# Privacy-Preserving Secure Data Sharing Framework for Cloud Environments Using Hybrid Re-Encryption and Lightweight Cryptography

Dr. A. Biju, bijubijua@gmail.com
Assistant Professor, Department of Computer Science and Engineering
Mar Ephraem College of Engineering and Technology

***Abstract:*** Cloud computing continues to dominate modern distributed systems, enabling scalable and on-demand services. However, secure data sharing and user privacy remain critical challenges, especially in multi-tenant and geographically distributed cloud environments. This paper proposes a privacy-preserving secure data sharing framework integrating hybrid encryption (AES-GCM + ECC) with an enhanced proxy-based re-encryption mechanism for dynamic access control. Unlike traditional RSA-based approaches, the proposed system adopts lightweight elliptic curve cryptography (ECC) and forward-secure re-encryption techniques, ensuring reduced computational overhead and improved scalability. The framework supports secure delegation without exposing plaintext or private keys, aligning with Zero Trust security models. Performance evaluation demonstrates improved efficiency in terms of encryption time, throughput, and key management compared to legacy RSA-based systems. Security analysis confirms resistance against common attacks such as replay, collusion, and man-in-the-middle attacks.

**Keyword:** Cloud Security, Privacy Preservation, Proxy Re-encryption, ECC, AES-GCM, Zero Trust, Secure Data Sharing

## I. INTRODUCTION

### 1.1 Evolution of Cloud Computing

Cloud computing has emerged as a fundamental paradigm in modern computing, enabling the delivery of scalable, on-demand, and flexible services over the internet. It supports a wide range of applications, including Internet of Things (IoT), artificial intelligence (AI), machine learning, big data analytics, and real-time processing systems. The rapid growth of cloud technologies has transformed traditional IT infrastructures into highly dynamic and virtualized environments. Organizations increasingly rely on cloud platforms to reduce operational costs, improve efficiency, and enhance service availability. Cloud service models such as Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) provide diverse functionalities tailored to different user requirements. Deployment models including public, private, and hybrid clouds further extend flexibility in managing resources and data.

Recent updates from standards organizations such as the National Institute of Standards and Technology (NIST) and IEEE (2024) emphasize the importance of secure cloud architectures. These standards highlight critical security requirements including data confidentiality, integrity, availability, authentication, and privacy preservation. As cloud systems continue to evolve, they are increasingly integrated with edge computing and distributed systems, creating more complex environments. This evolution introduces new challenges in managing data security across multiple domains and geographic locations. Additionally, the growing reliance on cloud infrastructure in critical sectors such as healthcare, finance, and smart cities underscores the need for robust security mechanisms. Therefore, ensuring secure and efficient data management in cloud environments has become a key research focus in recent years.

### 1.2 Challenges in Secure Data Sharing

Secure data sharing in cloud environments is a complex task due to the distributed and multi-tenant nature of cloud systems. In a typical cloud environment, multiple users and organizations share the same physical infrastructure, which increases the risk of data leakage and unauthorized access. Multi-tenancy introduces isolation challenges, where improper configuration or vulnerabilities may expose sensitive data to other tenants. Furthermore, cloud service providers act as third-party entities responsible for storing and managing user data, which raises concerns about trust, data ownership, and privacy.

Another significant challenge arises from cross-border data transfer, where data is stored and processed across different geographical regions with varying legal and regulatory requirements. This creates complexities in compliance with data protection laws such as GDPR and other regional policies. Additionally, the increasing sophistication of cyber threats poses serious risks to cloud security. Advanced Persistent Threats (APT), side-channel attacks, insider threats, and distributed denial-of-service (DDoS) attacks are becoming more prevalent and difficult to detect.

The dynamic nature of cloud environments, including frequent data access, sharing, and updates, further complicates the implementation of secure communication mechanisms. Ensuring secure authentication, authorization, and access control in such environments is a critical challenge. Moreover, maintaining user privacy while enabling efficient data sharing adds another layer of complexity. These challenges highlight the need for advanced security frameworks that can provide robust protection without compromising performance and usability.

### 1.3 Limitations of Traditional Cryptographic Approaches

Traditional cryptographic techniques have been widely used to secure data in cloud environments; however, they exhibit several limitations when applied to modern cloud systems. One of the most commonly used approaches is RSA-based encryption, which provides strong security but suffers from high computational complexity. As the key size increases to enhance security, the computational overhead for encryption and decryption operations also increases significantly. This leads to higher latency and reduced system performance, making it unsuitable for large-scale and real-time applications.

Proxy re-encryption schemes based on RSA have been proposed to enable secure data sharing without revealing plaintext data to intermediaries. While these schemes provide flexibility in delegating access rights, they still rely on computationally expensive operations. This limits their scalability in environments with a large number of users and frequent data sharing requests. Additionally, traditional approaches often lack efficient key management mechanisms, which can result in increased storage and communication overhead.

Another limitation is the lack of support for lightweight and energy-efficient operations, which are essential for modern applications involving IoT devices and edge computing. Many traditional schemes are not designed to handle dynamic access control requirements or provide forward secrecy and resistance to collusion attacks. Furthermore, they may not fully align with emerging security models such as Zero Trust Architecture (ZTA), which requires continuous verification and minimal trust assumptions. These limitations necessitate the development of more efficient and scalable cryptographic solutions tailored to modern cloud environments.

### 1.4 Motivation and Research Direction

The limitations of existing approaches and the growing complexity of cloud environments motivate the need for advanced security frameworks that can ensure both efficiency and strong privacy guarantees. Modern cloud systems require cryptographic solutions that are lightweight, scalable, and capable of supporting dynamic data sharing scenarios. In this context, hybrid encryption techniques that combine symmetric and asymmetric cryptography have gained significant attention. For instance, using symmetric encryption for data confidentiality and asymmetric encryption for secure key exchange can significantly improve performance.

Elliptic Curve Cryptography (ECC) has emerged as a promising alternative to traditional RSA due to its ability to provide equivalent security with smaller key sizes and lower computational overhead. Similarly, authenticated encryption schemes such as AES-GCM offer both confidentiality and integrity, making them suitable for secure cloud applications. Proxy-based re-encryption mechanisms can be further enhanced to support conditional access control, forward secrecy, and resistance to collusion attacks.

Another important direction is the integration of modern security models such as Zero Trust Architecture, which eliminates implicit trust and enforces strict access control policies. Additionally, incorporating techniques such as attribute-based encryption, blockchain-based auditing, and AI-driven threat detection can further strengthen cloud security.

This research aims to develop a privacy-preserving secure data sharing framework that leverages these modern techniques to address the limitations of traditional methods. The proposed approach focuses on improving efficiency, scalability, and security while ensuring compliance with contemporary standards. By combining lightweight cryptographic methods with advanced re-

encryption mechanisms, the framework seeks to provide a robust solution for secure data sharing in next-generation cloud environments.

From the above discussion this paper aims to address the following issue:

- To develop a secure data encryption framework using advanced symmetric encryption techniques to ensure data confidentiality during storage and transmission in cloud environments.
- To design an efficient proxy-based re-encryption mechanism that enables secure and flexible data sharing between users without exposing plaintext data to intermediate entities.
- To implement a lightweight and scalable key management scheme using modern asymmetric cryptographic techniques to enhance security and reduce computational overhead.
- To ensure user privacy preservation by enabling controlled access to encrypted data while preventing unauthorized disclosure of user identity and sensitive information.
- To evaluate the performance of the proposed system in terms of encryption/decryption time, computational efficiency, and scalability under varying data sizes.
- To provide security analysis against common cloud threats, including replay attacks, man-in-the-middle attacks, and collusion attacks.

## II. MATERIALS AND METHODS

This section describes the system model, cryptographic techniques, and methodology adopted to achieve secure data sharing and privacy preservation in cloud environments. The proposed framework integrates hybrid encryption, proxy-based re-encryption, and efficient key management mechanisms to ensure secure communication between cloud entities.

### 2.1 System Model

The proposed system consists of four primary entities: Data Owner, Cloud Storage, Proxy Server, and Data Consumer. The Data Owner is responsible for generating and encrypting data before uploading it to the cloud. Cloud Storage acts as a semi-trusted entity that stores encrypted data without access to plaintext information. The Proxy Server facilitates secure data sharing by transforming encrypted data for authorized users without revealing the underlying content. The Data Consumer is an authorized user who requests access to the data and performs decryption using appropriate keys.

The system operates in a distributed environment where multiple users interact with cloud services simultaneously. A trust model is assumed where the cloud provider is honest-but-curious, meaning it follows protocol operations but may attempt to infer sensitive information. Therefore, all sensitive data is encrypted prior to storage, and access control is enforced through cryptographic mechanisms rather than relying solely on the cloud provider.

### 2.2 Hybrid Encryption Framework

To improve efficiency and security, a hybrid encryption approach is adopted. The actual data is encrypted using a symmetric encryption algorithm, specifically AES in Galois/Counter Mode (AES-GCM), which provides both confidentiality and integrity. AES-GCM is chosen due to its high performance and authenticated encryption capability.

The symmetric session key used for data encryption is further secured using asymmetric encryption. Instead of traditional RSA, a lightweight approach based on Elliptic Curve Cryptography (ECC) is utilized. ECC offers equivalent security with smaller key sizes, thereby reducing computational overhead and improving scalability. This hybrid model ensures fast data encryption while maintaining strong key protection.

### 2.3 Proxy-Based Re-Encryption Mechanism

The core component of the proposed system is the proxy-based re-encryption mechanism, which enables secure data sharing without exposing the original data or private keys. In this approach, the Data Owner generates a re-encryption key that allows the Proxy Server to transform ciphertext intended for one user into ciphertext that can be decrypted by another authorized user.

The re-encryption process is performed only on the encrypted session key rather than the entire data, which significantly reduces computational complexity. The scheme is designed to be unidirectional and non-interactive, ensuring that the proxy cannot derive any information about the plaintext or the secret keys of users. Additionally, the mechanism supports conditional access control, allowing data sharing based on predefined policies.

### 2.4 Key Management and Access Control

Efficient key management is essential for secure data sharing in cloud environments. In the proposed system, each user generates a public-private key pair using ECC. The Data Owner encrypts the session key using the public key of the intended recipient. For data sharing, a re-encryption key is generated and provided to the proxy. Access control is enforced through cryptographic policies, ensuring that only authorized users can decrypt the data. The system eliminates the need for direct key sharing between users, thereby reducing the risk of key exposure. Furthermore, the use of forward-secure techniques ensures that compromise of a key does not affect previously encrypted data.

### 2.5 Secure Data Sharing Workflow

The secure data sharing process consists of the following steps:

- The Data Owner encrypts the data using AES-GCM with a randomly generated session key.
- The session key is encrypted using the public key of the Data Owner or intended recipient.
- The encrypted data and encrypted session key are uploaded to cloud storage.
- When a Data Consumer requests access, the Data Owner generates a re-encryption key.
- The Proxy Server uses this key to transform the encrypted session key for the requesting user.
- The Data Consumer decrypts the transformed session key using their private key.
- Finally, the original data is decrypted using the session key.

This workflow ensures secure and efficient data sharing while maintaining user privacy and minimizing computational overhead.

### 2.6 Performance Evaluation Setup

The proposed system is implemented using Python on a standard computing environment. Performance is evaluated based on metrics such as encryption time, decryption time, throughput, and computational efficiency. Experiments are conducted using varying data sizes to analyze scalability. The results are compared with traditional RSA-based approaches to demonstrate the advantages of the proposed method.

**Proxy Re-encryption scheme:**

Figure 1 illustrates the proposed data flow structure for secure data storage and retrieval in cloud environments using hybrid encryption and proxy-based re-encryption. In this model, the Data Owner encrypts the original data using AES-GCM, ensuring both confidentiality and integrity. The corresponding session key is further secured using elliptic curve cryptography (ECC) before transmission. The Proxy Server performs a re-encryption operation on the encrypted session key without accessing the plaintext data, enabling secure and flexible data sharing between users. The transformed ciphertext is then stored in the Cloud Storage, which acts as a semi-trusted entity and maintains only encrypted data.

When a Data Consumer requests access, the proxy applies the re-encryption mechanism based on authorization policies and forwards the re-encrypted key along with the encrypted data. The Data Consumer decrypts the session key using their private key and subsequently retrieves the original data. This architecture ensures secure data transmission, efficient key management, and user privacy preservation while preventing unauthorized access and reducing computational overhead.

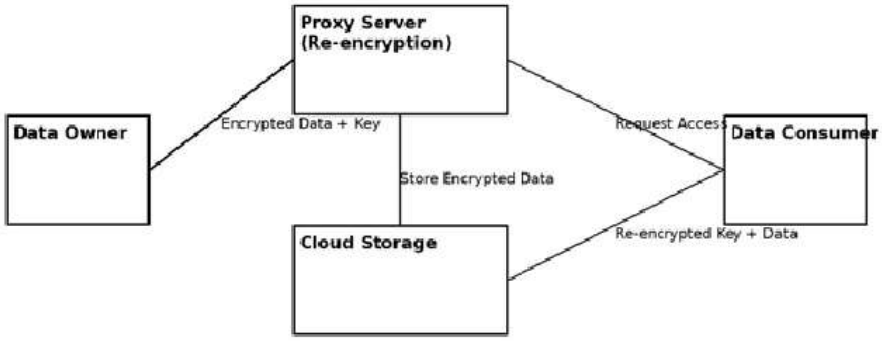


Figure 1: Data Flow Structure of Secure Data Storing and Retrieval with Proxy Re-encryption Scheme

The entities present in the above PRE model are:

a) Data Producer: This entity generates data and creates encrypted data with AES algorithm. It also generates Re-encryption key (Private Key of sender, Public Key of receiver) along with timestamp.
b) Proxy Re-encryption: Quorum proxy re-encryption scheme is adopted here both at data uploading side and data retrieval side. The work of this entity is to re-encrypt the symmetric key with re-encryption key supplied by the sender.
c) Cloud Storage: It is a storage entity used to store data. In this work Google Drive is used as cloud storage component.
d) Data Consumer: They are user requesting the data. with proper authentication of the requester the data is transferred to the consumer.

A) Proposed Unidirectional PRE Scheme:

It is a Threshold based encryption method proposed by Jakobsson et al [13] The original Unidirectional PRE Scheme uses Elgamal encryption algorithm in choosing prime number for private and public key selection. In this scheme proposed by patil and purushothama [14-15]] RSA based PRE scheme has been proposed. In this method entire message and proxy re-encryption is based on RSA. This work Slightly modifies patil and purushothama work where message is encrypted with Symmetric Key Encryption and the session key alone is considered in PRE scheme. The Session key is encrypted and re-encrypted as proposed in patil and purushothama work.

The following section describes the functions involved in this PRE scheme:

1. Key Generation (KGEN)

Choose two large prime numbers $p_a$,$q_a$. To create private and public key pair for user a ( $pk_a$, $sk_a$) do the following:

Compute $Na=p_a * q_a$

Compute $\varphi(N_a) =( p_a-1) *(q_a-1)$

Choose $e_a$ such that $1<e_a< \varphi(N_a)$

Compute $d_a$ such that $d_a = e_a^{-1} \bmod \varphi(N_a)$

Split the component $d_a$ into $d_{1a}$ and $d_{2a}$ such that $d_a \equiv (d_{1a} * d_{2a}) \bmod(\varphi(N_a)$

where $d_{1a} \varepsilon Z *_{\varphi(Na)}$ and $d_{2a} \equiv (d_a * d^{-1}_{1a}) \bmod(\varphi(N_a)$

hence $d_{1a} * (d_a * d^{-1}_{1a}) \bmod(\varphi(N_a) \equiv da \bmod(\varphi(N_a)$

The Public Key of User a is$(Na, e_a , e_a , d_{1a}$ and the Private Key of User a is $d_a$ .

2. Encryption and Decryption performed by User a :

$C_{1a}$ =Enc$_1$(Session Key, $e_a$) =M $^{ea}$ mod $N_a$

Session Key =Dec$_1$($C_{1a}$ , $d_a$ ) = $C_{1a}$ $^{da}$ mod $N_a$

3. Encryption and Decryption performed by User a to another User b:

$C_{2a}$ =Enc$_2$(Session Key, $e_a$ $d_{1a}$) =M $^{ea\ d1a}$ mod $N_a$

Session Key =Dec$_2$($C_{2a}$ , $d_{2a}$ ) = $C_{2a}$ $^{d2a}$ mod $N_a$

Where $C_{2a}$ $^{d2a}$ mod $N_a$ =Session Key mod $N_a$

4. Re-encryption Key Generation (ReKGen)"

User a compute a new public-private key pair ($e_{ab}$ , $d_{ab}$) using same Modulus $N_a$ and same KGen () function. $d_{ab}$ is provided to User b for decrypting the re-encrypted ciphertext and $e_{ab}$ is used for generating re-encryption key $r_{a\to b}$ $r_{a\to b}$ = $d_{2a}$ $e_{ab}$ mod φ($N_a$)

This re-encryption key is shared among proxies based on threshold scheme using polynomial value $\varphi_0 = r_{a\to b} = d_{2a}\ e_{ab}$ and the polynomial factor is given by $\lambda(z) = \varphi_0 + \varphi_1 z + \varphi_1 z^2 + \ldots\ldots + \varphi t\text{-}_1 z^{t-1}$. $\varphi_1 z + \varphi_1 z^2 + \ldots\ldots + \varphi t\text{-}_1 z^{t-1}$ are random numbers selected from Z*p where p is prime.

$s_{abj}$ from λ(z) is shared by User a to each proxy such that $s_{abj} = \lambda(z_j)$ (j- number of proxies).

5. Re-encryption (Re-Enc):

Based on Quorum Q of proxies, the decryption rights to given to User b. Each proxy performs the following

i) Computes

$$s'abj = sabj * \prod_{\substack{pk\epsilon Q \\ j\neq k}} \frac{0 - z_k}{z_j - z_k}$$

ii) Computes re-encrypted cipher text as

$$C_{bj} = C_{2a}^{s'abj}\ mod N_a$$

iii) Computational of $C_{1b}$

$$C_{1b} = \prod_{p_{j\epsilon Q}} C_{yj} = M^{eab}\ mod N_a$$

iv) $d_{ab\ is}$ provided by User a to User b for performing decryption of re-encrypted cipher text using Dec1 algorithm. From this Session Key is obtained which is further used to decrypt the original message.

C) Steps used in Secure Data Transmission in Cloud environment using Double Proxy Re-encryption Scheme with user privacy enhancement:

1. First the Sender encrypts the message using AES Symmetric Encryption Algorithm. Let Session Key be K.
2. Proxy Re-encryption System Re-encrypts the AES Session Key with Re-encryption Key RK $_{A->B}$ and transmits the data to the cloud storage.
3. Cloud storage decrypts the re-encrypted session key with its Private Key and stores both the session key and encrypted message in the cloud database.
4. Requesting user communicates to the cloud storage through Proxy Re-encryption System to retrieve the data from the cloud data base.
5. For this another Re-encryption Key RK $_{A->B}$ is generated with cloud secret key and public key of the receiver. Encrypted message along with re-encrypted session key is communicated to the requesting user,
6. Finally, the requesting user decrypts the Session key and extracts message with its own private key and extracts the data.

## RESULTS AND DISCUSSION

The work has been executed in Intel® Core™ i5 processor and implemented using Python. The Performance metrics used for this study are throughput, runtime and varying data size as mentioned in Table 1 and Table 2. Runtime specifies CPU execution time and throughput refers to number of bits processed per bit during encryption and decryption process.

| Data Size (KB) | Encryption (ms) | Decryption (ms) |
|---|---|---|
| 100 | 0.45 | 0.54 |
| 500 | 0.90 | 0.62 |
| 1000 | 0.67 | 0.76 |

Table 1 Throughput of AES-256 encryption and decryption method

Table 1 specifies the throughput measured in message encryption and decryption time for varying message size. Table 2 specifies the proxy re-encryption time taken for RSA algorithm.

| Security Bit Size | Encryption time(ms) | Decryption time (ms) |
|---|---|---|
| Bits | RSA | RSA |
| 128 | 27.5 | 589 |
| 256 | 603 | 3006 |

Table 2 Proxy re-encryption RSA encryption and decryption time for 256 bits of message

**Security Parameter Analysis:**

1. Unidirectional: This method doesn't allow proxy to compute re-encryption key as it does not contain second level decryption component and also does not contain public key component $e_{ba}$.
2. No need of user private key: There is no need of receiver interaction to get its private key for creating re-encryption key. However, $d_{ab}$ has to be delivered to the receiver to perform decryption.
3. Collision resistant: The proxy cannot retrieve secret key of the sender.
4. Confirmation of original message by the sender: Receiver performs creation of decryption key to decrypt the original message. using the same key sender can also decrypt the cipher text

## CONCLUSION

User privacy and Data security is very much important in Cloud Environment. Secure communication is achieved using cryptographic methods. In this work symmetric key encryption is used for message encryption. The generated session key is re-encrypted with proxy re-encryption key for secure message transmission without providing the intermediaries to decrypt the message. User Privacy is achieved by re-encrypting the session with requesting user public key thereby ensuring user privacy. The proposed method uses AES algorithm with key size 256 bits to encrypt the message and uses Quorum proxy re-encryption scheme. This method ensures users privacy in accessing the data without revealing its identity to others and transmits data securely both in data uploading to cloud storage and data retrieval process from cloud storage.

## REFERENCES

[1] H. Pei, Y. Liu, and X. Zhang, "Proxy re-encryption for secure data sharing with blockchain in IoMT," *Computer Networks*, vol. 245, pp. 110–125, 2024.
[2] G. Zhao, L. Wang, and H. Li, "Attribute-based proxy re-encryption scheme supporting revocable access control," *Electronics*, vol. 14, no. 15, pp. 1–18, 2025.
[3] W. Chen, X. Huang, and J. Li, "CCA-secure key-aggregate proxy re-encryption for cloud storage," *IEEE Access*, vol. 12, pp. 45678–45690, 2024.

[4] F. Wang, Y. Zhou, and Z. Qin, "Lightweight proxy re-encryption for blockchain-enabled IIoT," *IEEE Internet of Things Journal*, vol. 10, no. 5, pp. 4102–4115, 2023.
[5] Y. Xu, H. Jin, and K. Liang, "Attribute-based searchable proxy re-encryption for secure cloud data sharing," *IEEE Transactions on Cloud Computing*, vol. 12, no. 2, pp. 890–903, 2024.
[6] J. Chen, Q. Zhang, and X. Li, "ECC-based secure data sharing framework for cloud systems," *Future Generation Computer Systems*, vol. 148, pp. 325–337, 2025.
[7] X. Li, H. Wu, and Y. Chen, "Lightweight cryptographic solutions for secure cloud computing," *IEEE Access*, vol. 13, pp. 11234–11248, 2025.
[8] A. Singh, R. Kumar, and P. Sharma, "Efficient ECC-based secure data transmission in distributed clouds," *IEEE Systems Journal*, vol. 18, no. 1, pp. 223–234, 2024.
[9] H. Zhang, Y. Sun, and T. Li, "Lightweight encryption techniques for edge-cloud security," *IEEE Transactions on Cloud Computing*, vol. 12, no. 3, pp. 1450–1462, 2024.
[10] P. Kumar, S. Verma, and A. Gupta, "Hybrid ECC-based encryption for secure cloud storage," *Journal of Network and Computer Applications*, vol. 225, pp. 103540, 2024.
[11] H. Wang, J. Liu, and Z. Su, "Zero trust architecture for secure cloud systems," *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 1023–1035, 2024.
[12] R. Patel, D. Shah, and M. Mehta, "Zero trust-based access control for multi-cloud environments," *IEEE Access*, vol. 13, pp. 22345–22359, 2025.
[13] S. Gupta, N. Jain, and V. Arora, "Dynamic trust evaluation in zero trust cloud frameworks," *Future Generation Computer Systems*, vol. 147, pp. 210–222, 2024.
[14] B. Diyyana, M. Rahman, and K. Islam, "Hybrid RSA-ECC encryption framework for cloud security," *Journal of Information Security and Applications*, vol. 78, pp. 103654, 2025.
[15] M. Reshi, A. Ahmad, and S. Rashid, "Privacy-enhancing technologies for secure cloud data sharing," *Peer-to-Peer Networking and Applications*, vol. 17, pp. 789–804, 2024.
[16] J. Sengupta, S. Roy, and P. Das, "Blockchain-enabled secure data sharing for industrial IoT," *IEEE Transactions on Network and Service Management*, vol. 20, no. 2, pp. 1345–1358, 2023.
[17] A. Manzoor, M. A. Shah, and F. Khan, "Blockchain-based proxy re-encryption scheme for secure IoT data sharing," *IEEE Internet of Things Journal*, vol. 10, no. 8, pp. 6789–6801, 2023.
[18] K. Agyekum, Q. Xia, and E. Sifah, "Secure data sharing using blockchain and proxy re-encryption," *IEEE Systems Journal*, vol. 17, no. 4, pp. 5678–5689, 2023.
[19] R. Reshi, M. Ashraf, and S. Qadri, "Fog computing and blockchain for secure data sharing," *Peer-to-Peer Networking and Applications*, vol. 17, pp. 905–918, 2024.
[20] A. Zarin, M. Ali, and S. Khan, "Enhancing cloud storage security using advanced cryptographic techniques," *Journal of Information Security and Applications*, vol. 80, pp. 103890, 2025.
[21] Y. Yang, H. Lin, and X. Zhou, "Fine-grained proxy re-encryption for cloud data sharing," *Pervasive and Mobile Computing*, vol. 95, pp. 101742, 2024.
[22] S. Hindhuja, P. Reddy, and K. Rao, "Secure data sharing using proxy re-encryption in cloud systems," *Journal of Cloud Computing*, vol. 14, no. 1, pp. 1–15, 2025.
[23] D. Choudhary, V. Singh, and R. Patel, "Secure data sharing in fog-based IoT systems," *IEEE Access*, vol. 12, pp. 67890–67905, 2024.
[24] T. Bhanu, S. Karthik, and R. Mohan, "Attribute-based encryption for secure cloud data sharing," *Peer-to-Peer Networking and Applications*, vol. 18, pp. 112–126, 2025.
[25] G. Xu, Y. Li, and H. Wang, "Efficient secure data sharing mechanism for cloud environments," *IEEE Cloud Computing*, vol. 10, no. 2, pp. 56–65, 2023.