International Journal of Safety and Security Engineering
Vol. 16, No. 2, February, 2026, pp. 455-460
Journal homepage: http://iieta.org/journals/ijsse

ECDHSS: A Two-Party Key Generation Scheme Combining Elliptic Curve Diffie–Hellman and Shamir's Secret Sharing

Noor Muneam Abbas, Asmaa Rashid Salman, Hala Bahjat Abdulwahab, Nidaa Flaih Hassan, Mustafa Tareq Abd*

College of Computer Science, University of Technology, Baghdad 10066, Iraq

Corresponding Author Email: mustafa.t.abd@uotechnology.edu.iq

https://doi.org/10.18280/ijsse.160219

Received: 8 November 2025
Revised: 27 January 2026
Accepted: 16 February 2026
Available online: 28 February 2026

Keywords:
Elliptic Curve Diffie–Hellman, Shamir's Secret Sharing, key generation, key agreement, shared-secret expansion, statistical randomness evaluation

ABSTRACT

The security of cryptographic systems is fundamentally reliant on the establishment and augmentation of secret material. While Elliptic Curve Diffie–Hellman (ECDH) offers an effective method for establishing a shared secret across an unsecured channel, it often produces a single shared value that requires additional processing for key creation. Shamir's Secret Sharing (SSS) provides a threshold-based algebraic framework, often necessitating the transfer of shares or polynomial parameters. The current research introduces ECDHSS, a two-party key generation Scheme that integrates ECDH and SSS without the need to exchange SSS coefficients or shares. The two coordinates of the ECDH shared point are assigned to the coefficients of an SSS polynomial, and the SSS secret term is computed from these values modulo the field prime. Consequently, both parties autonomously formulate the same polynomial and produce congruent key sequences locally, without the need to send sensitive secret-sharing information. The strategy is therefore formulated to minimize exposure during key setup with respect to predictable consensus amongst the two parties. The produced keys were assessed by Shannon entropy, chi-square testing, Hamming distance, and the NIST SP 800-22 statistical examination suite, and were additionally compared to keys obtained from an ECDH+HKDF baseline. The findings indicate increased entropy, equitable bit distributions, significant inter-key variation, and NIST pass rates that are equivalent to the baseline method having NIST pass rate of 97.7% compared to 97.9% achieved by ECDH+HKDF baseline. These results demonstrate that ECDHSS may provide robust statistical cryptographic keys while minimizing the need to transmit sensitive key-generation information.

1. INTRODUCTION

The security of modern cryptographic systems depends mainly on how keys are generated, distributed, and protected throughout their lifecycle. Weaknesses in any of these processes can weaken strong encryption algorithms. Insufficient randomness level in the generated keys may lead to predictable patterns that can be exploited by attackers [1]. Traditional approaches often rely on transmitting keys, coefficients, or shares across communication links, which introduces risks of exposure and interception [2].

Scalability problems will arise because large-scale systems need a lot of keys and distribution operations. In order to overcome these problems, techniques that provide robust randomization and minimize the amount of sensitive keys or generating data to be transmitted between the parties are needed [3].

Because of its effectiveness and security, ECDH is one of the most popular and favored techniques for establishing a shared secret over unsecure channels. But using ECDH in a traditional way results in a single shared value that needs to be further managed and protected.

SSS provides a different concept, SSS splits a secret into multiple parts and distributes it to multiple parties, allowing a subset of shares to be used to reconstruct the original value. This threshold property offers resilience against insider compromise but still typically involves the exchange of shares or coefficients, which reintroduces exposure risks [4].

In this paper, a hybrid scheme is proposed, in this scheme an SSS polynomial is built based on the shared secret that was generated by ECDH between two parties, the generated shared secret from the ECDH process is an (X, Y) coordinates, these two values are used as the coefficients of a secret-sharing polynomial in addition to the required secret value to complete the equation. In this scheme, both parties independently compute the same polynomial without exchanging the coefficients or the secret, which acts as a seed to the generation process. The polynomial then serves as a synchronized key generator equation, from which each party can derive consistent cryptographic keys as needed.

The proposed method addresses several challenges in cryptography:

1- Key generation strength: secured by the proven hardness of elliptic curve arithmetic and the unpredictability

of ECDH outputs.

2- Secure distribution without transmission: achieved by embedding the ECDH output into the structure of a secret-sharing polynomial that both parties can compute locally.

3- Scalability and resilience: enabled by the properties of secret sharing, which allow a high volume of generations.

Through this design, the scheme provides a practical and secure mechanism for generating keys while reducing reliance on communication channels for sensitive exchanges.

ECDH and SSS have been used before by researchers, but in different manners; for example, Jiang et al. [5] integrate the Chinese remainder theorem into the ECDH to construct a lightweight key agreement protocol for smart home systems. In their method, they used a hash function to identify the sensor nodes instead of mutual authentication to serve as the cost-reduced authentication phase, followed by the Chinese remainder theorem to enhance the security of the original form of ECDH key agreement. Performance analysis and experiments performed on their protocol showed that the protocol achieves high security with low communication and computation costs, to be used with IOT devices.

Hall et al. [6] presented a mechanism to create a pair of keys (private/public pair) by nesting SSS scheme, generating a decentralized key generator for applications such as message decryption, identity validation, and agreements.

Yang et al. [7] proposed a dynamic contributory Group Key Agreement protocol by using Short Signature and ECDH. Their protocol supports dynamic group membership, allowing users to join or leave without restarting the entire key agreement process.

For the problem of key generation, key distribution, and key management based on secret sharing, Zhang et al. [8] addressed the challenge of securely distributing keys to multiple parties using Quantum secret sharing, the authors propose an advanced Device-Independent Quantum Secret Sharing protocol that can combine three elements: the random key generation basis, noise preprocessing, and post-selection strategies. Their method aims to provide the highest level of security by resisting attacks from imperfect devices depending on quantum communication.

NIST randomness tests, Entropy, Chi-square, and Hamming distance were used to evaluate the generated keys in term of randomness and attack resistance, and the evaluation results showed that the statistical properties of the generated keys were strong confirming the robustness of the approach against common cryptanalytic attacks.

The proposed scheme is primarily designed as a two-party key generation scheme, where both parties independently derive identical key sequences without exchanging secret parameters. Extensions to multi-party environments are considered as future work.

The remainder of this paper presents the design and implementation of the proposed scheme, followed by an evaluation of its performance and security.

2. MATERIALS AND METHODS

2.1 Elliptic Curve Diffie–Hellman

ECDH is a key agreement protocol that allows two parties, each having an elliptic-curve public–private key pair, to establish a shared secret over an insecure channel. This shared secret may be directly used as a key or to derive another key. The key, or the derived key, can then be used to encrypt subsequent communications using a symmetric-key cipher [9].

Diffie–Hellman (DH) is one of the first public key protocols invented by Whitfield Diffie and Martin Hellman, DH is mathematical method that generates a shared encryption key between two parties over a public channel [10].

DH key exchange protocols are accomplished by the following routine:

Alice and Bob publicly agree on using a large prime number (p) and a base value (g).

Alice and Bob separately choose a secret value (a the secret value of Alice, and b the secret value of Bob) and exchange the result of the following equation with each other ($A= ga \bmod p$, $B=gb \bmod p$).

Alice computes s ($s=Ba \bmod p$), and Bob also compute s ($s= Ab \bmod p$), where s is considered the shared key.

Elliptic Curves (EC) represent a fundamental mathematical construct with major impact on modern cryptographic systems. These curves are defined over finite fields using the following equation ($y^2 \equiv x^3 + ax + b \; mod \; p$) where a, b are constants defining the curve and p is a prime number that defines the finite field, elliptic curves have unique algebraic properties that make them particularly suitable for secure key exchange protocols. The geometric interpretation of elliptic curves allows for the definition of a specialized addition operation, where combining two points on the curve produces another point belonging to that curve. This operation, combined with the existence of an identity element known as the point at infinity, forms the basis of an Abelian group structure that underlies their cryptographic applications [11].

The cryptographic strength of elliptic curves came from the complexity of mathematical problems defined over the structure of the curve. The elliptic curve discrete logarithm problem presents difficult computational challenges that make the key exchange mechanisms secure enough, and this is why protocols like the ECDH key agreement are secure. ECDH leverages scalar multiplication operations on the curve, where private keys are used to generate corresponding public keys through repeated point addition, and shared secrets are derived through carefully constructed combinations of these elements.

2.2 Shamir's Secret Sharing

Secret sharing is a cryptographic technique that takes a secret and breaks it up into multiple shares, and distributes these shares among multiple parties or locations. The secret can only be reconstructed when enough shares are combined in a mathematical process [12-14].

Sensitive data are often protected, and their security is improved by employing this method in combination with other methods. There are many useful applications of secret sharing, such as Key management, Data recovery, Password recovery, Access control, and Secure transactions [14]. Figure 1 shows the schema of secret sharing.

Secret Sharing is done over finite field arithmetic, particularly over a prime field GF(p) or Galois fields GF(2^m) to keep the shares within the field range. Through calculations in GF(p), SSS guarantees that all operations are feasible and consistent with the desired mathematical range.

SSS over GF(2^m) can be used to enable binary field operations, making them appropriate for digital systems that favor binary data representation. Utilizing the characteristics of Galois fields, SSS can be optimized for effective operation with digital data, providing adaptability in security

applications across various hardware platforms.

To generate shares using SSS, the following requirements are needed:

a. The secret (S).
b. The number of shares to generate (n).
c. Threshold, the minimum number of shares to reconstruct the secret (k) where $k < n$.
d. A large prime number GF(p).
e. Random coefficients (a_i) where i equal to k-1.

Shares are generated using the following equation:

$$f(x) = S + a_1 x + a_2 x^2 + \cdots + + a_{k-1} x^{k-1} mod\ GF(p)$$

where, $f(0) = S$ and each share consist of (xi, $f(xi)$).

To reconstruct the secret, the following information needs to be prepared:

a. At least k Shares.
b. A large prime number GF(p) that has been used in secret generating process.

Secret is reconstructed using the following equation:

$$S = f(0) = \sum_{i=0}^{k-1} y_i l_i(x)$$

where, l_i is found using the following equation:

$$l_i = \frac{x - x_0}{x_i - x_0} \times \ldots \times \frac{x - x_{k-1}}{x_i - x_{k-1}}$$

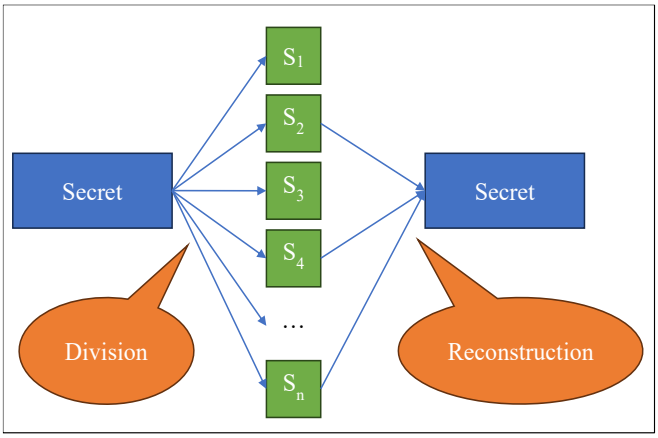

Figure 1. Secret sharing schema [15]

3. PROPOSED SCHEME

The proposed scheme starts by using ECDH protocol between parties A and B, for this protocol to work, A and B must agree on the following:

a. a publicly known and strong Elliptic Curve (e.g. secp256k1, secp256r1, secp384r1).
b. a generator point on the curve (G).
c. The order of the generator point (n).

The following example demonstrates the generation of the shared secret between A and B using ECDH over secp256r1:

1. $p\ (prime) = 2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$

p represents the large prime number that defines the finite field.

2. Using Elliptic Curve equation

$$y^2 \equiv x^3 + ax + b \bmod p$$

let a=-3 and b=0x5AC635D8AA3A93E7 B3EBBD55769886BC651D06B0CC53B0F63BC E3C3E27D2604B.

3. Using the following generator Point G:
X_G=0x6B17D1F2E12C4247F8BCE6E563A440 F277037D812DEB33A0F4A13945D898C296
Y_G=0x4FE342E2FE1A7F9B8EE7EB4A7C0F9E 162CBCE33576B315ECECBB6406837BF51F5.
4. n (Number of points in the curve): 0xFFFFFFFF00000000FFFFFFFFFFFFFFFFBC E6FAADA7179E84F3B9CAC2FC632551.
5. Each party randomly selects a private key d, which is a large integer less than n. let:
d_A= 0xA5B9CE14C3A89E4F
d_B= 0x3FAD239B47D2E3B5
6. Each party computes their public key by multiplying their private key with the generator point G:

- X_A=0x0C36F5B8A3D6F7C7E5F9D2C1A0B4E6 C2D4A1F9B8C7E6D5A4B3C2D1E0F9A8B6E
- Y_A=0x345A9F8C7E6D5C4B3A29181716151413 1211100F0E0D0C0B0A09080706050403
- X_B=0x6D4F8C2A9B3E7D5C1A2F3E4D5C6B7A 8F9E1D2C3B4A5968776655443322110 0A8
- Y_B=0x5C3A1F2E4D6B7C8A9E0F1D2C3B4A59 687766554433221100FFEEDDCCBBAA9985

7. Each party computes the same shared secret by multiplying their private key with the other party's public key.

X_S=0x8243F1C8A7B6D5E4F3A2918171615141312 11100F 0E0D0C0B0A09080706050403

Y_S=0xBC91E2D3C4B5A6978877665544332211 00FFEEDD CCBBAA998877665544332211

After these steps both parties will have the same shared secret (XS, YS).

This shared secret will be used as SSS coefficients (a and b) as follows:

$$f(x) = S + X_S x + Y_S x^2\ mod\ p$$

S in SSS polynomial is computed using the following equation:

$$S = X_S + Y_S \bmod P$$

S in the above example: S=0xE0D0FF66402304DEC2A07E6C5A4836241310FEE4 DAC8B6A492806E5C4A38261C, and P is the same prime used in ECDH.

The coordinates of the shared elliptic curve point (X_S, Y_S) are mapped directly to the polynomial coefficients within the same finite field GF(p), ensuring consistency between ECDH operations and SSS construction.

By solving the equation of SSS for consecutive generations, Table 1 gives a sample of the generated values:

Table 1. 20-generated values using the given setup

x	Generated Value
1	0xC1A1FECD804609BC8540FCD8B4906C482621FDC8B5916D492500DCB894704C39
2	0x1B96C3DE49D45BC758D047EF973EE68E3B32DA6629D17920C87017BF670EB67A
3	0xEEAF4E969CCDFB013D4E5FB10253A4F6524394BF3788DA2B7CCE1F70C21364DD
4	0x3AEB9EF97932E76732BB441CF5CEA7806B542CD0DEB79069421AF3CCA57E5765
5	0x004BB504DF0320FB3916F53371AFEE2C8664A29D1F5D9BDA185694D3114F8E10
6	0x3ECF90B8CE3EA7BD506172F475F778FAA374F623F97AFC7DFF810284058708DE
7	0xF677321546E57BAD789ABD6002A547EAC28527656D0FB254F79A3CDF8224C7CF
8	0x2742991C48F79CC9B1C2D47617B95AFCE395365F7A1BBD5F00A243E58728CAE5
9	0xD131C5CAD4750B14FBD9B836B533B23106A52315209F1D9C1A9917961493121D
10	0xF444B822E95DC68D56DF68A1DB144D872BB4ED846099D30C457EB7F12A639D79
11	0x907B702487B1CF32C2D3E5B7895B2CFF52C495AD3A0BDDAF815324F6C89A6CF9
12	0xA5D5EDCEAF7125063FB72F77C00850997BD41B90ACF53D85CE165EA6EF37809C
13	0x34543122609BC806CD8945E27F1BB855A6E37F2DB955F28F2BC865019E3AD863
14	0x3BF63A1E9B31B8356C4A28F7C6956433D3F2C0855F2DFCCB9A693806D5A4744D
15	0xBCBC08C35F32F5921BF9D8B7967554340301DF979E7D5C3B19F8D7B69574545A
16	0xB6A59D11AC9F801BDC985521EEBB88563410DC63774410DDAA774410DDAA788B
17	0x29B2F709837757D2AE259E36CF68009A671FB6E8E9821AB34BE47D15AE46E0E0
18	0x15E416A9E3BA7CB790A1B3F6387ABD009C2E6F28F53779BBFE4082C507498D58
19	0x7B38FBF2CD68EECA840C966029F3BD88D33D05239A642DF7C18B551EE8B27DF3
20	0x59B1A6E54082AE0A88664574A3D302330C4B78D7D908376695C4F4235281B2B2

These represent a subset of the generated 256-bit keys. These keys can be directly used as encryption keys and for the second party to decrypt the messages they need only to calculate the same SSS equation and generate the same key.

The system assumes two honest-but-curious parties that follow the protocol correctly but may attempt to infer additional information. The communication channel is considered insecure, and the security goal is to prevent exposure of secret parameters during key generation. No trusted third party is required.

Figure 2 shows the flowchart of the proposed scheme starting from the agreement on the curve, generating key pairs, exchanging public keys, computing the shared secret (X, Y), followed by using these values as coefficients for the Shamir polynomial, and finally generating the key sequences locally.

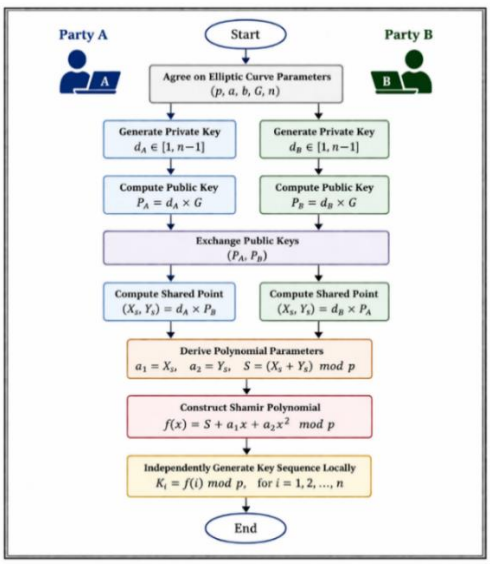

Figure 2. Proposed scheme flowchart

4. RESULTS AND DISCUSSION

To evaluate the strength of the shares in terms of key strength and randomness, the following tests have been used:

1. Entropy Test (Shannon Entropy) [16] Measures the unpredictability of the key by calculating how random it is, Shannon Entropy quantifies the uncertainty or randomness in a sequence of values. The higher the entropy, the less predictable the sequence is. Ideal randomness is achieved when entropy is equal to 1.0 (for a balanced binary string).
2. Key Uniformity (Chi-Square Test) [17] Verifies that the shares do not have biased distributions of 0s and 1 s, The Chi-square test compares the observed frequency distribution of bits with the expected distribution in a uniform random sequence (50% 0 s and 50% 1 s). For the Chi-square test, the critical value is 6.63, meaning that the sequence passes the test if Chi-Square value is less than 6.63.
3. Collision Resistance (Hamming Distance) [18] Measures how different two keys are when one bit is altered, A key with high collision resistance should have large differences when slightly changed. The Hamming distance between two keys should be close to 50% for an ideal cryptographic key.
4. NIST Statistical Test Suite (SP800-22) [19] is a collection of statistical tests designed to evaluate the randomness of binary sequences, the tests analyze various patterns and deviations from true randomness, these tests include commonly applied NIST SP800-22 tests such as Frequency (Monobit), Block Frequency, Runs, Longest Run of Ones, Discrete Fourier Transform, Template Matching, Serial, Approximate Entropy, and Cumulative Sums tests. A test is considered successful (Pass) if the p-value ≥ 0.01 (for a significance level of $\alpha = 1\%$). Some tests have additional criteria (e.g., proportion of sequence passing, uniformity of p-values). Table 2 gives the tests results for the generated 20 keys, while Table 3 gives the NIST STS test results over 100 generated values, the same ECDH shared secret was used with HMAC-based Extract-and-Expand Key Derivation Function (HKDF) [20] to generate 100 keys to compare with the proposed scheme generated keys.

Table 2. Evaluation results of the generated keys

Key	Shannon Entropy	Chi-Square Test	Hamming Distance
1	0.9872	4.5156	133
2	0.9972	1.0000	127
3	0.9978	0.7656	144
4	0.9964	1.2656	128
5	0.9989	0.3906	127
6	0.9937	2.2500	124
7	0.9984	0.5625	109
8	0.9996	0.1406	124
9	0.9964	1.2656	131
10	0.9998	0.0625	119
11	0.9989	0.3906	126
12	0.9978	0.7656	140
13	0.9996	0.1406	128
14	0.9996	0.1406	108
15	0.9964	1.2656	129
16	0.9984	0.5625	131
17	0.99996	0.0156	127
18	0.9998	0.0625	136
19	0.9984	0.5625	125
20	0.9901	3.5156	122

The evaluation was conducted on sequences generated using the proposed scheme, where 100 consecutive keys were produced and concatenated to form input sequences suitable for statistical testing. The NIST SP800-22 tests were applied using standard configurations, and all p-values were obtained based on the recommended significance level ($\alpha = 0.01$).

The generated keys present strong statistical randomness properties, indicating their suitability as statistically strong cryptographic keys. The Shannon Entropy values in Table 2 for the 20 samples used are close to 1, where 1 is the ideal balance between binary states. The Chi-square test results remain below the critical threshold (6.63), confirming the absence of statistical bias in the distribution of the bits.

The Hamming distance between generated keys is approximately 50% of the key length, indicating a high level of independence between keys and strong resistance to correlation-based attacks. This ensures that when applying a small variation in the input, this will lead to an output that is differ by a high variation, which is desirable in secure key generation schemes.

NIST SP800-22 statistical tests results in Table 3, which were done over 100 keys, also validate the randomness of the generated sequences. Most tests achieved high test-pass rates with p-values exceeding the required threshold (0.01), indicating no significant deviation from ideal randomness. Although slight variations appear in certain tests, such as the Non-overlapping Template test, the overall performance remains within acceptable limits and does not indicate weaknesses in the structure of the proposed scheme.

It should be noted that the value reported as 1.0000 in the Approximate Entropy test represents a rounded p-value rather than an exact value.

Table 3. NIST Statistical Test for 100 consecutive generated keys

Test	Pass Rate	P-Value	Pass Rate for HKDF	P-Value for HKDF
Frequency (Monobit)	100/100	0.090936	99/100	0.021999
Block Frequency	100/100	0.058984	99/100	0.003712
Runs Test	99/100	0.678686	99/100	0.108791
Longest Run of Ones	98/100	0.657933	100/100	0.137282
Discrete Fourier Transform	99/100	(0.008333 – 0.818546)	99/100	(0.001319 – 0.818546)
Non-overlapping Template	88/100	(0.000000 – 0.999983)	87-99/100	(0.000000 – 0.999983)
Serial Test	95/100	(0.498961, 0.498531)	96/100	(0.498961, 0.498531)
Approximate Entropy	100/100	1.0000*	100/100	1.0000*
Cumulative Sums (Forward)	98/100	0.145326	99/100	0.419021
Cumulative Sums (Reverse)	100/100	0.401199	99/100	0.066882

The results presented in Tables 2-3 confirm that the proposed scheme produces statistically strong and unpredictable keys. The scheme enhances security by eliminating the need to transmit secret shares or polynomial coefficients, thereby reducing exposure to interception or leakage. This makes the proposed scheme suitable for secure key generation and distribution in environments where minimizing communication of sensitive data is essential.

The proposed scheme achieved a NIST pass rate of 97.7%, while the standard ECDH + HKDF baseline achieved 97.9%, showing comparable statistical randomness characteristics.

Compared to conventional ECDH-derived keys which typically produce a single shared value requiring additional processing for key expansion, the proposed scheme directly generates multiple statistically strong keys from a synchronized polynomial structure.

5. CONCLUSIONS

This paper presented a two-party key generation Scheme that integrates Elliptic Curve Diffie–Hellman (ECDH) and Shamir's Secret Sharing (SSS) without the need to exchange SSS coefficients or shares. In this proposed scheme, the two coordinates of the ECDH shared point are assigned to the coefficients of an SSS polynomial, and the SSS secret term is computed from these values modulo the field prime, enabling both parties to independently generate identical key sequences without exchanging sensitive parameters.

The experimental evaluation, including entropy, Chi-square, Hamming distance, and NIST statistical tests, shows that the randomness level of the generated keys is high. These results confirm that the proposed scheme produces secure and unpredictable keys suitable for security applications.

The keys generated using the proposed scheme were compared with the keys generated using ECDH+HKDF based on the same ECDH-generated shared secret. NIST statistical tests results were almost identical, showing high competency with existing methods.

In addition to its statistical strength, the scheme improves security by reducing reliance on communication channels for key generation or distribution, minimizing the risk of interception and leakage. The scheme also supports scalability, as it can generate keys of different sizes and volumes

depending on system or application requirements.

Future research may investigate the adaptation of this framework to multi-party environments and assess its efficacy in practical applications, including secure storage systems, financial infrastructures, and distributed platforms. In this scheme, the generative parameters of SSS can be transmitted using a secure multiparty computation to add an extra level of trust and verification between the parties. The scheme can also work with a lightweight encryption method [21] to make random coefficients.

STATEMENT ON THE USE OF GENERATIVE AI

AI tools were used for language enhancement, grammar correction, and improving clarity of text. No AI tools were used to generate research content, results, data, or scientific contributions.

ACKNOWLEDGMENT

This research received no specific grant from any funding agency, commercial or not-for-profit sectors.

REFERENCES

[1] Zaki, R.M., Mahdi, Z.S., Abdulmunim, M.E. (2024). Survey: A study on image encryption using DNA in bioinformatics. Journal of Soft Computing and Computer Applications, 1(2): 5. https://doi.org/10.70403/3008-1084.1013

[2] Shareef, S.M., Hassan, R.F. (2025). Enhancing cybersecurity based on blockchain technology: A systematic review. Journal of Soft Computing and Computer Applications, 2(1): 2. https://doi.org/10.70403/3008-1084.1015

[3] Xu, W., Zhang, J., Huang, S., Luo, C., Li, W. (2021). Key generation for Internet of Things: A contemporary survey. ACM Computing Surveys (CSUR), 54(1): 1-37. https://doi.org/10.1145/3429740

[4] Hwang, J., Maji, H.K., Nguyen, H.H., Ye, X. (2025). Leakage-resilience of Shamir's secret sharing: Identifying secure evaluation places. In 6th Conference on Information-Theoretic Cryptography (ITC 2025), pp. 1-20. https://doi.org/10.4230/LIPIcs.ITC.2025.3

[5] Jiang, Y., Shen, Y., Zhu, Q. (2020). A lightweight key agreement protocol based on Chinese remainder theorem and ECDH for smart homes. Sensors, 20(5): 1357. https://doi.org/10.3390/s20051357

[6] Hall, J.L., Hertzog, Y., Loewy, M., Skerritt, M.P., Valladolid, D., Verma, G. (2023). Manifesting Unobtainable Secrets: Threshold Elliptic Curve Key Generation using Nested Shamir Secret Sharing. arXiv preprint arXiv:2309.00915. https://doi.org/10.48550/arXiv.2309.00915

[7] Yang, Z., Wang, Z., Qiu, F., Li, F. (2023). A group key agreement protocol based on ECDH and short signature. Journal of Information Security and Applications, 72: 103388. https://doi.org/10.1016/j.jisa.2022.103388

[8] Zhang, Q., Ying, J.W., Wang, Z.J., Zhong, W., et al. (2025). Device-independent quantum secret sharing with advanced random key generation basis. Physical Review A, 111(1): 012603. https://doi.org/10.1103/PhysRevA.111.012603

[9] Koppl, M., Siroshtan, D., Orgon, M., Pocarovsky, S., Bohacik, A., Kuchar, K., Holasova, E. (2021). Performance Comparison of ECDH and ECDSA. In 2021 2nd International Conference on Electronics, Communications and Information Technology (CECIT), pp. 825-829. https://doi.org/10.1109/CECIT53797.2021.00149

[10] Just, M., Adams, C. (2025). Diffie–Hellman key agreement. In Encyclopedia of Cryptography, Security and Privacy, pp. 656-658. https://doi.org/10.1007/978-3-030-71522-9_75

[11] Ullah, S., Zheng, J., Din, N., Hussain, M.T., Ullah, F., Yousaf, M. (2023). Elliptic Curve Cryptography; Applications, challenges, recent advances, and future trends: A comprehensive survey. Computer Science Review, 47: 100530. https://doi.org/10.1016/j.cosrev.2022.100530

[12] Sklavos, N. (2024). Book Review: Stallings, W. Cryptography and Network Security: Principles and Practice. Global Edition, 8th ed., Pearson, 2024.

[13] Shamsoshoara, A. (2019). Overview of blakley's secret sharing scheme. arXiv preprint arXiv:1901.02802. https://doi.org/10.48550/arXiv.1901.02802

[14] Hashim, A.T., Radeef, Z.M. (2017). Multiple image secret sharing based on linear system. Indian Journal of Science and Technology, 10(33): 1-17. https://doi.org/10.17485/ijst/2017/v10i33/113085

[15] Chanu, O.B., Neelima, A. (2019). A survey paper on secret image sharing schemes. International Journal of Multimedia Information Retrieval, 8(4): 195-215. https://doi.org/10.1007/s13735-018-0161-3

[16] Saraiva, P. (2023). On Shannon entropy and its applications. Kuwait Journal of Science, 50(3): 194-199. https://doi.org/10.1016/j.kjs.2023.05.004

[17] Wuensch, K.L. (2025). Chi-square tests. In International Encyclopedia of Statistical Science, pp. 466-468. https://doi.org/10.1007/978-3-662-69359-9_110

[18] Cherckesova, L.V., Safaryan, O.A., Lyashenko, N.G., Korochentsev, D.A. (2022). Developing a new collision-resistant hashing algorithm. Mathematics, 10(15): 2769. https://doi.org/10.3390/math10152769

[19] Park, H.J., Ngo, C.T., Lee, M.H., Hong, J.P. (2026). Parameter Optimization Method for Improving the Reliability of the NIST SP 800-22 Test Suite in TRNG Verification. https://doi.org/10.1109/ACCESS.2026.3667853

[20] Zhiqiang, H., Rauf, A., Nazir, A., Tchier, F., Aslam, A., Tola, K.A. (2025). Design and analysis of a secure image encryption algorithm using proposed non-linear RN chaotic system and ECC/HKDF key derivation with authentication support. Scientific Reports, 15(1): 39951. https://doi.org/10.1038/s41598-025-23592-w

[21] Khudhair, A.A.T., Maolood, A.T., Gbashi, E.K. (2024). A novel approach to generate dynamic s-box for lightweight cryptography based on the 3D hindmarsh rose model. Journal of Soft Computing and Computer Applications, 1(1): 4. https://doi.org/10.70403/3008-1084.1003