Hindawi
Complexity
Volume 2020, Article ID 1784276, 13 pages
https://doi.org/10.1155/2020/1784276

Research Article

A Group Identification Protocol with Leakage Resilience of Secret Sharing Scheme

Ping Li,[1,2] Shengjun Li,[3] Hongyang Yan,[2] Lishan Ke,[4] Teng Huang,[2] and Alzubair Hassan[2]

[1]*School of Computer Science, South China Normal University, Guangzhou 510631, China*
[2]*School of Computer Science and Cyber Engineering, Guangzhou University, Guangzhou 510006, China*
[3]*School of Information Science and Engineering, Qufu Normal University, Rizhao, China*
[4]*School of Mathematics and Information Science, Guangzhou University, Guangzhou 510006, China*

Correspondence should be addressed to Shengjun Li; qfnulsj@163.com

Received 7 November 2019; Revised 15 January 2020; Accepted 29 January 2020; Published 13 March 2020

Guest Editor: Xuyun Zhang

Secret sharing has been study for many years and has had a number of real-word applications. There are several methods to construct the secret-sharing schemes. One of them is based on coding theory. In this work, we construct a secret-sharing scheme that realizes an access structure by using linear codes, in which any element of the access structure can reconstruct the secret key. We prove that our scheme is a multiprover zero-knowledge proof system in the random oracle model, which shows that a passive adversary gains no information about the secret key. Our scheme is also a leakage-resilient secret-sharing scheme (LRSS) in the bounded-leakage model, which remain provably secure even if the adversary learns a bounded amount of leakage information about their secret key. As an application, we propose a new group identification protocol (GID-scheme) from our LRSS. We prove that our GID-scheme is a leakage-resilient scheme. In our leakage-resilient GID-scheme, the verifier believes the validity of qualified group members and tolerates l bits of adversarial leakage in the distribution protocol, whereas for unqualified group members, the verifier cannot believe their valid identifications in the proof protocol.

1. Introduction

The secret-sharing scheme, originally and independently introduced by Shamir [1] and Blakley [2], is a method in which a dealer selects a secret and distributes it as shares among a set of parties in the distribution stage. Only the predefined subsets of parties can reconstruct the secret from their shares, while others learning nothing about the secret in the reconstruction stage. These subsets are called qualified, and the monotonic collection of qualified subsets is called an *access structure* of the secret-sharing scheme. As a basic primitive in cryptography, the secret-sharing scheme has been used widely in security applications and protocols, such as threshold cryptography [3], secure multiparty computation [4], cloud computing [5–7], oblivious transfer [8], and access control [9].

In general, in the secret-sharing scheme (n parties and access structures are known in advance), there are two types of access structures: *threshold* and *nonthreshold*. In the *threshold* access structure, at least $t = t(n)$ qualified parties can reconstruct the secret key. In [10], the authors constructed an evolving secret-sharing scheme for a dynamic threshold access structure. In their scheme, the size of the qualified set increases if the number of parties increases. However, in the *nonthreshold* access structure, the size of the qualified set is not limited, i.e., any collection of the qualified subsets can reconstruct the secret key. If the nonthreshold access structure (a small monotonic span program) can be described, then an efficient secret-sharing scheme is realized [11]. For instance, given the forbidden graph access structure, Beimel et al. [12] proposed a linear secret-sharing scheme for forbidden graph access structures.

1.1. Secret-Sharing Scheme from Linear Codes. There is a natural correspondence between linear codes and a secret-sharing scheme. McEliece and Sarwate noted the relationship between Shamir–Blakley's secret-sharing scheme and Reed–Solomon codes in [13]. Since then, several secret-sharing schemes have been constructed in terms of linear error-correcting codes [14–16]. To construct a secret-sharing scheme from linear codes, Massey pointed out the relationship between the access structure and the minimal codewords of the dual code of the underlying code [17, 18]. Therefore, when designing a secret-sharing scheme based on linear codes, it is necessary to consider the open problem of how to determine the minimal codewords for certain linear codes. In this work, we assume that the codes and their dual codes are efficiently encodable and decodable, respectively. In this work, the relationship between secret-sharing scheme and linear codes is presented in Section 2.1.

1.2. Leakage-Resilient Secret-Sharing Scheme. In the application of secret-sharing scheme realizing the access structure, some parties exist that might cheat others by providing false secret keys under the sharing control, and the information about the secret key is leaked. To avoid this situation, it is necessary to consider the challenge of protecting the secret key of the dealer and the shares of parties against *information leakage*, i.e., in previous work, most researchers generally used *leakage-resilient cryptographic primitives* [19–24] and *leakage-resilient devices* [25, 26] to protect the security of the secret key. The cryptographic primitives and computational devices are said to be *leakage-resilient* if it remains secure in the presence of *bounded-leakage* of an internal (secret) state. In the work presented in [27], the authors defined the assumption that only the computation leaks information. In other words, there is no leakage without computation. However, this assumption does not guarantee security in the model because cold-boot attacks [28] may work [29]. Therefore, motivated by the work of Dziembowski and Pietrzak [30], we describe the leakage assumptions considered in this work as follows:

Independent leakage: the computation can be organized into rounds, and the leaks in each round are independent

Bounded leakage: in each round, the number of leakages are bound to some parameters,whereas the total leakage bit is bounded by l

Bounded domain: in fact, the leakage function takes as input *only* the secret state during the invocation

Formally, in the bounded-leakage model, an attacker can repeatedly and adaptively access to a *leakage oracle* and learn information about the secret key, as long as the total number of information leaked is bounded by some parameter l. An attacker chooses a sequence of polynomial-time-computable leakage functions $\{f_i : \{0,1\}^{|sk|} \longrightarrow \{0,1\}^{l_i}\}_{i=1}^{n}$ and obtains f (state), where state is the party P's secret state information at the end beginning of each round i and $\sum_i l_i \leq 1$.

1.3. Our Contributions. In this work, we construct a secret-sharing scheme for a given access structure arising from linear correcting codes. According to the definitions of security models and attack models, we prove that our protocol is an n-prover zero-knowledge proof system in the random oracle model, which reveals that the secret key can be shared repeatedly without leaking any information in the case of a passive attacker. Additionally, our protocol is a leakage-resilient secret-sharing scheme (LRSS) in the bounded-leakage model, which shows that it is leakage-resilient against (θ, n, l)-BCP.

In particular, as an application, our LRSS is a group identification scheme (GID-scheme); that is, all qualified parties can detect whether the dealer is cheating, and any verifier can detect whether unqualified parties are cheating. We also prove that our GID-scheme is leakage-resilient in the bounded-leakage model. The basic construction of our scheme relies on the following considerations:

Assume that a private channel exists in the distribution protocol of our protocol between every party and the dealer and that all the parties have an individual broadcast channel

Given the public key pk and l bits of secret key sk leakage, our protocol is performed between any probabilistic polynomial-time adversarial verifier V^* and an honest prover P, maintaining information-theoretic entropy and achieving security in the bounded-leakage model

For any adversarial prover, the corresponding secret key of the emulated identity's public key should be known

For one public key, the probability of an algorithm to find two distinct secret keys is negligible

1.4. Organization. This paper is organized as follows. In Section 2, we introduce some definitions and lemmas that are used in this work. We describe how to construct our protocol in Section 3 and provide several proofs of the properties of our protocol in Section 4. As an application, we propose a group identification protocol in Section 5. Finally, we provide the conclusions and future work on this topic in Section 6.

2. Preliminaries

Here, we introduce the notations and basic definitions used throughout this work. Let $\mathbb{N}$ and $\mathbb{R}$ be sets of natural numbers and real numbers, respectively. We write $[n]$ to indicate the set $\{1, \ldots, n\}$ of natural numbers $n \in \mathbb{N}$. Let $|x|$ denote the binary length of x.

2.1. Secret-Sharing Scheme from Linear Codes and Security Definitions. Let $\mathbb{F}_q$ denote a finite field where q is a prime. We write $\mathbb{F}_q^*$ for the set of nonzero elements of $\mathbb{F}_q$; then, $\mathbb{F}_q^*$ is a multiplicative cyclic group with $q-1$ elements, and any element in $\mathbb{F}_q^*$ has order dividing $q-1$. We use the symbol $\mathbb{F}_q^n$ to refer to an n-dimensional linear vector space over $\mathbb{F}_q$.

Let "‖" denote the concatenation of finite vector (bold letter), i.e., $\mathbf{x} = (x_1, x_2, \ldots, x_n) = (x_1 \| x_2 \| \ldots x_n) = (x_1 \| \mathbf{c}) \in \mathbb{F}_q^n$, where $\mathbf{c} = (x_2, \ldots, x_n) \in \mathbb{F}_q^{n-1}$.

Definition 1 (linear codes). An $[n, k]$ code C is a k-dimensional linear subspace of $\mathbb{F}_q^n$, which means that the sum of the two codewords of C is a codeword and that the product of any codeword by a field element is a codeword.

Definition 2 (generator matrix). A matrix $G = (\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_n)^\tau$ is called a generator matrix of an $[n+1, k]$ code C, if for every codeword z in C is a linear combination of the rows of G, i.e., $\mathbf{z} = r_0\mathbf{v}_0 + \cdots + r_{k-1}\mathbf{v}_{k-1} = (r_0, \ldots, r_{k-1})G$, where $\mathbf{v_i} = (g_{i0}, \ldots, g_{in})$, $g_{ij} \in \mathbb{F}_p$ for $0 \le i \le k-1, 0 \le j \le n$.

Let $\mathrm{P}_n = \{P_1, \ldots, P_n\}$ be a set of n parties. The definition of the access structure (monotone) is given as follows.

Definition 3 (access structure [31, 32]). *A* set

$$\mathrm{AS} = \left\{ A \subseteq 2^{\mathrm{P}_n} \,\middle|\, \mathrm{A\ can\ reconstruct\ the\ secret} \right\}, \quad (1)$$

is called an access structure, if it satisfies the monotone property, i.e., for any $A' \in \mathrm{AS}$ and $A' \subseteq A \subseteq 2^{\mathrm{P}_n}$, it holds $A \in \mathrm{AS}$.

Any subsets in AS are called qualified (or authorized), and the subsets that do not belong to AS are called unqualified (or unauthorized).

Definition 4 (secret-sharing scheme, SSS). Let Share be any probabilistic algorithm that takes as input a secret $s \in S$ and returns n shares $\mathbf{s} = (s_1, \ldots, s_n)$. Let Recon be a deterministic algorithm that takes as input the shares of a subset Λ and output a possible secret. Note that S is the domain of the secret key. We say that an (n, t)-secret-sharing schemeSSS = (Share, Recon) over field $\mathbb{F}_q$ for realizing an access structure AS, if it satisfies.

Correctness: for every secret $s \in S$ and every qualified set, $\Lambda \in \mathrm{AS}$ with $|\Lambda| \ge t$, and it has the equation $\mathrm{Recon}(\mathbf{s}_\Lambda, \Lambda) = s$

Security: for every unqualified set, $\Lambda \notin \mathrm{AS}$ with $|\Lambda| < t$, and two arbitrary distinct secrets $s^{(1)}, s^{(2)} \in S$, $\mathbf{s}_\Lambda^{(1)}$ is identically distributed to $\mathbf{s}_\Lambda^{(2)}$, where $\mathbf{s} \longrightarrow \mathrm{Share}(s)$ and $\mathbf{s}_\Lambda = \{s_i\}_{i \in \backslash \Lambda}$ are the completed shares of parties in Λ

Definition 5 (linear secret-sharing scheme, LSSS). An (n, t)-SSS = (Share, Recon) over field $\mathbb{F}_q$ is linear if the codomain of Share is the vector space $\mathbb{F}_q^n$, and Share is a $\mathbb{F}_q$-linear mapping and Share(s) is uniformly probability distributed over $\mathbb{F}_q^n$ for any $s \in \mathbb{F}_q$.

Based on the work in [17], we use a linear codes to construct an SSS as follows. An $[n+1, k]$ code C is a linear subspace of $\mathbb{F}_q^{n+1}$. Note that $G = (\mathbf{g}_0, \mathbf{g}_1, \ldots, \mathbf{g}_n)$ be the generator matrix for C, where $\mathbf{g}_i \in \mathbb{F}_q^k$ is the column vector of G, $0 \le i \le n$. In the SSS = (Share, Recon) constructed from C, the secret s is an element of $\mathbb{F}_q$, n parties $P_1, P_2, \ldots, P_n$ and a dealer D are involved. To compute the shares $s_1, \ldots, s_n$ of secret s, D performs the algorithm Share as follows.

The dealer D chooses a random codeword $\mathbf{u} = (u_0, u_1, \ldots, u_{k-1}) \in \mathbb{F}_q^k$ satisfying $s = \mathbf{u}\mathbf{g}_0$. Next, D computes the corresponding codeword by the following equation:

$$\mathbf{v} = (s_0, s_1, \ldots, s_n) = \mathbf{u}G. \quad (2)$$

Therefore, $(s_0, s_1, \ldots, s_n) \in C$ with $s_0 = \mathbf{u}\mathbf{g}_0 = s$. Let $\mathbf{s} = (s_1, \ldots, s_n)$ be the shares of s. Finally, D securely sends s_i to party P_i, for $i = 1, 2, \ldots, n$.

To reconstruct the secret s, the algorithm Recon of SSS is performed as follows: recall the fact that the dual code $C^\perp$ of C can be defined using the following formula:

$$C^\perp = \left\{ \mathbf{x} \in \mathbb{F}_q^{n+1} \,\middle|\, \mathbf{x}G = 0 \right\}. \quad (3)$$

Namely, a vector $\mathbf{x} \in \mathbb{F}_q^{n+1}$ belongs to $C^\perp$ if and only if $\mathbf{x}$ is orthogonal to any codeword in C. If $\mathbf{x} = (x_0, x_1, \ldots, x_n) \in C^\perp$ with $x_0 \ne 0$, for any codeword $(s_0, s_1, \ldots, s_n) \in C$, then we have

$$s = \sum_{i=1}^{n} -\frac{x_i}{x_0} s_i. \quad (4)$$

Using equation (1), the secret s can be reconstructed from the shares $s_1, s_2, \ldots, s_n$.

Lemma 1. *Assume that* $A = \{P_{c_1}, \ldots, P_{c_m}\} \subseteq P_n$, $1 \le c_1 < \cdots < c_m \le n$; *then, the members in A can reconstruct the secret s with their own shares* $s_{c_1}, \ldots, s_{c_m}$ *if and only if the vector* $\mathbf{g}_0$ *is a linear combination of* $\mathbf{g}_{c_1}, \ldots, \mathbf{g}_{c_m}$.

Proof. "⇒" The necessity is quite obvious.

"⇐" Assume that $\mathbf{g}_0$ is a linear combination of $\mathbf{g}_{c_1}, \ldots, \mathbf{g}_{c_m}$; then, there exists $a_{c_1}, \ldots, a_{c_m} \in \mathbb{F}_q$ such that

$$\mathbf{g}_0 = \sum_{i=1}^{m} a_{c_i} \mathbf{g}_{c_i}. \quad (5)$$

Then, the secret s is reconstructed by calculating

$$s = \mathbf{u}\mathbf{g}_0 = \mathbf{u}\left(\sum_{i=1}^{m} a_{c_i} \mathbf{g}_{c_i} \right) = \sum_{i=1}^{m} a_{c_i} \left(\mathbf{u}\mathbf{g}_{c_i} \right) = \sum_{i=1}^{m} a_{c_i} s_{c_i}. \quad (6)$$

The above mentioned SSS realizes that the access structure $\underline{\mathrm{AS}}$ is defined as follows:

$$\underline{\mathrm{AS}} = \left\{ A \subseteq \mathscr{P}_n \,\middle|\, \mathbf{g}_0 \in \mathrm{span}\left\{ \mathbf{g}_i \,\middle|\, P_i \in A \right\} \right\}, \quad (7)$$

where "*span*" means the linear space spanned by the element of the $\{\mathbf{g}_i \mid P_i \in A\}$ set.

Based on Definition 4 and the work of [19], we present the security definition of SSS and the adversarial model in the following.

In this work, to model adversarial leakage attacks on a secret key s, the adversary has an opportunity to adaptively access a *leakage oracle* and obtains information about the secret key s. The formal definition of leakage oracle is given as follows. □

Definition 6 (leakage oracle [20, 22]). Let $\mathcal{O}_{sk}^{\lambda,l}(\cdot)$ be a leakage oracle, which is parameterized by a prover's secret key sk, a leakage parameter l, and a security parameter λ. A query to the leakage oracle is constituted by a leakage function $\{f_j: \{0,1\}^{|sk|} \longrightarrow \{0,1\}^{l_j}\}_{j\in[n]}$, and the oracle responds with $f_j(sk)$. The oracle $\mathcal{O}_{sk}^{\lambda,l}(\cdot)$ is restricted in the total number of l bits. For all queries received, $\mathcal{O}_{sk}^{\lambda,l}(\cdot)$ only responds to the kth leakage query and computes the function $f_j(w)$ for at most $\text{poly}(\cdot)$ steps if $\sum_{j=1}^{k} l_j \leq l$, where $1 \leq j \leq k \leq n$. Otherwise, the oracle ignores the queries.

Remark 1. Note that $l = 0$ means that there is no information leaked to the simulator in an ideal setting, whereas $l \approx 1$ means that a malicious adversary (or verifier) learns nothing from the protocol other than obtaining the validity of the proven statement and obtaining the leakage information from an honest user (or prover).

Definition 7 (l-bounded adversary). Let Θ be a subset of $[n]$. We say that an adversary $\mathcal{A}$ is l-bounded, if the corruption set $P_\Theta: = \{P_i\}_{i\in\Theta}$ selected by $\mathcal{A}$ satisfies the following property; for each $P_i \in P_\Theta$, it holds that $\sum_{i\in\Theta} l_i \leq l$, where l_i denotes the length of the output of an arbitrary (leakage) function $\{f_i: \{0,1\}^* \longrightarrow \{0,1\}^{l_i}\}_{i\in\Theta}$.

Definition 8 (leakage-resilient secret sharing, LRSS). Let S be any secret key domain and AS be any access structure on parties $P_1, \ldots, P_n$. We say an SSS realizing AS is (Θ, l, ε)-leakage-resilient (or (Θ, l, ε)-LRSS), if for every leakage protocol Leak in (θ, n, l)-BCP, and for every pair of secrets $s^{(1)}, s^{(2)} \in S$, the following holds:

$$\{\text{Leak}_{\Theta,\mathbf{f}}(\text{Share}(s^{(1)})): \mathbf{s} \longleftarrow \text{Share}(s^{(1)})\} \approx_\varepsilon \{\text{Leak}_{\Theta,\mathbf{f}}(\text{Share} \cdot (s^{(2)})): \mathbf{s} \longleftarrow \text{Share}(s^{(2)})\}, \tag{8}$$

where Θ denotes the subset of $[n]$. That is, the distribution of transcript learned by $\mathcal{A}$ on sharing $s^{(1)}$ is statistically closed to the distribution of transcript learned by $\mathcal{A}$ on sharing $s^{(2)}$. In particular, an SSS = (Share, Rec) is said to be (θ, l, ε)-leakage-resilient (or (θ, l, ε)-LRSS) if it is (Θ, l, ε)-leakage-resilient for any subset $\Theta \subseteq [n]$ with $|\Theta| \leq \theta$.

In our work, the notation (θ, n, l)-BCP presented in Definition 4 is inspired by the work [33]. We give the program (θ, n, l)-bounded corrupted program (or (θ, n, l)-BCP) as follows. n parities $P_1, \ldots, P_n$ and $\theta \leq n$, where θ is an upper bound on the number of parties corrupted by adversary $\mathcal{A}$ in any round. Let l be the leakage bound. Let $\mathbf{f}$ be leakage function family $\mathbf{f} = (f_1, \ldots, f_n)$, where $f_j: \{0,1\}^* \longrightarrow \{0,1\}^{l_j}$ and $j \in [n]$. We write $\mathbf{f}(\mathbf{s}) = (f_1(s_1), \ldots, f_n(s_n))$ for the total leakage seen by adversary on the shares $\mathbf{s} = (s_1, \ldots, s_n)$ of secret s.

(θ, n, l)-BCP ON INPUT (s, Θ, n, l)

GENERATE shares $\mathbf{s} = (s_1, \ldots, s_n)$ of secret $s \in S$, SEND s_i to P_i for $i \in [n]$

Leak is empty at the begining of leakage-protocol

Leak is appended with the leakage and $|\text{Leak}| \leq l$

COMPUTE Leak←Leak$(\mathbf{f}(\mathbf{s}_\Theta))$ in each round, where $\mathbf{s}_\Theta: = \{s_i\}_{i\in\Theta}$ and $|\Theta| \leqslant \theta$

OUTPUT final transcript Leak as leakage

2.2. Zero-Knowledge

Definition 9 (negligible functions). A function $\mu: \mathbb{N} \longrightarrow \mathbb{R}$ is negligible if, for any positive polynomial $\text{poly}(\cdot)$, there exists $N \in \mathbb{N}$ such that, for all $n > N$,

$$\mu(n) < \frac{1}{\text{poly}(n)}. \tag{9}$$

Definition 10 (probability ensemble). X denotes a countable set. An ensemble indexed by X indicates a sequence of random variables indexed by X. For instance, $A = \{A_i\}_{i\in X}$ is an ensemble indexed by X, where each A_i is a random variable.

Definition 11 (polynomial-time indistinguishability). Let $A = \{A_n\}_{n\in n^*}$ and $B = \{B_n\}_{n\in n^*}$ be two difference ensembles indexed by n^*. If

$$|\Pr[D(A_n, 1^n)] - \Pr[D(B_n, 1^n)]| < \frac{1}{\text{poly}(n)} \tag{10}$$

holds for any probabilistic polynomial-time (PPT) algorithm D, any positive polynomial $\text{poly}(\cdot)$, and any sufficiently large n, then we say that A and B are indistinguishable in polynomial time.

In the following context, we use the terminology *computationally indistinguishable* instead of *indistinguishability in polynomial time*.

Definition 12. Let λ be a security parameter and let $X = X(\lambda)$ and $Y = Y(\lambda)$ be the ensemble sets. Let R be a witness relation associated with language L on $X \times Y$, where L is defined as $L = \{x: \exists y \quad \text{s.t.}\,(x, y) \in R\}$. R is polynomially bounded (i.e., $(x, y) \in R$ implies $|y| \leqslant \text{poly}(|x|)$) and recognized in polynomial time. Given $x \in L$, an element $y \in Y$ such that $(x, y) \in R$ is called a witness. Suppose that K is a PPT algorithm, which takes as input (1^λ) and outputs pairs (x, y) with $(x, y) \in R$.

We write P and V to denote the prover and the verifier, respectively, where P and V are two interactive probabilistic polynomial turning machines.

Definition 13 (interactive proof system, [34]). A pair of interactive machines (P, V) is called an interactive proof system for a language L if machine V is PPT and the following two conditions hold:

Completeness: for every $x \in L$, it holds that $\Pr[(P, V)(x) = 1] \geq (2/3)$

Soundness: for every $x \notin L$ and every interactive machine M, it holds that $\Pr[(M, V)(x) = 1] \leq (1/3)$

Now, we consider the interactive protocol consisting of infinitely powerful n machines $P_1, \ldots, P_n$ interacting with a PPT machine V.

Definition 14 (multiprover interactive proof system [35]). We say that the interactive turning machines $(P_1, \ldots, P_n, V)$ in the n-prover model are called multiprover interactive proof system for language L, if an interactive PPT Turning machine V exists and the following two properties are satisfied:

Completeness: for every $x \in L$, there is $\Pr[(P_1, \ldots, P_n, V)(x) = 1] \geq (2/3)$

Soundness: for every $x \notin L$ and all interactive machines $M_1, \ldots, M_n$, there are $\Pr[(M_1, \ldots, M_n, V)(x) = 1] \leq (1/3)$.

Definition 15 (multiprover computational zero-knowledge [34]). Let $(P_1, \ldots, P_n, V)$ be a multiprover interactive proof system with some language L. We say that $(P_1, \ldots, P_n, V)$ is computational zero-knowledge if, for every interactive PPT turning machine V^*, a PPT machine $\mathscr{M}$ exists such that, for all $x \in L$, ensemble $\text{View}\{(P_1, \ldots, P_n, V^*)(x)\}_{x \in L}$ and $\{\mathscr{M}(x)\}_{x \in L}$ are computationally indistinguishable.

In Definition 15, the ensemble $\text{View}\{(P_1, \ldots, P_n, V^*)(x)\}_{x \in L}$ denotes the view (or output) of the interactive machine V^* after interacting with n interactive machines $P_1, \ldots, P_n$ on common input x (namely, the transcript about the sequence of messages exchanged between V^* and $P_1, \ldots, P_n$). $\{\mathscr{M}(x)\}_{x \in L}$ denotes the output of $\mathscr{M}$ on input x. In this case, we call M is a simulator for the interaction of V^* with $P_1, \ldots, P_n$. The existence of $\mathscr{M}$ means that V^* does not gain any more knowledge than $\mathscr{M}$. Indeed, $\mathscr{M}$ does not access $P_1, \ldots, P_n$ and might not know whether $P_1, \ldots, P_n$ exists. However, it is able to simulate the interaction of V^* with $P_1, \ldots, P_n$.

Succinctly speaking, an interactive proof system $(P_1, \ldots, P_n, V)$ for language L is zero-knowledge if whatever can be efficiently computed by V after interacting with $P_1, \ldots, P_n$ on common input $x \in L$ can also be directly computed by the simulator with input x.

2.3. Identification Schemes and Security Definitions. In this section, we first recall the standard cryptographic concepts of $\sum$-protocols presented in [36, 37] and identification schemes (ID-scheme) presented in [20]. Then, we present the formal security definitions of the ID-scheme. We write $\text{TR} \leftarrow (P(y), V)(x)$ to denote the transcript TR that was generated through an interactive protocol (P, V), where y is the private input of P and x is the common input of P and V. The notation $\text{Out}_V(z)$ denotes the output of V with input $z \in \text{TR}$. V outputs $\text{Out}_V(z) = 1$ if he accepts the proof and $\text{Out}_V(z) = 0$ otherwise.

Definition 16 ($\sum$ −protocol [36, 37]). Let (P, V) be an interactive proof system for language L. The prover P takes as input $x \in L$ and sends a witness y to verifier V. V takes as input the common input x and the received y, and it outputs 1 if $|y| = \text{poly}(|x|)$ and $(x, y) \in R$.

A $\sum$-protocol for the relation R is a 3-round interactive proof system (P, V) that can be described as follows:

Step $P1$: P first sends a commitment α to V, i.e., $\alpha \leftarrow P(x, y)$

Step $V1$: V responds with a challenge β based on α, i.e., $\beta \leftarrow V(x, \alpha)$

Step $P2$: P returns with γ based on β, i.e., $\gamma \leftarrow P(x, y, \alpha, \beta)$

Step $V2$: V outputs a bit $b \in \{0, 1\}$, i.e., $b \leftarrow \text{Out}_V(x, \alpha, \beta, \gamma)$, where $b = 1$ if he accepts the proof and $b = 0$ otherwise

Here, TR: $= (\alpha, \beta, \gamma)$. A $\sum$-protocol satisfies the following properties:

Completeness: the verifier V outputs 1 with probability 1, i.e., $Pr[(x, y) \leftarrow \text{K}(1^\lambda); \alpha \leftarrow P(x, y); \beta \leftarrow V(x, \alpha); \gamma \leftarrow P(x, y, \alpha, \beta); 1 \leftarrow \text{Out}_V(x, \alpha, \beta, \gamma)] = 1$, where steps $P1$, $V1$, and $P2$ can be denoted by $\text{TR} \leftarrow (P(y), V)(x)$.

2-special soundness: sssume that an extractor *Extor* exists for every $x \in L$, given two valid transcripts (α, β, γ) and $(\alpha, \beta', \gamma')$, where $\beta \neq \beta'$ and $\gamma \neq \gamma'$. The *Extor* takes as input $(\alpha, \beta, \gamma, \beta', \gamma')$ and outputs a witness y for the relation R.

Honest verifier zero-knowledge: for every PPT turning machine V^*, a probabilistic algorithm $\mathscr{M}$ exists such that the two ensembles $\text{View}\{(P, V^*)(x)\}_{x \in L}$ and $\{\mathscr{M}(x)\}_{x \in L}$ are computationally indistinguishable.

Definition 17 (identification scheme [20]). An identification scheme (ID-scheme) contains four PPT procedures $\{\text{ParGen}, \text{KeyGen}, P, V\}$. The concrete description is as follows:

params $\leftarrow$ ParGen(1^λ): the parameter generation procedure takes as input security parameter λ and returns the system parameters of the identification scheme, denoted by params. params are common to all users and issued to KeyGen, P, and V as inputs.

$(pk, sk) \leftarrow$ KeyGen(): the key generation procedure takes as input params and outputs the public key pk and secret key sk.

$P(pk, sk), V(\text{TR}, pk)$: P denotes a prover and V denotes a verifier. V returns a judgement $b \in \{0, 1\}$ about P after executing the protocol. If $b = 1$, then V accepts P's identity. $b = 0$ otherwise.

To obtain a secure (group) ID-scheme, we define some security definitions and models.

Definition 18. We say that (group) ID-scheme $\{\text{ParGen}, \text{KeyGen}, P, V\}$ is secure if the following two conditions hold:

Completeness: for each party $P_i \in \text{AS}$, the probability of acceptance is

$$\Pr\left[\text{params}\leftarrow\text{ParGen}\left(1^{\lambda}\right);\ (pk_1,\ldots,pk_n,sk_i)\leftarrow\text{KeyGen}(),\right.$$
$$\left.\text{TR}\leftarrow(P(sk_i),V)(pk_1,\ldots,pk_m):\ 1\leftarrow\text{Out}_V(\text{TR})\right]=1. \tag{11}$$

Soundness: for any party $P_i \notin$ AS and any PPT algorithm, the advantage of P'

$$\Pr\left[\text{params}\leftarrow\text{ParGen}\left(1^{\lambda}\right);\ (pk_1,\ldots,pk_n)\leftarrow\text{KeyGen}()\right.$$
$$\left.\text{TR}\leftarrow(P'(\phi),V)(pk_1,\ldots,pk_n):\ 1\leftarrow\text{Out}_V(\text{TR})\right]-\frac{1}{2} \tag{12}$$

is negligible, where the private input ϕ of $P'(\cdot)$ is an empty string and TR is defined in Definition 16 and $i \in [m]$.

Before we formally define of the leakage-resilient ID-scheme, we first consider two security games (illustrated in Table 1), which are inspired by the work presented in [20]. The first game called *pre-emulation leakage security* is simulated by the attack game $\text{ID}^{\lambda}_{\text{pre}l}(\mathscr{A})$ and allows the adversary $\mathscr{A}$ to send leakage queries before the emulation attack. The second game called *arbitrary time leakage security* is simulated by the attack game $\text{ID}^{\lambda}_{\text{arb}l}(\mathscr{A})$ and allows the adversary $\mathscr{A}$ to adaptively execute leakage attacks at an arbitrary time during an emulation attack. The key generation phase and test phase are the same in the games $\text{ID}^{\lambda}_{\text{pre}l}(\mathscr{A})$ and $\text{ID}^{\lambda}_{\text{arb}l}(\mathscr{A})$. The emulation phase is divided into two separate games according to the definitions of $\text{ID}^{\lambda}_{\text{pre}l}(\mathscr{A})$ and $\text{ID}^{\lambda}_{\text{arb}l}(\mathscr{A})$, respectively.

Definition 19 (leakage-resilient ID-scheme). Let $\{\text{KeyGen}, P, V\}$ be an ID-scheme, which is parameterized by security parameter λ and holds the property of completeness. We say that $\{\text{KeyGen}, P, V\}$ is secure against the pre-emulation leakage l if the advantage of any PPT adversary $\mathscr{A}$ in the attack game $\text{ID}^{\lambda}_{\text{pre}l}(\mathscr{A})$ is negligible in λ. Additionally, it is secure against the arbitrary time leakage l if the abovementioned adversary $\mathscr{A}$ for the attack game $\text{ID}^{\lambda}_{\text{arb}l}(\mathscr{A})$ is negligible in λ.

Table 1: Attack game for defining the *pre-emulation leakage security* and *arbitrary time leakage security* of $\text{ID}^{\lambda}_{\text{pre}l}(\mathscr{A})$ and $\text{ID}^{\lambda}_{\text{arb}l}(\mathscr{A})$, respectively.

Attack Game $\text{ID}^{\lambda}_{\text{pre}l}(\mathscr{A})$	Attack Game $\text{ID}^{\lambda}_{\text{arb}l}(\mathscr{A})$
(1) Key generation phase: run two algorithms params$\leftarrow$ParGen(1^{λ}), $(pk, sk)\leftarrow$KeyGen() and give (params, pk) to the adversary $\mathscr{A}$	
(2) Test phase: the adversary $\mathscr{A}^{\mathcal{O}^{\lambda,l}_{sk}(\cdot),P(pk,sk)}$ can access oracles $\mathcal{O}^{\lambda,l}_{sk}(\cdot)$ (the leakage oracle) and $P(pk, sk)$ (an honest prover oracle), where $\mathcal{O}^{\lambda,l}_{sk}(f)$ returns $f(sk)$	
(3) Emulation phase: this phase is divided into two separate games	
(i) For the game $\text{ID}^{\lambda}_{\text{pre}l}(\mathscr{A})$: the adversary $\mathscr{A}$ is not allowed to access oracles $\mathcal{O}^{\lambda,l}_{sk}(\cdot)$ and honest prover oracle P. $\mathscr{A}$ performs an interactive protocol, denoted by $(\mathscr{A}, V(pk))$ with an honest verifier V	
	(ii) For the game $\text{ID}^{\lambda}_{\text{arb}l}(\mathscr{A})$: the adversary $\mathscr{A}$ only gets access to the leakage oracle $\mathcal{O}^{\lambda,l}_{sk}(\cdot)$ and performs the interactive protocol, denoted by $(\mathscr{A}^{\mathcal{O}^{\lambda,l}_{sk}(\cdot)}, V(pk))$ with an honest verifier V

3. Our Protocol

3.1. Protocol 1: The Basic Leakproof Secret-Sharing Scheme. For self-containedness, we recall the leakproof secret-sharing scheme [38] as follows. Let D be a dealer, V be a verifier, and $P_1, \ldots, P_n$ be n parties.

Initialization. In this stage, all the system parameters are generated. The dealer D obtains a public key corresponding to the public key encryption scheme Enc. In addition, suppose that D holds a private channel with every party and that D and every party keep a broadcast channel.

Distribution protocol. This protocol is divided into two steps:

(1) Distribution of the shares: this step is executed by the dealer D. First, for the master secret key s, D generates n shares $s_1, \ldots, s_n$ of s and sends them to $P_1, \ldots, P_n$ through individual private channels. Second, D calculates $\text{Enc}(s)$ and publishes it by his broadcast channel.

(2) Verification of the shares: every party P_j verifies the validity of share s_j received from D, where $j = 1, 2, \ldots, n$. If the verification condition is not satisfied, we will say that the dealer fails, and the protocol is aborted.

Proof protocol. This protocol is also divided into two steps:

(1) Proof of the secret s: let Λ be any qualified subset for parties $P_1, \ldots, P_n$. Each $P_i \in \Lambda$ holds the secret share $s_i \in \mathbf{s}_{\Lambda}$. Parties in Λ run a multiprover zero-knowledge argument of knowledge with the verifier V prove that they indeed share the secret s. During this interactive proof process, $\text{Enc}(s)$ is a common input of parties in Λ between V.

(2) Verification of the secret s: to check whether every $P_i \in \Lambda$ keeps a valid secret s_i, V verifies the following condition $s = F(\mathbf{s}_{\Lambda})$, where F stands for some certain deterministic polynomial-time function.

3.2. Protocol 2: Secret-Sharing Scheme via Linear Codes. In this section, we begin to describe how our secret-sharing scheme is constructed using linear codes in detail. First, suppose that we have obtained an access structure realized by linear codes C and that G is the corresponding generator matrix.

Initialization. Let λ be a security parameter and q be a large prime number, and p is a prime factor of $q - 1$. Let $\mathbb{F}_q$ be a finite field. We write $\langle g\rangle$ to indicate a cyclic group generated by an element $g \in \mathbb{F}_q^*$ with order $|\langle g\rangle| = p$ and $\langle g\rangle \subset \mathbb{F}_q^*$. Let C be a linear code over $\mathbb{F}_q$ with length $n+1$; its generator matrix $G = (g_{ij})_{k\times(n+1)}$, where $0 \le i \le k-1, 0 \le j \le n$.

Let D be a dealer and $\mathrm{P}_n = \{P_1, P_2, \ldots, P_n\}$ be a party set. V denotes a verifier with $V \notin \mathrm{P}_n$. Assume that a private channel between the dealer D and each $P_j\,(j \in [n])$ and a public channel between P_{j_1} and P_{j_2} exist, where $j_1 \neq j_2$ and $j_1, j_2 \in [n]$. In addition, the dealer D has a broadcast channel. We assume that the computing power of all individuals in this protocol is polynomial-time bound.

The encryption algorithm Enc is defined as $\mathrm{Enc}(x) = g^x \,(\mathrm{mod\ q})$, where $x \in \mathbb{F}_p$.

The dealer D randomly chooses s from $\mathbb{F}_p$ and $p > \max\{s, n\}$. The master private key $sk = s$ and the public key $pk = (p, q, g)$.

Distribution protocol. This protocol is divided into two steps:

(1) Distribution of shares: D defines $s_0 = s$ and calculates and publishes $B_0 = \mathrm{Enc}(s_0)$ by a broadcast channel. To distribute the master secret key s among n parties $P_1, \ldots, P_n$, D executes the algorithm Share of SSS = (Share, Recon) and gets $\mathbf{s} \longleftarrow \mathrm{Share}(s)$. More precisely, the algorithm Share is described by the following program of SECRET DISTRIBUTION:

D's SECRET DISTRIBUTION ON INPUT (s_0, pk, B_0, G):

CHOOSE random $\mathbf{u} = (u_0, u_1, \ldots, u_{k-1}) \in \mathbb{F}_p^k$ such that $u_{k-1} = ((s_0 - \sum_{i=0}^{k-2} u_i g_{i0})/g_{k-1,0} \,(\mathrm{mod}\ \ p))$
COMPUTE $B_i = g^{u_i} \bmod q$, where $i \in [k-1]$

COMPUTE $s_j = \sum_{i=0}^{k-1} u_i g_{ij} \,(\mathrm{mod}\ \ p)$ and set $sk_j = s_j$, where $j \in [n]$.

PUBLISH B_i, where $i \in [k-1]$.

GET a shares $\mathbf{s} = (s_1, \ldots, s_n)$ of secret $s_0 = s$

Then, D sends $s_1, \ldots, s_n$ to $P_1, \ldots, P_n$, respectively.

(2) Verification of the shares: for each $P_j \in \mathrm{P}_n$, let s_j denote the private key sk_j. Each $P_j \in \mathrm{P}_n$ performs the following program of SECRET VERIFICATION to check the validity of his own share for $j \in [n]$:

P_j's SECRET VERIFICATION ON INPUT (s_j, pk, B_0, B_i, G):

GET s_j from D. See if $s_j \in \mathbb{F}_p$; if not, halt.
IF $pk_j = g^{s_j} = B_0 \prod_{i=1}^{k-1} B_i^{g_{ij}} \,(\mathrm{mod}\ q)$, THEN
ACCEPT the share s_j
ELSE
REJECT the share s_j and repeat the program

Proof protocol. According to the abovementioned Distribution protocol, B_0, p, q, g, and G are common inputs of parties in P_n between V. Every party $P_j \in P_n$ has a secret input s_j for $j \in [n]$. Let $\mathrm{P}_{c_m} := \{P_{c_1}, \ldots, P_{c_m}\}$ be a size m subset of P_n, and every party P_{c_i} has valid secret input s_{c_i}, where $i \in [m]$, $1 \le c_1 < \cdots < c_m \le n$. To reconstruct the secret s, for all of $P_{c_1}, \ldots, P_{c_m}$, they need to determine the existence and uniqueness of the solutions to the system $\mathbf{g}_0 = d_{c_1}\mathbf{g}_{c_i} + \cdots + d_{c_m}\mathbf{g}_{c_m}$, where $\mathbf{g}_0, \mathbf{g}_{c_1}, \ldots, \mathbf{g}_{c_m}$ are the column vector of the generator matrix G of $[n+1, k]$ code C. The calculation is presented in Lemma 1.

This protocol is divided into two steps:

(1) Proof of the secret s: this step can be described as follows:

Step $P1$: every party $P_{c_i} \in P_{c_m}$ chooses a random $r_{c_i} \in \mathbb{F}_p$, computes that $x_{c_i} = g^{r_{c_i}} \,(\mathrm{mod}\ q)$, and sends x_i to V, where $i \in [m]$
Step $V1$: V chooses a random number $z \in \{1, 2\}$ and then sends z to all parties $P_{c_1}, \ldots, P_{c_m}$
Step $P2$: every party $P_{c_i} \in P_{c_m}$ computes $y_{c_i} = r_{c_i} - z d_{c_i} s_{c_i} \,(\mathrm{mod}\ p)$ and then sends y_{c_i} to V, where $i \in [m]$

(2) Verification of the secret s:

Step $V2$: if it holds that $g^{y_{c_1} + y_{c_2} + \cdots + y_{c_m}} B_0^z = \prod_{i=1}^{m} x_{c_i} \,(\mathrm{mod}\ \ q)$, then V believes that $P_{c_1}, \ldots, P_{c_m}$ share the secret s satisfying $B_0 = g^s \,(\mathrm{mod}\ \ q)$; otherwise, V rejects it.

4. Security Analysis of Protocol 2

In the following context, to illustrate and analyze the argument of our interactive protocol between parties $P_1, \ldots, P_n$ and verifier V, we use *prover* to replace *party*.

4.1. Properties from PROTOCOL 2 in Random Oracle Model

4.1.1. Completeness. Assume that $\{P_{c_1}, \ldots, P_{c_m}\}$ is a subset of $\mathrm{P}_n = \{P_1, \ldots, P_n\}$. $P_{c_1}, \ldots, P_{c_m}$ and V execute the abovementioned interactive protocol. V computes the following equation:

$$g^{y_{c_1} + y_{c_2} + \cdots + y_{c_m}} B_0^z = g^{y_{c_1} + y_{c_2} + \cdots + y_{c_m}} g^{sz} = g^{y_{c_1} + y_{c_2} + \cdots + y_{c_m}} g^{z \sum_{i=1}^{m} s_{c_i} d_{c_i}} = \prod_{i=1}^{m} g^{y_{c_i} + z d_{c_i} s_{c_i}} = \prod_{i=1}^{m} g^{r_{c_i}} = \prod_{i=1}^{m} x_{c_i} \,(\mathrm{mod}\, q). \tag{13}$$

If the last equation in equation (13) is satisfied, then V believes that $P_{c_1}, \ldots, P_{c_m}$ share the secret s satisfying $B_0 = g^s \,(\mathrm{mod}\ \ q)$.

4.1.2. Soundness. To prove the property of soundness, we consider the following three settings.

In the first setting, for all of the $P_n = \{P_1, \ldots, P_n\}$, let P_{c_0} be the only one prover corrupted by adversary $\mathcal{A}$. P_{c_o} pretends to be an honest prover $P_j \in \mathrm{P}_n$, where $i \in [n]$. After interacting with V in Step $P1$, he randomly selects $r_i \in \mathbb{F}_p$ and computes $x_j = g^{r_j} \,(\mathrm{mod}\ q)$; then, he sends x_j to V. Next, V sends the challenge z to P_{c_o}, who randomly chooses $y_j \in \mathbb{F}_p$, because he does not know the secret s_j (held by the honest prover P_j), where $j \in [n]$. Then, the success probability of equality

$$g^{y_1+y_2+\cdots+y_n} B_0^z = \prod_{j=1}^{n} x_j \pmod q \tag{14}$$

is $(1/q)$. If the interactive protocol executed for K times, the success probability is $(1/q^K)$.

In the second setting, the adversary $\mathcal{A}$ chooses a subset of $\Theta \subseteq [n]$ with $|\Theta| = \theta$ and get their shares. Let $P_\Theta := \{P_{c_1}, \ldots, P_{c_\theta}\}$ denote the subset of P_n, the adversary $\mathcal{A}$ chooses to corrupt in consecutive rounds. The honest parties are denoted by $P_{[n]\backslash\Theta}$. Assume that V knows the total number of all provers. Each $P_{c_i} \in P_\Theta$ disguises himself as the honest provers $P_j \in P_{[n]\backslash\Theta}$ to follow the interactive protocol with V, where $i \in [\theta]$, $j \in [n]\backslash\Theta$. In Step $P1$, P_{c_i} chooses a random element $r_{c_i} \in \mathbb{F}_p$, calculates $x_{c_i} = g^{r_{c_i}} \pmod q$, and sends x_{c_i} to V; in Step $V1$, V returns a challenge $z \in \{1, 2\}$ to P_{c_i}; in Step $P2$, after receiving the challenge z, P_{c_i} randomly chooses y_{c_i} from $\mathbb{F}_p$, since he does not know the secret key s_{c_i} of honest P_j; and in Step $V2$, V computes

$$g^{y_1+y_2+\cdots+y_n} B_0^z = \prod_{j=1}^{n} x_i \pmod q \tag{15}$$

with probability $(1/q)$. That is, the corrupted provers $P_{c_1}, \ldots, P_{c_\theta}$ choose $y_{c_1}, \ldots, y_{c_\theta}$, respectively, such that

$$\sum_{i=1}^{\theta} y_{c_i} = \sum_{i=1}^{\theta} r_{c_i} - \sum_{i=1}^{\theta} z d_{c_i} s_{c_i} \pmod p, \tag{16}$$

with probability $(1/q)$. If $P_{[n]\backslash\Theta} \cup P_\Theta$ and V sequentially perform the PROTOCOL 2 for K times, then the success probability of the following equality

$$g^{y_1+y_2+\cdots+y_n} B_0^z = \prod_{j=1}^{n} x_j \pmod q \tag{17}$$

falls to $(1/q^K)$.

In the third setting, the verifier V is not certain of the number of all provers. In this case, θ corrupted provers pretend to be $\nu(\neq\theta)$ honest provers. Based on the proof of the second setting, we can prove that the prover set $P_{[n]\backslash\Theta} \cup \{P_{c_1}, \ldots, P_{c_\nu}\}$ shares a secret s such that $B_0 = g^s \pmod q$ with the success probability $(1/q^K)$.

In light of the foregoing, for all sufficiently large K, the probability $(1/q^K)$ is negligible. Consequently, the property of soundness is matched.

4.1.3. Zero-Knowledge. To prove the property of zero-knowledge, we consider passive scenarios according to the power of the adversary.

A passive verifier V does not randomly choose z_η in the ηth time when running the protocol. Thus, he can use some $\{1, 2\}$-valued function f to compute z_η in deterministic polynomial time [39]. Therefore, we define $f(p, q, g, B_0, G, h, V_{\eta-1}, x_{1,\eta}, \ldots, x_{n,\eta}) = z_\eta$, where h is a secret input; $V_{\eta-1}$ denotes all data viewed when V executes the protocol in the $\eta - 1$-th time; $x_{1,\eta}, \ldots, x_{n,\eta}$ denotes a message sent to V in Step $P1$ when running PROTOCOL 2 in the ηth time. Suppose that a simulator M has been constructed with the input p, q, g, h, B_0, and G and has successfully simulated PROTOCOL 2 $\eta - 1$ times, then M will simulate the ηth time according to the following program:

DO FOREVER

z'_η: = a random number of $\{1, 2\}$

$y_{i,\eta}$: = a random number of $\mathbb{F}_p$ for $i \in [n]$

$d_{i,\eta}$: = a random number of $\mathbb{F}_p$ for $i \in [n-1]$

$s_{i,\eta}$: = a random number of $\mathbb{F}_p$ for $i \in [n-1]$

COMPUTE $r_{i,\eta} := y_{i,\eta} + z'_\eta d_{i,\eta} s_{i,\eta} \pmod p$, $x_{i,\eta} := g^{r_{i,\eta}} \pmod q$ for $i \in [n-1]$, and $x_{n,\eta} = (g^{y_{1,\eta}+y_{2,\eta}+\cdots+y_{n,\eta}} B_0^{z_\eta} / \prod_{i=1}^{n-1} x_{i,\eta}) \pmod q$

IF $z'_\eta = f(p, q, g, h, B_0, V_{\eta-1}, x_{1,\eta}, \ldots, x_{n,\eta})$ THEN

OUTPUT

$(x_{1,\eta}, x_{2,\eta}, \ldots, x_{n,\eta}, z'_\eta, y_{1,\eta}, y_{2,\eta}, \ldots, y_{n,\eta})$ and HALT

ELSE

GO back to the first step and repeat again

END DO

Let $(P_1, \ldots, P_n, V)$ be an interactive protocol performed among $(P_1, \ldots, P_n)$ and V. Suppose that $(P_1, \ldots, P_n, V)$ have been executed m times. According to the description of $V_{\eta-1}$, V_η is denoted by $\{(x_{1,1}, \cdots, x_{n,1}, z_1, y_{1,1}, \ldots, y_{n,1}), \ldots, (x_{1,\eta}, \ldots, x_{n,\eta}, z_\eta, y_{1,\eta}, \ldots, y_{n,\eta})\}$, and the mth output is denoted by $\mathrm{Out}_{V_m} = \mathrm{View}\{(P_1(c_1 s_1), \ldots, P_n(c_n s_n)), V(h)\}(p, q, g, B_0, G) = V_m$ (if $m = 0$, Out_{V_0} denotes an empty set). Suppose that the output of M for m times after the simulation is denoted by $M(p, q, g, h, B_0, G) = \{(x_{1,1}, \ldots, x_{n,1}, z'_1, y_{1,1}, \ldots, y_{n,1}), \ldots, (x_{1,m}, \ldots, x_{n,m}, z'_m, y_{1,m}, \ldots, y_{n,m})\}$.

According to Definition 15, we can determine whether ensembles $M(p, q, g, h, B_0, G)$ and $\mathrm{View}\{(P_1(c_1 s_1), \ldots, P_n(c_n s_n)), V(h)\}$ are *computational indistinguishability*, and then the protocol which we constructed is zero-knowledge. Hence, to prove that $\mathrm{M}(p, q, g, h, B_0, G)$ and $\mathrm{View}\{(P_1(c_1 s_1), \ldots, P_n(c_n s_n)), V(h)\}(p, q, g, B_0, G)$ are *computational indistinguishability*, we use a mathematical induction method to perform the following steps:

(1) Let the probability of Out_{V_η} be denoted by $P_{\mathcal{M}_V,\eta}$ if $\mathrm{Out}_{V_\eta} \subseteq \mathcal{M}(p, q, g, B_0, G)$ and the probability of Out_{V_η} be denoted by $\Pr_{P_1,\ldots,P_n,V,\eta}$ if $Out_{V_\eta} \subseteq View\{(P_1(d_1 s_1), \ldots, P_n(d_n s_n)), V(h)\}(p, q, g, B_0, G)$.

(2) When $\eta = 0$, in the initial state of interactive proof system $(P_1, \ldots, P_n, V)$ there is $\Pr_{\mathcal{M}_V,0} = \Pr_{P_1,\ldots,P_n,V,0}$ and $|\Pr_{\mathcal{M}_V,0} - \Pr_{P_1,\ldots,P_n,V,0}| < 3\mathrm{F}5$, where $\varepsilon > 0$ is an arbitrarily small constant.

(3) In the $\eta - 1$th operation, we assume that $|\Pr_{\mathcal{M}_V,\eta-1} - \Pr_{P_1,\ldots,P_n,V,\eta-1}| < \varepsilon$ is satisfied for each $\eta \geq 1$.

(4) In the ηth operation, we consider two models:

(I) In the real model, we consider $(P_1, \ldots, P_n, V)$. Assume that V chooses $z_\eta = 1$ with probability ϑ; then, V chooses $z_\eta = 2$ with probability $1 - \vartheta$. Every prover $P_i \in P_n$ selects $r_i \in \mathbb{F}_p$ with probability $(1/p)$ for $i \in [n]$. Additionally, the probability of $y_{i,\eta}$ is

$(1/p)$ since $y_{i,\eta}$ is computed from r_i, where $i \in [n]$. Therefore, the probability of all messages that V viewed in the ηth time, i.e., $(x_{1,\eta}, \ldots, x_{n,\eta}, z_\eta, y_{1,\eta}, \ldots, y_{n,\eta})$ is (ϑ/p^n) when $z_\eta = 1$ and $((1-\vartheta)/p^n)$ otherwise when $z_\eta = 2$.

(II) In the ideal model, we consider the simulator M. In the ηth simulation, M takes as input p, q, g, h, B_0, and G and chooses $y_{1,\eta}, y_{2,\eta}, \ldots, y_{n,\eta} \in \mathbb{F}_p$ with a probability of $(1/p^n)$. Later, V and M choose $z_\eta = 1$ and $z'_\eta = 1$ with a probability of $(\vartheta/2)$, respectively. Meanwhile, V and M choose $z_\eta = 2$ and $z'_\eta = 2$ with the probability of $(1-\vartheta/2)$, respectively. Suppose that $z_\eta = z'_\eta = 1$ or $z_\eta = z'_\eta = 2$; then, M outputs the effective output $(x_{1,\eta}, \ldots, x_{n,\eta}, t'_\eta, y_{1,\eta}, \ldots, y_{n,\eta})$ in the ηth time with probability (1/2). Hence, the probability of $(x_{1,\eta}, \ldots, x_{n,\eta}, 1, y_{1,\eta}, \ldots, y_{n,\eta})$ as an effective output of M is

$$\begin{aligned} &\frac{\vartheta}{p^n}\cdot\frac{1}{2} + \frac{\vartheta}{p^n}\cdot\frac{1}{2^2} + \frac{\vartheta}{p^n}\cdot\frac{1}{2^3} + \cdots \\ &= \frac{1}{2}\cdot\frac{\vartheta}{p^n}\left(1 + \frac{1}{2} + \cdots\right) = \frac{\vartheta}{p^n}, \end{aligned} \tag{18}$$

and the probability of $(x_{1,\eta}, \ldots, x_{n,\eta}, 2, y_{1,\eta}, \ldots, y_{n,\eta})$ as an effective output of M is

$$\begin{aligned} &\frac{1-\vartheta}{p^n}\cdot\frac{1}{2} + \frac{1-\vartheta}{p^n}\cdot\frac{1}{2^2} + \frac{1-\vartheta}{p^n}\cdot\frac{1}{2^3} + \cdots \\ &= \frac{1}{2}\cdot\frac{1-\vartheta}{p^n}\left(1 + \frac{1}{2} + \cdots\right) \\ &= \frac{1-\vartheta}{p^n}. \end{aligned} \tag{19}$$

In fact, z_η and $z'_\eta, y_{1,\eta}, \ldots, y_{n,\eta}$ are selected randomly and independently, the case $z = 1$ yields that $Pr_{P_1,\ldots,Pr_n,V,\eta} = P_{P_1,\ldots,P_n,V,\eta-1} \cdot (\vartheta/p^n)$, and $z = 2$ results in $Pr_{P_1,\ldots,Pr_n,V,\eta} = Pr_{P_1,\ldots,P_n,V,\eta-1} \cdot (1-\vartheta/p^n)$.

Suppose in the $(\eta-1)$th time that $|\Pr_{M_V,\eta-1} - \Pr_{P_1,P_2,\ldots,P_n,V,\eta-1}| < \varepsilon$; then, in the ηth time, we have

$$\begin{aligned} &\left|\Pr_{M_V,\eta} - \Pr_{P_1,\ldots,P_n,V,\eta}\right| \\ &= \left|\Pr_{P_1,\ldots,P_n,V,\eta-1}\cdot\frac{\vartheta}{p^n} - \Pr_{P_1,\ldots,P_n,V,\eta-1}\cdot\frac{1-\vartheta}{p^n}\right| \\ &= \left|\Pr_{M_V,\eta-1} - \Pr_{P_1,\ldots,P_n,V,\eta-1}\right|\cdot\frac{\max\{\vartheta, 1-\vartheta\}}{p^n} \\ &= \varepsilon\cdot\frac{\max\{\vartheta, 1-\vartheta\}}{p^n}. \end{aligned} \tag{20}$$

This is a negligible amount; thus, we know that $\mathrm{M}(p, q, g, h, B_0, G)$ and $\mathrm{View}\{(P_1(d_1 s_1), \ldots, P_1(d_n s_n)), V(h)\}(p, q, g, h, B_0, G)$ are computationally indistinguishable.

4.2. Theorems from PROTOCOL 2

Theorem 1. *PROTOCOL 2 is an n-prover computational zero-knowledge proof system.*

Proof. According to the context of Section 4.1 and Definition 15, PROTOCOL 2 satisfies completeness, soundness, and zero-knowledge. Consequently, PROTOCOL 2 is an n-prover computational zero-knowledge proof system. □

Theorem 2. *Let $C \subset \mathbb{F}_q^{n+1}$ be linear $[n+1, k]$ code. An adversary $\mathscr{A}$ is l-bounded. The corruption parties chosen by $\mathscr{A}$ is $P_\Theta := \{P_{c_1}, \ldots, P_{c_\theta}\}$, where Θ is a subset of $[n]$ with $|\Theta| = \theta$. Let $\mathbf{f}$ denote the family of leakage functions $(f_1, \ldots, f_n)$, where $f_i: \mathbb{F}_q \longrightarrow \{0,1\}^{l_i}$ with $\sum_{i\in\Theta} \le l$. Suppose $\widehat{l} := \max\{l_i\}_{i\in\Theta}$ and $const_{\widehat{l}} = (2^{l\sin(\pi/2^l)}/q\sin(\pi/q)) < 1$ (when $2^l < q$). If $l_i < \widehat{l}$, then add "0" on the left of the codeword $f_i(d_i)$ such that the length of codeword is equal to the length of $\widehat{l}$. There is*

$$\{\mathbf{f}(C)\} \approx_\epsilon \{\mathbf{f}(U_{n+1})\}, \tag{21}$$

where U_{n+1} is the uniform distribution on $\mathbb{F}_q^{n+1}$ and $\epsilon = (1/2)q^{n+1-k\widehat{l}^k}$. That is,

$$\left\{\{f_i(d_i)\}_{i\in[n]}: \left(d_0 \,\|\, \mathbf{d}\right) \longleftarrow C\right\} \approx_\epsilon \left\{\{f_i(d_i)\}_{i\in[n]}: \left(d_0 \,\|\, \mathbf{d}\right) \longleftarrow U_{n+1}\right\}. \tag{22}$$

Proof. Our secret-sharing scheme is a linear (also additive) secret-sharing scheme, and the detailed proof is similar to Theorem 4.5 in [24]. □

Theorem 3. *Under the bounded-leakage model, PROTOCOL 2 is an (θ, l, ϵ)-LRSS.*

Proof. Let C be an $[n+1, n]$ linear code. The secret $s \in \mathbb{F}_q$ is shared into n shares $\mathbf{s} = (s_1, \ldots, s_n)$ such that $s = \sum_{i=1}^n s_i$. Let $\mathbf{f} = (f_1, \ldots, f_n)$ be family of leakage functions where $f_i: \mathbb{F}_q \longrightarrow \{0,1\}^{l_i}$, $\widehat{l} := \max_{i\in[n]}\{l_i\}$. For $0 \in \mathbb{F}_q$, $\mathrm{Share}(0)$ is uniformly distributed on C and sample $\mathbf{z} \longleftarrow \mathrm{Share}(0)$. Note that $\mathbf{e} = (1, 0, \ldots, 0)$ and compute $s \cdot \mathbf{e} = (s, 0, \ldots, 0)$. Then, we can obtain the coset of $\mathrm{Share}(0)$ for the distribution $\mathrm{Share}(s)$. For any secret $s \in \mathbb{F}_q$, there is

$$\mathbf{f}(\mathrm{Share}(s)) = \mathbf{f}(\mathrm{Share}(0) + s\mathbf{e}) = \mathbf{f}'(\mathrm{Share}(0)), \tag{23}$$

where $\mathbf{f}' = (f'_1, f'_2, \ldots, f'_n)$ with $f'_1(x) = f_1(z+s)$ and $f'_i = f_i$, for $i = 2, \ldots, n$.

For the uniform distrubtion $U_{[n+1]}$ over C,

$$\mathbf{f}\left(U_{[n+1]}\right) = \mathbf{f}'\left(U_{[n+1]} - s\mathbf{e}\right). \tag{24}$$

Then, statistical distance between $\mathbf{f}'(\mathrm{Share}(0))$ and $\mathbf{f}'(U_{[n+1]} - s\mathbf{e})$ is $|\mathbf{f}'(\mathrm{Share}(0)) - \mathbf{f}'(U_{[n+1]} - s\mathbf{e})|_{\mathrm{SD}} \le (1/2)q^{\widehat{l}^n}$. According to the triangle inequality $|A - B| \le |A - X| + |X - B|$ and two secrets $s^{(1)}, s^{(2)} \in \mathbb{F}_q$, there is

$$\begin{aligned}&\left|\mathbf{f}\left(\text{Share}\left(s^{(1)}\right)\right)-\mathbf{f}\left(\text{Share}\left(s^{(2)}\right)\right)\right|_{\text{SD}}\\&\leq\left|\mathbf{f}\left(\text{Share}\left(s^{(1)}\right)\right)-\mathbf{f}\left(U_{[n+1]}\right)\right|_{\text{SD}}\\&\quad+\left|\mathbf{f}\left(U_{[n+1]}\right)-\mathbf{f}\left(\text{Share}\left(s^{(2)}\right)\right)\right|_{\text{SD}}\\&\leq q\widehat{l}^{n}.\end{aligned}\tag{25}$$

Let the leakage protocol Leak is denoted by $\text{Leak}_{\Theta,\mathbf{f}}(\mathbf{s})=(\mathbf{s}_{\Theta},\mathbf{f}(\|\mathbf{s}_{\Theta}\|\mathbf{s}_{[n]\setminus\Theta}))$. Then,

$$\left\{\text{Leak}_{\Theta,\mathbf{f}}(\mathbf{s}):\ \mathbf{s}\leftarrow\text{Share}\left(s^{(1)}\right)\right\}\approx_{\varepsilon}\left\{\text{Leak}_{\Theta,\mathbf{f}}(\mathbf{s}):\ \mathbf{s}\longleftarrow\text{Share}\left(s^{(2)}\right)\right\},\tag{26}$$

where $\varepsilon=(1/2)q\widehat{l}^{n}$. □

5. Group Identification via PROTOCOL 2

According to Def. 17 in Section 2.3 and PROTOCOL 2, we can construct a group identification protocol GID-scheme with the following properties: the valid identities can be believed by the verifier only for the qualified group members, not the unqualified members; the verifier gains nothing other than believing that the qualified members have valid identities. Let $P_n=\{P_1,\ldots,P_n\}$ be a group set, and let (P_n,V) be an n-prover interactive proof system. The GID-scheme $\{\text{ParGen},\text{KeyGen},P_n,V\}$ can be constructed by Table 2.

According to the work presented in [37, 40], our GID-scheme is secure against (classical) passive attacks.

Theorem 4. *Let $(ParGen, KeyGen, P_n, V)$ be a GID-scheme and $R=\{(\overline{pk},\overline{sk})\mid\overline{sk}=\{s_0\}\cup\{s_i\}_{i=1}^{n},\overline{pk}=\{pk_0\}\cup\{pk_i=g^{sk_i}\bmod q\}_{i=1}^{n}\}$ be a hard relation with key generator KeyGen. We write (P_n,V) to denote the prover set and the verifier in a $\sum$-protocol for R with 2-bit challenges. Suppose that the $\sum$-protocol is complete, 2-special sound, and honest verifier zero-knowledge. Then, our GID-scheme is secure against emulation under active attacks.*

Proof. If we write $(\overline{pk},\overline{sk})\longleftarrow\text{KeyGen}$ by the notation $(x,y)\longleftarrow\text{K}(1^{\lambda})$ and the transcript (x_i,t,y_i) by the notation $(\alpha_i,\beta_i,\gamma_i)$, then the $\sum-$ protocol and our GID-scheme are equal, where $i\in[n]$. For details, see Theorem 5 of [41].

In the bounded-leakage model, we have the following theorem. □

Theorem 5. *Under the discrete logarithm assumption, our GID-scheme is actively secure under pre-emulation attacks with leakage $l=(n-1)log(q)-\lambda\geqslant(1-(2/n))|sk|$ bits. It is secure against an arbitrary time leakage attack with $l^{*}=(1/2)l$ bits.*

Proof. Through the contradiction assumption, in the pre-emulation leakage attack game $\text{ID}_{\text{prel}}^{\lambda}\mathscr{A}^{*}$ there is an adversary $\mathscr{A}^{*}$ with a nonnegligible advantage. Then, we need to detect two different secret keys $\overline{sk}$ and $\overline{sk}^{*}$ for a public key $\overline{pk}$, for the randomly chosen $params^{*}$, since $params$ is not known. In fact, in the test phase, $\mathscr{M}$ randomly chooses the $(\overline{pk},t\overline{sk})$ key pair and utilizes $\overline{sk}$ to model the leakage oracle $\mathcal{O}_{\overline{sk}}^{\lambda,l}(\cdot)$ and an honest prover oracle P for the adversary $\mathscr{A}^{*}$. Later, in the emulation phase, Mperforms $\mathscr{A}^{*}$ twice (for attack games $\text{ID}_{\text{pre}_l}^{\lambda}\mathscr{A}^{*},\text{ID}_{\text{arb}_l}^{\lambda}\mathscr{A}^{*}$), with two distinct randomly chosen challenges β,β^{*}. Because $\mathscr{A}^{*}$ generates two valid transcripts (α,β,γ), $(\alpha,\beta^{*},\gamma)$ with a nonnegligible advantage, $\mathscr{M}$ can recover or find a secret key $\overline{sk}^{*}$ by using these two transcripts, according to the 2-special soundness property of the $\sum$-protocol.

We now need to analyze the probability of $\overline{sk}=\overline{sk}^{*}$. Table 3 presents three experiments that are performed by the adversary $\mathscr{A}^{*}$. In experiment Ext_0, $\mathscr{A}^{*}$ obtains $\overline{pk}$ and can access the oracle $O_{\overline{sk}}^{\lambda,l}(\cdot),P(\overline{pk},t\overline{sk})$; in experiment Ext_1, $\mathscr{A}^{*}$ obtains $\overline{pk}$ and only accesses the oracle $P(\overline{pk},t\overline{sk})$; in experiment Ext_2, A^{*} only obtains $\overline{pk}$. According to the construction of $\text{Ext}_0,\text{Ext}_1$, and Ext_2, we obtain the following inequality:

$$\begin{aligned}\tilde{H}_{\infty}\left(\overline{sk}\mid\text{Ext}_0\right)&\geq\tilde{H}_{\infty}\left(\overline{sk}\mid\text{Ext}_1\right)-l\\&\geq\tilde{H}_{\infty}\left(\overline{sk}\mid\text{Ext}_2\right)-l=\tilde{H}_{\infty}(\overline{sk}\mid\overline{pk})-l\\&\geq(n-1)\log(q)-l\geq\lambda.\end{aligned}\tag{27}$$

where $H_{\infty}(X)=-\log(\max_{x\in\mathcal{X}}\text{Pr}_X(x))$ denotes the mini-entropy of a random variable X with probability distribution Pr_X over X, and $\tilde{H}_{\infty}(X\mid\text{Ext})=-\log(\max_{\mathscr{A}}\text{Pr}_{\mathscr{A}^{Ext(\cdot)}}(x))$ denotes the (average-) conditional mini-entropy of a random variable X conditioned on experiment Ext. The detailed proof of equation (27) can be found in [20]. Due to the above mentioned equation, $\mathscr{M}$ outputs $\overline{sk}=\overline{sk}^{*}$ with the upper bound of probability $2^{-\lambda}$. Therefore, $\mathscr{M}$ generates two different secure keys $\overline{sk}$ and $\overline{sk}^{*}$ with a nonnegligible advantage.

For an arbitrary time leakage game $\text{ID}_{\text{arb}l}^{\lambda}(\mathscr{A})$, in the emulation phase, $\mathscr{M}$ can access the leakage oracle with l^{*} bits. Based on two distinct challenges, $\mathscr{M}$ performs the emulation phase twice; thus, the leakage bit is $2l^{*}$. Consequently, only $l^{*}=(1/2)l$ bits of arbitrary time leakage can be handled. □

5.1. Comparisons with Other Schemes. In this section, we compare our ID-scheme with several previous works. According to the construction of the GID-schemes, we summarize the main parameters in Table 4. The first column presents the compared related works. The second column indicates the size of the public parameters shared by all parties. The third and fourth columns indicate the size of the public key and secret key, respectively. The fifth column denotes the size of each party' s communication complexity. The last column shows the size of allowed leakage l, which was measured in bits.

The comparison of schemes in Table 4 is summaried as follows:

The $\text{Okamato}_m^{\lambda}$ scheme in [42] uses m generators and can tolerate the leakage bit $l=(1-(1/m))|sk|$, where λ denotes the security parameter. This scheme only provides an adequate method for *relative leakage*$1-(1/m)$, not for large *absolute leakage* l, and it

Table 2: Our GID-scheme.

– params⟵ParGen(1^λ): perform the Initialization of PROTOCOL 2 in Section 3, and set params = (B_0, p, q, g, G)
(i) $(\overline{pk}, \overline{sk})$⟵KeyGen: execute the distribution protocol of PROTOCOL 2 in Section 3, and set $\overline{sk} = \{s_0\} \cup \{s_i\}_{i=1}^n$ and $\overline{pk} = \{pk_0\} \cup \{pk_i = g^{sk_i} \bmod \ q\}_{i=1}^n$, where $sk = s = s_0$, $pk_0 = g^{s_0} \bmod \ q$ and $sk_i = s_i$ for $i \in [n]$
(ii) P, V: the PPT turning machines P, V run the following protocol:
*Step P*1: for $i \in [n]$, every P_i chooses $r_i \in F_q$, and sends $\alpha_i = g^{r_i} \bmod \ q$ to V
*Step V*1: choose a random number $\beta_i \in \{1, 2\}$ and send the challenge β_i to every P_i. Set $\beta_i = z$, for any $i, j \in [n]$
*Step P*2: for $i \in [n]$, every P_i computes $\gamma_i = r_i - zd_i sk_i \bmod \ p$ and sends γ_i to V
*Step V*2: for $i \in [n]$, V accepts the identification of P_i if and only if the transcript $(\alpha_i, \beta_i, \gamma_i)$ is satisfied $\mathrm{Out}_V(\alpha_i, \beta_i, \gamma_i) \stackrel{?}{=} 1$

Table 3: Experiments performed by the adversary $\mathscr{A}^*$.

Ext_0	Ext_1	Ext_2
(i) $\mathscr{A}^*$ holds the public key $\overline{pk}$ (ii) $\mathscr{A}^*$ allows access to the leakage oracle $\mathcal{O}_{\overline{sk}}^{\lambda,l}(\cdot)$ and an honest prover oracle $P(\overline{pk}, t\overline{sk})$	(i) $\mathscr{A}^*$ holds the public key $\overline{pk}$ (ii) $\mathscr{A}^*$ only allows access to the honest prover oracle $P(\overline{pk}, t\overline{sk})$	(i) $\mathscr{A}^*$ only obtains access to the public key $\overline{pk}$

Table 4: Comparison of our scheme with other schemes.

Scheme	params	pk	sk	Comm.	Leakage l bits
Okamato$_m^\lambda$ [42]	m	1	m	$m + O(1)$	$\approx(1 - (1/m))\lvert sk\rvert$
[20] DirProd$_{n,m,t}^\lambda$	m	1	mn	$O(tm)$	$\approx(1 - (1/m) - O(t/\lambda))\lvert sk\rvert$
[20] ComDirProd$_{n,m,t}^\lambda$	m	1	mn	$m + O(1)$	$\approx(1 - (1/m) - O(t/\lambda))\lvert sk\rvert$
LpSS [38]	n	1	n	$O(n \log p)$	=0
Local LRLSS [24]	n	1	n	$O(n)$	$=\log(p/4)$
Our scheme	n	1	n	$O(n \log q)$	$\approx(1 - (2/n))\lvert sk\rvert$

does not provide a proportional increase in communication complexity.

As an extension of the Okamato$_m^\lambda$ scheme, the researchers in [20] propose two schemes (DirProd$_{n,m,t}^\lambda$ and ComDirProd$_{n,m,t}^\lambda$), which add two additional parameters n (the number of Okamato$_m^\lambda$ key pairs stored) and t ($\approx O(m\lambda)$, the number of Okamato$_m^\lambda$ keys used). DirProd$_{n,m,t}^\lambda$ and ComDirProd$_{n,m,t}^\lambda$ can tolerate leakage bits $(1 - (1/m) - O(t/\lambda))|sk|$ and $(1 - (1/m) - O(t/\lambda))|sk|$, respectively.

In [38], the authors constructed a leakproof SSS by using threshold SSS, and they also proved that the master secret s can be shared an arbitrary number of times. However, their schemes do not consider the security of the scenario where there is some information leaked.

Local LRLSS [24] provides a local leakage-resilience of (additive or (n, t)-Shamir) SSS over field $\mathbb{F}_p$ (p is a large prime), which is secure under local leakage attacks when $\log(p/4)$ bits are leaked from every share.

Inspired by the work of [38], in our scheme, we construct a secret-sharing scheme by using linear codes for realizing a access structure, in which the master key can be shared as many times as designed in the random oracle model. Moreover, our scheme is (θ, l, ϵ)-LRSS under the bounded-leakage model. Based on our LRSS scheme, we construct a GID-scheme, which is proven to be leakage-resilient under the attacks $\mathrm{ID}_{\mathrm{pre}l}^\lambda(\mathscr{A})$ and $\mathrm{ID}_{\mathrm{arb}l}^\lambda(\mathscr{A})$.

6. Conclusions and Future Work

We proposed a secret-sharing scheme that realized access structure based on linear codes. According to the definitions of zero-knowledge proof system and security model, we proved that our protocol is a multiprover zero-knowledge proof system in the random oracle model. Our protocol is also leakage-resilient secret-sharing scheme (LRSS) in the bounded-leakage model. In our LRSS, the security is guaranteed even if the adversary learns leakage information is bounded by l bits.

Moreover, we presented a GID-scheme from our LRSS scheme, and it is leakage-resilient under the leakage attacks $\mathrm{ID}_{\mathrm{pre}l}^\lambda(\mathscr{A})$ and $\mathrm{ID}_{\mathrm{arb}l}^\lambda(\mathscr{A})$ in the bounded-leakage model. In our leakage-resilient GID-scheme, any authorized party sets can prove to the verifier that they share the secret key without leaking any information about their individual shares to adversary and can guarantee security even though l bits are retrieved by the malicious adversary; any authorized parties can prove themselves to keep the corresponding valid secret share.

In future work, we want to construct a practical dynamic secret-sharing scheme. In this dynamic secret-sharing scheme, there are more than one access structure, and we want to enable only one of them to be active to reconstruct the predefined secret.

Data Availability

No data were used to support this study.

Disclosure

This work is a major revision of a preprint of an article accepted by the International Symposium on Cyberspace Safety and Security (CSS 2019) and is subject to *Complexity*.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the Natural Science Foundation of China (Grant nos. 61702125 and 61702126).

References

[1] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612-613, 1979.

[2] G. R. Blakley, "Safeguarding cryptographic keys," in *Proceedings of the National Computer Conference*, vol. 48, pp. 313–317, NewYork, NY, USA, June 1979.

[3] Y. Desmedt and Y. Frankel, "Shared generation of authenticators and signatures," in *Proceedings of the Annual International Cryptology Conference*, pp. 457–469, Springer, Santa Barbara, CA, USA, August 1991.

[4] R. Cramer, I. Damgård, and U. Maurer, "General secure multi-party computation from any linear secret-sharing scheme," in *Advances in Cryptology—EUROCRYPT 2000*, B. Preneel, Ed., vol. 1807, pp. 313–334, Springer, Berlin, Germany, 2000.

[5] J. Li, Y. H. Huang, Y. Wei et al., "Searchable symmetric encryption with forward search privacy," *IEEE Transactions on Dependable and Secure Computing*, 2019.

[6] L. Qi, X. Zhang, W. Dou, C. Hu, C. Yang, and J. Chen, "A two-stage locality-sensitive hashing based approach for privacy-preserving mobile service recommendation in cross-platform edge environment," *Future Generation Computer Systems*, vol. 88, pp. 636–643, 2018.

[7] L. Qi, X. Zhang, W. Dou, and Q. Ni, "A distributed locality-sensitive hashing based approach for cloud service recommendation from multi-source data," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 636–643, 2017.

[8] T. Tassa, "Generalized oblivious transfer by secret sharing," *Designs, Codes and Cryptography*, vol. 58, no. 1, pp. 11–21, 2011.

[9] M. Naor and A. Wool, "Access control and signatures via quorum secret sharing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 9, no. 9, pp. 909–922, 1998.

[10] I. Komargodski and A. Paskin-Cherniavsky, "Evolving secret sharing: dynamic thresholds and robustness," in *Proceedings of the Theory of Cryptography Conference*, pp. 379–393, Springer, Baltimore, MD, USA, November 2017.

[11] M. Karchmer and A. Wigderson, "On span programs," in *Proceeding of the 8th IEEE Structure in Complexity Theory*, pp. 102–111, San Diego, CA, USA, May 1993.

[12] A. Beimel, O. Farràs, Y. Mintz, and N. Peter, "Linear secret-sharing schemes for forbidden graph access structures," in *Proceedings of the Theory of Cryptography Conference*, pp. 394–423, Springer, Baltimore, MD, USA, November 2017.

[13] R. J. McEliece and D. V. Sarwate, "On sharing secrets and reed-solomon codes," *Communications of the ACM*, vol. 24, no. 9, pp. 583-584, 1981.

[14] G. R. Blakley and G. A. Kabatianski, "Ideal perfect threshold schemes and mds codes," in *Proceedings of 1995 IEEE International Symposium on Information Theory*, p. 488, IEEE, Vancouver, Canada, September 1995.

[15] R. Cramer, I. B. Damgård, N. Döttling, S. Fehr, and G. Spini, "Linear secret sharing schemes from error correcting codes and universal hash functions," in *Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 313–336, Springer, Sofia, Bulgaria, April 2015.

[16] S. Mesnager, F. Özbudak, and A. Sınak, "Linear codes from weakly regular plateaued functions and their secret sharing schemes," *Designs, Codes and Cryptography*, vol. 87, no. 2-3, pp. 463–480, 2019.

[17] J. L. Massey, "Minimal codewords and secret sharing," in *Proceedings of the 6th Joint Swedish-Russian International Workshop on Information Theory*, pp. 276–279, Citeseer, Mölle, Sweden, August 1993.

[18] J. L. Massey, *Some Applications of Coding Theory in cryptography. Codes and Ciphers: Cryptography and Coding IV*, Eindhoven University Press, Eindhoven, Netherlands, 1995.

[19] S. Dziembowski and K. Pietrzak, "Intrusion-resilient secret sharing," in *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, pp. 227–237, IEEE, Providence, RI, USA, October 2007.

[20] J. Alwen, Y. Dodis, and D. Wichs, "Leakage-resilient public-key cryptography in the bounded-retrieval model," in *Proceedings of the 29th Annual International Cryptology Conference*, pp. 36–54, Springer, Santa Barbara, CA, USA, August 2009.

[21] E. Kiltz and K. Pietrzak, "Leakage resilient elgamal encryption," in *Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security*, pp. 595–612, Springer, Singapore, December 2010.

[22] C. Hazay, A. López-Alt, H. Wee, and D. Wichs, "Leakage-resilient cryptography from minimal assumptions," *Journal of Cryptology*, vol. 29, no. 3, pp. 514–551, 2016.

[23] V. Goyal and A. Kumar, "Non-malleable secret sharing for general access structures," in *Proceedings of the Annual International Cryptology Conference*, pp. 501–530, Springer, Santa Barbara, CA, USA, August 2018.

[24] F. Benhamouda, A. Degwekar, Y. Ishai, and T. Rabin, "On the local leakage resilience of linear secret sharing schemes," in *Proceedings of the 29th Annual International Cryptology Conference*, pp. 531–561, Springer, Santa Barbara, CA, USA, August 2018.

[25] S. Faust, T. Rabin, L. Reyzin, E. Tromer, and V. Vaikuntanathan, "Protecting circuits from leakage: the computationally-bounded and noisy cases," in *Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 135–156, Springer, French Riviera, France, 2010.

[26] M. Ajtai, "Secure computation with information leaking to an adversary," in *Proceedings of the Forty-Third Annual ACM Symposium on Theory of Computing*, pp. 715–724, ACM, New York, NY, USA, 2011.

[27] S. Goldwasser and G. N. Rothblum, "Securing computation against continuous leakage," in *Proceedings of the Annual Cryptology Conference*, pp. 59–79, Springer, Santa Barbara, CA, USA, August 2010.

[28] J. A. Halderman, S. D. Schoen, N. Heninger et al., "Lest we remember," *Communications of the ACM*, vol. 52, no. 5, pp. 91–98, 2009.

[29] S. Garg, A. Jain, and A. Sahai, "Leakage-resilient zero knowledge," in *Proceedings of the Annual Cryptology Conference*, pp. 297–315, Springer, Santa Barbara, CA, USA, August 2011.

[30] S. Dziembowski and K. Pietrzak, "Leakage-resilient cryptography," in *Proceedings of the 2008 49th Annual IEEE Symposium on Foundations of Computer Science*, pp. 293–302, IEEE, Philadelphia, PA, USA, October 2008.

[31] M. Liu, L. Xiao, and Z. Zhang, "Linear multi-secret sharing schemes based on multi-party computation," *Finite Fields and Their Applications*, vol. 12, no. 4, pp. 704–713, 2006.

[32] K. Okada and K. Kurosawa, "Mds secret-sharing scheme secure against cheaters," *IEEE Transactions on Information Theory*, vol. 46, no. 3, pp. 1078–1081, 2000.

[33] A. Kumar, R. Meka, and A. Sahai, "Leakage-resilient secret sharing," in *Electronic Colloquium on Computational Complexity (ECCC)*, vol. 25, p. 200, Weizmann Institute of Science, Rehovot, Israel, 2018.

[34] O. Goldreich, *Foundations of Cryptography Basic Tools*, Vol. 1, Cambridge University Press, Cambridge, UK, 2007.

[35] M. Ben-Or, S. Goldwasser, J. Kilian, and A. Wigderson, "Multi-prover interactive proofs: how to remove intractability assumptions," in *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, pp. 113–131, ACM, Chicago, IL, USA, 1988.

[36] I. Damgård, "On σ-protocols," in *Lecture Notes, University of Aarhus, Department for Computer Science*, Springer, Berlin, Germany, 2002.

[37] S. D. Galbraith and J. Silva, "Identification protocols and signature schemes based on supersingular isogeny problems," in *Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security*, pp. 3–33, Springer, Kobe, Japan, December 2017.

[38] C. Tang and S. Gao, "Leakproof secret sharing protocols with applications to group identification scheme," *Science China Information Sciences*, vol. 55, no. 5, pp. 1172–1185, 2012.

[39] S. Goldwasser, S. Micali, and C. Rackoff, "The knowledge complexity of interactive proof systems," *SIAM Journal on Computing*, vol. 18, no. 1, pp. 186–208, 1989.

[40] U. Feige, A. Fiat, and A. Shamir, "Zero-knowledge proofs of identity," *Journal of Cryptology*, vol. 1, no. 2, pp. 77–94, 1988.

[41] D. Venturi, "Zero-knowledge proofs and applications," University of Rome, Rome, Italy, 2015.

[42] T. Okamoto, "Provably secure and practical identification schemes and corresponding signature schemes," in *Proceedings of the International Cryptology Conference on Advances in Cryptology*, vol. 740, pp. 31–53, Springer, Santa Barbara, CA, USA, August 1992.