Zero : Jurnal Sains, Matematika, dan Terapan
E-ISSN : 2580-5754; P-ISSN : 2580-569X
Volume 10, Number 2, 2026
DOI: 10.30829/zero.v10i2.28985
Page: 411-423

Cheating Detection and Identification in Shamir Secret Sharing Scheme using Parity-Based Verification

[1]Mirza Farhan Azhari
School of Data Science, Mathematics, and Informatics, Bogor,16680, Indonesia

[2]Sugi Guritman
School of Data Science, Mathematics, and Informatics, Bogor,16680, Indonesia

[3]Jaharuddin
School of Data Science, Mathematics, and Informatics, Bogor,16680, Indonesia

[4]Teduh Wulandari Mas'oed
School of Data Science, Mathematics, and Informatics, Bogor,16680, Indonesia

Article Info

Article history:

Accepted: 10 July 2026

Keywords:
Cheating Detection;
Information-Theoretic Security;
Secret Sharing;
Shamir Threshold Scheme;
Vandermonde Matrix.

ABSTRACT

Shamir's (k, n)-threshold secret sharing scheme provides information-theoretic secrecy against unauthorized subsets. However, it does not verify whether submitted shares are genuine during reconstruction. This study proposes a parity-based modification of Shamir's scheme. In the proposed construction, each share is extended into a triplet over $\mathbb{Z}_p$. The additional component c_i serves as an authenticated verification parameter bound to the participant identity and share value. Under the authenticated parity model, the proposed scheme detects and identifies forged shares before reconstruction. Once a forged share is identified, its original value can be restored locally from the authenticated parity value. The cheating success probability is bounded by $1/p$. The construction also attains the Ogata–Kurosawa–Stinson lower bound on share size and preserves the threshold reconstruction property. The secret is embedded as the leading coefficient a_{k-1} and recovered using a recursive divided-difference formulation, requiring $O(k^2)$ field operations and $O(k)$ memory after verification. Runtime evaluation over a 256-bit prime field shows that reconstruction remains practical for large reconstruction sets. The additional parity component yields information rate $\rho = 1/2$. It also requires only one extra field element per participant. A blockchain wallet key-distribution case study is included to illustrate how authenticated parity values can support share verification and correction in institutional key recovery. Compared with prior cheating-detection schemes, the proposed construction achieves an OKS-bound cheating probability within a Shamir-based framework. It also supports recursive coefficient recovery and post-identification share correction.

This is an open access article under the CC BY-SA license.

Corresponding Author:

Mirza Farhan Azhari,
Department of Applied Mathematics,
IPB University
Email: mimir29azhari@apps.ipb.ac.id

1. INTRODUCTION

Secret sharing distributes a secret among participants. Only qualified subsets can reconstruct it. The concept was independently introduced by Shamir [1] and Blakley [2] through two constructions. Shamir's scheme uses polynomial interpolation over a finite field. Blakley's scheme uses intersections of hyperplanes. In a typical protocol, a trusted dealer generates and distributes the shares. The subsets allowed to reconstruct the secret form the authorized set. The remaining subsets are unauthorized. Together, these subsets define the access structure [3]. In a (k, n)-threshold scheme, any subset of at least k participants can reconstruct the secret. Any subset of fewer than k participants obtain no information about it. Shamir's scheme provides information-theoretic security against unauthorized subsets. It does not, however, detect forged shares during reconstruction. Dishonest participants may submit invalid shares without being identified by the original scheme.

Prior work on cheating detection and identification follows three main directions. Tompa and Woll [4] first showed that Shamir's scheme is vulnerable to cheating attacks. Dishonest participants can submit forged shares. They may obtain the correct secret while causing honest participants to reconstruct an invalid one. Their modified scheme preserves information-theoretic security. However, it requires shares larger than the theoretical minimum. Carpentieri et al. [5] established a fundamental lower bound. Any (k, n) threshold scheme with cheating probability below ε must assign shares of size at least $|S| + log(1/\varepsilon)$. This result formalizes the irreducible storage cost of cheating resistance. Liu [16] later constructed a linear (k, n) scheme under the OKS model. The scheme achieves cheating probability $\varepsilon = 1/p$ by embedding a second polynomial as a verification component. It does not correct corrupted shares after cheaters are identified. Harn and Lin [6] retained the original Shamir share size. Their method detects cheaters through polynomial inconsistency across multiple share subsets. It achieves an ideal information rate $\rho = 1$ without additional storage. However, its detection capability can fail when the reconstruction set closely approaches the threshold. From coding theory, McEliece and Sarwate [7] established that Shamir's scheme is equivalent to a Reed-Solomon code. This equivalence links share corruption resilience to error-detection bounds. Chor et al. [8] formalized verifiable secret sharing. Their construction enables consistency verification before reconstruction. Becerra and Vega [9] proposed check-digit techniques to reduce verification overhead. Munfa'ati et al. [10] introduced a recursive Gaussian-elimination algorithm over the Vandermonde structure of Shamir's scheme. Their method supports polynomial-time reconstruction with cheating detection. It reduces computational overhead compared with full matrix inversion. However, its security guarantee $\varepsilon = 2^{-\alpha\ell}$ does not attain the OKS lower bound. The scheme also lacks correction capability after cheater identification. Additional contributions include hierarchy-based detection [11], fair threshold schemes [12], symmetric bivariate polynomial constructions [13], and linear reconstruction with verification parameters [14]. These works address specific aspects of cheating resistance. They do not jointly provide tight security bounds, share correction, and efficient cheater identification within one framework.

Secret sharing is used in domains that require confidentiality and resilience against single-point failure. These domains include healthcare, cloud storage, electronic voting, IoT systems, secure multiparty computation, and blockchain-based financial security [15], [16], [17], [18]. In these settings, secret sharing protects sensitive data by distributing information across multiple entities. This improves privacy, availability, and fault tolerance [16], [17]. In blockchain and cryptocurrency applications, threshold cryptographic techniques can secure digital assets such as Bitcoin wallets. They avoid placing the entire signing key under the control of a single party [18]. Recent security vulnerabilities show why robust cheating detection is important in threshold systems. For instance, CVE-2023-33242 exposed improper share handling in threshold wallet implementations. Such weaknesses can allow full private key recovery by a malicious party. Related weaknesses in the GG18 and GG20 threshold ECDSA protocols show a similar risk. Forged or manipulated shares during reconstruction may lead to complete key compromise [19], [20]. Recent threshold ECDSA schemes improve round efficiency, public verification, and proactive share refresh [21], [22], [23]. However, they do not provide an algebraic share-integrity check at the reconstruction layer. These cases show that scheme-level secrecy must be supported by robust share verification during reconstruction.

Motivated by these developments, this study enhances Shamir's scheme with a parity-based verification parameter. The parameter detects inconsistent shares during reconstruction. A recursive procedure based on divided-difference interpolation improves secret recovery efficiency. The proposed approach addresses forged-share detection and reconstruction efficiency. It produces a Shamir-based scheme with stronger reconstruction-layer integrity. It also preserves the threshold reconstruction property.

This study addresses three research questions. First, how is the parity-based verification parameter generated and bound to each share in the linear Shamir threshold scheme? Second, how does the authenticated parity model detect and identify corrupted shares during reconstruction? This question also covers the conditions needed for correction and secret recovery. Third, what computational overhead is introduced by recursive reconstruction? This question evaluates the trade-off between share storage efficiency and reconstruction performance against the OKS lower bound.

2. RESEARCH METHOD

2.1 Threshold and Linear Secret Sharing Schemes

A (k, n)-threshold secret sharing scheme divides a secret Sinto $n$shares $v_1, \dots, v_n$. Any subset containing at least $k$shares can reconstruct S, whereas any subset containing fewer than k shares obtain no information about it. The schemes considered in this article are based on Shamir's (k, n)-threshold secret sharing scheme [1]. Shamir's scheme achieves the threshold property by evaluating a polynomial of degree $k-1$ over a finite field $\mathbb{Z}_p$. In the original construction, the secret is embedded as the constant term of the polynomial. In this work, the secret is instead embedded as the highest-degree coefficient a_{k-1}. This modification supports the parity-binding mechanism introduced in the proposed construction.

A (k, n)-threshold scheme is also a linear secret sharing scheme when the shares are linear combinations of polynomial coefficients over a finite field [24]. Let

$$f(x) = a_0 + a_1 x + \cdots + a_{k-1} x^{k-1} (\mathrm{mod} p),$$

where $a_0, \dots, a_{k-2}$ are chosen uniformly at random and $a_{k-1} = S$. For each participant P_i with distinct public identity x_i, the dealer assigns the share

$$v_i = y_i = f(x_i) (\mathrm{mod} p).$$

For any selected subset of k participants, the reconstruction problem can be written as the following linear system:

$$Y = Ta,$$

where

$$Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_k \end{bmatrix}, a = \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{k-1} \end{bmatrix},$$

And

$$T = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^{k-1} \\ 1 & x_2 & x_2^2 & \cdots & x_2^{k-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_k & x_k^2 & \cdots & x_k^{k-1} \end{bmatrix} \quad (1)$$

The matrix T is a Vandermonde matrix. It is nonsingular when the identities $x_1, \dots, x_k$ are distinct. Hence, any $k$valid shares uniquely determine the coefficient vector a, including the secret coefficient $s = a_{k-1}$. Conversely, fewer than k shares leave at least one coefficient undetermined. The secret remains information-theoretically hidden. The proposed recursive reconstruction uses this linear Vandermonde structure. It recovers a_{k-1}without explicitly computing a full matrix inverse.

2.2 Secret Sharing with Cheating Detection

Secret sharing schemes with cheating detection aim to prevent dishonest participants from manipulating reconstruction. Cheating occurs when one or more participants submit forged shares. In this case, the reconstructed secret s'appears valid but differs from the original secret s, i.e., $s' \neq s$. A scheme is called a (k, n, ϵ)-secure secret sharing scheme if it satisfies the usual (k, n)-threshold property. It also bounds the probability that up to $k-1$ dishonest participants can deceive an honest participant by $\epsilon > 0$. These schemes are commonly analyzed under the Ogata–Kurosawa–Stinson (OKS) model or the Carpentieri–De Santis–Vaccaro (CDV) model.

In a cheating-detection scheme, reconstruction includes a verification step. Given submitted shares, the reconstruction algorithm outputs the recovered secret or a rejection symbol $\perp$ when cheating is detected. Suppose $P_1, \dots, P_k$ participate in reconstruction, where $P_1, \dots, P_t$ submit forged shares $V_c = \{v_1', \dots, v_t'\}$, and the remaining participants submit valid shares $V_h = \{v_{t+1}, \dots, v_k\}$. If s' is the reconstruction output, the cheating success probability is defined as $\epsilon = \Pr[s' \notin \{s, \perp\}]$, . This probability measures the chance that an invalid secret is accepted as valid.

Cheating-detection schemes differ in security model, share size, information rate, reconstruction method, and correction capability. Tompa and Woll [4] first showed the vulnerability of Shamir's scheme under cheating attacks. Carpentieri et al. [5] established the lower bound $| V | \geq | S | / \epsilon$. Harn and Lin [6] retained the original Shamir share size but obtained attack-dependent detection guarantees. Liu [25] achieved $\epsilon = 1/p$ and $\rho = 1/2$

under the OKS model, but did not correct corrupted shares. Munfa'ati et al. [10] introduced recursive reconstruction with $\rho = 1/2$, although their cheating probability does not attain the OKS bound. Their scheme also does not support correction. The proposed scheme achieves $\epsilon = 1/p$. It attains the OKS lower bound on share size, uses recursive divided-difference reconstruction, and supports correction after cheating identification. Table 1 summarizes these comparisons.

Table 1. Comparison of cheating-detection secret sharing schemes.

Scheme	Security Model	Cheating Prob. ε	Share Size $\|V\|$	Info. Rate ρ	Recursive Recon.	Cheating Correction
Tompa & Woll [4]	CDV	$\frac{(s-1)(k-1)}{p-k}$	$> \|S\|/\varepsilon$	< 1/2	No	No
Carpentieri et al. [5]	CDV	Lower bound only	$\geq \|S\|/\varepsilon$	–	No	No
Harn & Lin [6]	Multiple	Attack-dependent	$\|S\|$	1	No	No
Liu [25]	OKS	$1/p$	$p^2 = \|S\|/\varepsilon$	1/2	No	No
Munfa'ati et al. [10]	Unconditional	$2^{-\alpha b}$	$\|S\|^2$	1/2	Yes	No
Proposed	OKS	$1/p$	$p^2 = \|S\|/\varepsilon$	1/2	Yes	Yes

2.3 Authenticated Parity Model

The proposed parity-based verification mechanism is adapted from check-digit and parity-bit principles in error-correcting codes. In such systems, a redundant value is computed from a data word. The value is later used to verify integrity [7]. The proposed scheme adopts this principle in algebraic form. Each share is bound to a multiplicative verification parameter c_i. This parameter depends on the share value and on a public polynomial evaluation determined by the participant identity. This transfers the error-detection role of parity codes into secret sharing. It does not require additional communication rounds or cryptographic assumptions beyond finite-field arithmetic. Since h_i is entirely determined by the publicly known and immutable identity x_i, the verification relation defines a bijection between y_i and c_i for any fixed x_i. The valid triple (x_i, y_i, c_i) is the unique element consistent with the stored parity value. Any modification of y_i, or simultaneous modification of both y_i and c_i, is detectable with probability $1 - 1/p$. Passing verification requires knowledge of the authenticated parity relation. The valid share value remains uniformly distributed over $\mathbb{Z}_p$ from the adversary's perspective.

The operational realization requires that c_i be protected from post-distribution modification. After distribution, the dealer publishes $\{c_1, \ldots, c_n\}$ to an authenticated storage channel. This channel is modeled as a write-once functionality. It accepts writes only from the dealer and later serves read-only queries to the combiner. Standard deployment components can instantiate this model. Examples include HSM-backed registries, blockchain -based append -only logs, or trusted bulletin boards consistent with the blockchain key-management scenario.

2.4 Information Rate

Information rate is used to evaluate the storage efficiency of secret sharing schemes. It compares the size of the secret with the size of the shares distributed to participants. In ideal secret sharing schemes, the shares are not significantly larger than the secret. A scheme is called ideal when the secret and shares belong to domains of the same size. In that case, each share stores the same amount of information as the secret [26]. This notion provides a standard measure for comparing secret sharing constructions. This paper uses it to compute the information rate of the proposed scheme.

Let P denote the set of all participants, $S$the set of all possible secret values, and V the set of all possible shares assigned to a participant $p \in P$. According to [27], the information rate of a secret sharing scheme F is defined as

$$\rho(F) = \frac{\log | S |}{\max_{p \in P} \log | V |}. \quad (2)$$

This quantity (2) is the ratio between the bit length of the secret and the maximum bit length of any participant share. Because shares cannot be smaller than the secret, the information rate satisfies. $\rho(F) \leq 1$.The value $\rho(F) = 1$ corresponds to the optimal case.

More complex access structures often require extra information in each share. The same issue appears when a scheme adds verification, cheating detection, or identification. The share size then increases, and the information rate decreases. The information rate reflects a balance between storage efficiency, structural flexibility, and security functionality. The proposed scheme attaches a parity verification parameter $c_i$to each share. The share domain expands from $\mathbb{Z}_p$ to $\mathbb{Z}_p^2$, yielding an information rate of $\rho = 1/2$. The OKS lower bound shows that this storage overhead is necessary for any scheme achieving $\varepsilon = 1/p$.

3. RESULT AND ANALYSIS

3.1 Our Proposed Scheme

This section presents the proposed scheme through two phases. The distribution phase explains how the dealer generates the polynomial and distributes shares. The reconstruction phase explains how participants recover the secret. Unlike the original Shamir scheme, the proposed method adds a parity-check mechanism. This mechanism verifies share consistency during reconstruction and identifies dishonest participants who submit invalid shares.

3.1.1 Secret Distribution Phase

The proposed scheme is derived from a linear secret sharing scheme based on Shamir's method. In this scheme, a trusted dealer P_0 distributes secret shares among n participants. The dealer constructs a polynomial over the finite field $\mathbb{Z}_p$. Random coefficients $a_0, a_1, \dots, a_{k-2}$ are chosen uniformly from $\mathbb{Z}_p$, while the coefficient a_{k-1} is determined from a random value T such that $a_{k-1} = T \pmod p$ with the condition $a_{k-1} \neq 0$. The secret is embedded as the highest-degree coefficient of the polynomial. Each participant receives one share and one parity value. The parity value enables verification during reconstruction. The process is described in Algorithm 1.

Algorithm 1: Share Distribution with Parity Check

Input: bit-length b, number of participants n, threshold k

1: Generate a random prime $p \in [0, 2^b]$and set the finite field $\mathbb{Z}_p$
2: Ensure $n < p$ with $n \in \mathbb{Z}^+$and $k < n$ with $k \in \mathbb{Z}^+$
3: Choose $a_0, a_1, \dots, a_{k-2} \leftarrow \mathbb{Z}_p$uniformly at random
4: Generate $T \leftarrow \mathbb{Z}_p^*$and set $a_{k-1} \leftarrow T(\mathrm{mod}p)$
5: Construct $f(x) \leftarrow a_0 + a_1x + \cdots + a_{k-2}x^{k-2} + a_{k-1}x^{k-1}(\mathrm{mod}p)$
6: Set $S \leftarrow a_{k-1}$
7: **for** $i = 1$ **to** n **do**
8: $y_i \leftarrow f(x_i)(\mathrm{mod}p)$
9: $h_i \leftarrow \sum_{j=0}^{k-1} x_i^j \ (\mathrm{mod}p)$
10: $c_i \leftarrow y_i \cdot h_i^{-1}(\mathrm{mod}p)$
11: $v_i \leftarrow (x_i, y_i, c_i)$
12: Securely distribute $v_i$to participant P_i
13:**end for**

Output ordered triples $v_1, v_2, \dots, v_n$ where $v_i = (x_i, y_i, c_i)$

3.1.2 Reconstruction Phase

After the secret shares v_i have been distributed to each participant in the set P, a trusted party referred to as the Pooler/Combiner collaborates with at least t participants to reconstruct the original secret S. In the original Shamir scheme, reconstruction uses at least the threshold number of shares. The combiner then constructs an interpolation polynomial. The original secret is recovered from the constant coefficient of that polynomial. In the proposed scheme, reconstruction also includes cheating detection and identification. The mechanism verifies each submitted share before secret recovery. It can identify participants who submit invalid shares. The reconstruction stages are described as follows.

Algorithm 2: Reconstruction and Verification Protocol

Input: secret shares $v_{i_1}, v_{i_2}, \dots, v_{i_m}$with $m \geq k$
1: Combiner selects subset $G = \{P_{i_1}, P_{i_2}, \dots, P_{i_m}\} \subseteq P$ with $k \leq m \leq n$
2: **for** $i = 1$**to** m **do**
3: Receive submitted share $v_i = (x_i, y_i, c_i)$ from participant P_i

4: Compute $h_i \leftarrow \sum_{j=0}^{k-1} x_i^j \pmod{p}$
5: Compute $c_i^* \leftarrow y_i \cdot h_i^{-1} \pmod{p}$
6:**end for**
7:**if** $c_i^* = c_i$ for all $i = 1, \dots, m$ **then**
8:Initialize $y_i^{(0)} \leftarrow y_i$for all $i = 1, \dots, m$
9:**for** $\ell = 1$ **to** $k - 2$ **do**
10:**for** $i = \ell + 1$ **to** k **do**
11: $y_i^{(\ell)} \leftarrow \frac{y_i^{(\ell-1)} - y_\ell^{(\ell-1)}}{x_i - x_\ell} \pmod{p}$
12:**end for**
13:**end for**
14: $a_{k-1} \leftarrow \frac{y_k^{(k-2)} - y_{k-1}^{(k-2)}}{x_k - x_{k-1}} \pmod{p}$
15:**return** $S \leftarrow a_{k-1}$
16:**else**
17:**return** proceed to Algorithm 3
18:**end if**
Output reconstructed secret S and polynomial if no cheating detected; otherwise proceed to Algorithm 3

If Algorithm 2 detects an inconsistency, the protocol proceeds to cheating detection and identification. Algorithm 3 determines the cheater set using the authenticated parity relation. Algorithm 4 then restores corrupted share values from the authenticated parity records.

Algorithm 3: Cheating Detection and Identification

Input: Modified or tampered secret share $(v_1^*, v_2^*, \dots, v_j^*)$, authenticated parity values c_i from write-once storage
1: **Initialize** $C \leftarrow \emptyset$, $H \leftarrow \emptyset$
2: **for** each secret share holder v_i^* where $m \geq k$ **do**
3: Retrieve c_i from authenticated storage channel
4: Compute $h_i \leftarrow \sum_{j=0}^{k-1} x_i^j \pmod{p}$
5: Compute $c_i^* \leftarrow y_i^* \cdot h_i^{-1} \pmod{p}$
6: if $c_i^* \neq c_i$ **then**
7: $C \leftarrow C \cup \{P_i\}$
8: else
9: $H \leftarrow H \cup \{P_i\}$
10: **end if**
11: **end for**
if $C = \emptyset$ **then**
return $H \leftarrow G$
else
return C, H; proceed to Algorithm 4
end if
Output: Cheater set $C$and honest participant set $H = G \setminus C$; if $C \neq \emptyset$ proceed to Algorithm 4

After correction, the combiner merges the restored shares with the honest set. The secret can then be recovered when at least the required number of authenticated parity-bound shares is available.

Algorithm 4: Share Correction

Input: Members of cheater set C, authenticated values c_i from write-once storage, public identities x_i, prime p, threshold k
1: **for** each $P_i \in C$ **do**
2: Retrieve c_i from authenticated storage
3: Compute $h_i \leftarrow \sum_{j=0}^{k-1} x_i^j \pmod{p}$

4: Compute $\widehat{y_i} \leftarrow c_i \cdot h_i \ (mod\ p)$
5: Update y_i^* with $\widehat{y_i}$ to form corrected triplet $(x_i, \widehat{y_i}, c_i)$
6: **end for**
7: Merge corrected shares with honest set H to recover the secret
8: **return** $S \leftarrow a_{k-1}$
Output: Corrected secret shares $(x_i, \widehat{y_i}, c_i)$ for each $P_i \in C$; The secret $S \leftarrow a_{k-1}$ is recovered if at least k authenticated parity-bound share records are available.

3.2 Recursive Formulation of the Secret Reconstruction Phase of the Scheme

This subsection presents a recursive method for computing the polynomial coefficients from the linear system $T\mathbf{a} = Y$that arises during reconstruction, grounded in Newton's divided difference interpolation framework. Instead of performing full matrix inversion or standard Gaussian elimination, the proposed approach reduces the system dimension recursively through two elementary row operations applied at each level: subtracting row ℓ from each subsequent row $i$for $i = \ell + 1, \dots, k$, then factoring out the common term $(x_i - x_\ell)$ from each affected row. Repeating this procedure from $\ell = 1$ to $\ell = k - 2$ yields the general recurrence

$$y_i^{(\ell)} = \frac{y_i^{(\ell-1)} - y_\ell^{(\ell-1)}}{x_i - x_\ell}, i = \ell + 1, \dots, k \tag{3}$$

with initialization $y_i^{(0)} = y_i$. After $k - 2$ levels a final 2×2system remains, from which the leading coefficient is recovered as

$$a_{k-1} = \frac{y_k^{(k-2)} - y_{k-1}^{(k-2)}}{x_k - x_{k-1}} \tag{4}$$

and the next coefficient as $a_{k-2} = y_{k-1}^{(k-2)} - \left(\sum_{i=1}^{k-1} x_i\right) a_{k-1}$. The remaining coefficients are recovered recursively in reverse order, and the secret is finally obtained as $S = a_{k-1}(\mathrm{mod} p)$.
The correspondence between Equations (3) and (4) and Newton's divided difference framework follows directly from their recursive interpolation structure. As described by Voudouris et al. [8], the divided difference $f[x_i, \dots, x_{i+j}]$is computed recursively from lower-order divided differences, namely $f[x_i, \dots, x_{i+j}] = \frac{f[x_{i+1}, \dots, x_{i+j}] - f[x_i, \dots, x_{i+j-1}]}{x_{i+j} - x_i}$. The leading coefficient of the degree-$(k-1)$ interpolating polynomial is precisely the $(k-1)$-th order divided difference $f[x_1, \dots, x_k]$. In this setting, each application of Equation (3) advances the computation by one divided-difference order, while Equation (4) directly recovers $a_{k-1} = f[x_1, \dots, x_k]$, which represents the secret-carrying leading coefficient. Because each level depends only on the immediately preceding level, it is sufficient to store only the current divided-difference values during the computation. Consequently, the reconstruction requires $O(k)$ storage and $O(k^2)$ field operations.

3.3 Cheating Detection and Identification Capability of the Scheme

Theorem 1.
Let the secret S be uniformly distributed over $\mathbb{Z}_p$. If up to $k - 1$ cheaters are present during the reconstruction phase, the proposed method detects cheating with success probability characterized by

$$| S | = p, \varepsilon = \frac{1}{p}, | V | = \frac{| S |}{\varepsilon},$$

and the share size achieves the lower bound under the OKS model.

Proof.
Under the authenticated parity model, let $S = \mathbb{Z}_p$and $C = \mathbb{Z}_p$, so that $| S | = p$. A share is defined as $v_i = (x_i, y_i, c_i)$ where $y_i \in S$and $c_i \in C$, hence $| V | = | S | \cdot | C | = p^2$.
The parity value computed during reconstruction is:

$$c_i^* = \frac{y_i^*}{\sum_{j=0}^{k-1} x_i^j} (\mathrm{mod} p).$$

If cheating occurs, success requires $c_i^* = c_i$, which means the adversary must submit y_i^*satisfying:

$$y_i^* = c_i \cdot h_i (\mathrm{mod} p).$$

Since $c_i^{\text{original}}$is unknown to the adversary under Authenticated parity model and is uniformly distributed over $\mathbb{Z}_p$, there is exactly one value of y_i^*satisfying this equation out of $p$possible values. It follows that:

$$\varepsilon = \Pr[c_i^* = c_i \mid y_i^* \neq y_i] = \frac{1}{p}.$$

Consequently, $\mid V \mid = p/(1/p) = \mid S \mid /\varepsilon = p^2$, and the bit-length of a share is $\lceil \log_2 \mid V \mid \rceil = \log_2(p^2)$. Under the OKS lower bound:

$$\mid V_{\text{OKS}} \mid = \left\lceil \log_2 \left(\frac{\mid S \mid -1}{\varepsilon} + 1 \right) \right\rceil = \lceil \log_2 (p^2 - p + 1) \rceil.$$

Since $\log_2(p^2 - p + 1) = \log_2(p^2) + \log_2\left(1 - \frac{1}{p} + \frac{1}{p^2}\right)$ and $-1 \leq \log_2\left(1 - \frac{1}{p} + \frac{1}{p^2}\right) \leq 0$, it follows that $\lceil \log_2 \mid V_{\text{OKS}} \mid \rceil = \log_2(p^2)$. Hence the proposed scheme attains the OKS lower bound on share size.

Theorem 2. Under authenticated parity model, no adversary can produce a forged pair (y_i, c_i^*) with $y_i^* \neq y_i$ that passes the parity verification, except with probability at most $\varepsilon = 1/p$.

Proof. Suppose an adversary submits a forged pair (y_i^*, c_i^*) with $y_i^* \neq y_i$. Verification succeeds if and only if $c_i^* = y_i^* \cdot h_i^{-1} \equiv c_i (\mathrm{mod} p)$. This requires the adversary to select $y_i^* = c_i \cdot h_i \, (mod \, p)$, which demands knowledge of c_i. Under Authenticated parity model, c_i is stored under an authenticated protocol inaccessible to participants, and is uniformly distributed over $\mathbb{Z}_p$ from the adversary's perspective. For any fixed value chosen by the adversary, the probability of satisfying the verification equation is exactly $1/p$. Simultaneous modification of both the share and parity value does not improve the adversary's success probability beyond this bound under the authenticated parity model.

3.4 Time Complexity and Storage Analysis

The computational cost of the proposed reconstruction phase consists of parity verification, share correction, and recursive coefficient recovery. For each submitted share (x_i, y_i^*, c_i), the combiner verifies $c_i^* = y_i^* h_i^{-1} (\mathrm{mod} p)$ against the authenticated value c_i. If h_i is computed directly during reconstruction, verification over the submitted shares requires $O(mk)$ field operations. Since $h_i$is public and fixed for each participant, both $h_i$and h_i^{-1}can be precomputed. With this precomputation, the verification stage is reduced to $O(m)$. When a mismatch is detected, the corrupted value is restored locally by $\hat{y}_i = c_i h_i (\mathrm{mod} p)$, requiring only constant operations per corrupted share and no additional participant interaction. The secret-bearing coefficient is then recovered from any valid or corrected shares using the recursive formulation in Section 3.2, which requires $O(k^2)$ field operations and $O(k)$ memory. The overall reconstruction complexity is $O(mk + k^2)$ without precomputation and $O(m + k^2)$ with precomputed verification factors.

To evaluate the practical runtime behavior, the proposed scheme was implemented over a 256-bit prime field with the threshold varied at $m = 128, 256, 512, 1024, 2048, 4096,$ and 8192. The reconstruction set size was varied as, and 50% of the submitted shares were corrupted in each configuration. Each experiment was repeated 30 times after 5 warm-up runs, and the execution time was reported as the mean with a 95% confidence interval. The measured runtime includes parity verification, local correction of corrupted shares, and recursive recovery of the secret coefficient $S = a_{k-1}$. As shown in Figure 1A, the mean execution time increases from 0.000336 seconds at $m = 128$ to 0.012849 seconds at $m = 8192$, confirming that the proposed reconstruction procedure remains computationally feasible even for large reconstruction sets. This increase is consistent with the theoretical complexity because the dominant varying factor is the parity verification over all submitted shares.

The storage and communication overhead of the proposed scheme comes from the additional parity component c_i. In standard Shamir reconstruction, each participant submits one field element y_i, whereas the proposed scheme submits two field elements, (y_i, c_i), with x_i treated as public identity metadata. Over a 256-bit prime field, each field element requires 32 bytes, so each submitted share requires $2 \times 32 = 64$ bytes. This gives information rate $\rho = 1/2$.

Figure 1B shows that the total share-related communication grows linearly with m. The total requirement ranges from 8192 bytes at $m = 128$ to 524288 bytes at $m = 8192$, which is approximately 0.5 MB. This overhead is modest for modern storage systems, network transmission, HSM-backed registries, and blockchain-based audit logs. The parity mechanism does not introduce additional communication rounds because participants submit their share triplets once and the combiner performs verification and correction locally.

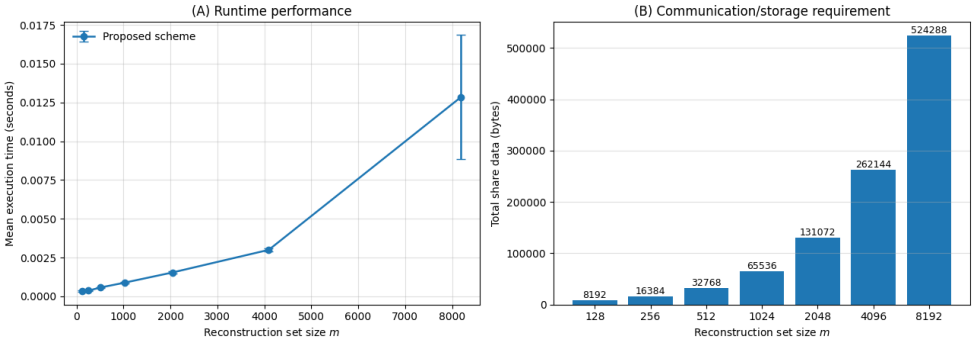

Figure 1. Computational performance and communication/storage overhead of the proposed scheme. (A) Mean execution time of the proposed reconstruction procedure for varying reconstruction set size m, reported with 95% confidence intervals. (B) Total share-related communication/storage requirement in bytes for the proposed share structure (y_i, c_i) over a 256-bit prime field.

3.5 Numerical Implementation: A Case Study on Blockchain Key Distribution

To illustrate the operational applicability of the proposed scheme, this section presents a simplified institutional blockchain wallet key distribution scenario. The purpose of this case study is to demonstrate how the proposed parity-based verification mechanism operates during share distribution, cheating detection, share correction, and secret reconstruction. The computational runtime evaluation under cryptographically realistic parameters is reported separately in Section 3.4. The present subsection focuses only on the numerical traceability of the proposed mechanism.

In this scenario, a Hardware Security Module (HSM) acts as the trusted dealer that generates the wallet private key. The key is then divided using a $(k, n) = (5,8)$ threshold secret sharing structure, meaning that at least five valid shares are required to reconstruct the secret. After share generation, the original private key is assumed to be securely erased from the HSM memory, so that no single entity permanently stores the complete key. The resulting shares are distributed to eight trusted organizational entities; each assigned a distinct public identity x_i for use in the reconstruction process.

Table 3. Participants, organizational roles, and assigned identities x_i in the blockchain wallet key distribution scheme.

Participant	Role	Identity x_i
P_1	Chief Executive Officer (CEO)	1
P_2	Chief Financial Officer (CFO)	2
P_3	Head of Security	3
P_4	Treasury Manager	4
P_5	Backup Security Node	5
P_6	Disaster Recovery Node	6
P_7	External Custodian	7
P_8	Cold Storage Backup	8

Each participant holds exactly one share of the secret and has no knowledge of the shares held by other participants. Consequently, no single participant possesses sufficient information to recover the wallet key independently. The master key can only be reconstructed when at least $k = 5$ participants collaborate and submit their shares to the secure recovery environment. This distributed control mechanism enhances security by eliminating single points of failure and preventing unauthorized unilateral access to the blockchain wallet.

The following numerical example is provided only as a pedagogical illustration. A small prime field $\mathbb{F}_{509}$ is intentionally used so that all modular arithmetic operations can be displayed explicitly and verified by hand. This parameter is not intended to represent a cryptographically secure deployment. In practical implementations, the

scheme should be instantiated over a cryptographically sized prime field, such as a 256-bit prime field, as considered in the runtime evaluation of Section 3.4.

Step 1: Share Generation. Let the participant identities be $x_1 = 1$, $x_2 = 2$, $x_3 = 3$, $x_4 = 4$, $x_5 = 5$, $x_6 = 6$, $x_7 = 7$, $x_8 = 8$. The arithmetic field is defined over $\mathbb{Z}_{509}$. The dealer sets the wallet secret as the leading coefficient $S = a_4 = 402$, and constructs the polynomial $f(x) = 123 + 77x + 250x^2 + 19x^3 + 402x^4 (\text{mod} 509)$.Evaluating the polynomial at each identity gives $(y_1, \dots, y_8) = (362,226,49,142,284,231,225,485)$.Then the dealer computes the parity value $c_i = y_i h_i^{-1} (\text{mod} 509), h_i = \sum_{j=0}^{4} x_i^j \, (\text{mod} 509)$. The resulting distributed shares are

$$\{v_1 = (1,362,276),\ v_2 = (2,226,270),\ v_3 = (3,49,379),\ v_4 = (4,142,187),$$
$$v_5 = (5,284,278),\ v_6 = (6,231,390),\ v_7 = (7,225,150),\ v_8 = (8,485,183)\}.$$

Step 2: Reconstruction with Cheating Detection. During reconstruction, all shares $v_i$for $i = 1, \dots, 8$are collected by the combiner. In this scenario, participants P_1, P_3, P_4, P_5, P_6,and P_7 modify their share values y_i, while their identities $x_i$and authenticated parity values $c_i$remain unchanged. For example, the submitted shares are

$$\{v_1^* = (1,111,276),\ v_2 = (2,226,270),\ v_3^* = (3,222,379),\ v_4^* = (4,333,187),$$
$$v_5^* = (5,444,278),\ v_6^* = (6,55,390),\ v_7^* = (7,66,150),\ v_8 = (8,485,183)\}.$$

The combiner recomputes $c_i^* = y_i^* h_i^{-1} (\text{mod} 509)$ for each submitted share. The mismatches occur for P_1, P_3, P_4, P_5, P_6, and P_7. Hence, the cheater set is $H = \{P_1, P_3, P_4, P_5, P_6, P_7\}$. Since $H \neq \emptyset$ and the authenticated parity values are available, the combiner restores each corrupted share using $\hat{y}_i = c_i h_i (\text{mod} 509)$. After correction, the consistent share set becomes $\{(1,362), (2,226), (3,49), (4,142), (5,284), (6,231), (7,225), (8,485)\}$. After the correction stage, the secret is recovered from any $k = 5$ valid shares. Choose the first five corrected shares:

$$(1,362), (2,226), (3,49), (4,142), (5,284).$$

Using the proposed recursive formulation, with initialization $y_i^{(0)} = y_i$, the recursive values are$\left(y_1^{(0)}, y_2^{(0)}, y_3^{(0)}, y_4^{(0)}, y_5^{(0)}\right) = (362,\ 226,\ 49,\ 142,\ 284)$, $\left(y_2^{(1)}, y_3^{(1)}, y_4^{(1)}, y_5^{(1)}\right) = (373,\ 98,\ 266,\ 235)$, $\left(y_3^{(2)}, y_4^{(2)}, y_5^{(2)}\right) = (234,\ 201,\ 463)$, $\left(y_4^{(3)}, y_5^{(3)}\right) = (476,\ 369)$
Then, using Equation (4), the leading coefficient is recovered as

$$a_4 = \frac{y_5^{(3)} - y_4^{(3)}}{x_5 - x_4} = \frac{369 - 476}{5 - 4} \equiv 402 (\text{mod} 509).$$

The example shows that the cheater set is correctly identified, the corrupted shares are restored, and the wallet secret is successfully recovered as $S = 402$. It also demonstrates the operational role of authenticated parity values in verifying submitted shares and enabling local correction before reconstruction.

3.6 Security Analysis and Practical Feasibility

This section analyzes the proposed scheme in terms of retrieval correctness, integrity, flexibility, and deployment feasibility. The analysis follows the authenticated parity model introduced in Section 2.4. In this model, each parity value $c_i$must remain authentic and protected from unauthorized modification. If the authenticated parity storage is compromised, the proposed construction no longer provides an independent integrity-verification layer and reduces to standard Shamir reconstruction.

Test 1: Retrieval of Information.

The proposed scheme preserves the threshold property of Shamir's construction. Any subset of at least k valid or corrected shares can recover the secret, whereas fewer than $k$shares reveal no information about it. Corrupted submissions can be identified per participant as long as the participant identity, parity value, and dealer-generated distribution state remain authentic. Recovery remains possible when at least $k$authenticated parity-bound share records are available.

Test 2: Integrity.
Integrity is enforced through the parity relation $c_i = y_i h_i^{-1} (\mathrm{mod} p)$. Under the authenticated parity model, a forged value y_i' passes verification only if it remains consistent with the dealer-generated c_i. Since the adversary cannot modify this authenticated reference value, the probability of accepting an invalid share is at most $1/p$. This guarantee no longer holds if the authenticated parity channel is compromised.

Test 3: Flexibility.
The reconstruction process does not depend on a fixed participant order. The combiner may use any subset of $m \geq k$ participants, provided that enough authenticated share records remain available. This property is useful in distributed recovery settings, where participant availability may vary. In practice, the reconstruction group should be selected conservatively to ensure that at least k authentic records can support recovery.

Test 4: Storage Requirement and Deployment Feasibility.
The proposed scheme stores an additional parity component c_i for each share. Hence, each participant holds (y_i, c_i) instead of only y_i. Consequently, the information rate is $\rho = \frac{1}{2}$. This constant-factor overhead is the trade-off for enabling cheating detection, identification, and local correction. Its quantitative impact has been evaluated in Section 3.4. In practical deployment, the main requirement is not storage capacity, but the authenticity of c_i. This requirement can be supported through a trusted bulletin board, append-only log, blockchain registry, or HSM-backed storage.

Overall, the proposed scheme provides reconstruction-layer integrity checking with constant-factor storage overhead. Its guarantees depend on the authenticity of the stored parity values. When this condition holds, inconsistent submissions can be handled before recursive reconstruction is applied. Without the authenticated parity layer, the proposed method retains Shamir's threshold reconstruction property but no longer provides the stated verification guarantee.

4. CONCLUSION

This study addressed forged-share submissions in Shamir-based reconstruction by introducing a parity-based verification component. Under the authenticated parity model, the proposed construction allows the combiner to verify submitted shares before reconstruction. Shares that fail the parity check are identified as corrupted. These corrupted shares can then be restored locally from the authenticated parity values. The cheating success probability is bounded by $1/p$. The scheme preserves the threshold reconstruction property of Shamir's scheme, attains the OKS lower bound on share size, and yields information rate $\rho = 1/2$. The recursive divided-difference formulation supports direct recovery of the secret-bearing coefficient with $O(k^2)$ field operations and $O(k)$ memory.

The construction is suitable for distributed key-management settings where share integrity during recovery is critical. Its main deployment requirement is an authenticated storage layer for parity values, such as a trusted bulletin board, append-only registry, HSM-backed database, or blockchain log.

Future work should focus on reducing the reliance on the authenticated parity model by developing a self-contained information-theoretic protection mechanism for c_i. Further extensions may also consider general access structures, proactive share refresh, multi-secret sharing, and integration with threshold signature protocols.

ACKNOWLEDGEMENT

The authors express their sincere appreciation to the anonymous reviewers for their careful evaluation and insightful comments, which have substantially contributed to the improvement of this manuscript. The authors also acknowledge the academic environment and institutional support provided by their affiliated institution, which facilitated the completion of this research.

5. REFERENCES

[1] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979, doi: 10.1145/359168.359176.

[2] G. R. Blakley, "Safeguarding cryptographic keys," in *1979 International Workshop on Managing Requirements Knowledge (MARK)*, New York, NY, USA: IEEE, Jun. 1979, pp. 313–318. doi: 10.1109/MARK.1979.8817296.

[3] D. R. Stinson, *Cryptography: theory and practice*, 3. ed. in Discrete mathematics and its applications. Boca Raton, Fla.: Chapman & Hall/CRC, 2006.

[4] M. Tompa and H. Woll, "How to share a secret with cheaters," *J. Cryptology*, vol. 1, no. 3, pp. 133–138, Oct. 1989, doi: 10.1007/BF02252871.

[5] M. Carpentieri, A. De Santis, and U. Vaccaro, "Size of Shares and Probability of Cheating in Threshold Schemes," in *Advances in Cryptology — EUROCRYPT '93*, vol. 765, T. Helleseth, Ed., in Lecture Notes in Computer Science, vol. 765. , Berlin, Heidelberg: Springer Berlin Heidelberg, 1994, pp. 118–125. doi: 10.1007/3-540-48285-7_10.

[6] L. Harn and C. Lin, "Detection and identification of cheaters in (t, n) secret sharing scheme," *Des. Codes Cryptogr.*, vol. 52, no. 1, pp. 15–24, Jul. 2009, doi: 10.1007/s10623-008-9265-8.

[7] R. J. McEliece and D. V. Sarwate, "On sharing secrets and Reed-Solomon codes," *Commun. ACM*, vol. 24, no. 9, pp. 583–584, Sep. 1981, doi: 10.1145/358746.358762.

[8] B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch, "Verifiable secret sharing and achieving simultaneity in the presence of faults," in *26th Annual Symposium on Foundations of Computer Science (sfcs 1985)*, Portland, OR, USA: IEEE, 1985, pp. 383–395. doi: 10.1109/SFCS.1985.64.

[9] D. Becerra and G. Vega, "Secret Sharing Scheme with Efficient Cheating Detection," in *Proceedings of the 2nd International Conference on Networking, Information Systems & Security*, Rabat Morocco: ACM, Mar. 2019, pp. 1–7. doi: 10.1145/3320326.3320331.

[10] R. H. Munfa'ati, S. Guritman, and B. P. Silalahi, "Application of Recursive Algorithm on Shamir's Scheme Reconstruction for Cheating Detection and Identification," *Jambura J. Math*, vol. 4, no. 1, Jan. 2022, doi: 10.34312/jjom.v4i1.12001.

[11] S. Banerjee, D. S. Gupta, and G. P. Biswas, "Hierarchy-based cheating detection and cheater identification in secret sharing schemes," in *2018 4th International Conference on Recent Advances in Information Technology (RAIT)*, Dhanbad: IEEE, Mar. 2018, pp. 1–6. doi: 10.1109/RAIT.2018.8389094.

[12] C.-S. Laih and Y.-C. Lee, "V-fairness (t, n) secret sharing scheme," *IEE Proc., Comput. Digit. Tech.*, vol. 144, no. 4, p. 245, 1997, doi: 10.1049/ip-cdt:19971223.

[13] Y. Han *et al.*, "Protected Fair Secret Sharing Based Bivariate Asymmetric Polynomials in Satellite Network," *Computers, Materials & Continua*, vol. 72, no. 3, pp. 4789–4802, 2022, doi: 10.32604/cmc.2022.027496.

[14] Z. Nur Ahzan, S. Guritman, and B. P. Silalahi, "Deteksi dan Identifikasi Pelaku Kecurangan Skema Pembagian Rahasia Linear Berbasis Skema Shamir," *Jurnal Karya Pendidikan Matematika*, vol. 7, no. 1, p. 27, Apr. 2020, doi: 10.26714/jkpm.7.1.2020.27-41.

[15] A. K. Chattopadhyay, S. Saha, A. Nag, and S. Nandi, "Secret sharing: A comprehensive survey, taxonomy and applications," *Computer Science Review*, vol. 51, p. 100608, Feb. 2024, doi: 10.1016/j.cosrev.2023.100608.

[16] P. Sarosh, S. A. Parah, G. M. Bhat, A. A. Heidari, and K. Muhammad, "Secret Sharing-based Personal Health Records Management for the Internet of Health Things," *Sustainable Cities and Society*, vol. 74, p. 103129, Nov. 2021, doi: 10.1016/j.scs.2021.103129.

[17] V. Attasena, J. Darmont, and N. Harbi, "Secret sharing for cloud data security: a survey," *The VLDB Journal*, vol. 26, no. 5, pp. 657–681, Oct. 2017, doi: 10.1007/s00778-017-0470-9.

[18] R. Gennaro, S. Goldfeder, and A. Narayanan, "Threshold-Optimal DSA/ECDSA Signatures and an Application to Bitcoin Wallet Security," in *Applied Cryptography and Network Security*, vol. 9696, M. Manulis, A.-R. Sadeghi, and S. Schneider, Eds., in Lecture Notes in Computer Science, vol. 9696. , Cham: Springer International Publishing, 2016, pp. 156–174. doi: 10.1007/978-3-319-39555-5_9.

[19] R. Canetti, R. Gennaro, S. Goldfeder, N. Makriyannis, and U. Peled, "UC Non-Interactive, Proactive, Threshold ECDSA with Identifiable Aborts," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, Virtual Event USA: ACM, Oct. 2020, pp. 1769–1787. doi: 10.1145/3372297.3423367.

[20] R. Gennaro and S. Goldfeder, "Fast Multiparty Threshold ECDSA with Fast Trustless Setup," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, Toronto Canada: ACM, Oct. 2018, pp. 1179–1194. doi: 10.1145/3243734.3243859.

[21] R. Cohen, J. Doerner, Y. Kondi, and A. Shelat, "Secure Multiparty Computation with Identifiable Abort via Vindicating Release," in *Advances in Cryptology – CRYPTO 2024*, vol. 14927, L. Reyzin and D. Stebila, Eds., in Lecture Notes in Computer Science, vol. 14927. , Cham: Springer Nature Switzerland, 2024, pp. 36–73. doi: 10.1007/978-3-031-68397-8_2.
[22] H. W. H. Wong, J. P. K. Ma, and S. S. M. Chow, "Secure Multiparty Computation of Threshold Signatures Made More Efficient," in *Proceedings 2024 Network and Distributed System Security Symposium*, San Diego, CA, USA: Internet Society, 2024. doi: 10.14722/ndss.2024.24601.
[23] B. Koziel, S. D. Gordon, and C. Gentry, "Fast Two-party Threshold ECDSA with Proactive Security," in *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, Salt Lake City UT USA: ACM, Dec. 2024, pp. 1567–1580. doi: 10.1145/3658644.3670387.
[24] J. Pieprzyk and X.-M. Zhang, "Cheating Prevention in Linear Secret Sharing," in *Information Security and Privacy*, vol. 2384, L. Batten and J. Seberry, Eds., in Lecture Notes in Computer Science, vol. 2384. , Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 121–135. doi: 10.1007/3-540-45450-0_9.
[25] Y. Liu, "Linear (k , n) secret sharing scheme with cheating detection," *Security Comm Networks*, vol. 9, no. 13, pp. 2115–2121, Sep. 2016, doi: 10.1002/sec.1467.
[26] E. F. Brickell, "Some Ideal Secret Sharing Schemes," in *Advances in Cryptology — EUROCRYPT '89*, vol. 434, J.-J. Quisquater and J. Vandewalle, Eds., in Lecture Notes in Computer Science, vol. 434. , Berlin, Heidelberg: Springer Berlin Heidelberg, 1990, pp. 468–475. doi: 10.1007/3-540-46885-4_45.
[27] Y. Kim, J. Kwon, and H.-S. Lee, "On Ideal Secret-Sharing Schemes for k-homogeneous access structures," 2023, *arXiv*. doi: 10.48550/ARXIV.2309.07479.