confidential database is to enable performing computations over the data. To this end, it is useful for the secure distributing mechanism to have *homomorphic* properties. We now describe the main ideas concerning homomorphic encryption systems. Assume π is a cryptographic system, m is a message and c its encryption, denoted $\mathrm{Enc}_\pi(m)$. Let f be a function defined over the message space of π. How, if at all, can c be publicly converted into an encryption of $f(m)$? This interesting question is a main subject of research in cryptography. If we assume that the messages and the ciphertexts are elements of a field or a ring (as is often the case), then a more specific form of that question is as follows. Let $m_1, \ldots, m_d$ be messages, and $c_1, \ldots, c_d$ their encryptions. Can $c_1, \ldots, c_d$ be used to publicly generate $c_{\mathrm{add}} = \mathrm{Enc}_\pi\left(\sum_{i=1}^d m_i\right)$ or $c_{\mathrm{mult}} = \mathrm{Enc}_\pi\left(\Pi_{i=1}^d m_i\right)$? If it is possible to use $c_1, \ldots, c_d$ to publicly generate $c_{\mathrm{add}} = \mathrm{Enc}_\pi\left(\sum_{i=1}^d m_i\right)$ (respectively, $c_{\mathrm{mult}} = \mathrm{Enc}_\pi\left(\Pi_{i=1}^d m_i\right)$), then π is *additively homomorphic* (respectively, *multiplicatively homomorphic*). If both tasks can be carried out, then π is a *fully homomorphic encryption* (FHE) system. There are several single-operation homomorphic schemes, such as the RSA cryptosystem, which is multiplicatively homomorphic (but cannot support homomorphic additions, and is only computationally secure), and Shamir's secret sharing scheme, which is additively homomorphic (but cannot support homomorphic multiplications).

Midway between single-operation homomorphic encryption systems and FHE systems are *somewhat homomorphic encryption systems*. These systems are additively homomorphic and also support a bounded number of homomorphic multiplications (or multiplicatively homomorphic and support a bounded number of homomorphic additions). To make the concept 'somewhat homomorphic' clear, we modify the above definitions. In the above definition of an additively (respectively, multiplicatively) homomorphic cryptographic system, d could be arbitrarily large. Now we define a cryptographic system to be d_1-*additively homomorphic* (respectively, d_1-*multiplicatively homomorphic*) if it is additively (respectively, multiplicatively) homomorphic for $d \leq d_1$. Thus, additively homomorphic systems are ∞-additively homomorphic, and multiplicatively homomorphic systems are ∞-multiplicatively homomorphic. In this work we suggest a somewhat homomorphic scheme. We support a single multiplication of secrets, followed by homomorphic additions of, practically, any number of such multiplied secrets.

Party Interaction

In the distributed approach, the security of the private data is maintained as long as the number of servers that collaborate in an adversarial attempt to reveal the secrets is less than some integer t. To carry computations over the secret shared data, the servers are often required to communicate with each other. Such interaction between the servers increases the risk of generation of large adversarial coalitions that can compromise the security of the data. To minimize that risk, one may allow no communication between the servers. With no communication between them, the servers can remain utterly oblivious to the number and identity of other servers participating in the scheme. The outsourcing scheme we suggest in this work is communicationless and has $t = N - 1$, hence significantly reducing the risk of generation of effective adversarial coalitions.

Related Work

Our work is based on Shamir's secret sharing scheme. In Shamir's secret sharing scheme, the secret s is an element of a finite field denoted $\mathbb{F}_p$ and is shared by a *dealer* among a set of N parties (where $p > N$) in the following way. Each party P_i, $1 \le i \le N$, is assigned by the dealer with an element α_i of $\mathbb{F}_p^{\times}$, where the α_is are distinct. Random elements a_j of $\mathbb{F}_p$, $1 \le j \le t-1$, are picked by the dealer. Let f be the polynomial defined by $f(x) = s + \sum_{j=1}^{t-1} a_j x^j$. Each party P_i gets the value $f(\alpha_i)$. It was proved by Shamir [24] that, in this way, every group of t parties will be able to reconstruct s, but no group of $t-1$ parties gains any information about s. These properties are derived directly from the fact that a polynomial of degree $t-1$ is uniquely determined by its values at t points.

Shamir's secret sharing scheme is one of the most influential schemes. It may be used to build schemes that enable IT-secure outsourcing of computations [11, 13–15]. Dolev et al. [13] used Shamir's standard scheme for IT-secure distributed evaluation of RAM programs, where the parties obliviously run a given RAM program over given input. Their solution is based on secure multi-party computation and requires ongoing communication between parties and an external entity called *reducer*, and hence implies high overhead. Earlier, Sander et al. [23] proposed a system to evaluate NC1 circuits on encrypted values, considering the following case: Alice holds an input x, and Bob is holding a circuit C. We would like Alice to be able to compute C(x), while keeping her input x private, and Bob keeping his circuit C private as well. A main drawback of the system is that the ciphertext length grows exponentially in the depth of the circuit. Their system is based on random self-reducible probabilistic encryption, which may be either computationally or information-theoretically secure. Either way, under the suggested protocol, Alice's input is computationally private, while Bob's circuit remains information-theoretically private.

In 2005, Boneh et al. [8] proposed a computationally secure public key encryption scheme and showed how it can be used to evaluate 2-DNF circuits over ciphertexts. Their scheme is somewhat homomorphic − additively homomorphic and 2-multiplicatively homomorphic. The first FHE scheme was proposed by Gentry [16], followed by several revisions and improvements [2, 9, 17, 18, 25–27]. Unfortunately, the time complexity of the current implementations of FHE scheme is too high to make the scheme practical. Brakerski and Perlman [9] suggested a computationally secure FHE scheme that can be carried out by an unbounded number of parties. Let N be the number of parties whose ciphertexts have been introduced into the computation so far (inputs from more parties can join the computation later). They used the multi-key approach and obtained a scheme with fully dynamic properties, $O(N)$ ciphertext expansion, and $O(N)$ space complexity for an atomic homomorphic operation. Unfortunately, their results are only theoretical, as their scheme is time-wise impractical.

To conclude, one may characterize solutions for the outsourcing problem according to the following criteria.

- Is the scheme distributed (or employs a single server, creating a SPOF and a need to manage keys)?
- Is the scheme information-theoretically secure (or only computationally secure)?
- Is the scheme dynamic (or adding new records over time requires storing a plaintext copy of the database)?
- Is the scheme, at least, somewhat homomorphic (or at most single-operation homomorphic)?
- If the scheme is based on the distributed approach, is it communicationless and enables the servers to remain utterly oblivious to the number and identity of one another (or does the scheme require communication between servers, hence increasing the risk of creation of large adversarial coalitions)?

No system is known that answers 'yes' to all of the questions mentioned above. Centralized fully homomorphic systems, such as Gentry's [16] and Brakerski and Perlman's [9], are neither IT-secure nor are they practical. No communicationless IT-secure scheme is fully homomorphic, and no practical scheme is fully homomorphic. The scheme suggested by Boneh et al. [8] is a centralized, somewhat homomorphic, computationally secure and practical scheme.

Our Contribution

The main result we obtain in this work is a distributed, information-theoretically secure, dynamic, somewhat homomorphic, and communicationless solution for the outsourcing problem. Our scheme is based on a new *function sieving* method we present here. Our method yields 1-*homomorphic* *multiplicative pair*s of polynomials, which enables us to adjust Shamir's secret sharing scheme to support one homomorphic multiplication of secrets. The secrets are shared using polynomials of degree $N-1$ among N parties, and our scheme provides perfect security against an attack of a single curious party and statistical security against an attack of a coalition of up to $N-2$ curious parties. We note that the level of security achieved in our scheme is optimal (in our setting). This follows from the result of Barkol et al. [3], who showed that perfectly secure t-private d-multiplicative secret sharing among N players is possible if and only if $N > dt$. Our scheme enables homomorphic multiplication of two secrets (i.e., $d = 2$) while keeping the secrets safe against coalitions of less than $t = N - 1$ out-of N servers and hence can achieve at most statistical security.

Of course, one can support homomorphic multiplications of secrets in Shamir's scheme by taking polynomials of a smaller degree to-begin-with. For example, one can use Shamir's original scheme to share two secrets among four parties using linear polynomials, enabling one homomorphic multiplication of secrets, but in this way, the security will be compromised since any coalition of two parties can easily determine the exact value of the secrets. In our scheme, for example, we can use cubic polynomials to share secrets among four parties in such a way that no coalition of two parties can find the secrets. Our scheme is based on a sophisticated way of choosing the polynomials in a correlated way.

One can support homomorphic evaluation of quadratic functions and 2-CNF circuits by sharing, along with each pair of secrets, their product, (or using Beaver's pre-processing method, as suggested in [4]). Nevertheless, in this way, if new secrets are expected to be joined with the primary ones, then one must keep all the primary secrets in memory to enable the homomorphic computations over the enlarged set of secrets. Our scheme enables additional secrets to be shared over time, while in each stage: (a) quadratic functions and 2-CNF circuits over the new set of secrets can be homomorphically and securely evaluated; (b) the dealer is not required to store the values of the already-shared secrets in memory, but only the non-free (secret-independent) coefficients of the polynomial that are meant to be used to encrypt the future secrets.

To the best of our knowledge, our scheme suggests the first efficient solution for the outsourcing problem while maintaining all the following attributes: *IT-secure, dynamic, somewhat homomorphic,* and *communicationless.*

Organization

In the next section, we introduce the function sieving method and our scheme for secret sharing and multiplication of two secrets among N servers using polynomials of degree $N-1$. In the following section, we prove the correctness of the scheme and discuss its security against an attack of one curious server and against an attack of a coalition of up to $N-2$ curious servers. Before the concluding section, we describe how to use our scheme to distribute a confidential database to a set of semi-honest servers while enabling homomorphic evaluation of quadratic functions and 2-CNF circuits dynamically. The final section concludes the work.

Homomorphic Multiplication of Secret Shares

In this section, we introduce our secret sharing scheme based on Shamir's secret sharing scheme. The scheme will enable us to share two secrets among N servers (parties) using polynomials of degree $N-1$, perform one homomorphic multiplication of the secrets and consecutive homomorphic additions with further secrets, without increasing the number of parties required to extract the result. We will show that the scheme has perfect security against an attack of a single party. We also prove that our scheme is statistically secure against coalitions of up to $N-2$ parties.

We begin with a brief overview of our methods and constructions. Assume s_1 and s_2 are two secrets that were shared by Shamir's scheme among N parties, P_j, $1 \le j \le N$, using two polynomials of degree $N-1$, f_1 and f_2, respectively. For convenience, we denote from now on $n = N - 1$. Each P_j holds a share of each of the secrets: $\left(\alpha_j, f_1(\alpha_j)\right)$ and $\left(\alpha_j, f_2(\alpha_j)\right)$. As Shamir's scheme is additively homomorphic, the points $\left(\alpha_j, f_1(\alpha_j) + f_2(\alpha_j)\right)$ for $1 \le j \le n+1$ are shares of $s_1 + s_2$. Interpolation of these points will yield the unique polynomial of degree $\le n$ going through them, which is $f_1 + f_2$, whose value at 0 is $s_1 + s_2$. Now, as Shamir's scheme is not multiplicatively homomorphic, the points $\left(\alpha_j, f_1(\alpha_j) \cdot f_2(\alpha_j)\right)$ are in general not shares of $s_1 \cdot s_2$. The polynomial $f_1 \cdot f_2$ is of degree $\le 2n$. Hence, $2n+1$ points are required to determine it, so that the $n+1$ points we have do not suffice, i.e., no information regarding $s_1 \cdot s_2$ may be gained from the $n+1$ points $\left(\alpha_j, f_1(\alpha_j) \cdot f_2(\alpha_j)\right)$ $(1 \le j \le n+1)$. If one insists on interpolating the points $\left(\alpha_j, f_1(\alpha_j) \cdot f_2(\alpha_j)\right)$, that interpolation will yield some polynomial g of degree $\le n$. It might be the case, though, that $g(0) = s_1 \cdot s_2$. When does it happen? We seek pairs of polynomials to be used with Shamir's scheme that yield $g(0) = s_1 \cdot s_2$. We call this procedure *function sieving*, and as we will show below, it yields 1-*homomorphic multiplicative pair*s of polynomials, which are pairs of polynomials that meet the required condition. We will show that, given the α_js, these pairs are independent of the secrets and can be determined according to the other coefficients of the polynomials (i.e., all coefficients except for the free terms, which are the secrets).

Function Sieving

Assume that the field $\mathbb{F}_p$, in which the secrets s_1 and s_2 reside, is such that $p \equiv 1 \pmod{n+1}$. In that case, since $\mathbb{F}_p^\times$ is cyclic, it contains a primitive root of unity of order $n+1$. Let α be such a root. For $1 \le j \le n+1$ denote $\alpha_j := \alpha^j$, and assign to each party P_j the value α_j.

Let $a_i, b_i \in \mathbb{F}_p, 1 \le i \le n$, and consider the polynomials

$$f_1(x) = s_1 + \sum_{i=1}^{n} a_i x^i, \quad f_2(x) = s_2 + \sum_{i=1}^{n} b_i x^i,$$

in $\mathbb{F}_p[x]$. Share the secrets s_1, s_2 among the parties using f_1, f_2. Namely, distribute to each P_j the values $f_1(\alpha_j), f_2(\alpha_j)$. Let

$$y_j = f_1(\alpha_j) \cdot f_2(\alpha_j), \quad 1 \le j \le n+1.$$

The pairs $(\alpha_j, y_j) \in \mathbb{F}_p^2$ are $n+1$ distinct points through which the polynomial $(f_1 \cdot f_2)(x)$ passes. Since $f := f_1 \cdot f_2$ is of degree $\le 2n$, it is uniquely determined by $2n+1$ points. Since there are only $n+1$ points (α_j, y_j), interpolation of them will certainly not yield $(f_1 \cdot f_2)(x)$. Nevertheless, let $g(x)$ be the interpolation polynomial for the $n+1$ points, (α_j, y_j). Obviously, g is of degree $\le n$.

Now, let

$$\psi(x) = \prod_{j=1}^{n+1} (x - \alpha_j).$$

Since f and g agree on the roots of ψ, we have $g(x) \equiv f(x) \pmod{\psi(x)}$. Since the α_js are all the roots of unity of order $n+1$, we have

$$\psi(x) = x^{n+1} - 1. \tag{1}$$

Hence, it is easy to compute g. In fact, denote

$$f(x) = s_1 s_2 + \sum_{i=1}^{2n} c_i x^i.$$

We have $x^{n+1} \equiv 1 \pmod{\psi(x)}$, and therefore,

$$g(x) \equiv f(x) \equiv s_1 s_2 + c_{n+1} + \sum_{i=1}^{n} (c_i + c_{n+1+i}) x^i \pmod{\psi(x)}.$$

This in turn implies that $g(0) = s_1 s_2 + c_{n+1}$.

Thus, if we take f_1 and f_2 such that $c_{n+1} = 0$, we get $g(0) = f(0)$. Now, $c_{n+1} = \sum_{i=1}^{n} a_i b_{n+1-i}$. This observation yields a useful variant of Shamir's secret sharing scheme. Instead of picking the coefficients of f_1 and f_2 uniformly at random, one may pick them in such a way that $c_{n+1} = 0$. This is, in essence, the function sieving process. Instead of using Shamir's secret sharing scheme with random polynomials from $\mathbb{F}_p[x]$, we use it with polynomials f_1, f_2, for which $c_{n+1} = 0$, which compels $g(0) = f(0)$. Such a pair (f_1, f_2) is a 1-*homomorphic multiplicative pair* of polynomials.

We define the set of acceptable coefficients for these pairs

$$\mathcal{V}_p := \left\{ (a_1, \dots, a_n, b_1, \dots, b_n) \in \mathbb{F}_p^{2n} \,\middle|\, \sum_{i=1}^{n} a_i b_{n+1-i} = 0, \right.$$
$$\left. \bar{a} \neq \bar{0} \neq \bar{b} \right\} \cup \left\{ \bar{0} \in \mathbb{F}_p^{2n} \right\},$$

where $\bar{a} = (a_1, \dots, a_n)$ and $\bar{b} = (b_1, \dots, b_n)$.[1]

Next, since elements should be picked from $\mathcal{V}_p$, we must define a probability measure on it. First, we compute the cardinality of $\mathcal{V}_p$.

Proposition 1 $|\mathcal{V}_p| = (p^n - 1)(p^{n-1} - 1) + 1$.

Proof The element $\bar{0} \in \mathbb{F}_p^{2n}$ contributes 1 to $|\mathcal{V}_p|$. The n-tuple $(a_1, \dots, a_n)$ may be chosen in $p^n - 1$ different ways. For each of these, the n-tuple $(b_1, \dots, b_n)$ is required to satisfy

$$\sum_{i=1}^{n} a_i b_{n+1-i} = 0.$$

Since $(a_1, \dots, a_n) \neq \bar{0}$, this equation has $p^{n-1} - 1$ non-zero solutions $\bar{b}$. All in all, we get $(p^n - 1)(p^{n-1} - 1) + 1$ elements in $\mathcal{V}_p$. □

Define a probability measure Q on $\mathcal{V}_p$ by:

$$Q(v) = \begin{cases} \frac{1}{p^n}, & v = \bar{0} \in \mathbb{F}_p^{2n}, \\ \frac{1}{p^n(p^{n-1}-1)}, & v \neq \bar{0}. \end{cases}$$

One verifies readily, using Proposition 1, that Q is indeed a probability.

The set $\mathcal{V}_p$ and the probability measure Q are used in the next section, where we present the multiplication scheme.

The Scheme

We now present our secret sharing scheme. A single homomorphic multiplication of two secrets is supported, to which further secrets can be added homomorphically. Assume a dealer D has two secrets $s_1, s_2 \in \mathbb{F}_p$ and private connection channels with N servers P_i, $1 \le j \le N$. As a preliminary

[1] Each of the $\bar{0}$s refers to the zero vector of the vector space it belongs to. We include these zero vectors in $\mathcal{V}_p$ for technical reasons explained below.

phase, the dealer D assigns to each server P_j an $\alpha_j = \alpha^j \in \mathbb{F}_p^{\times}$, where α is a primitive root of unity of order N. The scheme stages are as follows:

Algorithm 1: Sharing secrets and computing product

1. The dealer D picks an element $(a_1, \ldots, a_n, b_1, \ldots, b_n) \in \mathcal{V}_p$ according to the distribution Q.
2. D sets $f_1(x) = s_1 + \sum_{i=1}^{n} a_i x^i$ and distributes the value $f_1(\alpha_j)$ to each server P_j.
3. D sets $f_2(x) = s_2 + \sum_{i=1}^{n} b_i x^i$ and distributes the value $f_2(\alpha_j)$ to each server P_j.
4. Each P_j computes $y_j = f_1(\alpha_j) \cdot f_2(\alpha_j)$ in $\mathbb{F}_p$ and sends y_j back to D.
5. D finds the unique polynomial $g(x)$ over $\mathbb{F}_p$ that goes through (α_j, y_j).
6. D calculates $s = g(0)$.

As one can see, we use here a polynomial of degree n to represent each of the secrets, and yet we are able to reconstruct their product with only $n+1$ parties (versus $2n+1$ that would be needed originally).

Regarding stage 1 of the protocol, a simple way to Q-pick a suitable element is to create an array with the elements of the set $\mathcal{V}_p$ and insert the element $\overline{0} \in \mathbb{F}_p^{2n}$ into the array $p^{n-1} - 2$ more times. Then, picking an element uniformly at random from that array is equivalent to Q-picking an element of $\mathcal{V}_p$.[2] In stage 5, since $1 \le i \le n+1$, the polynomial g is obviously of degree $\le n$.[3]

Example We provide a simple example. Let $p = 17$ and consider a dealer that holds the secret elements $s_1 = 3$ and $s_2 = 4$ in $\mathbb{F}_{17}$. Let $N = 4$ and assign four parties with the x-values $\alpha_1 = 4, \alpha_2 = 16, \alpha_3 = 13$ and $\alpha_4 = 1$. Here, $\psi(x) = \prod_{1 \le i \le 4}(x - \alpha_i) = x^4 - 1$. Let $v = (1,3,2,5,2,1) \in \mathcal{V}_{17}$, which implies $f_1(x) = 2x^3 + 3x^2 + x + 3$ and $f_2(x) = x^3 + 2x^2 + 5x + 4$. Here, $f(x) = f_1(x)f_2(x) = 2x^6 + 7x^5 + 11x^3 + 6x^2 + 2x + 12$. When the dealer shares the secrets s_1 and s_2 among the four parties, the parties obtain the following values.

Party	P_1	P_2	P_3	P_4
x value	4	16	13	1
$f_1(x)$	13	3	4	9
$f_2(x)$	1	0	3	12

Multiplying the y values, the parties obtain:

Party	P_1	P_2	P_3	P_4
x value	4	16	13	1
$f_1(x) \cdot f_2(x)$	13	0	12	6

Now, let g be the polynomial of degree (at most) three determined by the four points $(\alpha_i, f_1(\alpha_i)f_2 \cdot (\alpha_i))$. Here, these are the points: (4, 13), (16, 0), (13, 12), (1, 6). The polynomial g, of course, can be obtained using Lagrange interpolation. Nevertheless, since the (non-free) coefficients of f_1 and f_2 were Q-picked from V_{17}, the polynomial g may also be computed by dividing f by ψ and taking the residue. Indeed, using polynomials division one finds

$$\begin{aligned} f(x) &= 2x^6 + 7x^5 + 11x^3 + +6x^2 + 2x + 12 \\ &= (2x^2 + 7x)(x^4 - 1) + 11x^3 + 8x^2 + 9x + 12 \\ &= (2x^2 + 7x) \cdot \psi(x) + g(x), \end{aligned}$$

i.e., $g(x) = 11x^3 + 8x^2 + 9x + 12$, and $g(0) = 12 = s_1 \cdot s_2 = f(0)$. It is easy to check that g is the only polynomial of degree (at most) three that goes through the four points (4, 13), (16, 0), (13, 12) and (1, 6). In our scheme, the dealer computes g from the four points received from the parties, and we prove that (following our scheme) the value of g at zero always equals $s_1 \cdot s_2$.

The Main Results

In this section, we discuss the correctness and security of our secret sharing scheme. We begin with correctness.

The Scheme Correctness

We prove the following proposition:

Proposition 2 *The value s, calculated at stage 6 of Algorithm 1, is equal to $s_1 \cdot s_2$.*

Proof The proposition follows directly from the function sieving process, described in "Homomorphic multiplication of secret shares". The coefficients of the polynomials

[2] Clearly, one can use the proof of Proposition 1 to implement stage 1 in time $O(n)$.

[3] In fact, given the y_js, $g(0)$ can be computed without finding g. That procedure is not of our main interests.

f_1,f_2 were picked from $\mathcal{V}_p$, and hence $\sum_{i=1}^{n} a_i b_{n+1-i} = 0$. By (1), the α_js were picked in such a way that $\psi(x) = x^{n+1} - 1$. In stage 5 of the scheme, the dealer finds a polynomial g of degree $\leq n$ such that $g(\alpha_j) = y_j$ for $1 \leq j \leq n+1$. This implies that

$$g(x) \equiv (f_1 \cdot f_2)(x) = s_1 s_2 + \sum_{i=1}^{2n} c_i x^i$$
$$\equiv s_1 s_2 + c_{n+1} + \sum_{i=1}^{n} (c_i + c_{n+1+i}) x_i \pmod{\psi(x)}.$$

Hence, $g(0) = s_1 s_2 + c_{n+1} \equiv s_1 s_2 \pmod{\psi(x)}$. □

Note that g may now be treated as if it was originally used to share $s_1 \cdot s_2$ among N parties since each of them is now holding y_j. Hence, further secrets can be shared and homomorphically added to $s_1 \cdot s_2$ as in Shamir's standard scheme.

The Scheme Security

We now analyze the scheme security against curious parties' attacks. We follow standard security definitions that can be found in literature (e.g., in [20]). We will show that our scheme has perfect passive security against one party attack and statistical security against an attack of a coalition of size up to $N-2$. To conclude such arguments, first, we must make our assumptions clear. We assume the following:

- *Assumption 1:* The pair of secrets $(s_1, s_2) \in \mathbb{F}_p^2$ is arbitrary. To be precise, we assume they are picked according to an arbitrary distribution Γ, on which we have no assumptions.
- *Assumption 2:* The prime p, the distribution Γ, the set $\mathcal{V}_p$ and the distribution Q over it are public. Namely, if we denote by S_1 and S_2 the $\mathbb{F}_p$-valued random variables indicating the Γ-picked secrets, then the probability $P[(S_1, S_2) = (s_1, s_2)]$ is known for each pair $(s_1, s_2) \in \mathbb{F}_p^2$.
- *Assumption 3:* The element $(a_1, \ldots, a_n, b_1, \ldots, b_n) \in \mathcal{V}_p$, that is Q-picked during stage 1 of the scheme, is kept secret. So are the values $f_1(\alpha_j)$ and $f_2(\alpha_j)$, $1 \leq j \leq N$, that D sends to each party P_j at stages 2 and 3 of the scheme. In the single party attack scenario, P_j does not know $f_1(\alpha_i)$ and $f_2(\alpha_i)$ for $i \neq j$. In the scenario of an attack of a coalition of k parties, we assume, without loss of generality, that $P_1, \ldots, P_k$ are curious parties that join their shares in an attempt to find the secrets, but they do not know the shares of other parties.

Perfect Security Against Single Party Attack

To show that our scheme has perfect security against one curious party attack, we need to show that, when P_j receives information from D during stages 2 and 3 of the scheme, he gains absolutely no information about the values of s_1 and s_2. We can summarize the information that P_j receives during stages 2 and 3 of the scheme by the following equations:

$$\begin{aligned} s_1 + \sum_{i=1}^{n} a_i \alpha_j^i &= y_j, \\ s_2 + \sum_{i=1}^{n} b_i \alpha_j^i &= y_j'. \end{aligned} \tag{2}$$

The unknowns in these equations are s_1, s_2, a_i and b_i, $1 \leq i \leq n$, while all other quantities are known parameters to P_j. We start with

Theorem 1 *For an arbitrary fixed* $\alpha \in \mathbb{F}_p^\times$ *denote* $u = \begin{pmatrix} \sum_{i=1}^{n} a_i \alpha^i \\ \sum_{i=1}^{n} b_i \alpha^i \end{pmatrix}$. *Under the above assumptions,* $P[u = \begin{pmatrix} x \\ y \end{pmatrix}] = \frac{1}{p^2}$, *for every* $\begin{pmatrix} x \\ y \end{pmatrix} \in \mathbb{F}_p^2$.

Proof of Theorem 1 Call u *the result vector*. Since p and α are set, u depends only on the Q-choice of $v \in \mathcal{V}_p$. For $v = (a_1, \ldots, a_n, b_1, \ldots, b_n) \in \mathcal{V}_p$, denote

$$M_v = \begin{pmatrix} a_1 & \ldots & a_n \\ b_1 & \ldots & b_n \end{pmatrix} \in M_{2\times n}(\mathbb{F}_p).$$

We define a mapping $\mu_\alpha : \mathcal{V}_p \to \mathbb{F}_p^2$ by

$$\mu_\alpha(v) = M_v \begin{pmatrix} \alpha \\ \vdots \\ \alpha^n \end{pmatrix}.$$

For convenience denote $\mu = \mu_\alpha$. Thus,

$$P[u = \begin{pmatrix} x \\ y \end{pmatrix}] = P[\mu(v) = \begin{pmatrix} x \\ y \end{pmatrix}].$$

To compute $P[u = \begin{pmatrix} x \\ y \end{pmatrix}]$, we first partition $\mathbb{F}_p^2$ into four subsets U_j, $1 \leq j \leq 4$:

- $U_1 = \{\begin{pmatrix} 0 \\ 0 \end{pmatrix}\} \subset \mathbb{F}_p^2$.
- $U_2 = \{\begin{pmatrix} x \\ 0 \end{pmatrix} \in \mathbb{F}_p^2 \mid x \neq 0\}$.
- $U_3 = \{\begin{pmatrix} 0 \\ y \end{pmatrix} \in \mathbb{F}_p^2 \mid y \neq 0\}$.
- $U_4 = \{\begin{pmatrix} x \\ y \end{pmatrix} \in \mathbb{F}_p^2 \mid x \neq 0, y \neq 0\}$.

We will compute $P[u = \begin{pmatrix} x \\ y \end{pmatrix}]$ for $\begin{pmatrix} x \\ y \end{pmatrix} \in U_j$ for each j separately.

Starting with $j = 1$. We look for elements $v \in \mathcal{V}_p$ such that

$$\mu(v) = \begin{pmatrix} 0 & \\ & 0 \end{pmatrix}. \tag{3}$$

Of course, $v = \overline{0} \in \mathbb{F}_p^{2n}$ is a solution of (2). Assume $v = (a_1, \dots, a_n, b_1, \dots, b_n) \in \mathcal{V}_p$ is such that $v \neq \overline{0}$ and M_v is a solution of (2). Namely:

$$\begin{aligned} &\text{I} && \sum_{i=1}^{n} a_i \alpha^i = 0, \\ &\text{II} && \sum_{i=1}^{n} b_i \alpha^i = 0, \\ &\text{III} && \sum_{i=1}^{n} a_i b_{n+1-i} = 0, \end{aligned} \tag{4}$$

where $(a_1, \dots, a_n) \neq \overline{0} \neq (b_1, \dots, b_n)$. Each solution for (3) gives a suitable element of $\mathcal{V}_p$. Now, (3)I is a linear equation in n variables a_i. Since the trivial solution is not acceptable, it has $p^{n-1} - 1$ possible solutions $(a_1, \dots, a_n)$. For each of these solutions, (3)II-(3)III is a linear system of two equations in n variables b_i. If the equations are independent, the system has $p^{n-2} - 1$ non-trivial solutions $(b_1 \dots, b_n)$. Can they be dependent? If they are, there is a $c \in \mathbb{F}_p$ such that $c \cdot \alpha^i = a_{n+1-i}$ for $1 \leq i \leq n$. By (3)I we get then $\sum_{i=1}^{n} c \cdot \alpha^{n+1-i} \cdot \alpha^i = 0$, so that $n \cdot c \cdot \alpha^{n+1} = 0$. Each of the factors is non-zero, and hence (3)II-(3)III are independent. All in all, we get $(p^{n-1} - 1)(p^{n-2} - 1)$ solutions $(a_1, \dots, a_n, b_1, \dots, b_n) \neq \overline{0}$.

We conclude that

$$P[u = \begin{pmatrix} 0 \\ 0 \end{pmatrix}] = 1 \cdot Q(\overline{0}) + (p^{n-1} - 1)(p^{n-2} - 1) \cdot Q(v_0),$$

where v_0 is any non-zero element of $\mathcal{V}_p$. That is

$$P[u = \begin{pmatrix} 0 \\ 0 \end{pmatrix}] = 1 \cdot \frac{1}{p^n} + \frac{(p^{n-1} - 1)(p^{n-2} - 1)}{p^n(p^{n-1} - 1)} = \frac{1}{p^2}.$$

We move to U_2. Thus, we are looking for elements $v \in \mathcal{V}_p$ such that

$$\mu(v) = \begin{pmatrix} x & \\ & 0 \end{pmatrix}, \quad (x \neq 0). \tag{5}$$

Similar to the computation of $\left| \mu^{-1}\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix} \right) \right|$, we get the system

$$\begin{aligned} &\text{I} && \sum_{i=1}^{n} a_i \alpha^i = x, \\ &\text{II} && \sum_{i=1}^{n} b_i \alpha^i = 0, \\ &\text{III} && \sum_{i=1}^{n} a_i b_{n+1-i} = 0, \end{aligned} \tag{6}$$

where $(a_1, \dots, a_n) \neq \overline{0} \neq (b_1, \dots, b_n)$, $x \neq 0$, and each solution of (4) gives a suitable element of $\mathcal{V}_p$. (4)I is a non-homogenous linear equation in n variables a_i, and hence has p^{n-1} solutions, $\overline{0}$ is not one of which. For each of these solutions, (4)II-(4)III is a system of two linear equations in n variables b_i. If they are independent, it has $p^{n-2} - 1$ non-zero solutions for b_i. Assume they are dependent. Hence, there is $c \in \mathbb{F}_p$ such that $c \cdot \alpha^{n+1-i} = a_i$ for $1 \leq i \leq n$. By (4)I we get then $\sum_{i=1}^{n} c \cdot \alpha^{n+1-i} \cdot \alpha^i = x$. Then $n \cdot c \cdot \alpha^{n+1} = x$, which gives $c = xn^{-1}$. Hence, there is exactly one solution a_i for (4)I that yields dependent equations (4)II-(4)III. Namely, for $a_i = c \cdot \alpha^{-i} = xn^{-1}\alpha^{-i}$ the system (4)II-(4)III is dependent, and hence has $p^{n-1} - 1$ non-zero solutions. All in all, we get that

$$\begin{aligned} &\left| \mu^{-1}\left(\begin{pmatrix} x \\ 0 \end{pmatrix} \right) \right| \\ &\quad = (p^{n-1} - 1) \cdot (p^{n-2} - 1) + 1 \cdot (p^{n-1} - 1) = p^{n-2}(p^{n-1} - 1). \end{aligned}$$

We use that and the fact that the trivial solution is not in $\mu^{-1}\left(\begin{pmatrix} x \\ 0 \end{pmatrix} \right)$ to compute

$$\begin{aligned} P[u = \begin{pmatrix} x \\ 0 \end{pmatrix}] &= P[\mu(v) = \begin{pmatrix} x \\ 0 \end{pmatrix}] = P[v \in \mu^{-1}\left(\begin{pmatrix} x \\ 0 \end{pmatrix} \right)] \\ &= \frac{p^{n-2}(p^{n-1} - 1)}{p^n(p^{n-1} - 1)} = \frac{1}{p^2}. \end{aligned}$$

The computation of $P[u = \begin{pmatrix} x \\ y \end{pmatrix}]$ for $\begin{pmatrix} x \\ y \end{pmatrix} \in U_3$ is analogous, which implies $P[u = \begin{pmatrix} 0 \\ y \end{pmatrix}] = \frac{1}{p^2}$ for $y \neq 0$.

Now, knowing $\left| \mu^{-1}\left(U_j \right) \right|$ for $1 \leq j \leq 3$, we subtract from $\left| \mathcal{V}_p \right|$ and get $\left| \mu^{-1}\left(U_4 \right) \right| = (p - 1)^2 \cdot p^{n-2}(p^{n-1} - 1)$. Observe that so far, for a specific $j \in \{1, 2, 3\}$, all elements of U_j had the same size of preimage under μ. If we show that the same holds for U_4 as well, then together with the fact that $\left| U_4 \right| = (p - 1)^2$ we get that $\left| \mu^{-1}\left(\begin{pmatrix} x \\ y \end{pmatrix} \right) \right| = p^{n-2}(p^{n-1} - 1)$ for $\begin{pmatrix} x \\ y \end{pmatrix} \in U_4$. This in turn will imply that

$$\begin{aligned} P[u = \begin{pmatrix} x \\ y \end{pmatrix}] &= P[\mu^{-1}(v) = \begin{pmatrix} x \\ y \end{pmatrix}] = p^{n-2}(p^{n-1} - 1) \cdot Q(v) \\ &= \frac{p^{n-2}(p^{n-1} - 1)}{p^n(p^{n-1} - 1)} = \frac{1}{p^2}. \end{aligned}$$

for $\begin{pmatrix} x \\ y \end{pmatrix} \in U_4$. Thus, all that is left is to show is that all elements of U_4 actually have the same size of preimage under μ.

To this end, we define a family of transformations $T_{k,l}$ over $\mathcal{V}_p$. For arbitrary fixed $k, l \in \mathbb{F}_p^\times$, let $T_{k,l} : \mathcal{V}_p \to \mathcal{V}_p$ be defined by

$$T_{k,l}(a_1, \dots, a_n, b_1, \dots, b_n) = (ka_1, \dots, ka_n, lb_1, \dots, lb_n).$$

The map $T_{k,l}$ is clearly bijective. In fact, the number and positions of zeros in v (if any) are the same as in $T_{k,l}(v)$. The set $\mathcal{V}_p$ and some of its subsets have important properties regarding $T_{k,l}$:

- $\mathcal{V}_p$ is $T_{k,l}$-invariant: If $v = (a_1, \dots, a_n, b_1, \dots, b_n) \in \mathcal{V}_p$, then $\sum_{i=1}^n a_i b_{n+1-i} = 0$. It immediately follows that $T_{k,l}(v) = kl \sum_{i=1}^n a_i b_{n+1-i} = 0$. Hence $T_{k,l}(v)$ is indeed in $\mathcal{V}_p$.
- The sets $\mu^{-1}(U_j)$ are $T_{k,l}$-invariant: If $v = (a_1, \dots, a_n, b_1, \dots, b_n) \in \mathcal{V}_p$, and $\mu(v) \in U_j$ for a certain j, then

$$\mu(v) = \begin{pmatrix} a_1 & \dots & a_n \\ b_1 & \dots & b_n \end{pmatrix} \begin{pmatrix} \alpha \\ \vdots \\ \alpha^n \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^n a_1 \alpha^i \\ \sum_{i=1}^n b_1 \alpha^i \end{pmatrix} \in U_j.$$

We have

$$\mu\big(T_{k,l}(v)\big) = \begin{pmatrix} ka_1 & \dots & ka_n \\ lb_1 & \dots & lb_n \end{pmatrix} \begin{pmatrix} \alpha \\ \vdots \\ \alpha^n \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^n ka_1 \alpha^i \\ \sum_{i=1}^n lb_1 \alpha^i \end{pmatrix}.$$

Then

$$\mu\big(T_{k,l}(v)\big) = \begin{pmatrix} k \sum_{i=1}^n a_1 \alpha^i \\ l \sum_{i=1}^n b_1 \alpha^i \end{pmatrix}.$$

Since $k, l \neq 0$, an entry of $\mu(v)$ vanishes if and only if the corresponding entry of $\mu\big(T_{k,l}(v)\big)$ does. Namely, if $\mu(v) \in U_j$, then $\mu\big(T_{k,l}(v)\big) \in U_j$. We conclude that the sets $\mu^{-1}(U_j) \subseteq \mathcal{V}_p$ are invariant under $T_{k,l}$.

Now, let $\begin{pmatrix} x \\ y \end{pmatrix}, \begin{pmatrix} x' \\ y' \end{pmatrix} \in U_j$ for some $1 \le j \le 4$. Take $v = (a_1, \dots, a_n, b_1, \dots, b_n) \in \mu^{-1}\left(\begin{pmatrix} x \\ y \end{pmatrix}\right)$. We have $\mu(v) = \begin{pmatrix} \sum_{i=1}^n a_1 \alpha^i \\ \sum_{i=1}^n b_1 \alpha^i \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix}$. Put

$$k = \begin{cases} \frac{x'}{x}, & x \neq 0, \\ 1, & x = 0, \end{cases}, \qquad l = \begin{cases} \frac{y'}{y}, & y \neq 0, \\ 1, & y = 0. \end{cases}$$

We get $\mu\left(T_{k,l}(v)\right) = \begin{pmatrix} k \sum_{i=1}^n a_1 \alpha^i \\ l \sum_{i=1}^n b_1 \alpha^i \end{pmatrix} = \begin{pmatrix} \frac{x'}{x} x \\ \frac{y'}{y} y \end{pmatrix} = \begin{pmatrix} x' \\ y' \end{pmatrix}$.

Thus, for every $v \in \mu^{-1}\left(\begin{pmatrix} x \\ y \end{pmatrix}\right)$ we have $T_{k,l}(v) \in \mu^{-1}\left(\begin{pmatrix} x' \\ y' \end{pmatrix}\right)$ for appropriate k, l. This implies that $\left|\mu^{-1}\left(\begin{pmatrix} x \\ y \end{pmatrix}\right)\right| = \left|\mu^{-1}\left(\begin{pmatrix} x' \\ y' \end{pmatrix}\right)\right|$ for $\begin{pmatrix} x \\ y \end{pmatrix}, \begin{pmatrix} x' \\ y' \end{pmatrix} \in U_j$. To conclude, for a given j, all elements of U_j have the same probability. □

We use Theorem 1 to prove the perfect security of our scheme in this scenario. We claim now

Proposition 3 $P[(S_1, S_2) = (s_1, s_2) \mid (1)] = P[(S_1, S_2) = (s_1, s_2)]$.

Proof Denote

$$\theta = P[(S_1, S_2) = (s_1, s_2) \mid (1)].$$

Explicitly[4],

$$\theta = P\left[(S_1, S_2) = (s_1, s_2) \,\middle|\, \begin{matrix} s_1 + \sum_{i=1}^n a_i \alpha^i = y \\ s_2 + \sum_{i=1}^n b_i \alpha^i = y' \end{matrix}\right].$$

Hence,

$$\theta = P\left[(S_1, S_2) = (s_1, s_2) \middle| u = \begin{pmatrix} y - s_1 \\ y' - s_2 \end{pmatrix}\right]$$
$$= \frac{P\left[(S_1, S_2) = (s_1, s_2) \cap u = \begin{pmatrix} y - s_1 \\ y' - s_2 \end{pmatrix}\right]}{P\left[u = \begin{pmatrix} y - s_1 \\ y' - s_2 \end{pmatrix}\right]}.$$

According to Theorem 1, we have $P\left[u = \begin{pmatrix} x \\ y \end{pmatrix}\right] = \frac{1}{p^2}$. Hence, the values of u are independent of (S_1, S_2), so that

$$\theta = \frac{P[(S_1, S_2) = (s_1, s_2)] \cdot \frac{1}{p^2}}{\frac{1}{p^2}} = P[(S_1, S_2) = (s_1, s_2)].$$

□

Security Against Coalitions of $k < N - 1$ Curious Parties

We now turn to analyze the scheme's security against a coalition of k parties for $k < N - 1$. Without loss of generality, we consider the coalition $\{P_1, \dots, P_k\}$. We will refer to this coalition as *the adversary*. As in the preceding scenario, we

[4] We omit the index j and write α, y, y'.

can summarize the information the adversary is holding by the system of $2k$ equations:

$$\begin{aligned} s_1 + \sum_{i=1}^{n} a_i\alpha_1^i &= y_{11}, \quad \dots \quad , s_1 + \sum_{i=1}^{n} a_i\alpha_k^i = y_{1k}, \\ s_2 + \sum_{i=1}^{n} b_i\alpha_1^i &= y_{21}, \quad \dots \quad , s_2 + \sum_{i=1}^{n} b_i\alpha_k^i = y_{2k}. \end{aligned} \tag{7}$$

The unknowns in these equations are a_i, b_i, s_1, s_2, while all other parameters are known to the adversary. We will now prove two useful results concerning this scenario. First, given (5), all p^2 options for $(s_1, s_2) \in \mathbb{F}_p^2$ are possible. Second, given a pair of secrets, the shares $y_{11}, \dots, y_{1k}, y_{21}, \dots, y_{2k}$ distribute almost uniformly. We will soon make this statement precise by analyzing how the matrix $\begin{pmatrix} y_{11} & \cdots & y_{1k} \\ y_{21} & \cdots & y_{2k} \end{pmatrix}$ is distributed over $M_{2\times k}(\mathbb{F}_p)$, given a pair of secrets (s_1, s_2), and show that this distribution is statistically close to the uniform distribution. Let (s_1, s_2) be a pair of secrets, and $Y_{(s_1,s_2)}$ be the $M_{2\times k}(\mathbb{F}_p)$-valued random variable indicating the matrix $\begin{pmatrix} y_{11} & \cdots & y_{1k} \\ y_{21} & \cdots & y_{2k} \end{pmatrix}$ induced by (s_1, s_2). We will show that the statistical difference [20] between the distributions $Y_{(s_1,s_2)}$ and the uniform distribution over $M_{2\times k}(\mathbb{F}_p)$ is $\approx \frac{1}{p^{n-k}}$. Since statistical difference is a metric, we will conclude by the triangle inequality that the statistical difference between two such distributions, $Y_{(s_1,s_2)}$ and $Y_{(s_1',s_2')}$, is no more than $\approx \frac{2}{p^{n-k}}$.

To this end, we need the following theorem. Denote

$$U = \begin{pmatrix} \sum_{i=1}^{n} a_i\alpha_1^i, & \dots & , \sum_{i=1}^{n} a_i\alpha_k^i \\ \sum_{i=1}^{n} b_i\alpha_1^i, & \dots & , \sum_{i=1}^{n} b_i\alpha_k^i \end{pmatrix}.$$

We call U *the result matrix.*

Theorem 2 *The distribution of the result matrix is given by*

$$P\left[U = \begin{pmatrix} y_1 & \cdots & y_k \\ y_1' & \cdots & y_k' \end{pmatrix}\right] = \begin{cases} \frac{1}{p^n} + \frac{(p^{n-k}-1)(p^{n-k-1}-1)}{p^n(p^{n-1}-1)}, & \begin{pmatrix} y_1 & \cdots & y_k \\ y_1' & \cdots & y_k' \end{pmatrix} = \begin{pmatrix} 0 & \dots & 0 \\ 0 & \dots & 0 \end{pmatrix}, \\ \frac{p^{n-k-1}(p^{n-k}+p-1)}{p^n(p^{n-1}-1)}, & \begin{pmatrix} y_1 & \cdots & y_k \\ y_1' & \cdots & y_k' \end{pmatrix} \in \Omega, \\ \frac{p^{n-k-1}(p^{n-k}-1)}{p^n(p^{n-1}-1)}, & \text{otherwise}, \end{cases}$$

where Ω is a proper subset of $M_{2\times k}(\mathbb{F}_p)$, with cardinality of $(p^k - 1)(p^{k-1} - 1)$.

Proof of Theorem 2 Since p and $\alpha_1, \dots, \alpha_k$ are set, the result matrix U depends only on the Q-choice of $v \in \mathcal{V}_p$. Using the same notation for M_v as in the proof of Theorem 1, we state the connection between U and v. For $\alpha_1, \dots, \alpha_k \in \mathbb{F}_p^\times$, we define a mapping $\rho : \mathcal{V}_p \to M_{2\times k}(\mathbb{F}_p)$ by

$$\rho(v) = M_v \begin{pmatrix} \alpha_1 & \dots & \alpha_k \\ \vdots & & \vdots \\ \alpha_1^n & \dots & \alpha_k^n \end{pmatrix}.$$

Thus,

$$P\left[U = \begin{pmatrix} y_1 & \cdots & y_k \\ y_1' & \cdots & y_k' \end{pmatrix}\right] = P\left[\rho(v) = \begin{pmatrix} y_1 & \cdots & y_k \\ y_1' & \cdots & y_k' \end{pmatrix}\right].$$

Let $\begin{pmatrix} y_1 & \cdots & y_k \\ y_1' & \cdots & y_k' \end{pmatrix} \in M_{2\times k}(\mathbb{F}_p)$. We compute $P\left[U = \begin{pmatrix} y_1 & \cdots & y_k \\ y_1' & \cdots & y_k' \end{pmatrix}\right]$ by finding the number of elements $v \in V_p$ for which $\rho(v) = \begin{pmatrix} y_1 & \cdots & y_k \\ y_1' & \cdots & y_k' \end{pmatrix}$, and use the probability Q defined above. These elements are exactly the elements $(a_1, \dots, a_n, b_1, \dots, b_n) \in V_p$ with $(a_1, \dots, a_n) \neq \overline{0} \neq (b_1, \dots, b_n)$ that solve the system of equations

$$\begin{array}{lll} \mathrm{I}_1 & \sum_{i=1}^{n} a_i\alpha_1^i = y_1, & \\ \vdots & \vdots & \\ \mathrm{I}_k & \sum_{i=1}^{n} a_i\alpha_k^i = y_k, & \\ \mathrm{II}_1 & \sum_{i=1}^{n} b_i\alpha_1^i = y_1', & (8) \\ \vdots & \vdots & \\ \mathrm{II}_k & \sum_{i=1}^{n} b_i\alpha_k^i = y_k', & \\ \mathrm{III} & \sum_{i=1}^{n} a_i b_{n+1-i} = 0. & \end{array}$$

We solve (6) and analyze the number of solutions for given $y_1, \dots, y_k, y_1', \dots, y_k'$. The sub-system (6)$\mathrm{I}_1$-$\cdots$-(6)$\mathrm{I}_k$ consists of k independent equations with n variables $a_1, \dots, a_n$. Its independence follows from the fact that the matrix of the coefficients $(\alpha_j^i)_{i,j}$ is a sub-matrix of Vandermonde matrix with distinct generators $\alpha_1, \dots, \alpha_k$. Hence, (6)$\mathrm{I}_1$-$\cdots$-(6)$\mathrm{I}_k$ has p^{n-k} solutions $(a_1, \dots, a_n)$. For each of them, the system (6)II_1-$\cdots$-(6)II_k-(6)III consists of $k+1$ equations with n variables $b_1, \dots, b_n$. Is this system independent? The equations (6)II_1-$\cdots$-(6)II_k are independent for the same reason that (6)I_1-$\cdots$-(6)I_k are. Hence, we only need to find out whether (6)III is dependent of (6)II_1-$\cdots$-(6)II_k. This may happen only if there exist $c_1, \dots, c_k \in \mathbb{F}_p$, such that $a_{n+1-i} = \sum_{j=1}^{k} c_j \cdot \alpha_j^i$ for all $1 \leq i \leq n$. Replacing i for $n+1-i$ and using the fact that $\alpha_j^{n+1} = 1$, we get equivalently that $a_i = \sum_{j=1}^{k} c_j \cdot \alpha_j^{-i}$. Now, a_i must satisfy (6)I_1-$\cdots$-(6)I_k, so we replace each a_i in (6)I_1-$\cdots$-(6)I_k with $\sum_{j=1}^{k} c_j \cdot \alpha_j^{-i}$ and get

$$\begin{aligned} &\mathrm{I}_1 && \sum_{i=1}^{n} \alpha_1^i \cdot \left(\sum_{j=1}^{k} c_j \cdot \alpha_j^{-i} \right) = y_1, \\ &\vdots && \vdots \\ &\mathrm{I}_k && \sum_{i=1}^{n} \alpha_k^i \cdot \left(\sum_{j=1}^{k} c_j \cdot \alpha_j^{-i} \right) = y_k. \end{aligned} \tag{9}$$

Given $y_1, \dots, y_k$, this is a system of k equations with k unknowns $c_1, \dots, c_k$. Write (6.1) in the form

$$\begin{aligned} &\mathrm{I}_1 && \sum_{j=1}^{k} c_j \cdot \sum_{i=1}^{n} \left(\frac{\alpha_j}{\alpha_1} \right)^i = y_1, \\ &\vdots && \vdots \\ &\mathrm{I}_k && \sum_{j=1}^{k} c_j \cdot \sum_{i=1}^{n} \left(\frac{\alpha_j}{\alpha_k} \right)^i = y_k. \end{aligned} \tag{10}$$

Now,

$$\sum_{i=1}^{n} \left(\frac{\alpha_j}{\alpha_l} \right)^i = \begin{cases} \sum_{i=1}^{n} 1 = n, & j = l, \\ \frac{\alpha_j}{\alpha_l} \cdot \frac{1-\left(\frac{\alpha_j}{\alpha_l}\right)^n}{1-\frac{\alpha_j}{\alpha_l}} = \frac{\alpha_j \cdot \left(1-\left(\frac{\alpha_j}{\alpha_l}\right)^{-1}\right)}{\alpha_l-\alpha_j} = -1, & j \neq l. \end{cases}$$

Hence, we may write (6.2) in the form

$$\begin{pmatrix} n & \dots & -1 \\ \vdots & \ddots & \vdots \\ -1 & \dots & n \end{pmatrix} \begin{pmatrix} c_1 \\ \vdots \\ c_k \end{pmatrix} = \begin{pmatrix} y_1 \\ \vdots \\ y_k \end{pmatrix}. \tag{11}$$

The matrix A on the left-hand side of (7) has ns on the main diagonal and -1 elsewhere. Namely, it can be generated by cyclic permutations of its first row (or column). A matrix like that is a *circulant matrix*. We compute its determinant using [19] (or directly) to get $\det(A) = (n-k+1)(n+1)^{k-1}$. Since $k < n < p$, we have $\det(A) \neq 0$, and hence A is invertible. Denote $\overline{c} = (c_1, \dots, c_k)^T$ and $\overline{y}$ the result vector of (7). We solve (7) to get the unique solution of this system

$$\overline{c} = A^{-1}\overline{y}. \tag{12}$$

For given $\overline{y} = (y_1, \dots, y_k)^T$, set $\overline{c} = A^{-1}\overline{y}$. Then $\overline{a}_0 = (a_1, \dots, a_n)$ with $a_i = \sum_{j=1}^{k} c_j \alpha_j^{-i}$ is a solution for (6)I_1-⋯-(6)I_k for which the left-hand side of (6)III is dependent of the left-hand side of (6)II_1-⋯-(6)II_k. Any other solution $(a_1, \dots, a_n) \neq \overline{a}_0$ of (6)I_1-⋯-(6)I_k yields an independent system (6)II_1-⋯-(6)II_k-(6)III. For such $\overline{a}_0$, the right-hand side of (6)III will be dependent of the right-hand side of (6)II_1-⋯-(6)II_k if $\sum_{j=1}^{k} y'_j \cdot c_j = 0$. Denoting $(y'_1, \dots, y'_k)^T = \overline{y'}$, we write that condition equivalently as $\langle \overline{y'}, \overline{c} \rangle = 0$.

To conclude, given $\begin{pmatrix} y_1 \dots y_k \\ y'_1 \dots y'_k \end{pmatrix} \in M_{2\times k}(\mathbb{F}_p)$, set $\overline{c} = A^{-1}\overline{y}$ and $\overline{a}_0 = (a_1, \dots, a_n)$ with $a_i = \sum_{j=1}^{k} c_j \alpha_j^{-i}$. If $\langle \overline{y'}, A^{-1}\overline{y} \rangle = 0$ then $\overline{a}_0$ is a solution of (6)I_1-⋯-(6)I_k for which (6)II_1-⋯-(6)II_k-(6)III has p^{n-k} solutions. If $\langle \overline{y'}, A^{-1}\overline{y} \rangle \neq 0$ then $\overline{a}_0$ is a solution of (6)I_1-⋯-(6)I_k for which (6)II_1-⋯-(6)II_k-(6)III has no solutions.

We can now count the total number of solutions $(a_1, \dots, a_n, b_1, \dots, b_n)$ of (6) in each of the following cases.

- *Case 1.* $\overline{y} = \overline{y'} = \overline{0}$. In this case, one solution is the trivial solution, $(a_1, \dots, a_n, b_1, \dots, b_n) = \overline{0}$. By (12) we get here $\overline{c} = \overline{0}$, implying $\overline{a}_0 = \overline{0}$. Now, (6)$\mathrm{I}_1$-⋯-(6)$\mathrm{I}_k$ has p^{n-k} solutions $(a_1, \dots, a_n)$. The solution $\overline{a}_0$ yields p^{n-k} solutions $(b_1, \dots, b_n)$ for (6)II_1-⋯-(6)II_k-(6)III. Among them, only $\overline{b} = \overline{0}$ is acceptable, but we have already counted it. So we are left with $p^{n-k} - 1$ solutions $\overline{a}$ for (6)I_1-⋯-(6)I_k. Each of these yields p^{n-k-1} solutions $\overline{b}$ for (6)II_1-⋯-(6)II_k-(6)III. The vector $\overline{b} = \overline{0}$ is always one of them, so we omit it. All in all we get a total of $1 + (p^{n-k} - 1)(p^{n-k-1} - 1)$ valid solutions for (6).
- *Case 2.* $\overline{y} = \overline{0}, \overline{y'} \neq \overline{0}$.
  By (12), we get again $\overline{c} = \overline{0}$, implying $\overline{a}_0 = \overline{0}$. Since $\overline{y'} \neq \overline{0}$, $\overline{b} = \overline{0}$ is not a solution of (6)II_1-⋯-(6)II_k, we obtain no valid solutions for $\overline{a}_0 = \overline{0}$. Each of the other $p^{n-k} - 1$ solutions $\overline{a}$ of (6)I_1-⋯-(6)I_k yields p^{n-k-1} solutions $\overline{b}$ of (6)II_1-⋯-(6)II_k-(6)III, all of which are valid. All in all we get a total of $p^{n-k-1}(p^{n-k} - 1)$ valid solutions for (6).
- *Case 3.* $\overline{y'} = \overline{0}, \overline{y} \neq \overline{0}$.
  Analogous to Case 2.
- *Case 4.* $\overline{y} \neq \overline{0} \neq \overline{y'}$ with $\langle \overline{y'}, A^{-1}\overline{y} \rangle \neq 0$.
  In this case there are no solutions with $\overline{a} = \overline{0}$ or $\overline{b} = \overline{0}$. Here, $\overline{a}_0$ is a solution for (6)I_1-⋯-(6)I_k which yields no solution of (6)II_1-⋯-(6)II_k-(6)III. For each of the other $p^{n-k} - 1$ solutions of (6)I_1-⋯-(6)I_k there are p^{n-k-1} solutions of (6)II_1-⋯-(6)II_k-(6)III. Hence, we get a total of $p^{n-k-1}(p^{n-k} - 1)$ valid solutions for (6).
- *Case 5.* $\overline{y} \neq \overline{0} \neq \overline{y'}$ with $\langle \overline{y'}, A^{-1}\overline{y} \rangle = 0$.
  As in the previous case, there are no solutions with $\overline{a} = \overline{0}$ or $\overline{b} = \overline{0}$. Here, $\overline{a}_0$ is a solution of (6)I_1-⋯-(6)I_k which yields p^{n-k} solutions of (6)II_1-⋯-(6)II_k-(6)III. For each of the other $p^{n-k} - 1$ solutions of (6)I_1-⋯-(6)I_k there are p^{n-k-1} solutions for (6)II_1-⋯-(6)II_k-(6)III. Hence, we get a total of $p^{n-k-1}(p^{n-k} - 1) + p^{n-k} = p^{n-k-1}(p^{n-k} + p - 1)$ valid solutions for (6).

Denote

$$\Omega = \left\{ \begin{pmatrix} y_1 \dots y_k \\ y'_1 \dots y'_k \end{pmatrix} \in M_{2\times k}(\mathbb{F}_p) \middle| \overline{y} \neq \overline{0} \neq \overline{y'}, \langle \overline{y'}, A^{-1}\overline{y} \rangle = 0 \right\}.$$

To compute $|\Omega|$, observe that $\overline{y}$ can be chosen in $p^k - 1$ different ways. For each of these, the condition $\langle \overline{y'}, A^{-1}\overline{y} \rangle = 0$ is a linear equation with p^{k-1} solutions. We omit the trivial solution and get $|\Omega| = (p^k - 1)(p^{k-1} - 1)$. By the definition of Q, the rest follows. □

An immediate consequence of Theorem 2 is that, given (5), all p^2 options for $(s_1, s_2) \in \mathbb{F}_p^2$ are indeed possible: if the adversary is holding $\begin{pmatrix} y_{11} & \dots & y_{1k} \\ y_{21} & \dots & y_{2k} \end{pmatrix} \in M_{2\times k}(\mathbb{F}_p)$, then, for each of the p^2 possible pairs of secrets $(s_1, s_2) \in \mathbb{F}_p^2$, there is a single suitable $\begin{pmatrix} y_1 & \dots & y_k \\ y'_1 & \dots & y'_k \end{pmatrix} \in M_{2\times k}(\mathbb{F}_p)$. This matrix is $\begin{pmatrix} y_1 & \dots & y_k \\ y'_1 & \dots & y'_k \end{pmatrix} = \begin{pmatrix} y_{11} - s_1 & \dots & y_{1k} - s_1 \\ y_{21} - s_2 & \dots & y_{2k} - s_2 \end{pmatrix}$.

Since all matrices $\begin{pmatrix} y_1 & \dots & y_k \\ y'_1 & \dots & y'_k \end{pmatrix} \in M_{2\times k}(\mathbb{F}_p)$ occur with positive probability, the adversary simply does not have enough information to determine the secrets. Now, not all elements $\begin{pmatrix} y_1 & \dots & y_k \\ y'_1 & \dots & y'_k \end{pmatrix}$ have the same probability. According to Theorem 2, exactly $(p^k - 1)(p^{k-1} - 1) + 1$ out of the p^{2k} elements of $M_{2\times k}(\mathbb{F}_p)$ have a slightly larger probability. We use the statistical difference function to measure the leakage of information: if (s_1, s_2) is a pair of secrets, we denote by $Y_{(s_1,s_2)}$ the $M_{2\times k}(\mathbb{F}_p)$-valued random variables indicating the matrix $\begin{pmatrix} y_{11} & \dots & y_{1k} \\ y_{21} & \dots & y_{2k} \end{pmatrix}$ induced by (s_1, s_2), over the Q-picking of v from V_p. We compute the statistical difference $\mathrm{SD}(Y_{(s_1,s_2)}, \mathbb{U})$ between the distribution $Y_{(s_1,s_2)}$ and the uniform distribution over $M_{2\times k}(\mathbb{F}_p)$:

$$\begin{aligned}
\mathrm{SD}(Y_{(s_1,s_2)}, \mathbb{U}) &= \frac{1}{2} \cdot \sum_{Y \in M_{2\times k}(\mathbb{F}_p)} \left| P\left[Y_{(s_1,s_2)} = Y\right] - P\left[\mathbb{U} = Y\right] \right| \\
&= \frac{1}{2} \cdot \sum_{\begin{pmatrix} y_1 & \dots & y_k \\ y'_1 & \dots & y'_k \end{pmatrix}} \left| P \left[\begin{pmatrix} s_1 + \sum_{i=1}^n a_i\alpha_1^i & \dots & s_1 + \sum_{i=1}^n a_i\alpha_k^i \\ s_2 + \sum_{i=1}^n b_i\alpha_1^i & \dots & s_2 + \sum_{i=1}^n b_i\alpha_k^i \end{pmatrix} = \begin{pmatrix} y_1 & \dots & y_k \\ y'_1 & \dots & y'_k \end{pmatrix}\right] - \frac{1}{p^{2k}} \right| \\
&= \frac{1}{2} \cdot \sum_{\begin{pmatrix} y_1 & \dots & y_k \\ y'_1 & \dots & y'_k \end{pmatrix}} \left| P \left[\begin{pmatrix} \sum_{i=1}^n a_i\alpha_1^i & \dots & \sum_{i=1}^n a_i\alpha_k^i \\ \sum_{i=1}^n b_i\alpha_1^i & \dots & \sum_{i=1}^n b_i\alpha_k^i \end{pmatrix} = \begin{pmatrix} y_1 - s_1 & \dots & y_k - s_1 \\ y'_1 - s_2 & \dots & y'_k - s_2 \end{pmatrix}\right] - \frac{1}{p^{2k}} \right|.
\end{aligned}$$

Using Theorem 2, a straightforward computation yields

$$\begin{aligned}
\mathrm{SD}(Y_{(s_1,s_2)}, \mathbb{U}) &= \frac{(p^k - p^{k-1} + 2)(p^k - 1)(p^{k-1} - 1)}{p^{2k}(p^{n-1} - 1)} \\
&\approx \frac{p^{3k-1}}{p^{2k} \cdot p^{n-1}} = \frac{1}{p^{n-k}}.
\end{aligned}$$

Since the statistical difference is a metric, by the triangle inequality we get that

$$\mathrm{SD}(Y_{(s_1,s_2)}, Y_{(s'_1,s'_2)}) \approx \frac{2}{p^{n-k}}$$

for any couple of distributions induced by pairs of secrets, $(s_1, s_2), (s'_1, s'_2) \in \mathbb{F}_p^2$.

IT-Secure Dynamic Somewhat Homomorphic Database Outsourcing

Our scheme can be used to perform homomorphic evaluation of quadratic functions over variables $s_1, \dots, s_m$, and arbitrarily long 2-CNF circuits. A quadratic function over the variables $s_1, \dots, s_m$ is of the form

$$F(s_1, \dots, s_m) = \sum_{1 \le i,j, \le m} r_{ij} s_i s_j + \sum_{k=1}^{m} t_k s_k + c,$$

with $r_{ij}, t_k, c \in \mathbb{F}_p$. There are $p^{\frac{1}{2}(m^2+3m+2)}$ such functions. We can use our scheme to homomorphically evaluate F. For each of the $\frac{m^2+m}{2}$ pairs of variables s_i, s_j, use our scheme to generate a pair of 1-homomorphic-multiplicative-polynomials f_{ij}, f_{ji}, and distribute s_i, s_j among the parties. This pre-processing stage requires the user send to the servers $O(m^2)$ data, but now F can be homomorphically evaluated in a straightforward way. Each party P_l simply evaluates F over its shares of the secrets and sends the result y_l to the dealer. The dealer in turn calculates the polynomial g going through the points (α_l, y_l) and finds $g(0) = F(s_1, \dots, s_m)$.

The space complexity of the aforementioned scheme may be reduced in the cost of lower security parameters. We now show how one can adjust the suggested scheme and achieve a scheme with $O(m)$ cyphertext instead of $O(m^2)$. Pick an element $\overline{v} = (a_1, \dots, a_n, b_1, \dots, b_n)$ from $\mathcal{V}_p$ under the condition that $\sum_{i=1}^n a_i\alpha_l^i \neq 0 \neq \sum_{i=1}^n b_i\alpha_l^i$ for $1 \le l \le N$. Pick $k_1, \dots, k_m, l_1, \dots, l_m$ from $\mathbb{F}_p$ uniformly at random and set $f_j(x) = s_j + k_j \sum_{i=1}^n a_i x^i$, and $h_j(x) = s_j + l_j \sum_{i=1}^n b_i x^i$, $1 \le j \le m$. Distribute to party P_l the $2m$ vector $\Big(f_1(\alpha_l), \dots, f_m(\alpha_l), h_1(\alpha_l), \dots, h_m(\alpha_l)\Big)$. Now, each party evaluates F over his shares of the secrets. The linear parts of F are computed by each party using either f_k or h_k. The quadratic parts of F are evaluated by each party as $f_i(\alpha_l) \cdot h_j(\alpha_l)$.

This scheme is perfectly secure against a single party attack, but is insecure against coalitions of two or more parties.

In various applications, the number of variables is growing over time. In that case, the method described above can be modified to allow new variables to be joined with the primary ones. Explicitly, assume a dealer has $s_1, \ldots, s_m \in \mathbb{F}_p$, and $s_{m+1}, \ldots, s_{m+k}$ are k more variables whose value may not be determined yet, and are expected to be determined and joined with $s_1, \ldots, s_m$ in the future. We wish to share $s_1, \ldots, s_m$ among N parties, in a way that (a) enables homomorphic evaluation of quadratic functions over the m variables; (b) will enable to share, in the future, the k additional variables among the parties; (c) will enable homomorphic evaluation of quadratic functions over the $m + k$ variables. We wish to achieve all that without keeping $s_1, \ldots, s_m$ in memory.

We now demonstrate how these dynamic properties are obtained. For each of the pairs of variables s_i, s_j, $1 \leq i \leq m, i \leq j \leq m$, use our scheme to generate a 1-homomorphic-multiplicative-pair of polynomials, f_{ij},f_{ji}, and distribute s_i, s_j among N parties. As in the non-dynamic version, quadratic functions over $s_1, \ldots, s_m$ can now be homomorphically evaluated. For each of the pairs s_i, s_j, $1 \leq i \leq m$, $m + 1 \leq j \leq m + k$, use our scheme to generate a 1-homomorphic-multiplicative-pair of polynomials, f_{ij},f_{ji}. Assuming $s_{m+1}, \ldots, s_{m+k}$ are not known yet, for $m + 1 \leq j \leq m + k$ let the free coefficient of f_{ji} be zero, and keep f_{ji} in memory. Distribute s_i to the parties using the first of each pair of 1-homomorphic-multiplicative polynomials, i.e., using f_{ij}. Now, when the values of s_j, $m + 1 \leq j \leq m + k$, are determined, add each of them to the corresponding polynomial f_{ji}, $1 \leq i \leq m$, and distribute s_j among the parties. In addition to that, for each pair of variables s_i, s_j, $m + 1 \leq i \leq m + k, i \leq j \leq m + k$, generate a 1-homomorphic-multiplicative-pair of polynomials, f_{ij},f_{ji}, and distribute s_i, s_j among the parties. Now, quadratic functions over the $m + k$ variables, $s_1, \ldots, s_{m+k}$, can be homomorphically evaluated in a straightforward way as in the non-dynamic version described above.

A 2-CNF expression over literals $s_1, \ldots, s_m$ is an expression of the form $(s_{i_1} \vee s_{i_2}) \wedge \cdots \wedge (s_{i_{2t-1}} \vee s_{i_{2t}})$. As we work in $\mathbb{F}_p$, we replace the logic values *True* and *False* with the elements 1 and 0 in $\mathbb{F}_p$, respectively (other elements of $\mathbb{F}_p$ are not logically defined). Logic operations are replaced with $\mathbb{F}_p$ operations as follows. Given literals s_1 and s_2, disjunction is implemented by $s_1 + s_2 - s_1 s_2$ and conjunction is considered as addition in $\mathbb{F}_p$. Negation of s_1 is $1 - s_1$. Then, a 2-CNF expression of length $2t$ is a multi-variable quadratic function, and is assigned *True* if the function is evaluated to $t \in \mathbb{F}_p$, and *False* otherwise. There are 2^{2m^2+m} such expressions that can be homomorphically evaluated using our scheme.

Known IT-Secure Somewhat Homomorphic Solutions are Not Dynamic

We now review several conventional methods for IT-secure somewhat homomorphic outsourcing and examine their (non-) dynamic features.

One may consider using Shamir's standard scheme and supporting homomorphic multiplication of secrets by just taking the polynomials to be of lower degree to-begin-with. However, such a solution yields a smaller threshold, e.g., if one runs Shamir's standard secret sharing scheme with four parties, and would like to be able to extract a product of two secrets, he/she would be obligated to work with linear polynomials. In that case, if an adversary manages to discover two of the shares of a certain secret, then the secret is revealed. If one tried to work with quadratic polynomials in the standard scheme (to achieve security against coalitions of two parties), then the product polynomial would be of degree 4, and it requires five parties to extract the product. Hence, this method is not somewhat homomorphic. In our scheme, even if an adversary manages to reveal two out of four shares of a certain secret, the secret is information-theoretically kept. We proved that each of the parties holding two correlated secret shares gains absolutely no information about the secrets. We also proved that a coalition of up to $N - 2$ curious parties still cannot reveal the exact value of (s_1, s_2), and that the statistical difference is negligible.

Now, to achieve a somewhat homomorphic effect, one may consider using Shamir's standard scheme and sharing, for each pair of secrets, their product. This method enables homomorphic evaluation of quadratic functions and 2-CNF circuits over m secrets using $O(m^2)$ ciphertext. Nevertheless, it is not dynamic since, in this solution, to allow new secrets to be joined with the primary ones, the user must keep the old secrets in memory. In our scheme, the primary secrets are not required to be stored in memory once they were shared. For example, assume a dealer holds three elements s_1, s_2, s_3 of a finite field $\mathbb{F}_p$. Following the simple scheme described above, the dealer computes the products $s_1 s_2, s_1 s_3, s_2 s_3$ and shares the six elements $s_1, s_2, s_3, s_1 s_2, s_1 s_3, s_2 s_3$ among N parties. After some time, the dealer obtains a fourth secret, s_4 (that was not known beforehand). To enable evaluation of quadratic functions over $\{s_1, s_2, s_3, s_4\}$ the dealer must compute the products $s_1 s_4, s_2 s_4, s_3 s_4$ and share them among the parties. To this end, the dealer must store s_1, s_2, s_3 in memory. In contrast, using our scheme, once the dealer shared s_1, s_2, s_3 among the parties (using pairs of 1-homomorphic-multiplicative-polynomials), there is no need to keep the secrets in memory by the dealer. Instead, the dealer prepares pairs of such 1-homomorphic-multiplicative polynomials for future computation of the products with the new secret: $s_1 s_4, s_2 s_4, s_3 s_4$. To support computation of the product $s_i s_4$ ($1 \le i \le 3$), the user Q-picks from $\mathcal{V}_p$ a $2n$-tuple $(a_1, \ldots, a_n, b_1, \ldots, b_n)$, uses $a_1, \ldots, a_n$ as the non-free coefficients for sharing s_i, and keeps in memory $b_1, ..., b_n$ to be used as the non-free coefficients of the polynomials used for sharing s_4 in the future.

One may consider using (a variant of) Beaver's multiplication trick [4] to enable homomorphic multiplication of m secrets as follows. First, the user N-out-of-N secret shares s_k (for $1 \le k \le m$) among the servers using an additive secret sharing scheme. E.g., for each secret s_k, the user randomly uniformly picks $N-1$ elements of $\mathbb{F}_p$ and sets an N'th share to satisfy the condition that the sum of all N elements equals s_k. Then, for each pair of secrets (s_i, s_j) the user generates independent one-time secret $\mathbb{F}_p$-triples $(\rho_{ij}, \sigma_{ij}, \rho_{ij} \cdot \sigma_{ij})$, N-out-of-N secret shares the triples among the parties using the same additive secret sharing scheme, and reveals $s_i + \rho_{ij}$ and $s_j + \sigma_{ij}$ to the network. Now, since $s_i \cdot s_j = (s_i + \rho_{ij} - \rho_{ij}) \cdot (s_j + \sigma_{ij} - \sigma_{ij}) = (s_i + \rho_{ij}) \cdot (s_j + \sigma_{ij}) - \rho_{ij}(s_j + \sigma_{ij}) - \sigma_{ij}(s_i + \rho_{ij}) + \rho_{ij}\sigma_{ij}$, IT-secure communicationless evaluation of quadratic functions over the set of secrets is possible. However, this scheme is not dynamic. If new secrets are to be joined with the primary ones then, to enable homomorphic multiplication of old and new secrets, the user must generate an independent triple for the new and old secrets and publish the corresponding values.

For example, assume (again) that a dealer holds three elements s_1, s_2, s_3 of $\mathbb{F}_p$. Following the variant of Beaver's trick described above, the dealer N-out-of-N secret shares s_1, s_2, s_3 among the parties, generates and secret shares independent random triples $(\rho_{ij}, \sigma_{ij}, \rho_{ij} \cdot \sigma_{ij})$ (for $1 \le i \le j \le 3$), and for each pair of secrets reveals the corresponding values to the network. After some time, the dealer obtains a fourth secret, s_4 (that was not known beforehand). To enable homomorphic multiplication of, say, s_1 and s_4, the dealer must generate a random triple $(\rho_{14}, \sigma_{14}, \rho_{14} \cdot \sigma_{14})$, publish $s_1 + \rho_{14}$ and $s_4 + \sigma_{14}$, and secret share $\rho_{14} \cdot \sigma_{14}$. Now, to publish $s_1 + \rho_{14}$, the user must keep s_1 in memory, which results in a non dynamic scheme. In an attempt to avoid it, assume that the dealer picked ρ_{14} before s_4 was known and already published $s_1 + \rho_{14}$ to the network. Now, when the time comes and s_4 is known, the user should publish $s_4 + \sigma_{14}$ to the network and secret share $\rho_{14} \cdot \sigma_{14}$. To this end, the user must keep ρ_{14} in memory. Now, since $s_1 + \rho_{14}$ is public, keeping ρ_{14} in memory is (security-wise) equivalent to storing s_1 in memory. The fact that the user is required to store in memory all the ρ_{ij}s and that the values $s_i + \rho_{ij}$ are all public creates a SPOF and makes this scheme non-dynamic. We conclude that this simple variation of Beaver's multiplication trick cannot be used to construct a dynamic solution for the outsourcing problem.

Conclusions

We have proposed a scheme to perform a homomorphic multiplication over secret shares without increasing the number of parties required to extract the product. In our scheme, we have dealt with N parties and used polynomials of degree $N-1$. We have shown how to use our scheme to perform homomorphic and information-theoretically secure evaluation of quadratic functions and 2-CNF circuits over a dynamic database of m secrets with $O(m^2)$ ciphertext.

Our scheme has several practical applications. For example, every problem in 2-SAT is reducible to solving a 2-CNF Boolean formula. Solving well-known problems in 2-SAT privately can be very useful. Conflict-free placement of geometric objects, data clustering, scheduling, and discrete tomography are but several out of many interesting and practical problems in 2-SAT. Our scheme may also suggest an alternative for applications in which Beaver's multiplication trick is used to enable homomorphic multiplication of shared secrets.

The scheme we suggest here is somewhat surprising. We multiply two secrets that were shared via degree $N-1$ polynomial and manage to extract the product using no more than the N parties we began with, proving it to be IT-secure. The innovation is in the function sieving method, and in the way that we built the set $\mathcal{V}_p$ and defined the probability Q over it.

One of the main advantages of our scheme is being dynamic. To emphasize the virtue of dynamic schemes, consider a scenario in which a user holds a database containing highly confidential information. It can be, e.g., a database containing private medical information of patients of a medical institution, biomedical information of citizens of a specific country, financial information regarding stocks in a market, or bank account details of clients of a big bank, etc. Keeping the entire database on a single server is risky since it creates a single point of failure (SPOF). If that server is breached, then the privacy of the entire information is compromised. An alternative solution is secret sharing the entire database among several servers and keep no plaintext copy of the database anywhere. This way, each server holds zero (or a negligible amount of) information, and security is maintained even if several servers (up to the threshold of the secret sharing scheme used) have been breached. Secret shared databases enable storing confidential information in a distributed fashion with no SPOF risk. Furthermore, in our scheme, the servers do not communicate with each other and hence each server may remain utterly oblivious to the number and identity of other servers that participate in the scheme. This fact reduces the risk of adversarial attacks of large coalitions of servers. In the distributed approach, when the user needs to observe a specific record in the database, she may retrieve the corresponding shares from the servers and reconstruct the plaintext.

Two essential requirements arise in this setting. The first is adding new records to the database over time, and the second is performing computations over the shared data. Distributing the database using Shamir's secret sharing scheme (as it is) enables adding new records to the database over time, but supports no homomorphic multiplication of secrets. If one also computes all the possible products of secrets and secret shares them among the servers, it enables evaluating quadratic functions over secrets, but, if new records are added to the database, they cannot be homomorphically multiplied with the primary secrets, since the corresponding products were not known when the database was shared. To support multiplications of old and new secrets, the user must keep a copy of the entire database in memory, which again creates a SPOF and contradicts the purpose of the distribution process. A dynamic scheme, like the one we suggest, enables the user to add new records to the secret shared database and evaluate quadratic functions over the entire set of secrets, old and new.

Finally, we believe that our approach and proof techniques may open an opportunity for fruitful research on secure distributed computing, as well as other applications.

Acknowledgements With pleasure, we thank Amos Beimel and Niv Gilboa for useful inputs.

Funding Research partially supported by the Lynne and William Frankel Center for Computer Science, the Rita Altura Trust Chair in Computer Science and also supported by a grant from the Ministry of Science, Technology and Space, Infrastructure Research in the Field of Advanced Computing and Cyber Security, the Israel & the Japan Science and Technology Agency (JST), and the German Research Funding Organization (DFG, Grant#8767581199).

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Applebaum B, Brakerski Z, Tsabary R. Perfect secure computation in two rounds. In: Theory of cryptography conference. New York: Springer; 2018. p. 152–174.
2. Akavia A, Gentry C, Halevi S, Leibovich M. Setup-free secure search on encrypted data: faster and post-processing free. New York: Technical report. Cryptology ePrint Archive Report; 2018.
3. Barkol O, Ishai Y, Weinreb E. On d-multiplicative secret sharing. J Cryptol. 2010;23(4):580–93.
4. Beaver D. Efficient multiparty protocols using circuit randomization. In: Annual international cryptology conference. New York: Springer; 1991. p. 420–432.
5. Ben-Or M, Goldwasser S, Wigderson A. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In: Providing sound foundations for cryptography: on the work of Shafi Goldwasser and Silvio Micali . 2019. p. 351–371.
6. Bitan D, Dolev S. Invited paper: Homomorphic operations techniques yielding communication efficiency. In: Devismes S, Mittal N editors. Stabilization, safety, and security of distributed systems—22nd international symposium, SSS 2020, Austin, TX, USA, November 18–21, 2020, proceedings. Lecture notes in computer science, vol 12514. New York: Springer; 2020. p. 16–28.
7. Blakley GR. Safeguarding cryptographic keys. In: 1979 international workshop on managing requirements knowledge (MARK). New York: IEEE; 1979. p. 313–318.
8. Boneh D, Goh E-J, Nissim K. Evaluating 2-dnf formulas on ciphertexts. In: Theory of cryptography conference. New York: Springer; 2005. p. 325–341.
9. Brakerski Z, Perlman R. Lattice-based fully dynamic multi-key the with short ciphertexts. In: Annual cryptology conference. New York: Springer; 2016. p. 190–213.
10. Chaum D, Crépeau C, Damgard I. Multiparty unconditionally secure protocols. In: Proceedings of the twentieth annual ACM symposium on theory of computing. New York: ACM; 1988. p. 11–19.
11. Dawson E, Donovan D. The breadth of Shamir's secret-sharing scheme. Comput Secur. 1994;13(1):69–78.
12. Damgård I, Ishai Y. Constant-round multiparty computation using a black-box pseudorandom generator. In: Annual international cryptology conference. New York: Springer; 2005. p. 378–394.
13. Dolev S, Li Y. Secret shared random access machine. In: Algorithmic aspects of cloud computing. New York: Springer; 2016. p 19–34.
14. Dolev S, Lahiani L, Yung M. Secret swarm unit reactive k-secret sharing. In: International conference on cryptology in India. New York: Springer; 2007. p. 123–137.
15. Dolev S, Gilboa N, Li X. Accumulating automata and cascaded equations automata for communicationless information theoretically secure multi-party computation. In: Proceedings of the 3rd international workshop on security in cloud computing. New York: ACM; 2015. p. 21–29.
16. Gentry C. A fully homomorphic encryption scheme. Stanford: Stanford University; 2009.
17. Gentry C, Halevi S, Smart NP. Fully homomorphic encryption with polylog overhead. In: Annual international conference on the theory and applications of cryptographic techniques. New York: Springer; 2012. p. 465–482.
18. Gentry CB, Halevi S, Smart NP. Homomorphic evaluation including key switching, modulus switching, and dynamic noise management. US Patent 9281941. 2016.
19. Gray RM et al. Toeplitz and circulant matrices: a review. Found Trends® Commun Inf Theory 2006; 2(3):155–239.
20. Goldreich O. Foundations of cryptography. Vol. 2. Basic applications. Cambridge: Cambridge University Press; 2009.
21. Goldreich O, Micali S, Wigderson A. How to play any mental game. In: Proceedings of the nineteenth annual ACM symposium on theory of computing. New York: ACM; 1987. p. 218–229.
22. Rivest RL, Adleman L, Dertouzos ML, et al. On data banks and privacy homomorphisms. Found Secur Comput. 1978;4(11):169–80.
23. Sander T, Young A, Yung M. Non-interactive cryptocomputing for nc/sup 1. In: 40th annual symposium on foundations of computer science (Cat. No. 99CB37039). New York: IEEE; 1999. p. 554–566.
24. Shamir A. How to share a secret. Commun ACM. 1979;22(11):612–3.
25. Smart Nigel P, Vercauteren Frederik. Fully homomorphic encryption with relatively small key and ciphertext sizes. In *International Workshop on Public Key Cryptography*, pages 420–443. Springer, 2010.

26. Van DM, Gentry C, Halevi S, Vaikuntanathan V. Fully homomorphic encryption over the integers. In: Annual international conference on the theory and applications of cryptographic techniques. New York: Springer; 2010. p. 24–43.
27. Xu J, Wei L, Zhang Y, Wang A, Zhou F, Gao C. Dynamic fully homomorphic encryption-based Merkle tree for lightweight streaming authenticated data structures. J Netw Comput Appl. 2018;107:113–24.