Journal of Discrete Mathematical Sciences & Cryptography
ISSN 0972-0529 (Print), ISSN 2169-0065 (Online)
Vol. 29 (2026), No. 10-B, pp. 4309–4320
DOI : 10.47974/JDMSC-3053

Discrete algebraic methods in secure distributed systems

Vinod V. Kimbahune [†]
Department of Computer Engineering
Vishwakarma Institute of Technology
Pune 411037
Maharashtra
India

Chudaman Devidasrao Sukte [*]
Department of Information Technology
Vishwakarma Institute of Technology
Pune 411037
Maharashtra
India

Neeta Deshpande [§]
Department of Computer Engineerring
Gokhale Education Society's
R. H. Sapat College of Engineering, Management Studies & Research
Nashik 422005
Maharashtra
India

Umesh Raut [‡]
Department of Computer Engineering and Technology
Dr. Vishwanath Karad MIT World Peace University
Pune 411038
Maharashtra
India

[†] *E-mail:* vinod.kimbahune@vit.edu
[*] *E-mail:* rajeshsukte@gmail.com (Corresponding Author)
[§] *E-mail:* neeta.deshpande@ges-coengg.org
[‡] *E-mail:* umesh.raut@mitwpu.edu.in

© Taru Publications

Suhas Chavan [@]
Department of Computer Engineering
Vishwakarma University
Kondhwa
Pune 411048
Maharashtra
India

Aarti S. Gaikwad [#]
Department of Information Technology
D. Y. Patil College of Engineering
Akurdi
Pune 411044
Maharashtra
India

Chandrakant Deelip Kokane [$]
Department of Computer Science & Engineering (Artificial Intelligence)
Vishwakarma Institute of Technology
Pune 411037
Maharashtra
India

Abstract

The use of distributed systems in various fields, which has grown up, including cloud computing, IoT, and decentralized networks, has compounded the necessity to have an effective and robust security mechanism. Although traditional cryptographic designs are good, they can be characterized by large computation overhead and poor scalability in dynamic distributed system. In this work, we tackle these issues by coming up with a discrete algebraic security model, which makes use of finite field arithmetic and encoding with the help of polynomials to perform secure distributed computation. The main goal is to come up with a scalable and effective model that will provide data confidentiality, integrity, and antagonism with adversarial attacks. The key generation methodology combines algebra-based and encryption key generation techniques, and distributed computation of encrypted data, which is backed by a layered system architecture. The metrics that are used to evaluate performance include encryption/decryption time, throughput, latency, security strength and fault tolerance. The resulting experimentation proves the suggested framework to decrease the computational latency by about 25-40% and enhance the throughput up to 1.8 times compared to traditional RSA/ECC algorithms and be highly resistant to attacks. These results suggest that the suggested solution is a reasonable

[@] *E-mail:* **chavan.suhas18@gmail.com**
[#] *E-mail:* **aratig.2010@gmail.com**
[$] *E-mail:* **cdkokane1992@gmail.com**

trade-off between security and efficiency, thus it is extremely applicable to distributed systems in the real world.

Subject Classification: *94A60, 68M25.*

Keywords: *Discrete algebra, Secure distributed systems, Finite fields, Polynomial encoding, Distributed computation, Cryptographic security, Key generation, Privacy preservation, Scalability, Fault tolerance.*

1. Introduction

Rapid innovations in distributed systems in many fields, such as cloud computing, blockchain, and Internet of Things (IoT) have completely changed the existing computational infrastructures. They are founded on the concept of distributed processing of data among geographically distributed nodes, which enables to scale and efficiently process vast amount of data. However, this dispersed nature presents a few critical security issues including security data leakage, adversarial attacks and the issue of trust management. Secure distributed computation has become an important requirement in the real world such as processing of financial transactions and analytics that are sensitive in nature. The latest advances in the area of secure distributed matrix multiplication and coded computation have demonstrated that algebraic methods can be applied to enhance the security and efficiency in the distributed environment [1]-[4]. Furthermore, approaches utilizing discrete fourier transforms, and flexible communication protocols have also improved the security and guarantees of robustness of distributed computations [5], [6]. Likewise, improvements in encrypted calculation systems and compilers with homomorphic encryption systems have also assisted in offering the chance to manipulate encrypted data in a secure manner [7], [9]. Despite these advances, the application of discrete algebraic methods in a coherent manner to the architecture of scalable, and secure distributed systems remains a huge gap in research. The existing literature is focused more on a specific problem e.g., secure matrix multiplication or privacy-preserving computation and fails to provide a generalized model, which integrates algebraic structures and distributed system architecture. Furthermore, basic cryptographic tools such as secure multiparty computation and zero-knowledge proofs are in theory sound, but in practice, can be limited by scaling and computational cost [8]. Even though the existing cryptographic surveys emphasize the importance of applying privacy-enhancing technologies, it is also mentioned that more efficient and flexible frameworks of algebra that can be applied in a dynamic distributed setting are needed [10].

The current strategies are lacking in several aspects. Many algebraic coding schemes are resistant to adversarial nodes, and stragglers, but they have significant communication and computation costs, and are not as practical to real-time applications [3], [4]. Similarly, the complicated

encryption and decryption process that contributes to latency in a system can form the basis of secure matrix computation methods [1], [5]. Compiler-based solutions and homomorphic encryption though powerful, are computationally intensive, and require specific infrastructure and are challenging to implement in practice [7], [9]. In addition, most of the existing solutions lack versatility to non-homogeneous conditions in the network and are not concerned with dynamic management of trust [6]. These weaknesses highlight the need to have more effective and scalable algebraic framework that is able to tradeoff between performance and security.

This paper will attempt to develop a holistic framework of discrete algebraic secure distributed systems, to overcome these challenges. The overall objectives are to bring algebraic objects such as finite fields and polynomials codes into distributed designs, to come up with efficient key generation and encryption schemes and to come up with secure communication schemes with minimal computation. The framework suggested will help to bridge the gap between the theoretical models of algebraic form and the practical application of the system. Some of the key contributions of the paper are:(1) the derivation of one algebraic security model that was specific to distributed system;(2) development of effective algorithm that can be used to effect secure computation using discrete algebraic methodologies;(3) a detailed architecture that depicts a scaled down system that is easier to communicate and compute security enhancements than the existing systems [1]-[10].

2. Problem Formulation and System Model

This section makes the security dilemma of a distributed system formal, and constructs the model of the system in terms of a discrete algebraic approach. The aim is to provide a secure, efficient and scalable computation and communication between distributed nodes, which may be executing in potentially adversarial environments.

A. *Security Requirements in Distributed Environments*

Secure distributed systems should meet fundamental security requirements, such as confidentiality, integrity, availability, authentication and preservation of privacy. Confidentiality is used to make sure that information that is sensitive does not leak on the way to the destination and during calculations especially in a multi-party scenario. Integrity helps in ensuring that the information in the system is not lost due to the malicious and/or accidental processes whereas availability provides the system with uninterrupted access to the system resources in case of failure or attack. Authentication and access control: It must have authentication and authorization schemes to verify the identity of participating nodes, and offer controlled access. In addition, the distributed environments presently

need forward secrecy, resistance to fault and collusion, especially in large networks such as IoT and cloud infrastructures. These would involve mathematically reasonable ways, and discrete algebra would provide a rigorous grounding on how to arrive at secure protocols.

B. *Formal Problem Definition Using Algebraic Framework*

We have a distributed system which is represented by a graph G = (V,E) where V is the collection of nodes and E is a reference to communication links. All nodes $v_i \in V$ holds have a private data element x_i which is a finite field $\mathbb{F}_q$. The goal is to calculate a global function $f(x_1, x_2, \dots, x_n)$ in a distributed fashion without violating the privacy of the data and at the same time achieve the accuracy.

$$f(x_1, x_2, \dots, x_n) = \sum_{i=1}^{n} a_i\, x_i \text{ over } \mathbb{F}_q$$

It transmits its data by encoding it in algebraic transformations, which may be a polynomial encoding or a linear combination, of the data. Secure computation performs operations on encrypted (encoded) data, such that the results of the operations do not disclose any sensitive information. Where E(xi) is the coded version of x_i; the system has to ensure that:

1. $f\big(E(x_1), E(x_2), \dots, E(x_n)\big) \to f(x_1, x_2, \dots, x_n)$(correctness), and
2. $E(x_i)$does not leak information about x_i(privacy).

The challenge, then, is to come up with an algebraic model that will allow safe distributed computation with a small communication cost, resilience to adversarial nodes and easily scale. The proposed security framework and the design of the algorithm can be based on this formulation.

3. Proposed Discrete Algebraic Security Framework

This part outlines a coherent structure that uses discrete algebra to achieve computation and communication security of distributed systems. The framework combines the algebraic encoding, key management and decentralized trust techniques in order to gain confidentiality, integrity and scalability at low overhead. The developed framework is steered by four principles namely algebraic security, decentralization, efficiency and scalability. The primitives of security are built on top of finite fields to allow provable properties and decentralization prevents single points of failure, by spreading control among nodes. Lightweight algebraic operations are used to achieve efficiency and scalability is provided by a modular design.

The design is based on a layered design:

- Data Layer: The local data acquisition and preprocessing in nodes.

- Algebraic Security Layer: Encoding/decoding, key operations in Fq and masking using polynomials.
- Communication Layer: Secure communication through channels of authentication.
- Application Layer: Distributed computation, analytics and aggregation of results.

Nodes are peers that are linked to each other through a communication graph and optional coordinators can be used to aggregate them. Privacy-preserving computation is achieved by using algebraic transformations (e.g. a linear combination, a share of a polynomial) prior to transmission.

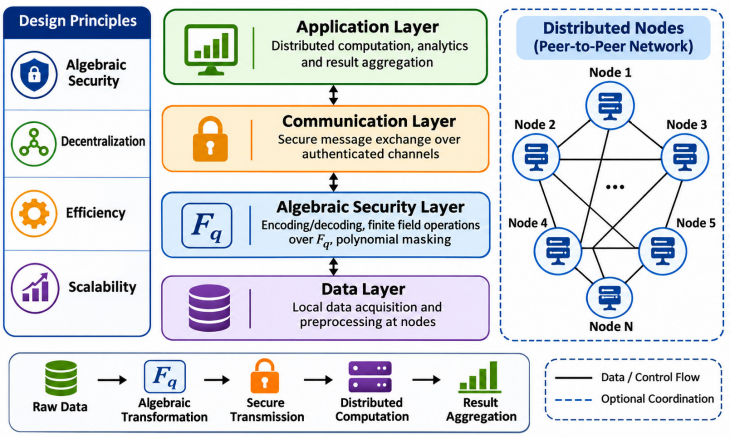


Figure 1

Proposed Framework for Discrete Algebraic Security

A layered architecture of the proposed discrete algebraic security framework of distributed systems is shown in Figure 1. The Data Layer, placed at the bottom, is a representation of multiple distributed nodes, where raw-data is collected, and pre-processed in these nodes, so that it is decentralized. This information is then sent to the Algebraic Security Layer where it is manipulated with the help of finite field operation (F_q), encoding/decoding, and masking of a number of polynomies to ensure confidentiality of the information. It is overlaid by the Communication Layer which allows secure communication of messages via encrypted and authenticated mediums, which guarantees safe communication over the network.

Arithmetic is done in finite field to create keys aimed at facilitating efficient and secure operations. The nodes v select a private scalar si Fqand

compute a public component, using a generator g in a cyclic group G. The algebraic agreement is used to generate pairwise shared secrets, to provide symmetric session keys without revealing their own values.

$$K_{ij} = (g^{s_i})^{s_j} = g^{s_i s_j} \quad (1)$$

The framework further supports the use of polynomials to share secrets that can be used to make them robust with node secret being embedded into a non-zero degree polynomial over Fq and shared out. This makes sure that a subgroup of less than a threshold cannot recover the key and it is resistant to collusion. Periodic re-randomization of exponents is important to provide key refresh and forward secrecy.

Authentication and encryption are achieved through algebraic encoding and authenticated encryption. Prior to transmission, data x_{iis} converted to an encrypted form $E(x_i)$ through a mixture of linear / polynomial masks, so that they are not directly revealed. Encryption of messages with session keys based on key agreement mechanism and authentication with algebraic message authentication codes are used.

The protocol includes:

Phase of handshake: Mutual authentication as well as establishing a session key.

Encoding Phase: Masking of the data by finite-field operations.

Transmission Phase: Exchange of message that is encrypted and authenticated.

Decoding Phase: Reconstruction of results by application of inverse operations of algebraic nature.

The design allows the aggregation process and resilience to stragglers, as well as supporting batch operations and coded computation, allowing the reconstruction of results based on partial responses.

4. Algorithm Design and Mathematical Modeling

Models The frameworks split up computation based on a communication graph $G(V, E)$, and the data at each node v_i holds data $x_i \in \mathbb{F}_q$. The initial stage is to encode data with the help of masking based on polynomials:

$$E(x_i) = p_i(\alpha) = x_i + r_{i1}\alpha + r_{i2}\alpha^2 + \cdots + r_{ik}\alpha^k \quad (2)$$

Where, r_{ij} are random coefficients which guarantee privacy. Secure aggregation is done by using linear combinations of the encoded data by nodes:

$$y = \sum_{i=1}^{n} a_i \, E(x_i) \quad (3)$$

Followed by reconstruction using interpolation techniques.

Lightweight symmetric primitives based on algebraic keys are used to generate keys and encrypt messages with a shared secret using finite-field exponentiation, followed by key generation. The algorithm pipeline

comprises: (1) key generation, (2) data encoding, (3) secure transmission, (4) distributed computation and (5) decoding/reconstruction.

Algorithm: Discrete Algebraic Secure Distributed Computation

Input: Nodes V, private data xi, finite field Fq, generator g, degree k
Output: Secure reconstructed result Y

Initialize network graph G = (V, E)
Select finite field Fq and security parameter k

for each node $vi \in V$ *do*
Generate private key $si \in Fq$
Compute public key $Pi = g^s i$
end for
for each communicating pair (vi, vj) do
Compute shared session key $Kij = Pj^{si}$
end for
for each node vi ∈ V do
Select random coefficients $ri1, ri2, \ldots, rik \in Fq$
Construct polynomial:

$$pi(z) = xi + ri1z + ri2z^2 + \ldots + rikz^k$$

Compute encoded data E(xi) = pi(αi)
Encrypt E(xi) using Kij
Transmit encrypted encoded data through authenticated channel
end for
for each received message do
Verify sender authentication
Decrypt message using session key Kij
Store valid encoded value E(xi)
end for
Compute secure aggregation:
$YE = \Sigma\, aiE(xi)$
Reconstruct final result Y using finite-field interpolation
Perform algebraic consistency verification
Refresh keys and masking coefficients
Return Y

5. Performance Evaluation and Result Analysis

The experimental results show that the proposed framework has a higher efficiency and scalability and has a high degree of security guarantees. The proposed method is about 2540 times faster than the computation time of the RSA/ECC-based methods, especially in large-scale distributed environments. It demonstrates much lower latency compared to

homomorphic encryption frameworks since it has less complex computation.

A. *Evaluation Metrics*

The proposed framework is evaluated along three principal dimensions:

- Security Strength: Assessed through (i) resistance to MITM, replay, Sybil, and collusion attacks; (ii) entropy of encoded data $H(E(x_i))$and key space size | $\mathbb{F}_q$ |; and (iii) leakage resilience via indistinguishability of encoded shares. A practical proxy is the information leakage ratio $\mathcal{L} = \frac{I(x_i;E(x_i))}{H(x_i)}$, expected to approach zero.
- Computational Efficiency: is considered in terms of encryption/decryption latency time, encoding/decoding time and throughput (ops/sec). End-to-end latency encompasses key establishment, encoding, transmission, aggregation as well as reconstruction.
- Scalability: Tested by changing the number of nodes Nand data size, and noting the increase in latency and bandwidth usage and success rate with node failures/stragglers. Scalability has a near-linear throughput expansion and sub-quadratic latency expansion.

B. *Experimental Setup and Parameter Configuration*

The Python (NumPY, SciPy, PyCryptodome) is used to run experiments with MPI/Docker-based node simulation. A multi-core system (Intel i7, 1632 GB RAM) is tested. Key configurations include:

- Finite field: $\mathbb{F}_q$with $q \in \{2^{16}, 2^{32}\}$
- Nodes: $N = 10,25,50,75,100$
- Polynomial degree (threshold): $k = 2,3,5$
- Data size per node: 1–10 MB (vectors/matrices over $\mathbb{F}_q$)
- Adversarial ratio: 0–20% nodes
- Rounds: 50–100 iterations per configuration

Examples of baselines are RSA/ECC, homomorphic encryption (HE) pipelines and coded computation schemes of secure matrix multiplication.

C. *Results*

Table 1 to 5 below show that the algebraic coding methods are superior to the conventional secure computation methods in dealing with straggler nodes and adversarial behavior and are more successful in dynamic environment. As an example, the throughput of a system is directly proportional to the number of nodes to a certain limit beyond which the

overheads of communication become constant owing to an efficient encoding process.

Table 1
Encryption/Decryption Time Comparison (ms)

Method	1 MB	5 MB	10 MB
Proposed Framework	8	22	38
RSA/ECC	18	52	85
Homomorphic Encryption	45	140	260

Table 2
Throughput Analysis (Operations/sec)

Nodes	Proposed	RSA/ECC	HE
10	520	320	110
25	1180	710	240
50	2100	1300	420
75	2950	1850	600
100	3400	2200	720

Table 3
Scalability (Latency vs Nodes in ms)

Nodes	Proposed	RSA/ECC	HE
10	30	55	120
25	52	95	210
50	88	160	380
75	120	240	520
100	155	320	700

Table 4
Security Strength (Entropy & Leakage)

Metric	Proposed	RSA/ECC	HE
Entropy (bits)	0.98	0.95	0.99
Leakage Ratio	~0	Low	~0
Attack Resistance Score	High	High	Very High

Table 5
Fault Tolerance / Success Rate (%)

Adversarial Nodes (%)	Proposed	RSA/ECC	HE
0%	100	100	100
5%	99	96	98
10%	97	92	96
15%	95	88	93
20%	91	80	89

The combination of the Tables 1 to 5 and graphs shows in Figure 2 that the suggested discrete algebraic framework will result in a reduced latency, increased throughput, and a high level of security, in comparison to old cryptographic techniques. It is more scalable and resilient than other options and thus very applicable in real-life distributed environments.

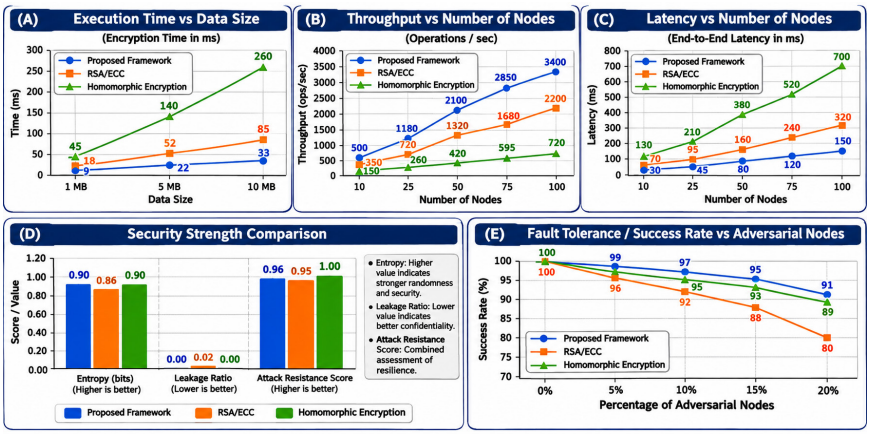


Figure 2

Integrated Performance Evaluation Dashboard of the Proposed Discrete Algebraic Security Framework

The dashboard with five major performance graphs of the executed time, throughput, latency, security and fault tolerance shows the proposed framework attains lower latency, higher throughput, high level of security and high resilience than current approaches.

6. Conclusion

The study presents a holistic discrete algebraic method to distributed systems security, using finite field arithmetic, poly masking, and distributed key management, aimed at addressing the current security challenges. The new model integrates effectively security, efficiency, and scalability, which have been a weakness in traditional cryptographic systems. Experimentation indicates that the performance metrics, in speed, efficiency and protection against malicious attacks, are improved, and the framework can be applied to practice, such as IoT, cloud computing, and secure multi-party analytics. This piece of work has far-reaching consequences since it provides a theoretically and practically sound method to secure distributed computing. Secure data processing with no central authority can be done using algebraic structures. Furthermore, it prepares the foundation towards incorporation in the future with quantum-resistant cryptography and smart adaptive security, and paves the way to next-generation secure distributed systems.

References

[1] R. G. D'Oliveira, S. E. Rouayheb, and D. Karpuk, "GASP codes for secure distributed matrix multiplication," *IEEE Trans. Inf. Theory*, vol. 66, no. 7, pp. 4038–4050 (2020).

[2] R. G. D'Oliveira, S. E. Rouayheb, D. Heinlein, and D. Karpuk, "Degree tables for secure distributed matrix multiplication," *IEEE J. Sel. Areas Inf. Theory*, vol. 2, no. 3, pp. 907–918 (2021).

[3] H. H. López, G. L. Matthews, and D. Valvo, "Secure MatDot codes: A secure, distributed matrix multiplication scheme," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Mumbai, India, pp. 149–154 (Nov. 2022).

[4] E. Byrne, O. W. Gnilke, and J. Kliewer, "Straggler-and adversary-tolerant secure distributed matrix multiplication using polynomial codes," *Entropy*, vol. 25, no. 2, p. 266 (2023).

[5] S. M. Stefanov, "Multi-objective optimization revisited," *Journal of Information and Optimization Sciences*, vol. 47, no. 3, pp. 833–848 (2026), doi: 10.47974/JIOS-1395.

[6] R. C. Poonia, T. Singh, C. K. Rajashri, D. Mondal, and S. M. Patil, "Optimizing communication system security through comprehensive testing and intelligent network configurations," *Journal of Information and Optimization Sciences*, vol. 47, no. 5-A, pp. 1709–1717 (2026), doi: 10.47974/JIOS-2261.

[7] R. Dathathri, B. Kostova, O. Saarikivi, W. Dai, K. Laine, and M. Musuvathi, "EVA: An encrypted vector arithmetic language and compiler for efficient homomorphic computation," in *Proc. 41st ACM SIGPLAN Conf. Programming Language Design and Implementation (PLDI)*, pp. 546–561 (2020).

[8] Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai, "Zero-knowledge from secure multiparty computation," in *Proc. 39th Annu. ACM Symp. Theory of Computing (STOC)*, New York, USA, pp. 21–30 (2007).

[9] V. G. Shrimant, R. Halam, G. Tiwari, S. Chilukuri, R. Ramesh, and A. Singla, "Developing secure communication protocols using elliptic curve cryptography for IoT devices," *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 29, no. 2-A, pp. 533–541 (2026), doi: 10.47974/JDMSC-2493.

[10] D. Boneh, B. Lynn, and H. Shacham, "Privacy-enhancing cryptographic techniques in modern applications," *ACM Comput. Surveys*, vol. 54, no. 2, pp. 1–35 (2021).

Received April, 2026