Evaluating the Effectiveness of Secret Sharing Algorithms in Distributed Cloud Environments

Ani Azizyan
National Polytechnic University of Armenia
Yerevan, Armenia
e-mail: aniazizyan.tt055-2@polytechnic.am

***Abstract*—This paper evaluates the effectiveness of two classical secret sharing algorithms — Shamir's Secret Sharing and Blakley's Scheme — within distributed cloud storage environments. Secret sharing provides strong confidentiality and fault tolerance by splitting secrets into shares recoverable only through threshold reconstruction, addressing key management challenges in cloud systems. A comparative analysis highlights the mathematical foundations, computational efficiency, storage overhead, and resilience of both schemes under diverse scenarios. The findings offer practical insights into selecting suitable secret sharing methods for secure and scalable cloud infrastructures.**

***Keywords*—Computer science, secret sharing, Shamir's secret sharing, Blakley's scheme.**

I. Introduction

With the rapid growth of data-centric applications, cloud storage has become essential to modern computing. The exponential increase in data from individuals, organizations, and IoT devices has driven widespread adoption of cloud platforms, offering scalable, flexible, and cost-effective solutions for storing and managing large volumes of information. However, as sensitive data increasingly resides in cloud infrastructure, ensuring confidentiality, availability, and resilience has become a critical security challenge.

Traditional encryption provides confidentiality by converting plaintext into unreadable formats, but it does not fully address key management challenges, especially in distributed and dynamic cloud environments. Centralized key management systems from major cloud providers can create single points of failure, insider threats, and regulatory compliance issues across multi-region deployments [1].

Secret sharing schemes offer an advanced cryptographic approach to enhance security and fault tolerance in distributed cloud storage. These schemes divide a secret—such as an encryption key—into multiple shares distributed across independent nodes. The original secret is reconstructed only when a predefined threshold of shares is collected, while smaller subsets reveal no information, ensuring strong protection against partial compromise. Threshold-based schemes also improve resilience, allowing data recovery even when some nodes fail [2].

Among various secret sharing algorithms, Shamir's Secret Sharing (SSS) and Blakley's Scheme are foundational. Shamir's method uses polynomial interpolation over finite fields, encoding the secret as the constant term of a polynomial. Blakley's approach uses geometric principles, representing the secret as the intersection of multiple hyperplanes. Both provide perfect secrecy, but they differ in implementation, storage requirements, and computational efficiency.

This paper evaluates Shamir's and Blakley's schemes in distributed cloud storage, motivated by the need for robust protection against breaches, hardware failures, or network partitions. A comparative analysis examines mathematical foundations, security guarantees, computational complexity, storage overhead, and fault tolerance under diverse cloud scenarios. The findings highlight trade-offs between algebraic and geometric constructions, illustrating how these differences affect performance, reliability, and integration. Selecting an appropriate secret sharing method depends on system constraints, performance goals, and deployment models. This research contributes to understanding secure distributed storage and provides practical guidance for designing resilient and efficient cloud-based systems using threshold cryptography.

II. Theoretical Foundations

Shamir's Secret Sharing (SSS), introduced by Adi Shamir in 1979, is a pioneering algorithm grounded in the mathematics of polynomial interpolation over finite fields. The core idea is to represent the secret as the constant term of a polynomial of degree k−1, where ***k*** is the threshold number of shares required to reconstruct the secret.

Process: A random polynomial f(x) of degree k−1 is generated such that:

$$f(x) = s + a_1 x + a_2 x^2 + \cdots + a_{k-1} x^{k-1}$$

where **s** is the secret embedded as the constant term and $a_1, a_2, \dots, a_{k-1}$ are random coefficients.

- Each share corresponds to a unique evaluation of this polynomial at a non-zero point x_i, producing pairs $(x_i, f(x_i)$, Figure 1 Shamir's Secret sharing

https://doi.org/10.51408/csit2025_46

- The security guarantee arises because knowing fewer than k shares provides no information about the secret due to the properties of polynomial interpolation [3].

Security:

Shamir's Secret Sharing guarantees perfect secrecy from an information-theoretic standpoint: any adversary with fewer than the threshold k shares gains no information about the secret. The distribution of the secret remains uniform and independent of partial shares, regardless of computational power. This security arises because k points uniquely define a polynomial of degree k−1; with fewer, infinitely many polynomials remain possible, leaving the secret undetermined.

Reconstruction:

When at least k shares are available, the secret can be recovered by interpolating the polynomial using *Lagrange interpolation*. The secret corresponds to the polynomial's value at x=0.

Threshold Property:

The secret reconstruction process requires at least k shares, each representing a point (x_i, y_i) on the polynomial curve f(x). Using these points, the original polynomial f(x) can be uniquely reconstructed through *Lagrange interpolation*. This mathematical technique constructs the polynomial as a weighted sum of Lagrange basis polynomials, each designed to be 1 at one known point and 0 at all others. Once the polynomial is reconstructed, evaluating it at x=0 yields the secret s=f(0), since the secret was embedded as the polynomial's constant term. The interpolation process is both efficient and reliable, ensuring exact recovery of the secret without loss or distortion when the threshold is met.

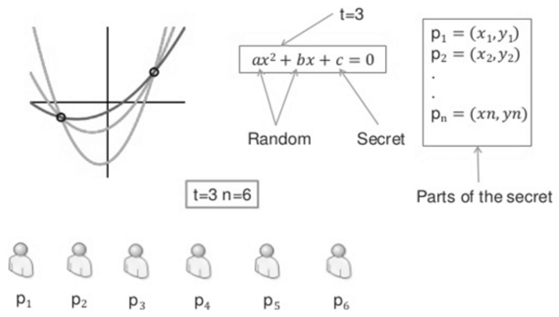

Figure 1. Shamir's Secret sharing

Blakley's Secret Sharing, proposed by George Blakley in 1979, approaches secret sharing through geometric concepts rather than algebraic ones. It models the secret as the unique intersection point of multiple hyperplanes in a k-dimensional vector space.

Process:

- The secret is encoded as a point in k - dimensional space.
- Each share corresponds to a hyperplane - represented as a linear equation - that passes through this secret point.
- Distribution of shares involves providing the coefficients defining these hyperplanes [4]. Figure 2 Blakley's Secret Sharing

Security:

Blakley's secret sharing guarantees perfect secrecy using geometric principles. Each share is a hyperplane in a k-dimensional space, and the secret is the unique point where all hyperplanes intersect. With fewer than k shares, the intersection forms a subspace of infinitely many points, so the adversary cannot narrow the secret to any finite set. Missing hyperplanes mean missing constraints, ensuring the secret remains completely hidden.

Reconstruction:

To reconstruct the secret, at least k hyperplanes (shares) are required, each expressed as a linear equation in k-dimensional space. These equations form a solvable system whose unique solution is the secret point. Using linear algebra methods such as Gaussian elimination or matrix inversion, the secret can be recovered exactly once the threshold is met. This geometric approach ensures both security and reliable reconstruction.

Drawbacks:

- **Storage Overhead:** Each share requires storing a vector of coefficients describing the hyperplane, which can be larger in size compared to Shamir's scalar shares.
- **Computational Complexity:** Solving a system of linear equations, especially in higher dimensions, may introduce computational overhead.

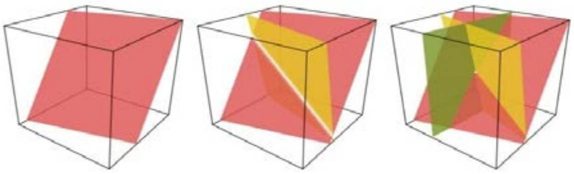

Figure 2. Blakley's Secret Sharing

III. Key Management and Distributed Encryption

In the realm of cloud security, **key management** is not merely a supporting function - it is the *keystone* of any secure encryption system. While cryptographic algorithms such as AES, RSA, and ECC are mathematically robust, the overall security of these systems is critically dependent on how the cryptographic keys are managed, stored, and accessed. In distributed cloud environments, this becomes even more significant due to the inherently decentralized, dynamic, and multi-tenant nature of such infrastructures.

Traditional key management solutions rely heavily on **centralized key repositories** or **Key Management Systems (KMS)**. Prominent cloud providers, such as Amazon Web Services (AWS KMS), Microsoft Azure Key Vault, and Google Cloud KMS, offer centralized interfaces for key creation, storage, versioning, and rotation. These platforms typically use hardware security modules (HSMs) to protect the keys from external tampering [5].

However, **centralized key management introduces several critical risks**:

- **Single Point of Failure:** If the central key store becomes unavailable due to a system outage, denialof-service attack, or internal fault, data decryption and access are completely blocked.

- **Insider Threats:** A rogue administrator or compromised privileged account can potentially exfiltrate encryption keys and access all protected data.
- **Regulatory and Jurisdictional Constraints:** Centralizing keys in a single location may violate data sovereignty or compliance requirements in multinational cloud deployments.
- **Scalability and Flexibility Limits:** Centralized architecture struggles to keep pace with the scale and heterogeneity of distributed applications, especially those spread across hybrid or multi-cloud environments.

These limitations necessitate **reimagining key management as a distributed service**, designed for resilience, security, and autonomy across nodes [6].

IV. Comparative Summary of SSS vs Blakley's Scheme

Both Shamir's Secret Sharing (SSS) and Blakley's Scheme are foundational threshold cryptographic methods that offer robust guarantees for confidentiality and fault tolerance. However, when evaluated within the context of distributed cloud environments, notable differences emerge in terms of practicality, efficiency, and implementation complexity.

Aspect	**Shamir's Secret Sharing (SSS)**	**Blakley's Scheme**
Mathematical Basis	Polynomial interpolation (modular arithmetic)	Geometry of hyperplanes
Secret Representation	Constant term of polynomial	Intersection of hyperplanes
Share Format	Pair (xi, f(xi))	Vector of hyperplane coefficients
Share Size	Small (~32B)	Larger (~96–128B for k=3)
Storage Overhead	Minimal	Higher
Bandwidth Cost	Low	High
Share Generation	O(n(k−1)); highly parallel	O(nk); less parallel
Secret Recovery	$O(k^2)$; Lagrange interpolation	$O(k^3)$; Gaussian elimination
Security	Perfect secrecy, robust to failure	Perfect secrecy, may suffer from rounding errors
Error Propagation	Low	Possible with floating-point math
Verification	Polynomial consistency	Requires solving systems
Use Case (Multi-Region)	Lower total overhead (~160B), <5ms recovery	Higher (~600B), ~20ms recovery

V. Security Considerations and Threat Models

In distributed cloud environments, secret sharing algorithms like Shamir's Secret Sharing (SSS) and Blakley's Scheme divide cryptographic secrets into multiple shares, ensuring confidentiality and fault tolerance. However, practical deployment introduces additional security challenges.

Security Guarantees and Limitations: Threshold cryptography ensures that only a minimum number of shares can reconstruct the secret. Security relies on strong randomness, secure storage, and protected communication channels (e.g., TLS).

Insider Threats and Collusion: Collusion among insiders is a risk. Mitigation includes role-based access control, separation of duties, audit monitoring, and integration with multi-party computation (MPC) to limit exposure.

Share Integrity: Malicious or corrupted shares can disrupt reconstruction. Verifiable Secret Sharing (VSS) allows checking share correctness without revealing the secret.

Fault Tolerance and Dynamic Management: Secret sharing tolerates up to n−k lost shares. Systems should support resharing when nodes change, proactively refresh shares to prevent leakage, and maintain backups for recovery.

Regulatory Compliance: Distributed shares must respect data sovereignty laws, and transparent audit logs are required to comply with GDPR, HIPAA, and other mandates.

VI. Challenges and Future Research Directions

Secret sharing schemes improve confidentiality and availability in cloud systems, but key challenges remain:

1. **Scalability and Overhead**
   Shamir's and Blakley's schemes face performance issues as participants grow. Even advanced models like VSS [3], [7] and proactive schemes add heavy computational and communication costs. Future work should optimize share generation and reconstruction, using lightweight cryptography or parallel processing for multi-cloud settings.
2. **Dynamic Access Structures**
   Traditional threshold models assume static participants, while cloud environments are highly dynamic. Flexible schemes such as ramp, hierarchical, and ABSS show promise but lack mature integration. Research should advance adaptive models that reconfigure trust without full resets.
3. **Security Against Advanced Threats**
   Secret sharing resists single-point failures but remains exposed to collusion, insider, and side-channel attacks [6]. Hybrid approaches combining TEEs, homomorphic encryption, and zero-knowledge proofs could strengthen resilience.
4. **Interoperability and Standards**
   Current implementations often lack compatibility with cloud standards. Development of interoperable libraries and alignment with frameworks like NIST is needed for broader adoption [8].

VII. Conclusion

In the context of rapidly evolving distributed cloud infrastructures, the need for robust, fault-tolerant, and efficient data protection mechanisms has never been more critical. This research has undertaken a comprehensive evaluation of two foundational secret sharing algorithms—**Shamir's Secret Sharing (SSS)** and **Blakley's Scheme**—with the goal of understanding their practical suitability in cloud-based environments.

Our findings clearly demonstrate that both SSS and Blakley's Scheme offer **perfect secrecy** and **threshold-based resilience**, making them theoretically sound for secure data distribution. However, their underlying mathematical constructs—algebraic for SSS and geometric for Blakley—lead to important trade-offs that influence their real-world application in distributed cloud storage systems.

From a performance and scalability perspective, Shamir's Secret Sharing significantly outperforms Blakley's Scheme. It offers compact share sizes, faster computation, and lower network overhead, which are critical benefits in large- scale or multi-region deployments. Its reliance on simple modular arithmetic makes it more parallelizable and easier to implement securely across heterogeneous cloud nodes.

Blakley's Scheme, while equally secure in theory, suffers from greater computational and storage overhead. Its use of hyperplane equations introduces complexity in both share generation and reconstruction, particularly in higher dimensions. Furthermore, its susceptibility to numerical inaccuracies (due to floating-point operations in linear systems) poses a potential risk in precision-critical environments.

That said, Blakley's geometric approach may offer educational and research value due to its mathematical intuitiveness and visual interpretability, especially in systems with low-dimensional thresholds (e.g., $k \leq 3$). It may also be suitable for niche applications where vector-based encoding aligns with system design constraints.

Ultimately, for cloud-native distributed storage systems, where efficiency, resilience, and implementation agility are paramount, Shamir's Secret Sharing emerges as the superior choice.

References

[1] B. Schneier, *Applied Cryptography: Protocols, Algorithms, and Source Code in C (2nd Ed.)*, New York, Wiley-Interscience, 1996.

[2] C. Cachin, S. Micali, and M. Stadler, "Computationally Private Information Retrieval with Polylogarithmic Communication", *Advances in Cryptology – EUROCRYPT 1999*, Prague, 1999.

[3] A. Shamir, "How to Share a Secret", *Communications of the ACM*, New York, 1979.

[4] G. R. Blakley, "Safeguarding cryptographic keys", *Proceedings of the National Computer Conference*, New York, 1979.

[5] Y. Desmedt, Y. Frankel, "Threshold Cryptosystems", *Advances in Cryptology — CRYPTO '89*, Santa Barbara, CA, 1989.

[6] A. Herzberg, "Proactive Secret Sharing or: How to Cope with Perpetual Leakage", *Advances in Cryptology – CRYPTO '95*, Santa Barbara, CA, 1995.

[7] V. Goyal, O. Pandey, A. Sahai, B. Waters, "Efficient Verifiable Secret Sharing Schemes", *Advances in Cryptology — EUROCRYPT 2007*, Barcelona, 2007.

[8] N. I. o. S. a. Technology,"*FIPS PUB 140-2: Security Requirements for Cryptographic Modules*", Gaithersburg, MD, U.S. Department of Commerce, NIST, 2001.