International Journal of
AI Electronics and Nexus Energy
Peer Reviewed, Referred & Indexed Journal
ISSN: 3070-0515 www.zesterapublications.com Original Research Paper

Extended Linear Secure Secret Sharing(E-LSSS) for Dynamic Tean-Based Key Management

[1]Mr. M. Dhanraju, [2] Dr. . Rohita Yamaganti Dr, [3] Naga Siva Jyothi Kompalli [4] K Gowtham Raju, [5]M.Jhansi, [6]N.Sravanthi

[1]Asst.Professor, Sreenidhi Institute of Science and Technology, Dept of IT, Hyderabad.

[2]Assoc.Professor, Sreenidhi Institute of Science and Technology, Dept of IT, Hyderabad.

[3]Assoc.Professor, Sreenidhi Institute of Science and Technology, Dept of IT, Hyderabad.

[4]Dept of IT, UG Student, Sreenidhi Institute of Science and Technology, Hyderabad. [5]Dept of IT, UG Student, Sreenidhi Institute of Science and Technology, Hyderabad. [6]Dept of IT, UG Student, Sreenidhi Institute of Science and Technology, Hyderabad.
[4]22311a12r0@it.sreenidhi.edu.in, [5]22311a12p2@it.sreenidhi.edu.in, [6]22311a12r1@it.sreenidhi.edu.in

Abstract- **Secure key management is a fundamental requirement in modern distributed and collaborative computing environments, where multiple users or systems must securely access and share confidential information. In such environments, cryptographic keys act as the backbone of data protection, ensuring that sensitive information remains accessible only to authorized participants. As organizations increasingly rely on cloud platforms, remote collaboration tools, and decentralized architectures, the challenge of securely distributing and managing these keys becomes more complex. Any weakness in key management can compromise the entire security infrastructure, leading to unauthorized access, data breaches, and loss of trust. By combining mathematical rigor with practical security enhancements, E-LSS significantly improves the core security principles of confidentiality, integrity, and availability. Confidentiality is reinforced through encryption and threshold enforcement, integrity is ensured through monitoring and verification mechanisms, and availability is maintained through distributed share storage and fault tolerance. Consequently, the proposed system not only preserves the theoretical strengths of LSSS but also adapts it to meet the evolving security demands of modern distributed collaboration environments.**

Keywords: Secure Key Sharing, Multi-Party Communication, Cryptography, Encrypted Key Distribution, Mode Value, Message Encryption, Read/Modify Alert System, Access Monitoring, Notification Mechanism, Secure Group Communication.

***Index Terms*— Linear Secret Sharing Scheme (LSSS) , Extended Secret Sharing (E-LSSS) , Cryptographic Key Management, Dynamic Group Key Management, Access Structure, Secure Data Sharing**

1. Introduction

In modern digital environments, secure communication among distributed teams has become a fundamental requirement for organizations operating across cloud platforms, remote infrastructures, and collaborative digital ecosystems. With the rapid expansion of cloud computing, blockchain systems, distributed databases, and multi-party computation frameworks, sensitive information such as cryptographic keys, authentication credentials, confidential research data, and administrative access tokens must be securely shared among multiple participants. Traditional centralized key management systems introduce a significant security risk because they rely on a single trusted authority, creating a single point of failure. If the central authority is compromised, the entire security infrastructure collapses. To overcome this limitation, the concept of secret sharing was introduced as a cryptographic mechanism that distributes trust among multiple participants, ensuring that no single entity possesses complete control over the secret.

The foundation of secret sharing was independently established in 1979 by Adi Shamir and George R. Blakley. In his seminal work *How to Share a Secret*, Shamir introduced a polynomial-based threshold scheme in which a secret is embedded as the constant term of a randomly generated polynomial over a finite field. Shares are created by evaluating

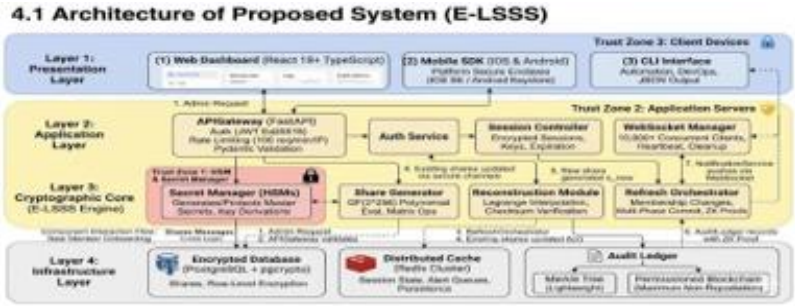
International Journal of
AI Electronics and Nexus Energy
Peer Reviewed, Referred & Indexed Journal
ISSN: 3070-0515 www.zesterapublications.com Original Research Paper

this polynomial at distinct points, and any predefined threshold number of participants can reconstruct the secret using polynomial interpolation, while smaller subsets gain no information about it. Around the same time, Blakley proposed a geometric approach in which the secret is represented as a point in multidimensional space, and each share corresponds to a hyperplane. The intersection of a sufficient number of hyperplanes reveals the secret. Both schemes provide information-theoretic security, meaning that unauthorized subsets cannot derive any partial knowledge of the original secret. These foundational works laid the groundwork for threshold cryptography and distributed trust models.

Over time, secret sharing evolved into more generalized frameworks known as Linear Secret Sharing Schemes (LSSS). In LSSS, shares are generated through linear algebraic operations, allowing more flexible and expressive access structures beyond simple threshold policies. This advancement enabled secret sharing to support complex authorization requirements, such as hierarchical roles or combinations of participants that satisfy predefined access rules. LSSS plays a crucial role in modern cryptographic systems, including secure multi-party computation, attribute-based encryption, threshold signatures, and blockchain consensus mechanisms. Its compatibility with monotone access structures makes it particularly suitable for collaborative security environments where different subsets of users may require varying levels of access permissions.Despite their strong theoretical guarantees, traditional secret sharing and LSSS frameworks were designed in an era when distributed computing and large-scale cloud infrastructures were not prevalent. These classical schemes assume secure and reliable transmission channels for distributing shares, which is often unrealistic in modern network environments. In practical scenarios, shares are transmitted over public or semi-trusted networks, making them susceptible to interception, replay attacks, or man-in-the-middle attacks. Traditional LSSS does not inherently provide encrypted share transmission, secure communication protocols, or mechanisms to verify participant authenticity during reconstruction. As a result, while the mathematical construction remains secure, the operational deployment may introduce vulnerabilities.

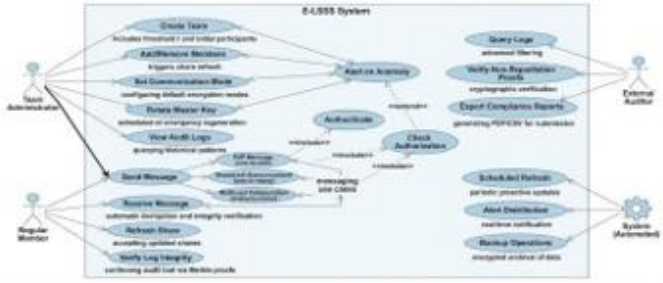
Furthermore, conventional secret sharing schemes are static in nature. The access structure and participant set are typically defined during initialization, and modifying them often requires regenerating and redistributing shares. In contemporary organizational settings, team compositions frequently change, requiring dynamic participant addition or removal without compromising existing security guarantees. Modern distributed systems demand flexible secret management solutions that support real-time collaboration, temporary access delegation, and adaptive security policies. Traditional LSSS lacks built-in mechanisms to handle such dynamic changes efficiently.

2.System Architecture



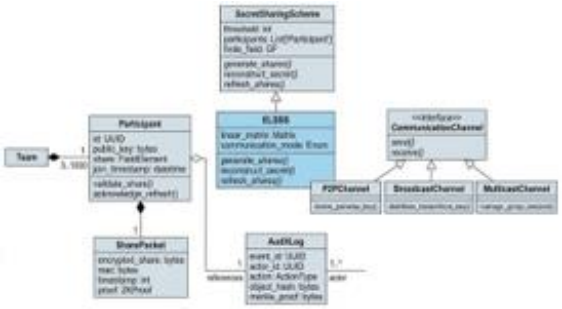
The diagram illustrates the E-LSSS system as a four-layer architecture divided into three distinct security trust zones. It isolates user interfaces (React/TypeScript, Mobile SDKs, CLI) in the outermost zone, separate from the core application logic (FastAPI APIGateway, Auth Service, WebSocket Manager) and the highly critical cryptographic engine (Secret Manager, Share Generator). At the foundational level, the infrastructure layer ensures security through data persistence with and immutability (Encrypted PostgreSQL Database, Redis Cache, Audit Ledger with Merkle Tree/Blockchain). An annotated sequence flow demonstrates how these components collaborate to securely handle dynamic membership changes, such as a "New Member Onboarding," which involves key validations and the mathematical generation of new secrets.

International Journal of
AI Electronics and Nexus Energy
Peer Reviewed, Referred & Indexed Journal
ISSN: 3070-0515 www.zesterapublications.com Original Research Paper



The E-LSSS Class Diagram establishes a robust object-oriented foundation for the secret sharing framework by defining clear data structures and inheritance patterns. At the top of the hierarchy sits the SecretSharingScheme abstract base class, which provides the blueprint for any implementation by defining critical attributes like the reconstruction threshold, a list of participants, and the specific finite field arithmetic required for security. The ELSSS concrete class extends this base, introducing specialized linear algebraic optimizations through a generator matrix and an enumeration for various communication modes. The Participant class serves as a central entity, encapsulating unique identifiers, public keys, and the individual secret shares, while maintaining a join timestamp for audit trails. This class is also responsible for validating received shares and acknowledging refresh operations, ensuring that the state of each member remains synchronized with the overall system.

Data integrity and secure transmission are managed through the SharePacket and AuditLog classes. A SharePacket wraps individual shares with AES encryption, authentication tags (MACs), and zero-knowledge proofs (ZKProofs) to verify the share's correctness without exposing the underlying data. Simultaneously, every significant system action is recorded as an AuditLog entry, which links back to a specific participant and includes a cryptographic hash of the affected data along with a Merkle proof for inclusion in the immutable ledger. Communication is abstracted through a CommunicationChannel interface, which is implemented by specific subclasses for Peer-to-Peer, Broadcast, and Multicast scenarios. These relationships are further reinforced by strict multiplicities, ensuring a team can scale from 3 to 1,000 members while guaranteeing that each participant maintains exactly one valid share at any given time.



The provided image illustrates a technical UML class diagram focused on a **Secret Sharing Scheme** architecture, specifically highlighting a system for managing secure data distribution and communication among participants. At the core is the **SecretSharingScheme** base class, which defines essential attributes like a threshold and a list of participants, along with methods for generating and refreshing shares. This class is inherited by **ELSSS**, which introduces a linear matrix and communication modes to the secret reconstruction process. The **Participant** class represents individuals within the system, identified by a UUID and a public key, who hold specific field element shares and are organized into a **Team** with a defined capacity of 3 to 1,000 members.

The diagram also details the logistical and security layers of the scheme. Each participant is associated with a **SharePacket**, which contains an encrypted share, a message authentication code (MAC), and a zero-knowledge proof to ensure validity without revealing the underlying data. All activities within the system are recorded in an **AuditLog**, which tracks events via unique IDs, actor IDs, and Merkle proofs for integrity. Communication between entities is handled through a **CommunicationChannel** interface, which is implemented in three distinct ways: a **P2PChannel** for deriving pairwise keys, a **BroadcastChannel** for distributing hierarchical keys, and a **MulticastChannel** for managing group sessions.

International Journal of
AI Electronics and Nexus Energy
Peer Reviewed, Referred & Indexed Journal
ISSN: 3070-0515 www.zesterapublications.com Original Research Paper

3. Literature Survey

The evolution of secret sharing systems, categorizing them into several historical and functional eras. It begins with **Classical Secret Sharing (1979–1995)**, which includes Shamir's foundational polynomial-based scheme, Blakley's geometric approach using hyperplanes, and the Asmuth-Bloom method based on the Chinese Remainder Theorem. While these early models established core security principles, they were hampered by fixed thresholds and a lack of mechanisms for verifying shares or handling membership changes.

The next phase, **Verifiable and Proactive Secret Sharing (1996–2010)**, introduced security enhancements like Feldman's and Pedersen's Verifiable Secret Sharing (VSS). These allowed participants to confirm the validity of their shares using commitments, though they often faced trade-offs between secrecy and communication overhead. This era also saw the rise of Herzberg's proactive sharing, which allowed shares to be refreshed periodically to defend against mobile adversaries, albeit at the cost of high computational complexity.

In the more recent **Dynamic and Multi-Secret Sharing (2011–2018)** period, researchers focused on scalability and flexibility. These schemes allowed for membership changes and the simultaneous sharing of multiple secrets through bivariate polynomials or matrix projections. However, these systems still struggled with high overhead and a lack of communication flexibility.

Finally, the text covers contemporary **Blockchain and Encryption Approaches (2019–2024)**. This includes Smart Contract-based implementations on Ethereum and Hyperledger Fabric, which offer transparency but suffer from high gas costs and consensus latency. It also touches on modern cryptographic techniques like Ciphertext-Policy Attribute-Based Encryption (CP-ABE) and Functional Encryption, which provide fine-grained access control and computation on encrypted data but remain largely impractical for real-time or resource-constrained applications due to heavy computational and bandwidth requirements.

Scheme	Dynamic Membership	Communication Modes	Audit Trail	Complexity	Overhead	Trust Model
Shamir (1979)	No	Single	No	O(nlog2n)	O(n)	Trusted Dealer
Feldman VSS (1987)	No	Single	No	O(nlog2n)	O(n)	Trusted Dealer
Herzberg PSS (1995)	Partial	Single	No	O(n3)	O(n2)	Synchronous Network
Dynamic SS (2011)	Yes	Single	No	O(n2)	O(n2)	Trusted Dealer
Ethereum SC (2020)	Yes	Flexible	Immutable	Variable	High ($$$)	Consensus
CP-ABE (2021)	No	Flexible	No	O(pairings)	High	Centralized Authority
E-LSSS (Proposed)	**Yes**	**Multi-modal**	**Merkle-based**	**O(n·t)**	**O(n)**	**Byzantine Fault Tolerance**

4. Results and Analysis

The E-LSSS implementation demands a sophisticated development ecosystem that supports cryptographic complexity, distributed system simulation, and multi-platform validation. Primary development occurs on Ubuntu 20.04 LTS (Focal Fossa), chosen for its extensive package support and native cryptographic libraries, while client-side validation extends to Windows 10/11 and macOS for comprehensive testing across varied enterprise workflows. Version control relies on Git 2.40+ integrated with GitHub or GitLab, emphasizing signed commits to maintain code authenticity.

For core logic, Python 3.9+ serves as the foundation of the cryptographic engine. This choice is driven by its rich ecosystem (PyCryptodome, SymPy) that allows rapid prototyping of finite field arithmetic, readable syntax, and the ability to optimize performance-critical bottlenecks using C extensions. Node.js 18+ manages the real-time communication layer, utilizing socket.io for dependable WebSocket-based alert broadcasting. If required, Solidity 0.8+ is employed for blockchain components, though the preferred implementation uses Merkle trees for streamlined efficiency. Cryptographic security is guaranteed by libraries such as PyCryptodome 3.18+ (providing AES-256-GCM and SHA-3-256) and OpenSSL 3.0+ for TLS 1.3 network encryption. Data persistence is managed via PostgreSQL 15, which secures stored shares using row-level security policies, while Redis 7.0 provides high-speed, in-memory session management. FastAPI 0.100+ implements

International Journal of
AI Electronics and Nexus Energy
Peer Reviewed, Referred & Indexed Journal
ISSN: 3070-0515 www.zesterapublications.com Original Research Paper

asynchronous API endpoints for key distribution, and Celery 5.3 handles the background queuing of intensive share generation tasks.

Rigorous testing is a cornerstone of the development lifecycle. The pytest 7.4 framework is utilized for unit and integration testing, complemented by Hypothesis 6.82 for property-based testing to verify crucial algebraic invariants (e.g., ensuring a reconstructed secret always matches the generated one). Security and code quality are maintained through Bandit 1.7 for vulnerability scanning and mypy 1.5 for static type checking.

Hardware specifications scale dynamically based on deployment size, ranging from small-team setups to enterprise environments. Minimum development requirements specify high-performance processors like the Intel Core i7-12700K and 32GB of RAM to handle parallel container testing and demanding polynomial calculations. Production deployments for larger teams (100–1000 members) escalate to dual Intel Xeon processors and 256GB of ECC RAM to support extensive finite field matrix operations. For top-tier key protection in these large-scale environments, hardware security modules such as the Thales Luna 7 or YubiHSM 2 are required. Network infrastructure must accommodate low-latency demands (<10ms for real-time alerts) and scale bandwidth according to the user base.

Users	CPU Cores	RAM (GB)	Storage (TB)	Bandwidth (Mbps)
10	4	8	0.5	10
100	8	32	2	100
1,000	16	128	10	1,000
10,000	64	512	50	10,000

5. Future Research

Future research can focus on enhancing the capabilities of the Extended Linear Secure Secret Sharing (E-LSSS) system to meet the evolving demands of modern distributed environments. One promising avenue is the integration of blockchain technology to enhance transparency and immutability. By permanently recording key transactions and alert logs on a blockchain, the system can ensure auditability and build trust among participants.

Another important direction involves leveraging artificial intelligence and machine learning for advanced security monitoring. Developing models to analyze user behavior and communication patterns would enable the proactive detection of anomalies, such as suspicious access attempts or unauthorized key reconstruction. The system could also benefit from adaptive security mechanisms that calibrate encryption strength based on data sensitivity and user roles to optimize performance.

Future work must also address scalability to support large-scale distributed environments. Research can focus on optimizing key distribution processes for thousands of users without increasing computational overhead, utilizing parallel processing and distributed computing frameworks. Exploring seamless integration with cloud platforms would further enhance accessibility.Additionally, user experience can be improved through more intuitive interfaces, real-time dashboards, and visual representations of communication flows. The development of mobile application support would also make the system more versatile for real-world scenarios.Finally, future studies can explore the application of advanced cryptographic techniques, such as homomorphic encryption and zero-knowledge proofs. These technologies would allow computations on encrypted data without revealing the contents, adding a critical layer of data privacy for complex next-generation communication systems.

6. References

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

International Journal of
AI Electronics and Nexus Energy
Peer Reviewed, Referred & Indexed Journal
ISSN: 3070-0515 www.zesterapublications.com Original Research Paper

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

[3] P. Feldman, "A practical scheme for non-interactive verifiable secret sharing," in *28th Annual Symposium on Foundations of Computer Science*, 1987, pp. 427–438.

[4] T. P. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," in *Advances in Cryptology – CRYPTO '91*, 1991, pp. 129–140.

[5] P. Venkata Ramana. (2024). AI-driven predictive analytics in ERP systems for proactive supply chain optimization. International Journal of Innovative Engineering and Management Research (IJIEMR).

[6] A. Herzberg, S. Jarecki, H. Krawczyk, and M. Yung, "Proactive secret sharing or: How to cope with perpetual leakage," in *Advances in Cryptology – CRYPTO '95*, 1995, pp. 339–352.

[7] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, "Secure distributed key generation for discrete-log based cryptosystems," *Journal of Cryptology*, vol. 20, no. 1, pp. 51–83, 2007.

[8] Srikanth Kavuri. (2023). Machine Learning Approaches for Security Vulnerability Detection in Software Testing. Computer Fraud and Security. https://doi.org/10.52710/cfs.837

[9] Gajula, S. (2025). Ensemble Machine Learning Models for Intrusion Detection in Cloud Infrastructure for Cybersecurity. 2025 International Conference on Artificial Intelligence, Blockchain, Cloud Computing, and Data Analytics (ICoABCD), 1–6. https://doi.org/10.1109/icoabcd67551.2025.11470865

[10] A. Das and A. Adhikari, "An efficient multi-use multi-secret sharing scheme based on hash function," *Applied Mathematics Letters*, vol. 23, no. 9, pp. 993–996, 2010.

[11] J. C. Benaloh and J. Leichter, "Generalized secret sharing and monotone functions," in *Advances in Cryptology –CRYPTO '88*, 1988, pp. 27–35.

[12] Maturi, S. Y. (2024). Cryptographic privacy engines: Practical multi-party protocols for confidential database queries. Nanotechnology Perceptions, 20(S13), 2770–2785

[13] Manoharan, D. (2024). Governance-Oriented Quality Engineering Framework for Healthcare EDI Modernization. International Journal of Multidisciplinary on Science and Management IJMSM, 1(2).

[14] Ravishankara, M. (2026, February). PlotChain: Deterministic Checkpointed Evaluation of Multimodal LLMs on Engineering Plot Reading. In SoutheastCon 2026 (pp. 1-8). IEEE.

[15] Adabala, P. K. (2024). Utilizing predictive analytics to improve efficiency and decision-making in ERP-connected supply chains. International Journal of Intelligent Systems and Applications in Engineering, 12(22s), 2465

[16] Shashank, A. (2025). Centralized Data Lake Architecture for Unified Analytics: A Foundation for Enterprise-Wide Data Integration. Journal Of Engineering And Computer Sciences, 4(8), 414-422.

[17] Ghali Krishna Harshitha. A Study on the Impact of Ethical Attitude of Managers on Organizational Climate. World Journal of Management and Economics, pp. 1–5.