A Cryptographic Framework Using AES-ECC with Threshold Key Management for Cloud Storage Systems

Abdulsalam Ibrahim Almirdasi, Mohamed Tahar Ben Othman
Department of Computer Science-College of Computer, Qassim University, Saudi Arabia

***Abstract*—Cloud storage systems have become an essential platform for storing and managing large volumes of data, but their security depends not only on confidentiality but also on integrity, controlled key management, and resistance to active attacks. Many existing protection approaches emphasize encryption of data while giving less attention to context–aware verification and controlled object recovery in untrusted cloud settings. This study proposes a novel hybrid cryptographic model for cloud storage systems, Object-Centric Threshold–Sealed Encryption with Two Keys (OCTET). The model integrates AES chunk-based encryption to protect data confidentiality, ECC to secure key exchange, threshold–based secrets to reconstruct, HKDF–derived per chunk, and Merkle root to enforce a verification–before–decryption policy. The implementation of the proposed model is in an emulated cloud storage system and examined on a dataset of large objects, then compared against baseline schemes, including symmetric and hybrid encryption models, under the same experimental environment. The main outcome demonstrates that the proposed model achieves practical performance, minor overhead, and superior resistance to the attack models. In general, this study demonstrates that the proposed model offers trade–offs between security and efficiency and a robust integrity technique for large objects in cloud storage systems.**

***Keywords*—Cloud storage system; hybrid encryption; AES; ECC; Object-Centric; threshold key management; integrity verification; Merkle root verification**

I. Introduction

Cloud computing systems are essential for providing trustworthy computation and storage resources, allowing organizations to easily manage data without using local servers [1]. This way of data storage brings several challenges related to potential data access and misuse [2]. While cloud solutions can be attractive for their flexibility, cost savings, and easier management, they also come with certain drawbacks, such as challenges related to multi–tenancy and having less direct control over the underlying hardware [3].

In addition, complicated service–based agreements and an external attack surface adds more complicated security. Any cloud data protection plan must, therefore, carefully balance these benefits against the risks involved. Although cloud–based systems provide advantages like flexibility, effectiveness, and enhanced control, these systems present challenges, such as multi–tenancy issues and limited hardware control [3]. Furthermore, complex agreements of service and a broad attack surface complicate security [2]. Any plan of data protection in cloud must balance these advantages and challenges.

Cryptographic schemes serve as a vital defense technique to secure data on the cloud and they use three different methods to provide security, which are symmetric, asymmetric, and hybrid key encryption. Symmetric methods utilize one key to encrypt/decrypt data, applying different schemes such as AES, Blowfish, and 3DES [1]. They are considered a viable option for handling large data because of their efficiency in rapid encryption. In the other hand, asymmetric encryption uses two different keys, where the public secret encrypts data and the private secret decrypts data, such as RSA and ECC, making it a proper option for key exchange due to their security robustness [4]. The hybrid models combine both methods, with a symmetric part for encrypting the large data and an asymmetric part to secure key exchange [5]. In the context of data encryption algorithms for cloud storage systems, it a viable choice to select hybrid encryption methods to provide the framework with fast data encryption by using a symmetric technique and secure key exchange between channels by using an asymmetric technique.

Furthermore, key management is an important concept in cryptography. If the keys are not effectively exchanged, stored, or generated, it means that the implementation of the cryptographic scheme is ineffective and cannot function successfully [1]. In cloud storage systems, keys should be generated with efficient randomness, linked with certain characteristics, and tracked via whole lifecycle. However, threshold–or distributed–based environments, key servers, and hardware security modules (HSM) each has distinguish advantages and disadvantages in terms of security, complexity, and availability [4]. As a growing number of organizations adopt cloud–based services, they need to carefully assess both their advantages and disadvantages to ensure that strong security measures are established. Understanding the main concepts and ideas of each strategy may significantly impact how efficient data security is.

For protecting cloud storage, there are some security concepts, which are confidentiality, integrity, and availability (CIA). Confidentiality is basically when authorized clients are able to read and access the data in cloud storage [4]. Cloud systems use cryptographic techniques and keeps the keys from being released to ensure confidentiality. Integrity means that the cloud storage model checks that the data objects have not been changed or deleted while they are stored [6]. The cloud storage model does integrity verification. Users frequently apply techniques such as authenticated encryption, hashing, and Merkle tree commitments to achieve integrity. The availability mainly is when real clients can retrieve and recover their data, whether there is an incident or partial leak

[3].

The main problem of this study is that many modern security techniques emphasize confidentiality instead of focusing on integrity and contextual relevance. It is difficult to manage keys and efficiently store data in cloud storage systems, where attacks commonly occur against users. CSPs must protect sensitive data and maintain the users' accessibility.

The aim of this study is to propose a new hybrid encryption model for cloud storage systems, which is called Object–Centric Threshold–Sealed Encryption with Two Keys (OCTET). The model encrypts data using AES and secures key exchange using ECC to balance between efficiency and security. The framework also has security features like chunk authentication and a secret sharing threshold. This study presents an evaluation and comparison of proposed model against baseline schemes in the context of security and efficiency. The following objectives are defined to achieve the main goal of this study:

- Develop OCTET, a hybrid encryption model that combines AES and ECC to enhance security and integrity through an object–centered approach.
- Adopt advanced security measures, including authenticated chunking, HKDF key derivation, and threshold secret sharing, to safeguard confidentiality, uphold integrity, and regulate access at both object and chunk levels.
- Simulate a reproducible OCTET prototype for cloud storage to assess its resilience against tampering and context–binding attacks, while comparing its efficiency and security with baseline schemes.

The main contribution of this study is to introduce OCTET, a cloud storage security framework that employs chunk encryption to maintain data integrity and user access control. By assigning specific secret keys to tenants and verifying data integrity before decrypting it, OCTET makes unauthorized access more complex. It combines several features, such as AES, ECC, and Shamir secret sharing, to properly manage systems with multiple users. Furthermore, it evaluates the efficiency and security trade–offs.

II. Literature Review

In this section, a review investigates the security of data in cloud computing systems, covering threats like violation of integrity and leakage of data. This review covers the symmetric and hybrid models, as well as the efficiency and security of these cryptographic models. Furthermore, it highlights existing frameworks and their limitations in the context of confidentiality, integrity, and key management flexibility.

A. Cloud Storage Security Challenges

Challenges of security in cloud computing arise due to the lack of control over infrastructure, applications, and data by users, which are handled by cloud service providers (CSP) via different locations. This way may cause some issues, such as loss of data, access for unauthorized users, delays, corruption, and potential risks for sensitive data. In [7], a review addresses the challenge of data leakage, which occurs due to vulnerabilities like cross attacks and admins that maliciously threaten confidentiality and integrity, highlighting the need for security in monitoring reliable computing and teamwork, while also investigating the dependence on new threats and untrusted systems. Also, in [8], another review highlights the significance of CIA as risks arise with Internet of Things (IoT) and cloud use, emphasizing techniques including access control and encryption. Furthermore, in [9], a comprehensive review shows that clients are losing direct control in cloud data centers, which leads to vulnerabilities, such as theft of data and disconnections of service. This demonstrates that more verifiable methods are needed to improve security robustness and efficient data protection.

These studies highlight persistent security issues in cloud storage, including data leaks, insider threats, or tenant attacks that influence CIA. Security measures such as encryption, access control, DLP, blockchain, and redundancy complicate business operations and increase costs. There is no single solution ensuring complete safety, and the proposed models reveal a research gap for integrated cloud–based security models that effectively combine strong encryption, key management, verifiable integrity, and performance efficiency.

These studies mainly define the landscape of threat at an overall level, like access authorization, data leakage, and control loss without providing cryptography designs that combine integrity and confidentiality with controlled recovery for stored data. Therefore, the proposed model (OCTET) is designed to cover these challenges via context-aware verification, threshold key management, and protection of objects without dealing with these security demands separately.

B. Existing Encryption Models for Cloud Storage

Recent trends in improving the security of cloud storage have resulted in different encryption schemes to preserve privacy while enabling search, access control, and other performance improvements. Several studies in the area of encryption models have significantly improved the security and accessibility of cloud data. In [10], SES–CSE enhances data confidentiality and availability by enabling efficient keyword searches in large encrypted cloud datasets while protecting plaintext information and demonstrating resilience against ciphertext and chosen–keyword attacks. Additionally, in [11], ECSES and ECRM make data more secure and available in cloud systems by using strong symmetric encryption and backup methods to lower the chances of problems from node failures or security breaches. Furthermore, in [12], an improved CP–ABE model makes it easier to control who can access encrypted cloud data in IoT environments, ensuring that only authorized users can get to the data and reducing security risks related to hashing functions and signature checks.

In summary, the studies demonstrate that encryption algorithms have been improved over time under cloud storage systems. Currently, encryption frameworks started using extra complicated tools such as searchable methods, redundancy–based check techniques, and attribute–based access control. So, the studies indicated the significance of cryptographic frameworks in terms of security of systems, protection of data, and access control policies.

Significant functions like confidentiality, backup support, searchable techniques, and access control are addressed as improved methods for cryptographic models. However, these functions are discussed as separate solutions without integrating chunking authentication with context-binding and data integrity. Furthermore, the proposed model (OCTET) is built to cover these limitations by using techniques such as AES for chunking and HKDF-derived keys with verification by Merkle root.

C. Hybrid Models and Key Management Mechanism

When combined with key management operations, hybrid encryption algorithms can efficiently alleviate the shortcomings of single cryptographic schemes in cloud computing. Such techniques generally utilize symmetric encryption to protect data while using asymmetric or distributed approaches for key management. Hybrid and key encryption models play a crucial role in enhancing data security in cloud environments. In [13], a blockchain–based key security management framework is proposed that utilizes an oblivious pseudo–random function to safeguard encryption keys in cloud storage, enhancing recovery from failures and ensuring immutable transaction records while demonstrating resilience against brute–force and collusion attacks with low computational costs for key generation. Furthermore, in [14], a hybrid model enhances cloud data protection by combining symmetric and asymmetric encryption algorithms with a private blockchain, utilizing AES for large files and RSA for securing the AES key, thereby improving key exchange and access monitoring. Finally, in [15], the SymECCipher framework combines ECC with AES to efficiently secure healthcare data in the cloud, enhancing key management and throughput protection tailored to the specific needs of cloud healthcare applications.

Finally, these investigations demonstrate the principles of how the proposed model functions. The main focus is that it is essential to supplement fast symmetric with asymmetric or decentralized key management. They also emphasize the advantages of threshold and secret–sharing schemes for secure key backup. The studies also show how blockchain technology or other equivalent techniques can be used to create a decentralized and tamper–proof way to monitor key management and access events in cloud storage systems.

These studies ensure the significance of integrating symmetric and asymmetric methods but still concern secure key exchange, key monitoring, and recovery instead of the combination of protection of chunks, key release, and verification-before-decryption policy. Therefore, key management techniques are improved, while data integrity and recovery control are not imposed simultaneously. On the other hand, the proposed model (OCTET) addresses this issue by using both AES and ECC with chunk-based derivation and threshold methods.

D. Integrity Mechanisms in Cloud Storage

Cloud storage involves transferring data from local devices to third–party servers, necessitating user trust in these providers to maintain data safety and accessibility. Concerns about data manipulation and loss are addressed through cryptographic integrity checks. Recent studies have explored innovative integrity mechanisms designed to bolster data verification and security across various domains. In [16], a study introduces a hierarchical Merkle hash tree for integrity auditing that improves data verification while reducing audit operations and expenses. Also, in [17], a model for checking the integrity of electronic health records (EHRs) in the cloud allows for fast checks without downloading all the data, using cryptographic tags to verify blocks and letting computations be done elsewhere to lessen the load on servers. Lastly, in [18], different storage–layer integrity techniques and the security problems they pose are grouped together, where the focus is on risks like insider threats and ways to improve efficiency.

Recent trends highlight a transition from basic checksum verifications to advanced cryptographic auditing methods that enhance efficiency and security. Hierarchical Merkle hash trees reduce audit costs and effectively detect tampering. Several studies establish methods that balance between vulnerabilities and costs of computations. In general, observations support cryptographic frameworks that provide greater security for large data stored on the cloud while maintaining performance.

These studies demonstrate that Merkle root and verification checks can detect tampered data and decrease the cost of verification. However, these techniques are mainly presented as mechanisms for checks and audits that come on the storage layer, where they are unable to show how integrity checks may be tied with the derivation of keys and control of decryption inside a cryptographic model. Therefore, the proposed model (OCTET) solves this gap by designing a step that ensures integrity before the decryption process using Merkle root verification and checking header authentication to prevent tampered data from the recovery process.

E. Limitations of Current Encryption Models

Although tools of cryptography in cloud environments are notably improved, they still lack solid approaches that encompass performance, supported cross–platform scalability in key management, and integrity assurance. Some of the difficulties addressed by security and encryption techniques are system robustness and user access control balance with performance. In [19], a blockchain-based model is proposed for data security in cloud environments using the combination of AES and RSA for encryption. The model is built using decentralized key management contracts and agreements to enhance confidentiality. Furthermore, in [20], the authors address security techniques using attribute–based cryptography, homomorphic encryption, and authentication of multi-factors. In addition, in [21], a hybrid cryptographic model is proposed using AES with OTP and RSA for cloud computing systems, which establishes dynamic key management with strict access control.

Overall, with the advancement of cryptographic models using blockchain methods, there are still challenges like processing delays and compatibility issues. Also, multi–layer cryptography systems faces challenges such as the increment of costs and complexity in key management. For hybrid models, they improve access control, yet their encryption process caused overhead data, which influenced the system performance.

Generally, these studies show that cloud encryption frameworks improve several aspects like access control, key manage-

ment, and confidentiality, while still lacking a balance between overhead, compatibility, integrated integrity verification, and scalability. Many encryption algorithms are strong in function isolation but still offer limited support for trade-offs in efficiency and security. On the other hand, the proposed model (OCTET) brings a mixture of techniques and behaviors as one cryptographic hybrid model for cloud storage systems.

III. Methodology

To propose a novel cryptographic model, OCTET, for cloud storage systems, the methodology of this research is a mixed–method approach. Qualitative method reviews current studies in cloud security and data encryption schemes to find out design demands. Quantitative method conducts the development and evaluation of OCTET, as well as the experiment setup with a real–world dataset. In terms of efficiency and security, a comparative analysis is also presented to compare the proposed model against baseline schemes.

A. Design of the Proposed Model

The Object–Centric Threshold–Sealed Encryption with Two Secrets (OCTET) model is a hybrid cryptographic framework designed to secure large data in cloud environments. It achieves its goal by binding data to its specific cloud location while employing dual secrets, threshold key release, and verifiable integrity. Firstly, OCTET splits data objects into several chunks of fixed size, then labels each chunk to be identified easily, which makes these labels linked to their corresponding encrypted chunks and linked to valid cloud identities, thus maintaining data security and integrity at its specific location.

OCTET generates two independent secret keys: the root object secret (ROS) and a second secret known as "Delta" (δ), which is deterministically derived from a key management system (KMS). ROS is stored using Shamir–style threshold sharing, allowing it to be reset through elliptic curve cryptography (ECC) after receiving a predetermined number of approvals (k–of–n). Both ROS and δ are processed through a hash–based message authentication key derivation function (HKDF) to produce advanced encryption standard (AES) parameters for each data chunk, preventing key reuse. AES encrypts per–chunk, resulting in ciphertext and a tag. The proposed model builds a Merkle tree from chunks of ciphertext hashes, creating one MerkleRoot that ensures the overall integrity of the object. When clients retrieve encrypted data (ciphertext), the framework reconstructs ROS and calculates δ, validating the Merkle root and header message authentication code (MAC) before it performs decryption, where the header is created containing Merkle root and other parameters, as well as deriving the two secret keys (ROS and δ) by MAC. Upon successful verification, keys of decryption are generated via HKDF, then chunks are decrypted, and objects are recovered into original plaintext. Fig. 1 illustrates the whole structure of proposed model (OCTET), showing the encryption and decryption workflows organized around chunking, key security, and integrity check, which clearly defines that the model is not retrieving the data, while it initially reconstructs the needed key material and verifies the integrity information.

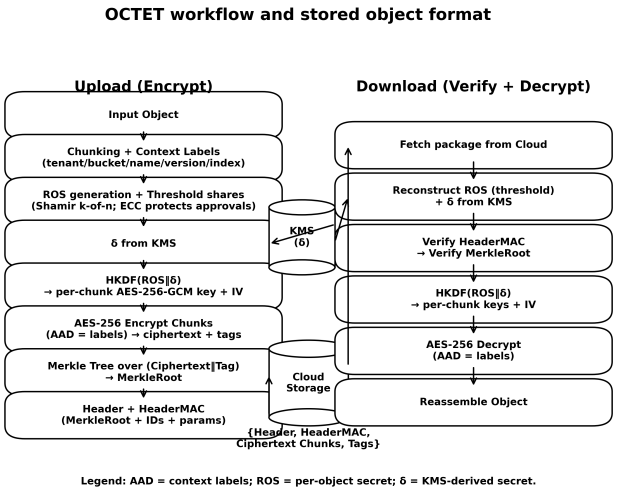

Fig. 1. Encryption and decryption in proposed model (OCTET).

B. Selected Encryption Algorithms

This study presents a comparative analysis of algorithms to assess OCTET for cloud data storage, focusing on key requirements such as data privacy, integrity verification, and key management. The study involves tested algorithms, including hybrid models that offer AEAD and symmetric schemes that utilize an encrypt–then–MAC scheme with HMAC–SHA256 for integrity verification. Under controlled scenarios, the evaluation is presented with consistency using the same settings and workloads for all the selected encryption algorithms to ensure fairness. The algorithms' configurations are listed below to recognize the procedures and roles for the comparison:

1) Proposed Model (OCTET): It employs a 60 MB chunk size for data encryption, utilizing AES–256–GCM for encryption and ECC P–256 for key encryption, which is linked to the object's storage context. Implements a 2–of–3 threshold to reconstruct the ROS per object with the secondary secret key δ, derived during the decryption process.

2) Blowfish: It is configured with a key size of 256 bits and an initialization vector (IV) with a size of 64 bits per chunk, which ensures the consistency for the scenarios measuring the efficiency of the encryption/decryption process.

3) Triple DES (3DES): It applies 3DES–EDE procedures using a setup of two secrets, providing the mode of CBC with 112 bits and PKCS#7 with 8 bytes.

4) Hybrid AES-RSA (AES-RSA): It utilizes AES–256–GCM with a tag size of 128 bits for encrypting data, while RSA–2048 is used to secure the keys of AES with RSAES–OAEP and SHA–256 to encrypt keys.

5) Hybrid AES-ECC (AES-ECC): It also utilizes AES–256–GCM with a tag size of 128 bits for encrypting data, while ECC (P–256) in the secp254r1 curve is used to secure key exchange with ECDH mode integrated with HKDF–SHA256 to derive keys.

The selected schemes offer a standard measurement for evaluation purposes of the proposed model (OCTET). The symmetric algorithms, including Blowfish and 3DES, are used

to present traditional cryptography behavior. The asymmetric algorithms are also selected to present the cryptography behavior. AES–RSA is chosen as a standard hybrid metric that shows the common process, where AES is utilized for the encryption of large data and RSA–OAEP is utilized for key-wrapping security. Additionally, AES–ECC is selected as the nearest hybrid model to OCTET due to both models relying on AES and ECC, which makes it a basic representation for the proposed model. However, AES–ECC is unlike OCTET because of the additional techniques in OCTET, like context-binding, the verification-before-decryption policy, and the threshold method.

C. Data Collection

In this study, the dataset is used for evaluation of OCTET under scenarios for cloud storage of bulk data instead of presenting all heterogeneous workloads in the cloud. The dataset is a collection of video files that focused on large object sizes, ranging from 30 to 300 MB with an average size of 120 MB and a total size of approximately 1.2 GB. It contains different sizes of objects, which allows lightweight measures on small files and enables costs scalability for larger objects via chunking processing. Before the encryption process, all objects are set and read in byte format for authenticity and accuracy of analysis. The evaluation solely considers the performance of OCTET for large objects, where smaller objects remain a demand for a broader workload generalization.

D. Implementation and Experimental Setup

In this work, experiments are introduced under control settings to efficiently assess the cryptographic algorithms, using Python 3.12, including its cryptographic libraries. The cloud storage system is simulated with a local drive to store data, using AES–256 in AES–256–GCM (AEAD) mode and SHA256 for hashing. The setup of hardware is a PC device of Windows 11 Pro using an Intel CPU i5, RAM with a size of 16 GB, and a hard disk with a size of 1 TB. The envelope encryption method is used to secure stored files locally and decrypt them for retrieval.

E. Evaluation Metrics

This study utilizes various evaluation metrics to assess the security and efficiency of the proposed model within a cloud storage system. Furthermore, other evaluation metrics are utilized to compare the proposed model against the selected algorithms under a consistent and unified environment. Overall, the evaluation metrics demonstrate the real costs of processing, the impact on storage, and the robustness of the system in controlled tests. The evaluation metrics used in this study are set below for both the OCTET assessment alone and the comparative analysis against the other algorithms.

1) OCTET assessment:

a) Encryption time: It measures the time taken by OCTET to perform the encryption process for different objects, starting from the initiation of plaintext preparation to the resultant packages of ciphertext, which reflects how fast OCTET is to encrypt data.

b) Decryption time: It measures the time taken by OCTET to perform the decryption process for different objects, starting from ciphertext retrieval and authorization material to plaintext recovery, including the two key derivations, the verification-before-decryption method, and AES decryption, which reflects how fast OCTET is to decrypt data.

c) Comparison of data sizes (overhead): It compares the difference between the original and encrypted data sizes to measure the overhead produced by OCTET, evaluating how much an encryption scheme costs, which reflects the storage impact.

d) Attack detection rate under threat model simulation: It assess the proposed encryption model to ascertain its effectiveness in detecting and preventing detrimental data modifications across various attack scenarios. The main goal of this security testing is to test how well OCTET can detect and block unauthorized data by simulating attacks such as replay attacks and ciphertext tampering. This work illustrates the techniques used to verify data integrity, bind context, and authorize access while blocking attempts at decryption after detecting any modifications.

2) Assessment for comparative analysis:

a) Encryption and decryption throughput: It evaluates the speed of cryptographic schemes to compare which algorithm is faster by computing the data processed per time. Furthermore, throughput determines how well a cryptographic algorithm can handle bulk data. Mainly, the speed in OCTET compares to other selected encryption algorithms, focusing on efficiency.

b) Performance metrics: It examines the baseline schemes using security outcome classification, including accuracy, precision, recall, and F1–score. These metrics are computed using simulated attack models against each selected cryptographic method, evaluating how reliably and accurately an encryption algorithm detects tampered data, which reflects the reliability of the decisions made during the decryption process.

c) Error metrics: It examines the selected encryption algorithms using error metrics like mean absolute error (MAE) and mean squared error (MSE). These two metrics are computed under controlled environments, determining the deviation of decrypted output from predicted behavior among different tests, which reflects how much a cryptographic algorithm is able to correctly or incorrectly return bytes under untrusted scenarios.

IV. Results

This section presents the experimental results of the hybrid cryptographic model (OCTET), tested on simulated cloud storage systems with various file sizes. It includes evaluations of OCTET's efficiency and security through speed measurements and robustness tests. Additionally, OCTET is compared against other encryption algorithms within the same cloud simulation using the same dataset.

A. Evaluation of the Proposed Model

To assess the efficiency and security of OCTET, several assessments are performed to evaluate the proposed model.

TABLE I. ENCRYPTION OPERATION TIMING FOR OCTET

Encryption Operation of (OCTET)	Encryption Time (s)
ROS generation	0.000005
Shamir threshold sharing	0.000078
ECC share packaging	0.008818
HKDF–SHA256	0.000216
AES encryption total	0.061667
TOTAL encryption path time	0.070784

In terms of efficiency, time is measured for the encryption and decryption process, as well as the overhead generated for OCTET. In terms of security, context-binding and tamper resistance are examined against attack models with manipulation to define how OCTET can detect data that has been tampered with and modified.

1) Average encryption time: As shown in Table I, OCTET efficiently distributes computing costs across encryption tasks, with the AES algorithm significantly influencing total encryption time due to its processing of all file data. Operations related to key handling and authorization have minimal impact, while per–chunk keys generated with HKDF show negligible performance changes. Despite minor overhead from ECC share protection and Shamir 2–of–3 threshold sharing, the results highlight OCTET's focus on rapid symmetric encryption and minimal key management. Fig. 2 illustrates the average encryption time for OCTET's operations, where the main cost is caused by the AES encryption step due to the encryption process of the whole object, where the key management steps involve less time, which means it demonstrates that OCTET retains its main cost in the main data path instead of the additional security techniques.

2) Average decryption time: Table II shows that OCTET handles the decryption process, which balances the workloads of retrieval and verification, where the highest time taken to decrypt data is decryption of AES and verification of tag. The policy of verification–before–decryption is applied on each chunk to ensure data integrity, which causes delays. Recomputing Merkle root is a verification operation that ensures that chunks are valid. Extra time is added due to per–chunk key regeneration, overhead by ECC management shares, approvals, and reconstruction of ROS. In general, the cost of decryption is still moderate, while providing vital security techniques that improve the integrity of cloud–object. Fig. 3 illustrates the average decryption time for OCTET's operations, showing that the integrity process has the main role in the decryption process, where the model checks data

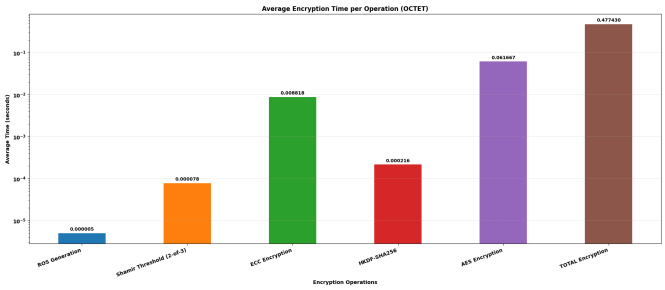

Fig. 2. Average encryption operations time.

TABLE II. DECRYPTION OPERATION TIMING FOR PROPOSED MODEL (OCTET)

Decryption Operation of (OCTET)	Decryption Time (s)
ECC approvals/shares handling	0.003104
ROS reconstruction	0.000030
Merkle verification	0.169917
HKDF–SHA256	0.000189
AES decryption + verification	0.073829
TOTAL decryption path time	0.247069

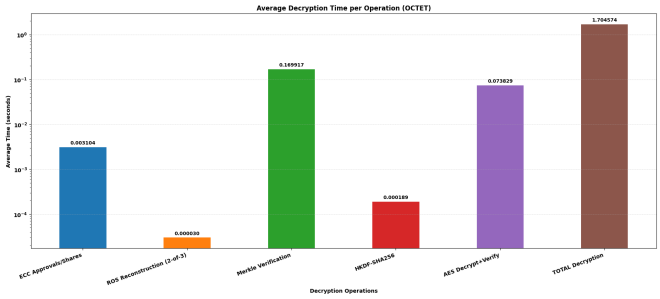

Fig. 3. Average decryption operations time.

TABLE III. COMPARISON OF DATA SIZES IN PROPOSED MODEL (OCTET)

Cryptographic Element	Size (Bytes)
Original Data	1,209,896,808
Encrypted Data	1,209,896,808
Authentication Tags	368
Header	4,378
Encrypted Key/ECC–Threshold	21,540
Total Overhead	26,286

correctness before release, which means that OCTET involves the verification-before-decryption policy.

3) Comparison of data sizes: Table III shows the data comparison sizes of OCTET, demonstrating that the storage efficiency maintains the size of encrypted data (ciphertext) same as original data (plaintext), which is essential for bulk data management in cloud environments. Authentication tags of AES showed the lowest overhead, where the headers of metadata and protected keys are still marginal comparing with the size of plaintext. Subsequently, OCTET obtained minor overhead and achieved data confidentiality, access control, and integrity, which indicates that the proposed model is a practical solution for cloud computing systems that handle large-scale data. Fig. 4 illustrates the comparison of the plaintext and ciphertext with the extra security methods involved in OCTET, showing that the size of the original and encrypted data is approximately similar, where the small storage increment occurred due to metadata, tags, and key material security, which mainly demonstrates that OCTET achieves minor overhead.

4) Attack detection rate under threat model: Table IV shows that OCTET achieved a high detection rate via several attack models, including tampering with ciphertext, integrity verification replay, and encryption authentication. It achieved a full rejection for the context–binding attack, demonstrating the strength of data–context labels. In contrast, the attack of mix–and–match faced difficulty in verification. Design of OCTET improves the security, where mix–and–match model tests the

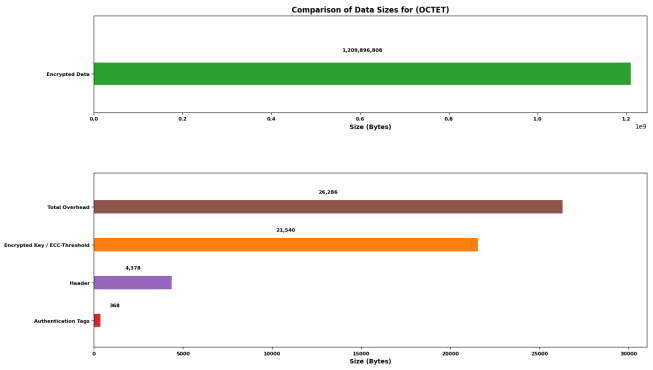

Fig. 4. Data sizes comparison and overhead of proposed model.

TABLE IV. ATTACK DETECTION RATE UNDER THREAT MODEL AGAINST PROPOSED MODEL (OCTET).

Attack Type	Detected Attacks	Detection Rate (%)
Ciphertext Tamper	98	98.00
Replay	99	99.00
Context Mismatch	100	100.00
Mix–and–Match	95	95.00

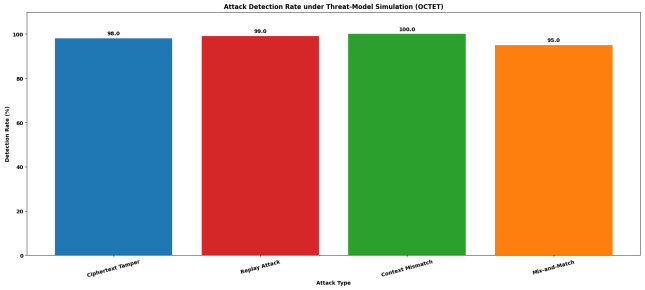

Fig. 5. Detection rate of proposed model (OCTET) against different attacks.

reliability and integrity of metadata. Fig. 5 illustrates how well OCTET stands against various attack models, showing that the model reliably achieves high rates for tampering threats and mismatches in context, which means that the robust integrity policy supports the security and performance enhancement of OCTET.

B. Comparative Analysis

The test compares OCTET against baseline schemes such as Blowfish, 3DES, hybrid AES–RSA, and hybrid AES–ECC. All of these were tested in the same way to find differences between the encryption schemes. The comparison uses metrics such as throughput to evaluate efficiency, while performance and error metrics test security.

1) Throughput of encryption and decryption: Throughput measures the speed of each selected algorithm to determine which algorithm is faster. As shown in Table V, the highest throughput is achieved by AES–ECC, while symmetric algorithms had the slowest throughput. On the other hand, OCTET achieved better throughput than symmetric schemes

TABLE V. AVERAGE ENCRYPTION AND DECRYPTION THROUGHPUT FOR DIFFERENT CRYPTOGRAPHIC ALGORITHMS.

Algorithm	Encryption Throughput	Decryption Throughput
OCTET	293.983	156.636
Blowfish	84.036	113.438
3DES	23.056	24.885
Hybrid AES–RSA	530.844	587.304
Hybrid AES–ECC	**549.668**	**555.848**

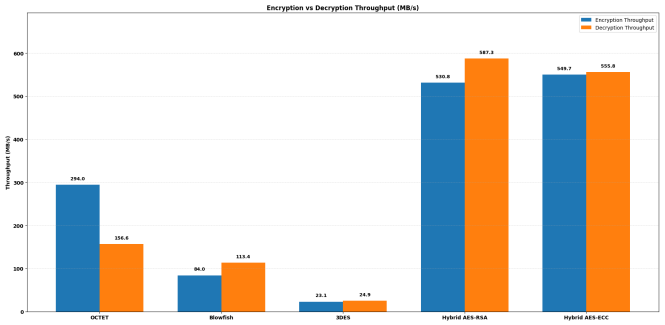

Fig. 6. Throughput of encryption and decryption for different cryptographic algorithms.

TABLE VI. PERFORMANCE METRICS FOR DIFFERENT CRYPTOGRAPHIC ALGORITHMS.

Algorithm	Accuracy	Precision	Recall	F1–Score
OCTET	**99.84**	**99.92**	**99.58**	**99.72**
Blowfish	95.12	95.78	94.08	94.92
3DES	92.23	93.08	90.48	91.76
Hybrid AES–RSA	97.76	98.18	97.08	97.63
Hybrid AES–ECC	96.98	97.38	96.28	96.83

and less than hybrids due to additional techniques for security. Fig. 6 illustrates the throughput for the comparison of selected algorithms, showing that OCTET does not achieve the highest speed among others but is still practical while remaining outperforming symmetric algorithms, which means that the extra security techniques cause a small cost, and OCTET still operates effectively for trusted cloud storage.

2) Performance metric: Performance metrics assessment assesses the reliability and correctness of cryptographic techniques under controlled conditions, focusing on robustness against simulated attacks. Key metrics include accuracy, precision, recall, and F1–score. As shown in Table VI, OCTET outperforms others in reliability and consistency, while other hybrid models show moderate strength. On the other hand, symmetric encryption schemes, including Blowfish and 3DES, achieve the least accuracy and security. OCTET offers superior object protection characteristics, making it the strongest choice among the selected encryption algorithms, with hybrid models as alternatives and symmetric algorithms as less effective. Fig. 7 illustrates the accuracy and reliability of OCTET against other encryption schemes, showing that OCTET achieves the highest score for reliability, including accuracy, precision, recall, and F1-score, among others.

3) Error metric: Error metrics examine how the encryption algorithm's output is correct and accurate by computing deviation via indicators like MAE and MSE. Table VII shows

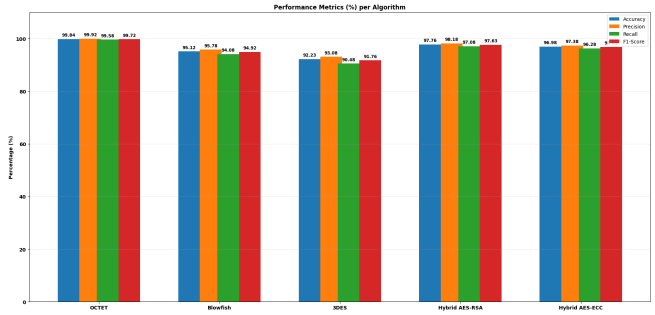

Fig. 7. Reliability of performance metrics for selected encryption schemes.

TABLE VII. ERROR METRICS FOR DIFFERENT CRYPTOGRAPHIC ALGORITHMS.

Algorithm	MSE	MAE
OCTET	**0.553**	**0.874**
Hybrid AES–RSA	0.611	0.921
Hybrid AES–ECC	0.628	0.938
Blowfish	0.742	1.041
3DES	0.815	1.109

that OCTET achieved the lowest values of both MAE and MSE, which means that the proposed model outperformed. However, symmetric algorithms had the highest rates, which indicates that they have less robustness than other algorithms. The policy of verification before decryption improves the plaintext accuracy, which leads to lower error rates than other standard encryption schemes. Fig. 8 illustrates the error rates and resilience behavior for the baseline schemes, showing that OCTET achieves the lowest score, which means that its security and verification steps decrease incorrect recovery under the experimental environment.

V. DISCUSSION

This discussion analyzes the implications of robust confidentiality, handling keys, and integrity verification. It emphasizes the efficiency and security of OCTET compared to baseline schemes, highlighting the usability of cloud systems for data objects verification and the balance of security measures and computational cost. Experimental results demonstrates that OCTET accomplished the goal of this study, showing the robustness and practical efficiency of OCTET. Understanding these findings helps readers to select ideal practices of security for cloud storage systems.

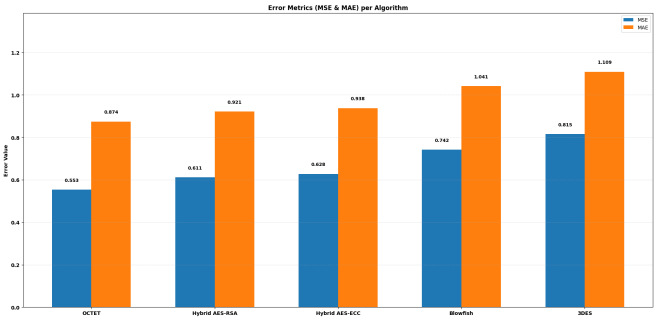

Fig. 8. Error metrics and resilience for different cryptographic algorithms.

The OCTET decryption process offers moderate throughput with high verification costs, primarily due to the extensive steps taken to ensure data integrity, such as checking header authenticity and validating object keys. While this procedure results in increased latency compared to other models, OCTET is advantageous for systems that can manage this overhead. It secures objects from manipulations of identities in cloud–based systems by decrypting data only after checking integrity. Selected algorithms don't have robust integrity verification, whereas OCTET's use of cloud identity labels enhances security, meaning that additional efficiency overheads are an essential option.

One of the limitations of this study is that the assessment dataset only consists of bulk objects. So, the resultant performance should be interpreted as evidence for large-object cloud storage environments instead of all types of cloud workloads. In the future, the evaluation can include different heterogeneous cloud workloads, like structured records, small documents, and other dataset types to examine the cryptographic model across heterogeneous storage data.

VI. CONCLUSION

In this study, a hybrid cryptographic model for cloud storage systems, called OCTET, is proposed, emphasizing robust integrity and confidentiality while allowing access of context–aware and key management in multi–user settings. OCTET combines chunk authentication, keys derivation procedure, threshold–protected key material, and verifiable integrity to make sure that ciphertext is connected to its cloud identity, which decrypts data when the verification process is completely performed. OCTET balances security and efficiency successfully, adding a minor overhead of 26,286 bytes totally due to the additional security techniques. The speed of encryption and decryption throughput is 293.983 and 156.636, respectively, in OCTET, which is practical comparing to the selected baseline schemes. It reliably and strongly detects several attack models with high rates, where it achieves 98.00% for tampering, 99.00% for replay, 100.00% for context mismatch, and 95.00% for mix-match, which indicates that the proposed model is a proper option for cloud storage systems while improving security with respect to standard cryptographic schemes.

At the end, the main outcomes imply that OCTET efficiently secures bulk objects with small overhead and maintains high throughput. It is especially beneficial for cloud–based storage systems that require access control policies, protection against active attacks, and strong data integrity. Future work includes cloud testing the proposed model distributed in a storage system with a wider range, discovering different risky behaviors, evaluating heterogeneous cloud workloads, and refining implementation, such as metadata handling and chunking in parallel to improve ease of application and efficiency.

REFERENCES

[1] A. Jula, E. Sundararajan, and Z. Othman, "Cloud computing service composition: A systematic literature review," *Expert Systems with Applications*, vol. 41, no. 8, pp. 3809–3824, 2014, doi: 10.1016/j.eswa.2013.12.017.

[2] H. Tabrizchi and M. K. Rafsanjani, "A survey on security challenges in cloud computing: issues, threats, and solutions," *The Journal of Supercomputing*, vol. 76, pp. 9493–9532, 2020, doi: 10.1007/s11227-020-03213-1.

[3] R. Nachiappan, B. Javadi, R. N. Calheiros, and K. M. Matawie, "Cloud storage reliability for big data applications: A state of the art survey," *Journal of Network and Computer Applications*, vol. 97, pp. 35–47, 2017, doi: 10.1016/j.jnca.2017.08.011.

[4] M. Campagna and S. Gueron, "Key Management Systems at the Cloud Scale," *Cryptography*, vol. 3, no. 3, Art. no. 23, 2019, doi: 10.3390/cryptography3030023.

[5] S. Ahmad, S. Mehfuz, and J. Beg, "Hybrid cryptographic approach to enhance the mode of key management system in cloud environment," *The Journal of Supercomputing*, vol. 79, pp. 7377–7413, 2023, doi: 10.1007/s11227-022-04964-9.

[6] J. Mao, Y. Zhang, P. Li, T. Li, Q. Wu, and J. Liu, "A position-aware Merkle tree for dynamic cloud data integrity verification," *Soft Computing*, vol. 21, pp. 2151–2164, 2017, doi: 10.1007/s00500-015-1918-8.

[7] B. M. Salih and O. K. J. Mohammad, "Cloud Data Leakage, Security, Privacy Issues and Challenges: Review," *Procedia Computer Science*, vol. 242, pp. 592–601, 2024, doi: 10.1016/j.procs.2024.08.113.

[8] M. E. Moudni and E. Ziyati, "Advances and Challenges in Cloud Data Storage Security: A Systematic Review," *International Journal of Safety & Security Engineering*, vol. 15, no. 4, pp. 653–675, 2025, doi: 10.18280/ijsse.150403.

[9] L. K. Suresh Kumar, "Cloud Computing Data Storage Security: A Comprehensive Review," *International Journal of Intelligent Systems and Applications in Engineering (IJISAE)*, vol. 12, no. 3, pp. 4443–4449, 2024.

[10] Y. Xiong and M. X. Luo, "Searchable Encryption Scheme for Large Data Sets in Cloud Storage Environment," *Radioengineering*, vol. 33, no. 2, pp. 223–235, Jun. 2024, doi: 10.13164/re.2024.0223.

[11] R. A., S. Kautish, S. Juneja, K. Mohiuddin, F. K. Karim, H. Elmannai, S. Ghorashi, and Y. Hamid, "Enhanced Cloud Storage Encryption Standard for Security in Distributed Environments," *Electronics*, vol. 12, no. 3, Art. no. 714, 2023, doi: 10.3390/electronics12030714.

[12] P. Chinnasamy, P. Deepalakshmi, A. K. Dutta, J. You, and G. P. Joshi, "Ciphertext-Policy Attribute-Based Encryption for Cloud Storage: Toward Data Privacy and Authentication in AI-Enabled IoT System," *Mathematics*, vol. 10, no. 1, Art. no. 68, 2022, doi: 10.3390/math10010068.

[13] J. Huang and J. Yi, "The key security management scheme of cloud storage based on blockchain and digital twins," *Journal of Cloud Computing*, vol. 13, no. 1, Art. no. 15, 2024, doi: 10.1186/s13677-023-00587-4.

[14] F. M. Khalaf and A. M. Sagheer, "A Hybrid Encryption Model with Blockchain Integration for Secure Cloud Data Storage and Retrieval," *Journal of Intelligent Systems and Internet of Things*, vol. 16, no. 2, pp. 236–245, 2025, doi: 10.54216/JISIoT.160217.

[15] P. Selvi and S. Sakthivel, "A hybrid ECC-AES encryption framework for secure and efficient cloud-based data protection," *Scientific Reports*, vol. 15, Art. no. 30867, 2025, doi: 10.1038/s41598-025-01315-5.

[16] Z. Liu, S. Wang, S. Duan, L. Ren, and J. Wei, "Dynamic Data Integrity Auditing Based on Hierarchical Merkle Hash Tree in Cloud Storage," *Electronics*, vol. 12, no. 3, Art. no. 717, 2023, doi: 10.3390/electronics12030717.

[17] P. Goswami, N. Faujdar, S. Debnath, A. K. Khan, and G. Singh, "Investigation on storage level data integrity strategies in cloud computing: classification, security obstructions, challenges and vulnerability," *Journal of Cloud Computing*, vol. 13, no. 1, Art. no. 45, 2024, doi: 10.1186/s13677-024-00605-z.

[18] Y. Wu, X. Tan, and Q. Xie, "Certificateless Provable Data Possession Scheme for Cloud-Based Electronic Health Records System," *Mathematics*, vol. 12, no. 24, Art. no. 3883, 2024, doi: 10.3390/math12243883.

[19] R. Li, W. Liang, S. Bo, X. Ran, X. Kong, and C. Wang, "Cloud Storage Privacy Data Protection Technology Based on Blockchain," *Procedia Computer Science*, vol. 262, pp. 1388–1394, 2025, doi: 10.1016/j.procs.2025.05.186.

[20] K. N. Mishra, R. K. Lal, P. N. Barwal, and A. Mishra, "Advancing Data Privacy in Cloud Storage: A Novel Multi-Layer Encoding Framework," *Applied Sciences*, vol. 15, no. 13, Art. no. 7485, 2025, doi: 10.3390/app15137485.

[21] D. Shivaramakrishna and M. Nagaratna, "A novel hybrid cryptographic framework for secure data storage in cloud computing: Integrating AES-OTP and RSA with adaptive key management and Time-Limited access control," *Alexandria Engineering Journal*, vol. 84, pp. 275–284, 2023, doi: 10.1016/j.aej.2023.10.054.