10.48047/jocaaa.2024.33.08.185

A Comprehensive Cloud Security Model for Enhanced Data Privacy and Access Control in Distributed Cloud Environments

Milind[1*], Nidhi Sharma[2], Amit Sharma[3]
[1*,2*,3]Department of Computer Science and Engineering, SCRIET,
Chaudhary Charan Singh University Meerut U.P
*Email: milindccsu@yahoo.com ,chotisharma301@gmail.com, amitccsu.net@gmail.com

Abstract

The pervasive adoption of cloud computing, while offering significant benefits in scalability and cost-efficiency, introduces complex challenges related to data security and privacy. Issues such as loss of control over data, inherent trust deficits with Cloud Service Providers (CSPs), and an evolving landscape of cyber threats necessitate robust and integrated security solutions. This research paper introduces a comprehensive cloud security model designed to address these critical concerns. The proposed model integrates three novel components: the Amino Hydropathy Index Encryption Algorithm (AHIEA) for efficient and secure data encryption, the Modified Chebyshev polynomial Based Access Control (MCBAC) scheme for multi-level authentication and fine-grained access, and the Privacy Based Data Publishing Algorithm (PBDPA) for optimizing privacy-utility balance in data sharing. The AHIEA leverages chaos-based key generation, demonstrating superior performance in key generation, encryption, and decryption times compared to existing symmetric algorithms like AES and DES. The MCBAC, employing Chebyshev polynomials and multi-level verification, exhibits significantly higher precision, recall, and detection rates against common authentication attacks, including password guessing, brute force, and dictionary attacks. Furthermore, the PBDPA, utilizing a Genetic Grey Wolf Optimization (Genetic-GWO) algorithm for k-anonymization, achieves a more effective balance between data privacy and utility, as evidenced by minimized Generalized Information Loss (GILoss) and Average Equivalence Value (CAvg) across diverse datasets. This integrated model significantly contributes to enhancing secure, private, and efficient cloud data management, particularly for sensitive information in distributed environments.

Keywords:
Cloud Computing, Data Security, Privacy, Amino Hydropathy Index Encryption Algorithm (AHIEA), Modified Chebyshev polynomial Based Access Control (MCBAC), Privacy Based Data Publishing Algorithm (PBDPA), Chaos-Based Key Generation, Multi-Level Authentication, k-Anonymization

1. Introduction

1.1 Overview of Cloud Computing

The landscape of computation has undergone a profound transformation, shifting from a two-tier centralized architecture to distributed computing, and subsequently evolving into a three-tier virtual centralization model. This evolution defines cloud computing as a paradigm where a vast array of computing resources is perceived and accessed as a "Cloud".[1] This cloud serves as a repository of complex and interconnected resources, designed to simplify technology for real-

10.48047/jocaaa.2024.33.08.185

world applications.[1] Businesses and individuals alike benefit from dynamic, scalable, and virtualized resources delivered through a "pay-per-use" service model, where "everything is distributed in the way of 'as a Service'".[1] This "as a Service" approach signifies the fine-grained reuse of components via the internet.[1]

The utility and rapid delivery of resources offered by cloud computing have made it indispensable across a wide spectrum of sectors. Its influence spans from government and military applications to satellite communication, and it garners significant interest from researchers, scientists, and hospitals globally.[1] The ability to instantly access applications such as healthcare, medicine, emergency management, data storage and publishing, and social networks from any location using heterogeneous devices has fundamentally reshaped resource delivery.[1]

While cloud computing presents a simplified user experience by abstracting away the underlying infrastructure complexities, it inherently centralizes and virtualizes resources. This fundamental design choice, though beneficial for scalability and cost-efficiency, introduces a critical shift in the locus of control and responsibility. End-users, by design, often relinquish direct control over their data and the technologies underpinning cloud services.[1] This lack of direct oversight, while simplifying operational aspects for the user, inadvertently introduces new and often more intricate security and trust challenges. For instance, the very mechanisms that enable cloud efficiency, such as multi-tenancy and virtualization, can generate novel attack vectors and impose a significant computational and complexity burden on ensuring information protection and disallowing server access to user data structures.[1] The perceived simplicity at the user interface level thus masks an underlying increase in technical and organizational complexity required to maintain data integrity, confidentiality, and privacy within this virtualized environment.

1.2 Cloud Security and Privacy Challenges

Despite the transformative benefits of cloud computing, its widespread adoption is significantly hindered by persistent and evolving security and privacy concerns. The concept of securing valuable information is not new, but its application in the digital realm, particularly within cloud environments, presents unique struggles.[1] The broad sharing of data, especially sensitive personal information, carries the potential for serious harm if compromised.[1] Consequently, robust access control models are essential to protect stored data against policy violations from both authorized and unauthorized users.[1] A core difficulty lies in simultaneously ensuring trust on the server-side while providing legitimate users with the necessary access.[1]

The inherent characteristics of cloud computing, such as its shared and on-demand nature, introduce novel security vulnerabilities that can potentially negate the advantages of migrating to cloud technology.[1] For example, the attributes of multi-tenancy and virtualization, which are foundational to cloud efficiency, also create different attack surfaces compared to conventional computing models.[1] The security posture of a distributed environment like the cloud is intrinsically tied to the weakest entity within its ecosystem.[1] This means that even minor vulnerabilities can have cascading effects, compromising the overall security strength.

The "as a Service" model, central to cloud computing, facilitates the reuse of components in a fine-grained manner across the internet.[1] This model underpins essential cloud characteristics like

10.48047/jocaaa.2024.33.08.185

resource pooling, which employs a multi-tenant approach to serve numerous consumers.[1] While these features drive economic and operational advantages such as reduced costs, enhanced scalability, and rapid elasticity, they concurrently expand the potential attack surface. A vulnerability within any shared component or layer can impact multiple tenants, leading to systemic risks that are amplified by the interconnected and on-demand nature of cloud services.[1] This establishes a direct relationship between the cloud's operational model and the emergence of unique, complex security challenges, demanding a proactive and adaptive approach to security that transcends traditional perimeter defenses.

2. Literature Review

A foundational understanding of cloud computing necessitates an examination of its architectural framework and service delivery models. The National Institute of Standards and Technology (NIST) defines cloud computing architecture as comprising several key entities: Cloud Service Providers (CSPs), Cloud Users, Cloud Carriers, Cloud Auditors, and Cloud Brokers.[1] Each entity plays a specific role, participating in processes and performing tasks within the cloud computing ecosystem.[1] CSPs deliver services to Cloud Users based on business relationships, while Cloud Auditors assess mechanisms implemented on cloud services.[1] Cloud Brokers facilitate connections between Cloud Suppliers and Cloud Buyers, and Cloud Carriers provide network connectivity and transport services from providers to consumers.[1]

The clear benefits derived from cloud computing systems include increased economic return through reduced operational and maintenance costs.[1] This is primarily achieved by transforming IT assets into utility costs, paid only for the duration of resource utilization.[1] This model particularly benefits small and new companies, allowing them to scale without large initial investments.[1] The essential characteristics of cloud computing that enable these benefits are Self-Service (instant access to resources), Broad Network Access (heterogeneous client platforms), Resource Pooling (multi-tenant model for dynamic provisioning), Rapid Elasticity (flexible provisioning based on demand), and Measured Service (automated control and optimization with billing).[1]

Cloud deployment models categorize the cloud based on location, ownership, and resource sharing. These include Public Cloud (dynamically supplied over the internet by third-party vendors, shared infrastructure), Private Cloud (dedicated to a single organization, high control over infrastructure), Hybrid Cloud (combining internal and external vendors, allowing sensitive data in private cloud and non-core apps in public cloud), and Community Cloud (shared resources among a group with common interests).[1]

Cloud computing services are broadly classified into three primary models: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS).[1] IaaS provides virtualized resources on-demand, including compute servers, operating systems, load balancers, and firewalls, allowing consumers to pay based on usage.[1] Major IaaS providers include Amazon Web Services (AWS), Google Compute Engine (GCE), and Microsoft Azure.[1] PaaS offers a deployment and development environment, providing frameworks, programming languages, editors, and persistent data storage for building and running applications.[1] Examples include Aneka, Microsoft Azure, and App Engine.[1] SaaS delivers applications to clients via a web-based

10.48047/jocaaa.2024.33.08.185

interface, minimizing software maintenance burden and improving application development and testing.[1] Salesforce.com and Googledocs are prominent examples.[1]

A fundamental observation in cloud computing is that the model does not grant clients full control over their information.[1] Unlike conventional computing, cloud computing allows service providers to activate control of data and servers.[1] This inherent shift of data and infrastructure management from the user to a third-party CSP directly results in a diminished level of direct control for the data owner. This absence of control contributes to greater information security risks than traditional processing models.[1] Enterprises often lack knowledge regarding the physical storage location of their data or the specific security mechanisms in place to protect it.[1] This lack of transparency regarding the CSP's internal practices, physical infrastructure, and data access logs naturally erodes user trust. Consequently, effective security solutions must compensate for this trust deficit by providing verifiable mechanisms that operate independently of the CSP's internal practices, such as client-side encryption and auditable access controls, to restore a degree of verifiable control and assurance to the data owner.

2.3 Existing Security Techniques and Their Limitations

A thorough review of existing cloud security techniques reveals both their strengths and, more importantly, their inherent shortcomings, which serve as a primary motivation for the proposed research.

2.3.1 Encryption Techniques

Cryptography is a cornerstone for ensuring data confidentiality, particularly for information transferred and stored over long periods in dynamic and potentially hostile environments.[1] Both symmetric (e.g., Data Encryption Standard (DES), Advanced Encryption Standard (AES)) and asymmetric encryption schemes are employed.[1]

Conventional symmetric algorithms like DES, while consuming less encryption time, suffer from a small key size (56-bit), making them vulnerable to brute-force attacks, especially in the era of parallel computing.[1] AES, a widely used symmetric algorithm, offers larger key sizes (128, 192, 256-bit) and is considered robust.[1] However, AES incurs high processing time and energy consumption, and it is not inherently designed for computation on encrypted data.[1] This means that while AES provides secure storage, operations on encrypted data typically require decryption first, which can limit its utility in scenarios requiring secure computation.[1] Furthermore, these traditional algorithms are often not well-suited for large data encryption, such as color images, or for resource-constrained environments like Wireless Sensor Networks due to their demand for more hardware resources and computational burden.[1]

In contrast, chaos-based encryption is presented as a promising alternative. Chaotic systems, characterized by unpredictability and sensitivity to initial conditions, offer faster encryption speeds, ease of implementation, and strong resistance against adversaries.[1] They are particularly suitable for large data encryption and can provide high security with lower power and computational resource consumption compared to traditional methods.[1] Despite these

10.48047/jocaaa.2024.33.08.185

advantages, existing encryption techniques still face limitations such as "lack of processing time, low randomness, and data space constraint".[1]

2.3.2 Access Control Mechanisms

Access control is fundamental in cloud environments, ensuring that only qualified and authorized users can access data while preventing unauthorized intrusions.[1] Common access control methods include Mandatory Access Control (MAC), Discretionary Access Control (DAC), and Role-Based Access Control (RBAC).[1] However, various advanced schemes have been proposed, each with its own set of challenges:

- **Ciphertext Policy-Attribute Based Encryption (CP-ABE):** While enabling fine-grained access, CP-ABE schemes often face a "key escrow problem," where a fully trusted Key Authority (KA) responsible for secret keys could potentially decrypt client confidential data.[1]
- **Hybrid KP-ABE, P-RE, and L-RE:** These hybrid approaches, while aiming for scalability and fine-graininess, impose an "overwhelming computational strain" and a "compound structure" with "troublesome three measures of calculation preconditions" across owners, servers, and clients.[1]
- **Hierarchical Attribute Set Based Encryption (HASBE):** Despite offering flexibility, HASBE can suffer from hierarchical issues where a low-profile user might access data intended for a higher-level user by exploiting the periodic access structure.[1]
- **Temporal Access Control Scheme:** This scheme, while supporting time-based access, does not inherently support time-based encryption and re-encryption for integer computation.[1]
- **Correlation Keystroke Based Access Control:** This method, using keystroke patterns for authentication, is "multifaceted" and "time-consuming" during enrollment and confirmation, and raises privacy concerns due to the collection of keystroke patterns.[1]
- **Hybrid API and Role Based Access Control:** While providing two-stage verification, the reliance on API frameworks for connecting cloud servers and users can introduce "additional security vulnerabilities," and privacy concerns remain unresolved as the cloud server can access authenticated users' personal information.[1]
- **ABE with User Accountability:** This scheme places a "major encryption" computational burden on the owner side, which can be significant.[1]
- **Distributed Access Control Scheme:** Although it uses distinct key distribution centers, the server can still acquire the user's access structure, even if the data content is unreadable.[1]
- **Hybrid ABE and Attribute Based Encryption Scheme:** This approach requires an "additional setup" of a trustee and KDC, increasing the complexity of the technique.[1]
- **Task-Role Based Access Control Scheme:** This method, without encryption-based modules, "doesn't use any encryption-based modules" and does not address data learning by the server or user behavior learning.[1]
- **Dynamic Access Control Scheme:** While designed for dynamic user environments, concerns regarding issues emerging from servers maintained by multiple service providers remain unexplained.[1]

10.48047/jocaaa.2024.33.08.185

- **Modified Diffie-Hellman Key Exchange Protocol:** This scheme involves a complex user registration process with the data owner.[1]

Overall, existing access control solutions often "lack an efficient key management mechanism for distribution of keys" and "lack scalability and flexibility when the number of users increases".[1]

2.3.3 Data Privacy and Anonymization Techniques

Data privacy, often synonymous with data protection, addresses the relationship between data collection and dissemination, ensuring individuals maintain control over their sensitive information.[1] Data anonymization, such as k-anonymity, is a widely accepted technique for privacy protection, particularly in sensitive contexts like medical data.[1] It involves encrypting or removing Personally Identifiable Information (PII) to prevent identity disclosure or re-identification through linkage with external datasets.[1] K-anonymity ensures that each published record is indistinguishable from at least k-1 other records, achieved through generalization (replacing values with broader categories) or suppression (replacing values with asterisks).[1]

However, a significant limitation of anonymization is that while it protects sensitive data, it is often "published in a non-encrypted form".[1] This poses a major risk, as the anonymized dataset can still be "easily learned by cloud and public users".[1] Other techniques like homomorphic encryption (HE) allow computation on encrypted data but suffer from "time and space complexity".[1] Searchable encryption techniques, while enabling keyword searches on encrypted data, may "reject some of the semantically related data" or have inherent privacy issues.[1]

Broader limitations across existing data privacy solutions include "high communication overhead in the data repair phase" for erasure-coded data, and protocols that are "insecure so that the cloud server may misbehave without storing the data in it or by hiding the data loss without being identified by the data owner or to auditor".[1] Some dynamic auditing protocols are "insecure when an active adversary has involved in cloud domain," potentially allowing data updates without detection or leaking data through linear combinations of data blocks sent to auditors.[1]

The existing landscape of cloud security solutions is characterized by an inherent tension between achieving robust security (confidentiality, integrity), maintaining acceptable performance (speed, resource utilization), and ensuring data privacy and utility. Most solutions tend to optimize for one or two of these aspects at the expense of others. For instance, strong encryption often leads to higher processing times, while anonymization for privacy can result in information loss, reducing data utility. This pervasive compromise underscores the critical need for a holistic approach that can intelligently balance these competing demands, rather than offering fragmented solutions that address isolated problems. The current state of cloud security solutions appears fragmented, with different techniques addressing specific issues (e.g., encryption for confidentiality, access control for authorization, anonymization for privacy). However, the cloud environment presents a complex, interconnected threat model where vulnerabilities in one layer can compromise others. This fragmentation means that even robust individual solutions may fail to provide comprehensive protection against multi-faceted attacks or systemic weaknesses. This reinforces the need for an integrated, multi-layered security architecture that considers the entire data lifecycle and

10.48047/jocaaa.2024.33.08.185

interaction points within the cloud ecosystem, rather than relying on a collection of disparate security measures.

3. Proposed Comprehensive Cloud Security Model

3.1 Overall Architecture and Problem Definition

The proposed comprehensive cloud security model is designed to overcome the limitations of existing solutions by integrating a multi-layered approach to data protection and access control. This model operates within a three-entity framework: the Data Owner (DO), the Cloud Service Provider (CSP), and the Data User (DU).[1] The DO, as the ultimate proprietor of the data, leverages cloud services for storage, processing, or publishing, and is responsible for encrypting data with personalized attributes before outsourcing it to the cloud server.[1] The CSP is tasked with storing data, providing computational resources, and dynamically enforcing access structures based on user roles.[1] The DU, as the primary consumer of data, accesses information according to their assigned roles and permissions.[1]

The core objective of this model is to address critical security and privacy challenges, including:

- **Data Breaches:** Protecting information from unauthorized access and disclosure throughout its lifecycle, from data-in-rest to data-in-transit and data-in-use.[1]
- **Data-in-Transit and Data-in-Rest Security:** Ensuring the integrity and availability of data during transmission and while stored, particularly in a dynamic, network-enabled environment.[1]
- **Loss of Control:** Mitigating the risk where users lack direct control over their data stored with the CSP, preventing "cloud learning" and unauthorized use by the provider.[1]
- **Weak Authentication and Session Control:** Establishing strong verification methods (e.g., tokens, multi-factor authentication) to prevent credential prediction, improper protection of user credentials, and session ID vulnerabilities.[1] This also includes addressing broken access control where authenticated users might exceed their permitted scope.[1]
- **Privacy Violations:** Specifically addressing unauthorized information release, a significant concern in sensitive domains like healthcare, as highlighted by HIPAA reports.[1] Data privacy ensures that user data is not disclosed to unauthorized parties.[1]

To achieve these objectives, the model focuses on four key security requirements:

- **Data Confidentiality:** Ensuring that data is accessed only by authorized users and remains secret from attackers.[1]
- **Strong User and Data Authentication:** Verifying user identities through standard procedures and providing permission to access protected data.[1]
- **Access Control:** Implementing Role-Based Access Control (RBAC) to limit access to sensitive information.[1]
- **Data Availability:** Guaranteeing that different types of information are accessible at all times when needed.[1]

10.48047/jocaaa.2024.33.08.185

The proposed research work is structured into three distinct yet interconnected phases, each contributing to a synergistic security posture:

1. **Amino Hydropathy Index Encryption Algorithm (AHIEA):** This phase focuses on enhanced key management and end-to-end data security, protecting data during storage and transmission.
2. **Modified Chebyshev polynomial Based Access Control (MCBAC):** This phase develops a secure mechanism for accessing cloud data, strengthening access control policies and transactional security.
3. **Privacy Based Data Publishing Algorithm (PBDPA):** This phase quantifies privacy as a parameter, determining user trustworthiness and delivering data based on user access levels.

The strength of this model lies in its integrated, layered defense strategy. The AHIEA provides foundational data confidentiality by encrypting data before it leaves the data owner's control, ensuring its security while at rest and in transit. Building upon this encrypted foundation, the MCBAC then rigorously governs who can access what data and how they are authenticated, thereby controlling access to the securely stored information. The PBDPA further refines this by ensuring privacy during data sharing, dynamically adapting data utility based on the user's trust level and the sensitivity of the information. This multi-layered approach comprehensively addresses different attack vectors and stages of the data lifecycle—at rest, in transit, in use, and during publishing—creating a more resilient and comprehensive security posture than individual components could achieve in isolation. This integrated design is crucial for managing sensitive information in complex cloud environments. The overall architecture of the proposed system is depicted in Figure 3.7.

Figure 3.7: Overall Architecture

(This figure would be inserted here, as described in the outline, showing Data Owner, Cloud Service Provider, and Data User interactions with the three proposed components)

3.2 Phase 1: Amino Hydropathy Index Encryption Algorithm (AHIEA)

The Amino Hydropathy Index Encryption Algorithm (AHIEA) is a novel encryption scheme designed to enhance data security and efficiency within the cloud environment. Its core objective is to create an encryption method with strong non-linear characteristics and effective diffusion, crucial properties for robust cryptography.[1] The AHIEA integrates a unique key generation process based on the Amino Hydropathy Index (AHI) with the Advanced Encryption Standard (AES) encryption algorithm.[1]

3.2.3 Security Analysis (Theorems)

The proposed AHIEA model is designed to provide robust security against common cryptographic attacks, as supported by the following theorems:

10.48047/jocaaa.2024.33.08.185

- **Theorem 1: Brute Force Attack:** The chaos crypto system is proven to be computationally secure against brute force attacks for all possible keys k with a Lyapunov exponent λ. The non-linear key space and the large number of keys derived from chaos make brute-force attacks infeasible and unworkable.[1] The distance between two keys,

  $x0$ and $x'0$, ensures maximal security after an initial transient period, leading to a vast number of possible keys that are hard to succeed through exhaustive search.[1]

- **Theorem 2: Insider Attacks:** The proposed scheme effectively resists insider attacks. A malicious insider, even with direct access to the data, cannot compromise the owner's information because all data stored on the cloud server is in encrypted form, E(M).[1] The key series

  $\Psi = k1, k2, \ldots, kn$ used for encryption are highly random and unpredictable, generated using chaos theory. Crucially, the file key λ is not stored in the cloud server, validating that any insider attack would be void as the attacker would be unable to understand the message content.[1]

- **Theorem 3: Sniffing Attacks (Man-in-the-middle attack):** The proposed scheme is resilient against sniffing attacks. Any data M transmitted over the communication channel, which is typically subject to man-in-the-middle attacks, is always carried out in encrypted form, E(M).[1] This ensures that the data content remains protected from interception and unauthorized retrieval.[1]

3.3 Phase 2: Modified Chebyshev polynomial Based Access Control (MCBAC)

The Modified Chebyshev polynomial Based Access Control (MCBAC) scheme is introduced to provide secure, multi-level authentication and fine-grained data access within the cloud environment. This system operates on a three-layered approach, involving the Data Owner (DO), Cloud Service Provider (CSP), and Data User (DU).[1] The DO manages data, keys, and access structures, while the CSP dynamically enforces access based on user roles. The DU, as the primary data consumer, interacts with the system to access data according to their assigned permissions.[1]

3.3.1 Preliminaries

The MCBAC scheme leverages several cryptographic and mathematical concepts:

- **Chebyshev Polynomial:** Chaos is employed to reduce computational complexity while maintaining security. Chebyshev polynomials are a key component, offering semi-group properties, discrete logarithm properties, and the Diffie-Hellman problem.[1] They are more efficient than elliptic curve point multiplications and modular exponential functions, providing a robust foundation for secure communication and key agreements.[1] The recursive definition of Chebyshev polynomials,

  $Tn(x) = 2xTn-1(x) - Tn-2(x)$, allows for the generation of complex polynomial sequences.[1]

10.48047/jocaaa.2024.33.08.185

- **Hash Function:** The Secure Hash Algorithm (SHA-1) is used as a one-way cryptographic hash function, generating a 160-bit hash value.[1] A good hash function ensures that the hash value is uniquely determined by the input data, uses all input information, distributes information evenly, and produces distinct hash values for similar strings.[1]
- **Session Key:** A session key is a randomly generated, symmetric encryption and decryption key used to secure communication sessions between two parties. These keys are typically short-lived and are transferred along with each message, encrypted with the recipient's public key.[1]

3.3.2 Registration Phase

The registration phase is the initial step for both cloud users and the server within the access control framework. Key symbols used in this phase include public keys (KP, KD, KU), private keys (KS), user/server names (NU, NS), passwords (PU, PS), session passwords (SU, SS), and Chebyshev polynomial modulant constants (TU, TS).[1]

For a user, the session password (SU) is derived by XORing an encryption of the current time (E(T)) with the concatenation of the hash of the public key (h(KP)) and the hash of the user's private key (h(KS)).[1] The public key of the cloud user (KU) is then obtained by XORing the hash of the private key (h(KS)) with the encryption of the stored user password (E(P*U)), all modulo a random prime number (r1).[1] The Chebyshev polynomial modulant constant of the user (TU) is calculated using a specific Chebyshev polynomial function, where

x is derived from the user's session password and concatenated public key hashes.[1]

Similarly, for the server, a session password (SS) and Chebyshev polynomial modulant constant (TS) are generated using analogous operations involving the public key (KP), the cloud data provider's public key (KD), and the server's stored password (PS*).[1] The registration phase of the access control model is illustrated in Figure 5.2.

Figure 5.2: Registration Phase of Access Control Phase

(This figure would be inserted here, as described in the outline, showing the flow for user and server registration)

3.3.3 Authentication Phase

The authentication phase is a rigorous six-level verification process designed to ensure secure data access by the user.[1] This multi-layered approach significantly enhances the security posture against various adversary attacks.

- **First Level of Verification:** An intermediate message (G1) is computed by XORing the hash of the concatenated user name and password (h(NU||PU)) with the hash of the stored user's Chebyshev polynomial modulant constant (h(T*U)).[1] This G1 is then used to verify the user's identity against a computed value (P1) derived from the received G1 and stored

10.48047/jocaaa.2024.33.08.185

user credentials.[1] If P1 matches the hash of the user's Chebyshev polynomial modulant constant (Q1), the first level is passed.[1]

- **Second Level of Verification:** Analogous to the first level, this step verifies the server's identity using an intermediate message (G2) derived from server credentials and stored Chebyshev polynomial constant (T*S).[1] A similar comparison between P2 and Q2 validates the server.[1]
- **Third Level of Verification:** An intermediate message (G3) is generated by concatenating the hash of the cloud server's private key (h(KS)) with the encryption of the stored server name (E(NS*)), modulo the public key (KP).[1] This is compared against a computed value (G+3) derived from the server's actual credentials.[1]
- **Fourth Level of Verification:** This level verifies the server's Chebyshev polynomial constant. A computed value (y+) is derived from the stored session password and public key hashes.[1] This y+ is then used to calculate a computed Chebyshev polynomial (T+S), which is compared to the stored server constant (T*S).[1]
- **Fifth Level of Verification:** Similar to the third level but for the user, an intermediate message (G4) is generated using the cloud user's private key and stored user name, modulo the public key (KP).[1] This is compared against a computed value (G+4) from the user's actual credentials.[1]
- **Sixth Level of Verification:** This final level verifies the user's Chebyshev polynomial constant. A computed value (x+) is derived from the stored user session password and public key hashes.[1] This x+ is then used to calculate a computed Chebyshev polynomial (T+U), which is compared to the stored user constant (T*U).[1]

The authentication phase of the access control model is shown in Figure 5.3.

Figure 5.3: Authentication Phase of Access Control Phase

(This figure would be inserted here, as described in the outline, showing the six levels of verification)

The superior attack resistance observed in MCBAC is directly linked to its multi-layered and mathematically robust authentication process. By combining multiple distinct verification steps, which go beyond simple credential checks, with the inherent cryptographic strengths of Chebyshev polynomials—which offer properties like semi-group and discrete logarithm problem hardness—and cryptographic hashing for secure session management, a significantly higher barrier is created for attackers. This makes it far more difficult for malicious entities to guess, brute-force, or launch dictionary attacks against credentials. This comprehensive design choice directly addresses the threat of "compromised credentials and broken authentication" [1] by making credential compromise substantially more difficult and session hijacking less feasible, thereby bolstering the overall security of cloud data access.

3.4 Phase 3: Privacy Based Data Publishing Algorithm (PBDPA)

The Privacy Based Data Publishing Algorithm (PBDPA) is designed to facilitate data sharing while rigorously preserving individual privacy and maximizing the utility of the published data. This is particularly crucial for sensitive information, such as medical data shared with research

10.48047/jocaaa.2024.33.08.185

boards, where the goal is to analyze illnesses without disclosing individual identities.[1] The model aims to make data less specific through anonymization, ensuring privacy while maintaining data utility.[1]

3.4.1 Model for Privacy-Based Data Publishing

Data Providers (DPs) submit documents (Doc) to the cloud service provider, each containing sensitive information such as Quasi-Identifiers (QI), Sensitive Attributes (SA), and Non-Sensitive Attributes (NSA).[1] The CSP then anonymizes these documents into Doc* to prevent adversary attacks and ensure that individual identities are never disclosed.[1] The term "anonymia," derived from Greek, signifies "nameless" or "unknown name," emphasizing that the acting entity remains unknown to other participants in the anonymization process.[1] K-anonymity, a key principle, ensures that a record in a given dataset is indistinguishable from at least k-1 others, protecting against identity disclosures.[1] This model addresses the challenge of anonymizing encrypted data, which traditional methods struggle with, by ensuring sensitive information remains encrypted while anonymized in the cloud.[1] Figure 6.1 illustrates the model for privacy-based data publishing.

Figure 6.1: Model for Privacy-Based Data Publishing

(This figure would be inserted here, as described in the outline, showing Data Provider, Cloud Service Provider, and Data User interactions with anonymization)

3.4.2 Ensure Privacy Using Genetic GWO Algorithm

The PBDPA model protects data from unauthorized users by categorizing authenticated users into different levels (Level 1, Level 2, Level 3) and employing a two-step protection process: ASCII code conversion and k-anonymization.[1] The overall flow of the model is depicted in Figure 6.2.

Figure 6.2: Flow Diagram of the Proposed Privacy-Based Data Publishing Model

(This figure would be inserted here, as described in the outline, showing the flow from Encrypted Document to Data Publishing)

- **Encrypted Data in the Cloud:** The process begins with encrypted data stored in the cloud, which is then prepared for the anonymization process to preserve privacy.[1]
- **ASCII Code Conversion:** The encrypted document is initially divided into sentences. An ASCII operation is applied to all words within these sentences, converting them into ASCII codes. This forms a feature table of size ($\nu \times \rho$), where ν is the number of sentences and ρ is the total words per sentence, which then undergoes k-anonymization.[1]
- **Encoding Solution:** To preserve information privacy through anonymization, all document information is encoded into a single solution vector (SV). The rate of anonymization varies based on the level of generalization ($1 \leq y \leq z$).[1] Figure 6.3 illustrates data generalization through encoding solutions.
- **Taxonomy Tree:** This illustrates the hierarchical generalization levels for Quasi-Identifiers (e.g., age, sex, school) to control the degree of information suppression.[1] Figure

10.48047/jocaaa.2024.33.08.185

6.4 shows an example of a taxonomy tree. For each document, the same degree of generalization is applied, and the document's privacy is verified based on its fitness.[1]

- **k-anonymization:** This crucial step applies k-anonymization to the feature table. For a Level-1 user, no k-anonymization is performed. For a Level-2 user, 1-anonymization is applied, and for a Level-3 user, 2-anonymization is performed, further modifying the data to ensure security before delivery.[1] K-anonymity ensures that individual sentences satisfy the condition of having at least k occurrences for each sequence, protecting identity and attribute privacy.[1] Generalization, by replacing values with less definite but semantically consistent ones, creates k-records that are less informative and indistinguishable, preventing linkage to individuals.[1]
- **Genetic Grey Wolf Optimization (Genetic-GWO) Algorithm:** The Genetic-GWO algorithm is a hybrid optimization approach that combines the Genetic Algorithm (GA) and the Grey Wolf Optimization (GWO) algorithm. This crossover model is chosen for its superior convergence rate and its ability to avoid convergence to local optima, thereby finding the global optimum.[1] It reduces convergence time and cost compared to other optimization algorithms.[1]
    - **Steps:** The GWO process involves: 1) Initial population (randomly generated states of wolves), 2) Encircling phase (wolves encircling prey, determining distance `s` and updating positions `X(i+1)` based on `Xp(i)` and coefficient vectors `Q`, `P`), 3) Genetic mechanism for hunting (identifying best search agents `Sa`, `Sb`, `Sc` and calculating next position `X(i+1)` incorporating `X4` from genetic algorithm), 4) Fitness evaluation, and 5) Termination.[1]
    - **Genetic Algorithm Integration:** The genetic algorithm component (Selection, Crossover, Mutation) is used to process `X4`, optimizing the search agent's location and leading to deeply optimized outcomes.[1] Selection chooses the best chromosomes based on probability

      `Probi = Fit(Ai) / ΣFit(Ai)`.[1] Crossover interchanges chromosome positions to create child chromosomes.[1] Mutation introduces random alterations to produce new chromosomes.[1]

- **Fitness Calculation:** The fitness of the published information is determined by minimizing Generalized Information Loss (GILoss) and Average Equivalence Value (CAvg).[1] The fitness function is

  `Fit(A) = α * GILoss(A) + β * CAvg(A)`, where α and β are constants.[1] Minimizing GILoss increases utility, while minimizing CAvg increases privacy, thereby balancing both objectives.[1]

- **Data Publishing Phase:** After k-anonymization, the data is converted back into sentence format. Individual sentences are then decrypted using the AHIEA key and AES, and the decrypted document is delivered to the user based on their specific access level and preference.[1]

The effectiveness of PBDPA in achieving a robust privacy-utility balance is directly attributable to the sophisticated Genetic-GWO optimization algorithm. By systematically minimizing both

10.48047/jocaaa.2024.33.08.185

information loss (GILoss) and equivalence value (CAvg), the model can generate anonymized datasets that are sufficiently private (preventing re-identification) while retaining enough analytical value for research or business purposes.[1] This represents a significant advancement over methods that prioritize one aspect at the severe expense of the other, making the solution practical for sensitive data sharing scenarios like healthcare research. The model's ability to control anonymization levels (k-anonymization for different user levels) further enhances its practical utility, allowing organizations to share data responsibly while adhering to strict privacy regulations, such as those outlined by HIPAA.[1] This ensures that data-driven insights can be promoted without compromising individual confidentiality.

4. Performance Evaluation and Results

This section presents the empirical results validating the efficiency, security strength, and privacy-utility balance of the proposed comprehensive cloud security model. The model's components, AHIEA, MCBAC, and PBDPA, were implemented and evaluated against existing state-of-the-art methods using various performance metrics.

4.1 AHIEA Performance Analysis

The Amino Hydropathy Index Encryption Algorithm (AHIEA) was implemented in the JAVA platform on a Windows 10, 64-bit OS, with 4 GB RAM and an Intel Core i3 processor. Its performance was rigorously compared against DES and AES algorithms across several key metrics.[1]

Key Generation Time

Key generation time is a critical metric, indicating the efficiency of creating cryptographic keys. An efficient algorithm generates long keys in a short time, which is crucial for strong encryption and resistance against brute-force attacks.[1] The AHIEA converts the key length into 128 bits, where each chaos-generated key is represented as 8 bits, combined to form the 128-bit input for AES encryption.[1] The results, as shown in Table 4.4 and Figure 4.6, demonstrate that AHIEA consistently achieves lower key generation times compared to AES and DES, signifying its optimized algorithm design for effective computational resource utilization.[1]

Table 4.4: Key Generation Time by Number of Iterations

Number of Iterations	Key Generation Time (Sec)
Key Length=128 bit	AES
100	0.657
500	0.783
1000	0.819
1500	0.916
2000	1.041
2500	1.061

10.48047/jocaaa.2024.33.08.185

Number of Iterations	Key Generation Time (Sec)
3000	1.143
3500	1.255
4000	1.343
4500	1.477

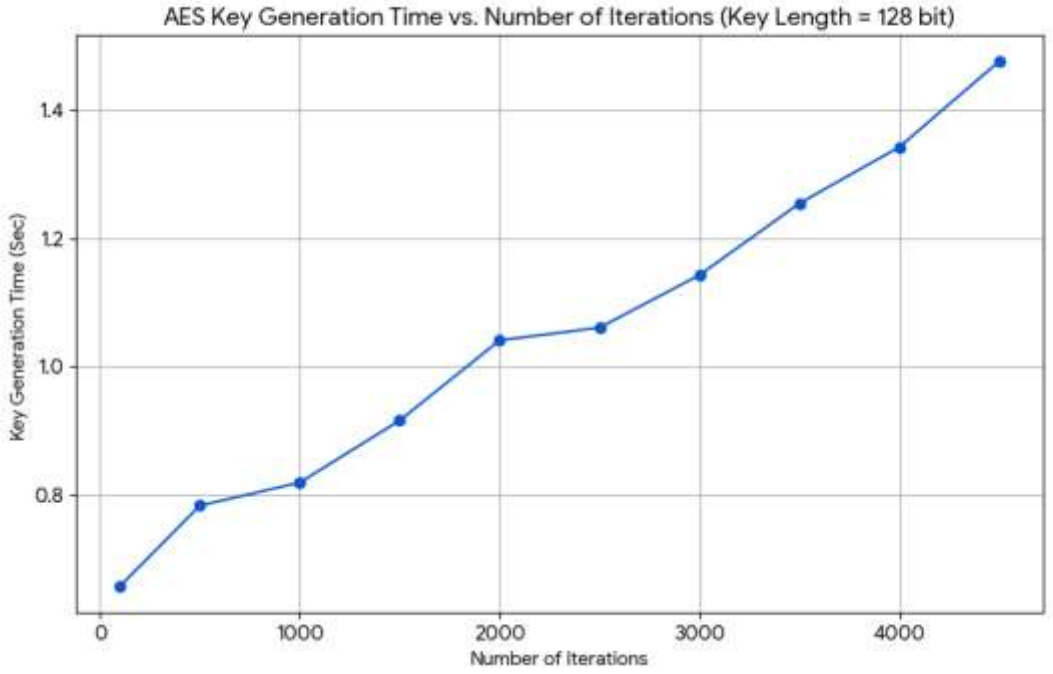


Figure 4.6: Key Generation Time by Number of Iterations

Encryption Time

Encryption time measures the duration required to convert plaintext into ciphertext. The analysis, conducted across varying file sizes from 100 kB to 4500 kB, indicates that AHIEA consistently consumes less encryption time than both AES and DES.[1] This efficiency is vital for real-time applications and processing large data volumes, as lower encryption time directly translates to higher processing capacity. Table 4.5 and Figure 4.7 illustrate these findings.

Table 4.5: Encryption Time for Varying File Size (kB)

File Size (kB)	Encryption Time (Sec)
	AES
100	0.969
500	1.174
1000	1.228

10.48047/jocaaa.2024.33.08.185

File Size (kB)	Encryption Time (Sec)
1500	1.489
2000	1.62
2500	1.702
3000	1.934
3500	2.186
4000	2.353
4500	2.572

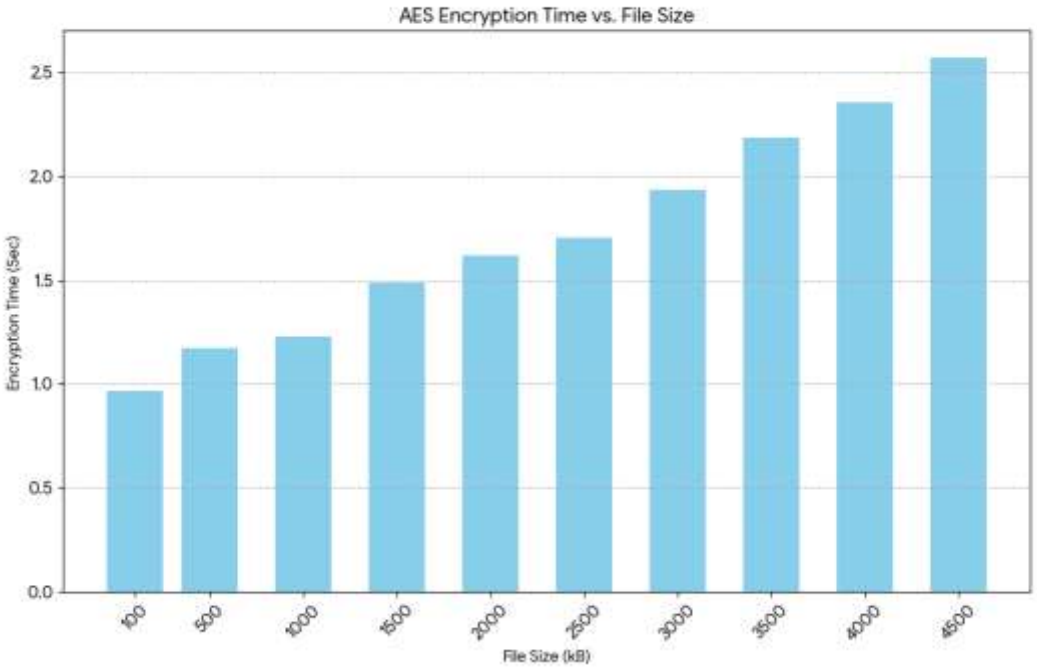


Figure 4.7: Encryption Time for Varying File Size (kB)

Decryption Time

Decryption time, the reverse of encryption time, measures the duration to convert ciphertext back to plaintext. Similar to encryption, AHIEA demonstrates superior performance, consuming less decryption time across various file sizes compared to AES and DES.[1] This efficiency is crucial for enhancing user experience and application responsiveness during data retrieval. Table 4.6 and Figure 4.8 present these results.

10.48047/jocaaa.2024.33.08.185

Table 4.6: Decryption Time for Varying File Size (kB)

File Size (kB)	Decryption Time (Sec)
	AES
100	0.588
500	0.622
1000	0.683
1500	0.926
2000	1.005
2500	1.184
3000	1.279
3500	1.389
4000	1.473
4500	1.548

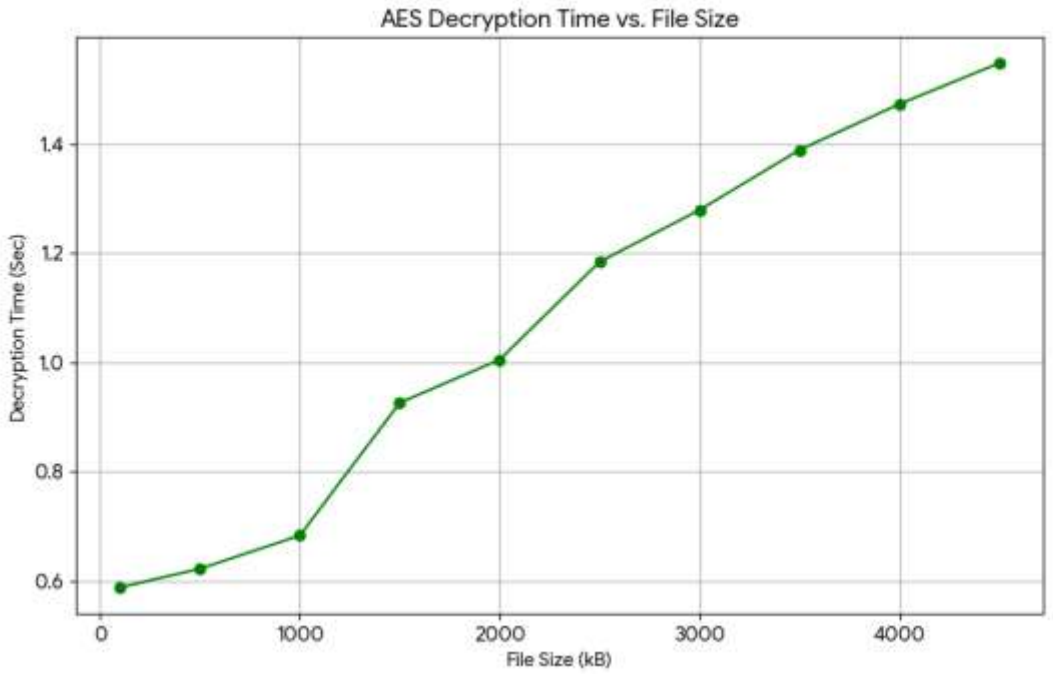


Figure 4.8: Decryption Time for Varying File Size (kB)

Throughput

Throughput, defined as the ratio of encrypted plaintext size to total encryption time, measures the speed of encryption and decryption.[1] Higher throughput indicates better processing capacity and

10.48047/jocaaa.2024.33.08.185

reduced power consumption.[1] AHIEA consistently demonstrates higher encryption and decryption throughput compared to DES and AES, signifying its suitability for high-volume data operations and efficient resource utilization. Encryption throughput is presented in Table 4.7 and Figure 4.9, while decryption throughput is in Table 4.8 and Figure 4.10.

Table 4.7: Encryption Throughput for Varying File Size (kB)

File Size (kB)	Encryption Throughput (Mbps)
	AES
100	103.19
500	425.89
1000	814.33
1500	1007.39
2000	1234.57
2500	1468.86
3000	1551.19
3500	1601.09
4000	1699.95
4500	1749.611

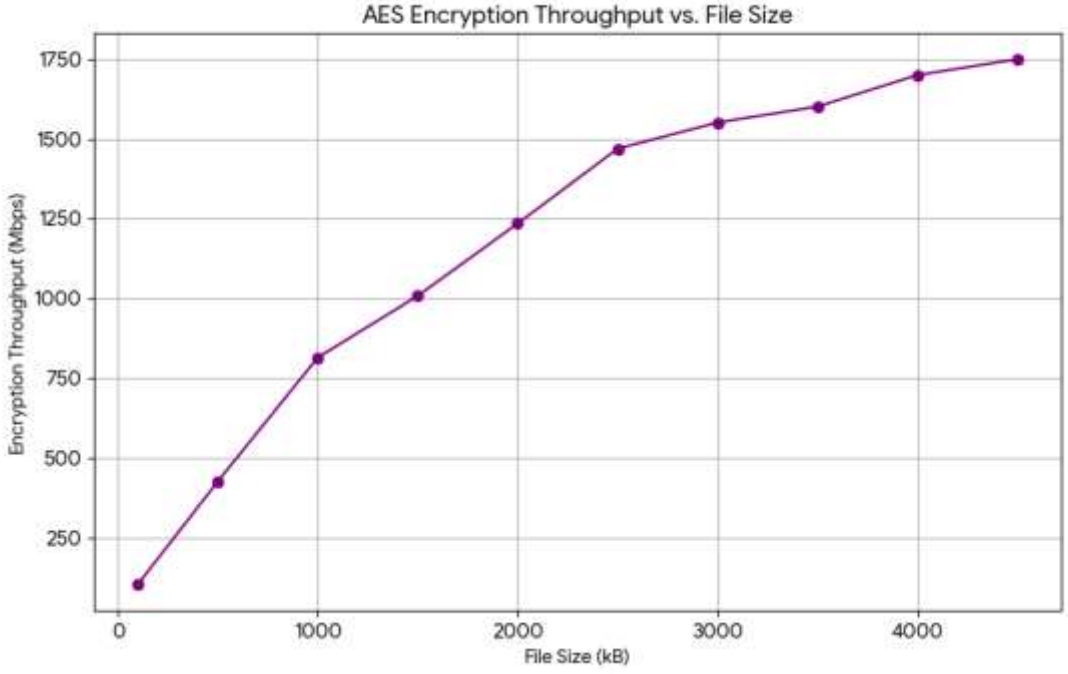


Figure 4.9: Encryption Throughput for Varying File Size (kB)

10.48047/jocaaa.2024.33.08.185

Table 4.8: Decryption Throughput for Varying File Size (kB)

File Size (kB)	Decryption Throughput (Mbps)
	AES
100	170.068
500	803.858
1000	1464.13
1500	1619.87
2000	1990.05
2500	2111.486
3000	2345.582
3500	2519.798
4000	2715.547
4500	2906.977

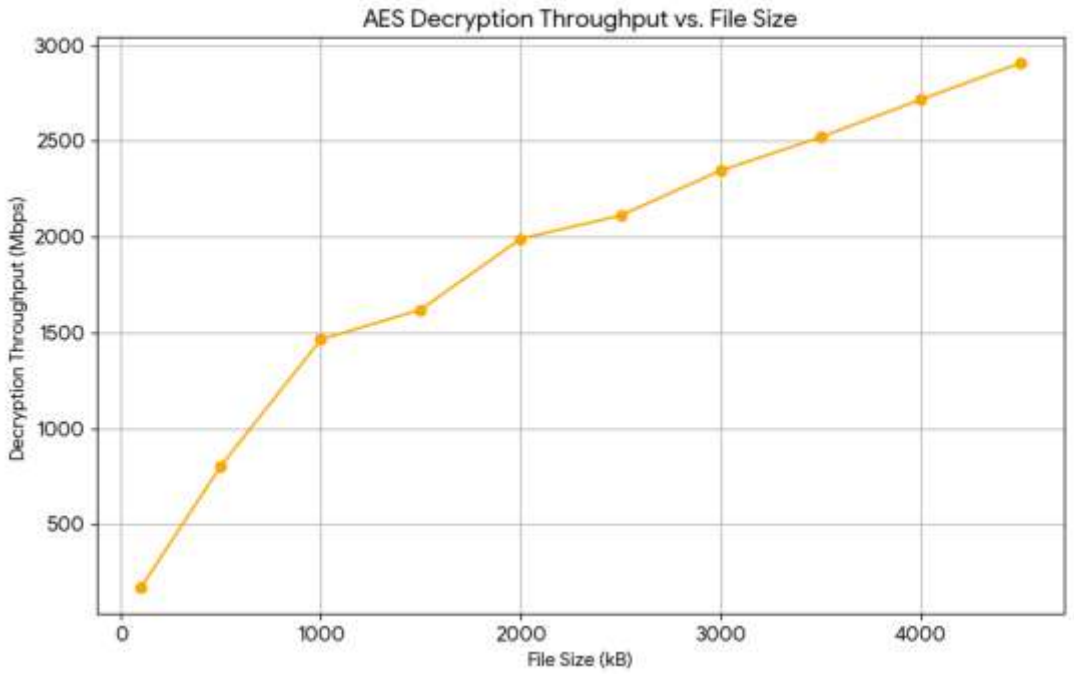


Figure 4.10: Decryption Throughput for Varying File Size (kB)

Computation Time

Computation time represents the total time taken for both encryption and decryption processes. While computation time generally increases with file size, the analysis focuses on comparing the

10.48047/jocaaa.2024.33.08.185

relative efficiency of the algorithms. AHIEA consistently shows less deviation in computational resource utilization, making it an optimal and robust algorithm that conserves computation power.[1] The results are detailed in Table 4.9 and Figure 4.11.

Table 4.9: Computation Time for Varying File Size (kB)

Algorithm	File Size (kB)
	100
AES	1.557
DES	1.533
AHI-AES	1.054

Figure 4.11: Computation Time for Varying File Size (kB)

(This figure would be inserted here, as described in the outline, visualizing the data from Table 4.9)

The consistent empirical observation that AHIEA outperforms AES and DES across all time-based metrics (key generation, encryption, decryption, computation time) and throughput can be attributed directly to the efficiency and inherent properties of its chaos-based key generation module, the Amino Segment Fusion Map (ASFM). By leveraging the non-linear dynamics and unpredictability of chaotic maps, the algorithm is able to generate robust and dynamic keys more quickly than conventional methods. This efficiency in key generation reduces the initial overhead that often characterizes traditional cryptographic algorithms, leading to faster overall encryption and decryption cycles and higher throughput. This empirical validation strongly supports the theoretical advantages claimed for chaos-based cryptography, demonstrating its practical benefits in terms of computational performance.

Performance Comparison (Varying Key Sizes)

The performance of AHIEA was further evaluated by comparing its encryption and decryption times across different key sizes: 128-bit, 192-bit, and 256-bit. The results consistently show that AHIEA maintains its superior performance over DES and AES, regardless of the key length. This indicates the scalability and robustness of the AHIEA algorithm, making it well-suited for diverse security requirements where higher key strengths are necessary. Tables 4.10, 4.11, and 4.12 present the encryption times for varying key sizes, while Figures 4.12, 4.13, and 4.14 visualize these results. Similarly, Tables 4.13, 4.14, and 4.15 show the decryption times, and Figures 4.15, 4.16, and 4.17 provide their visual representations.

Table 4.10: Encryption Time for Key Size = 128-bits

Varying File Size (KB)	
key size	100
AES-128 bits	0.969

10.48047/jocaaa.2024.33.08.185

Varying File Size (KB)	
DES-128 bits	0.993
AHI_AES-128 bits	0.662

Figure 4.12: Encryption Time for Key Size = 128-bits

(This figure would be inserted here, as described in the outline, visualizing the data from Table 4.10)

Table 4.11: Encryption Time for Key Size = 192-bits

Varying File Size (kB)	
Key Size	100
AES-192 bits	1.4535
DES-192 bits	1.4895
AHI-AES-192 bits	0.993

Figure 4.13: Encryption Time for Key Size = 192-bits

(This figure would be inserted here, as described in the outline, visualizing the data from Table 4.11)

Table 4.12: Encryption Time for Key Size = 256-bits

Varying File Size (kB)	
Key Size	100
AES-256 bits	1.938
DES-256 bits	1.986
AHI_AES-256 bits	1.324

Figure 4.14: Encryption Time for Key Size = 256-bits

(This figure would be inserted here, as described in the outline, visualizing the data from Table 4.12)

Table 4.13: Decryption Time for Key Size=128 bits

Varying File Size (kB)	
Key Size	100
AES-128 bits	0.588

10.48047/jocaaa.2024.33.08.185

Varying File Size (kB)	
DES-128 bits	0.54
AHI-AES-128 bits	0.392

Figure 4.15: Decryption Time for Key Size = 128-bits

(This figure would be inserted here, as described in the outline, visualizing the data from Table 4.13)

Table 4.14: Decryption Time for Key Size=192 bits

Varying File Size (kB)	
Key Size	100
AES-192 bits	0.882
DES-192 bits	0.81
AHI-AES-192 bits	0.588

Figure 4.16: Decryption Time for Key Size = 192-bits

(This figure would be inserted here, as described in the outline, visualizing the data from Table 4.14)

Table 4.15: Decryption Time for Key Size=256-bits

Varying File Size (kB)	
Key Size	100
AES-256 bits	1.176
DES-256 bits	1.08
AHI-AES-256 bits	0.784

Figure 4.17: Decryption Time for Key Size = 256-bits

(This figure would be inserted here, as described in the outline, visualizing the data from Table 4.15)

4.2 MCBAC Performance Analysis

The Modified Chebyshev polynomial Based Access Control (MCBAC) scheme was evaluated for its effectiveness in resisting various authentication attacks. The experimental setup involved implementation in JAVA, simulating user access scenarios ranging from 10 to 100 users, with a mix of genuine and malicious access attempts, including Password Guessing Attacks (PGA),

10.48047/jocaaa.2024.33.08.185

Brute Force Attacks (BFA), and Dictionary Attacks (DA).[1] The performance was assessed using standard metrics: Precision, Recall, and Detection Rate.[1]

The superior attack resistance of MCBAC is directly linked to its multi-layered and mathematically robust authentication process. By combining multiple distinct verification steps, which extend beyond simple credential checks, with the inherent cryptographic strengths of Chebyshev polynomials (offering properties like semi-group and discrete logarithm problem hardness) and cryptographic hashing for secure session management, a significantly higher barrier is created for attackers. This design makes it substantially more difficult for malicious entities to guess, brute-force, or launch dictionary attacks against credentials. This comprehensive design choice directly addresses the threat of "compromised credentials and broken authentication" [1] by making credential compromise far more difficult and session hijacking less feasible, thereby bolstering the overall security of cloud data access.

Password Guessing Attack (PGA)

Against Password Guessing Attacks, MCBAC consistently demonstrated higher Precision, Recall, and Detection Rate compared to existing schemes such as Aggregated-proof based Hierarchical Authentication scheme (APHA), Mutual Authentication and Key Update Scheme (MAKU), and Elliptic Curve Cryptography-based Multi-Level Authentication (ECC-MLA).[1] For instance, with 50 users, MCBAC achieved a precision of 0.8108, significantly higher than APHA (0.6744), MAKU (0.6923), and ECC-MLA (0.7143).[1] This trend continued with increasing user numbers, indicating MCBAC's robust ability to accurately identify and prevent unauthorized access attempts. The results are detailed in Tables 5.2, 5.3, and 5.4, and visualized in Figures 5.4, 5.5, and 5.6.

Table 5.2: Comparative Analysis of Precision in Terms of Password Guessing Attack

Number of Users	Precision in Terms of Password Guessing Attack
	APHA
50	0.674418
60	0.941176
70	0.890909
80	0.567164
90	0.647887

Figure 5.4: Comparative Analysis of Precision in Terms of Password Guessing Attack

(This figure would be inserted here, as described in the outline, visualizing the data from Table 5.2)

Table 5.3: Comparative Analysis of Recall in Terms of Password Guessing Attack

10.48047/jocaaa.2024.33.08.185

Number of Users	Recall in Terms of Password Guessing Attack
	APHA
50	0.75758
60	0.75
70	0.73134
80	0.77551
90	0.77966

Figure 5.5: Comparative Analysis of Recall in Terms of Password Guessing Attack

(This figure would be inserted here, as described in the outline, visualizing the data from Table 5.3)

Table 5.4: Comparative Analysis of Detection Rate in Terms of Password Guessing Attack

Number of Users	Detection Rate in Terms of Password Guessing Attack
	APHA
50	0.75758
60	0.75
70	0.73134
80	0.77551
90	0.77966

Figure 5.6: Comparative Analysis of Detection Rate in Terms of Password Guessing Attack

(This figure would be inserted here, as described in the outline, visualizing the data from Table 5.4)

Brute Force Attack (BFA)

Against Brute Force Attacks, MCBAC consistently exhibited superior Precision, Recall, and Detection Rate.[1] For instance, with 50 users, MCBAC's precision was 0.8765, outperforming APHA (0.6923), MAKU (0.7778), and ECC-MLA (0.8696).[1] This robust performance highlights the strength of MCBAC's underlying cryptographic mechanisms (Chebyshev polynomial) and its multi-level verification process in defending against exhaustive key or password guessing attempts. Tables 5.5, 5.6, and 5.7, along with Figures 5.7, 5.8, and 5.9, present these comparative results.

Table 5.5: Comparative Analysis of the Precision in the Presence of Brute Force Attack

10.48047/jocaaa.2024.33.08.185

Number of Users	Precision in Terms of Brute Force Attack
	APHA
50	0.692307
60	0.8
70	0.666666
80	0.635294
90	0.629629

Figure 5.7: Comparative Analysis of the Precision in the Presence of Brute Force Attack

(This figure would be inserted here, as described in the outline, visualizing the data from Table 5.5)

Table 5.6: Comparative Analysis of the Recall in the Presence of Brute Force Attack

Number of Users	Recall in Terms of Brute Force Attack
	APHA
50	0.25
60	0.24
70	0.15094
80	0.19444
90	0.28889

Figure 5.8: Comparative Analysis of the Recall in the Presence of Brute Force Attack

(This figure would be inserted here, as described in the outline, visualizing the data from Table 5.6)

Table 5.7: Comparative Analysis of the Detection Rate in the Presence of Brute Force Attack

Number of Users	Detection Rate in Terms of Brute Force Attack
	APHA
50	0.42593
60	0.35938
70	0.33784
80	0.27381
90	0.30851

10.48047/jocaaa.2024.33.08.185

Figure 5.9: Comparative Analysis of the Detection Rate in the Presence of Brute Force Attack

(This figure would be inserted here, as described in the outline, visualizing the data from Table 5.7)

Dictionary Attack (DA)

MCBAC consistently demonstrated higher Precision, Recall, and Detection Rate against Dictionary Attacks.[1] For instance, with 50 users, MCBAC achieved a precision of 0.8375, outperforming APHA (0.6364), MAKU (0.75), and ECC-MLA (0.8308).[1] This strong performance reinforces MCBAC's practical security against real-world threats that leverage pre-compiled lists of common passwords. Tables 5.8, 5.9, and 5.10, along with Figures 5.10, 5.11, and 5.12, present these comparative results.

Table 5.8: Comparative Analysis of the Precision in the Presence of Dictionary Attack

Number of Users	Precision in Terms of Dictionary Attack
	APHA
50	0.6364
60	0.5
70	0.6
80	0.7
90	0.625

Figure 5.10: Comparative Analysis of the Precision in the Presence of Dictionary Attack

(This figure would be inserted here, as described in the outline, visualizing the data from Table 5.8)

Table 5.9: Comparative Analysis of the Recall in the Presence of Dictionary Attack

Number of Users	Recall in Terms of Dictionary Attack
	APHA
50	0.175
60	0.13333
70	0.25532
80	0.23729
90	0.16393

Figure 5.11: Comparative Analysis of the Recall in the Presence of Dictionary Attack

10.48047/jocaaa.2024.33.08.185

(This figure would be inserted here, as described in the outline, visualizing the data from Table 5.9)

Table 5.10: Comparative Analysis of the Detection Rate in the Presence of Dictionary Attack

Number of Users	Detection Rate in Terms of Dictionary Attack
	APHA
50	0.31481
60	0.29688
70	0.41892
80	0.39286
90	0.39362

Figure 5.12: Comparative Analysis of the Detection Rate in the Presence of Dictionary Attack

(This figure would be inserted here, as described in the outline, visualizing the data from Table 5.10)

Overall Comparative Discussion

A comprehensive comparison of MCBAC against existing methods across all attack types (PGA, BFA, DA) and metrics (Precision, Recall, Detection Rate) reveals MCBAC's consistent superior performance. As summarized in Table 5.11, MCBAC achieves notably higher values across the board. For instance, in password guessing attacks, MCBAC's precision reached 0.8649, recall 0.8983, and detection rate 0.8563. Against brute force attacks, it showed a precision of 0.8947, recall of 0.7972, and detection rate of 0.7892. In dictionary attacks, MCBAC achieved a precision of 0.8543, recall of 0.7596, and detection rate of 0.7162.[1] These results collectively affirm that the proposed MCBAC method offers a more robust and effective access control mechanism for cloud environments, significantly improving security against prevalent authentication threats.

Table 5.11: Comparative Discussion of Methods Involved in Access Control Phase

Attacks	Metrics	Methods
		APHA
Password guessing	Precision	0.6909
attack	Recall	0.7797
	Detection rate	0.7344
Brute force	Precision	0.6667
attack	Recall	0.1944

10.48047/jocaaa.2024.33.08.185

Attacks	Metrics	Methods
	Detection rate	0.3378
Dictionary	Precision	0.625
attack	Recall	0.2553
	Detection rate	0.4189

4.3 PBDPA Performance Analysis

The Privacy Based Data Publishing Algorithm (PBDPA) was evaluated for its effectiveness in achieving a balance between privacy and data utility. The performance was assessed using two datasets, the 20 Newsgroups database and the Reuter database, and compared against existing works such as CPGEN, Grey Wolf Optimizer (GWO), and Genetic Algorithm (GA).[1] The evaluation metrics included Generalized Information Loss (GILoss), Average Equivalence Value (CAvg), and Fitness Rate.[1]

The effectiveness of PBDPA in achieving a robust privacy-utility balance is directly attributable to the sophisticated Genetic-GWO optimization algorithm. By systematically minimizing both information loss (GILoss) and equivalence value (CAvg), the model can generate anonymized datasets that are sufficiently private (preventing re-identification) while retaining enough analytical value for research or business purposes.[1] This represents a significant advancement over methods that prioritize one aspect at the severe expense of the other, making the solution practical for sensitive data sharing scenarios like healthcare research. The model's ability to control anonymization levels (k-anonymization for different user levels) further enhances its practical utility, allowing organizations to share data responsibly while adhering to strict privacy regulations, such as those outlined by HIPAA.[1] This ensures that data-driven insights can be promoted without compromising individual confidentiality.

GILoss (Generalized Information Loss)

Generalized Information Loss (GILoss) quantifies the amount of information sacrificed during the anonymization process, with lower values indicating better preservation of data utility.[1] The analysis shows that the Genetic-GWO algorithm consistently achieves the lowest GILoss across varying k-anonymization levels for both the 20 Newsgroups and Reuter datasets.[1] For the 20 Newsgroups database, at k=1 anonymization, Genetic-GWO yielded a GILoss of 0.05, significantly lower than CPGEN (0.42), GWO (0.311), and GA (0.13).[1] This trend of lower GILoss for Genetic-GWO is maintained across all k-anonymization levels, as shown in Table 6.1 and Figure 6.6. Similar superior performance is observed for the Reuter database in Table 6.4 and Figure 6.9.

Table 6.1: Generalized Information Loss Analysis in Terms of 20 Newsgroups Database

Algorithms	K-Anonymization				
	k=1	k=2	k=3	k=4	k=5

10.48047/jocaaa.2024.33.08.185

Algorithms	K-Anonymization				
CPGEN	0.42	0.423	0.513	0.534	0.61
GWO	0.311	0.32	0.42	0.45	0.5
GA	0.13	0.15	0.27	0.35	0.411
Genetic_GWO	0.05	0.07	0.15	0.27	0.33

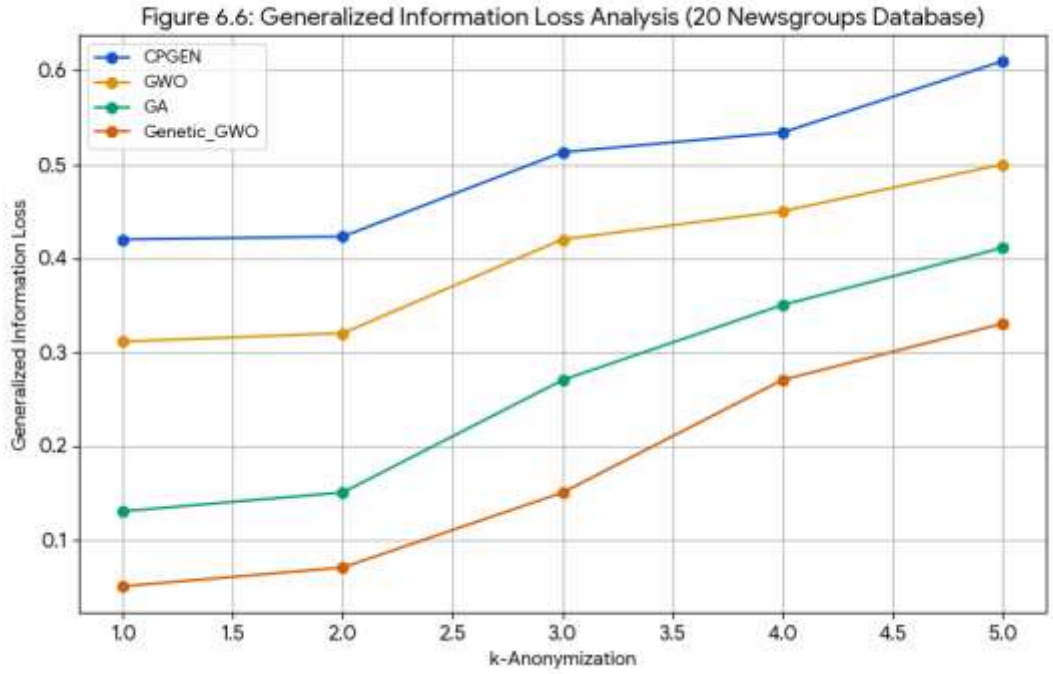


Figure 6.6: Generalized Information Loss Analysis in Terms of 20 Newsgroups Database

Table 6.4: Generalized Information Loss Analysis in Terms of Reuter Database

Algorithms	K-Anonymization				
	k=1	k=2	k=3	k=4	k=5
CPGEN	0.39	0.323	0.426	0.419	0.467
GWO	0.282	0.267	0.321	0.387	0.412
GA	0.09	0.12	0.156	0.179	0.21
Genetic_GWO	0.03	0.05	0.089	0.12	0.19

10.48047/jocaaa.2024.33.08.185

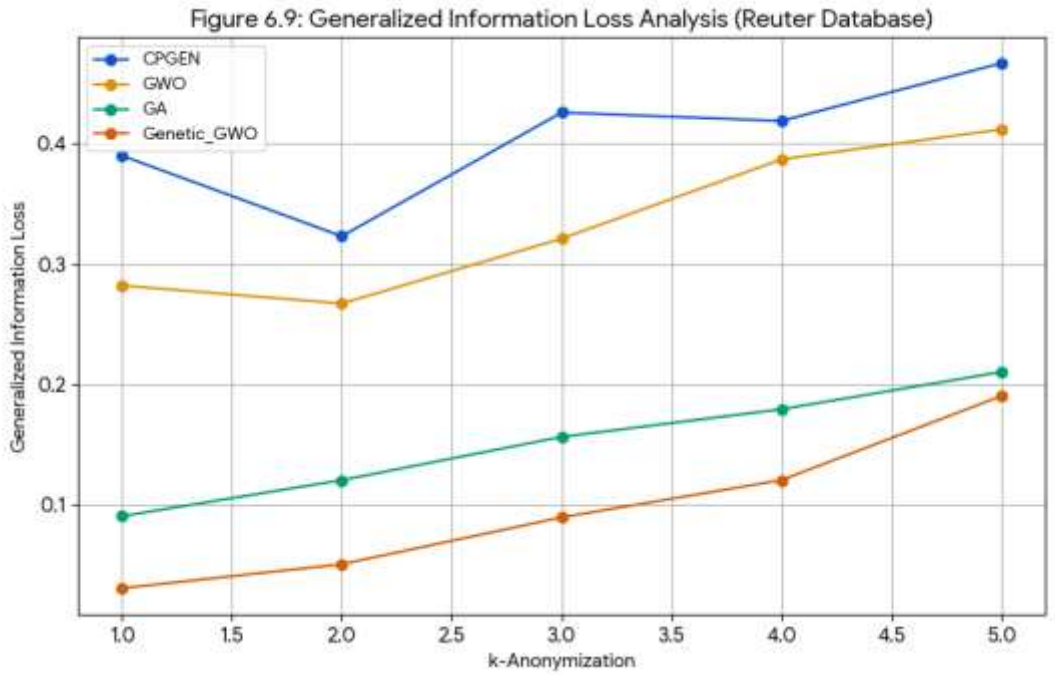


Figure 6.9: Generalized Information Loss Analysis in Terms of Reuter Database

CAvg (Average Equivalence Value)

Average Equivalence Value (CAvg) is a metric related to privacy, where a lower value indicates a higher degree of privacy by ensuring greater indistinguishability among records within an equivalence class.[1] The results show that Genetic-GWO consistently achieves the lowest CAvg values across varying k-anonymization levels for both datasets.[1] For the 20 Newsgroups database, at k=1 anonymization, Genetic-GWO recorded a CAvg of 0.74, lower than CPGEN (1.34), GWO (1.03), and GA (0.99).[1] This consistent reduction in CAvg with increasing anonymization levels for Genetic-GWO is presented in Table 6.2 and Figure 6.7. Similar trends are observed for the Reuter database in Table 6.5 and Figure 6.10.

Table 6.2: Average Equivalence Value Analysis in Terms of 20 Newsgroups Database

Algorithms	K-Anonymization				
	k=1	k=2	k=3	k=4	k=5
CPGEN	1.34	1.03	0.99	0.74	1.34
GWO	1.03	1.02	1	0.94	0.92
GA	0.99	0.95	0.95	0.92	0.91
Genetic_GWO	0.74	0.74	0.72	0.71	0.69

10.48047/jocaaa.2024.33.08.185

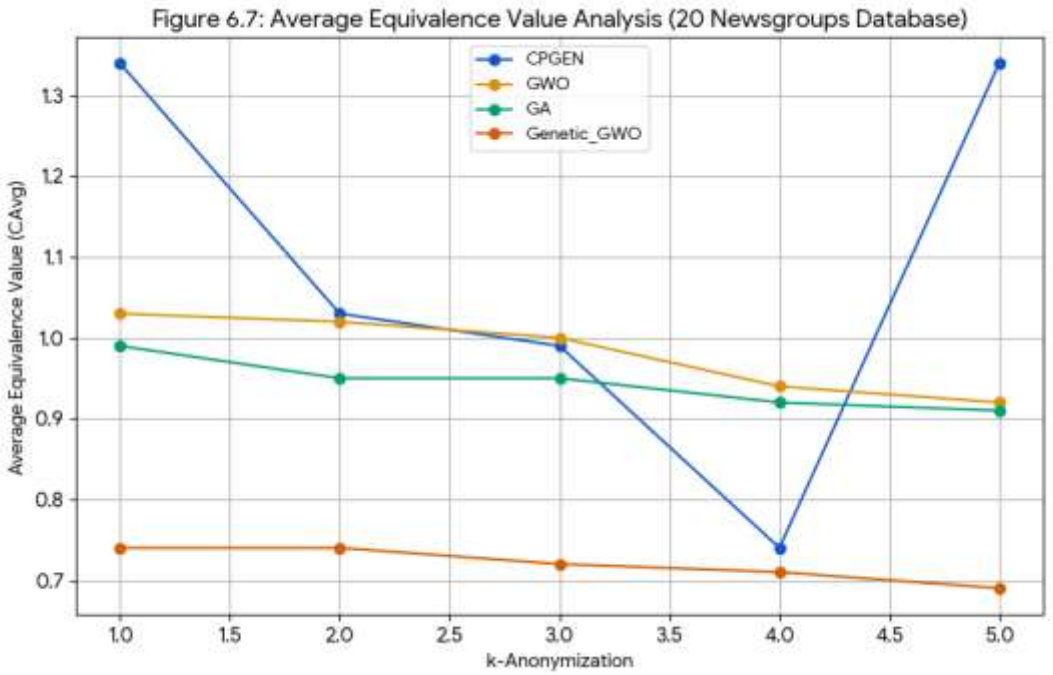


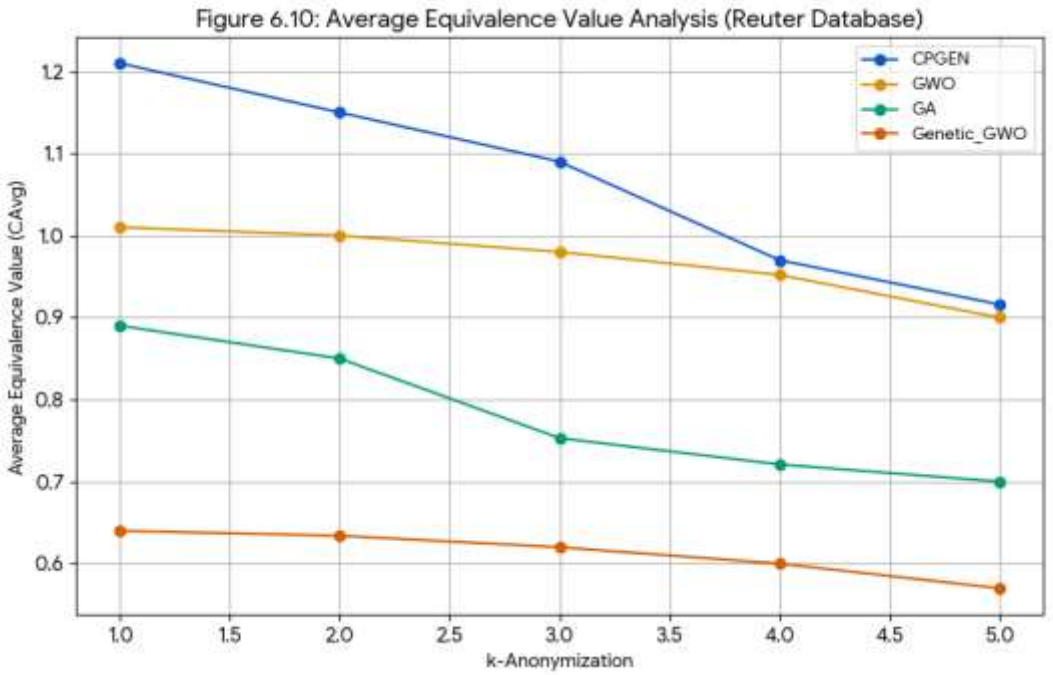
Figure 6.7: Average Equivalence Value Analysis in Terms of 20 Newsgroups Database

Table 6.5: Average Equivalence Value Analysis in Terms of Reuter Database

Algorithms	K-Anonymization				
	k=1	k=2	k=3	k=4	k=5
CPGEN	1.21	1.15	1.09	0.97	0.916
GWO	1.01	1	0.98	0.952	0.9
GA	0.89	0.85	0.753	0.721	0.7
Genetic_GWO	0.64	0.634	0.62	0.6	0.57

Figure 6.10: Average Equivalence Value Analysis in Terms of Reuter Database

10.48047/jocaaa.2024.33.08.185



Fitness Rate

The fitness rate is a composite measure of GILoss and CAvg, reflecting the algorithm's success in optimizing the privacy-utility trade-off.[1] Lower fitness values indicate a better-optimized solution.[1] Genetic-GWO consistently achieves the lowest fitness values across iterations for both datasets, demonstrating its superior ability to balance privacy and utility.[1] For the 20 Newsgroups database, at iteration 1, Genetic-GWO's fitness value was 8.821, lower than CPGEN (9.91), GWO (9.874), and GA (9.568).[1] This trend of decreasing fitness with increasing iterations, indicating optimal balance, is shown in Table 6.3 and Figure 6.8. Similar results are observed for the Reuter database in Table 6.6 and Figure 6.11.

Table 6.3: Fitness Value Analysis in Terms of 20 Newsgroups Database

Algorithm	Fitness Value Iterations
	1
CPGEN	9.91
GWO	9.874
GA	9.568
Genetic_GWO	8.821

10.48047/jocaaa.2024.33.08.185

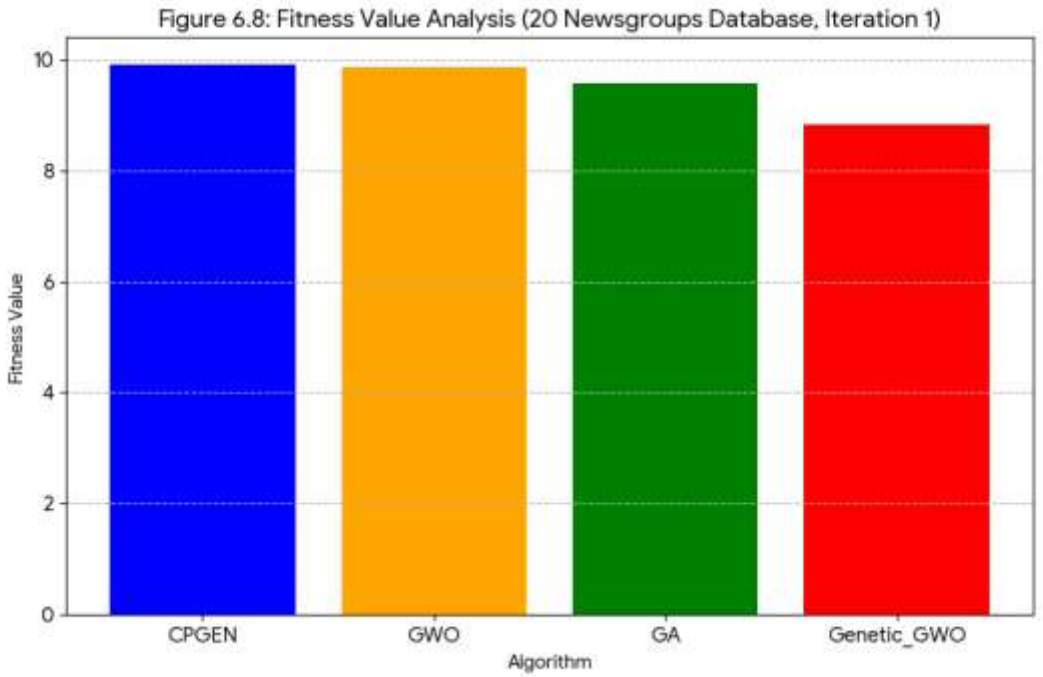


Figure 6.8: Fitness Value Analysis in Terms of 20 Newsgroups Database

Table 6.6: Fitness Value Analysis in Terms of Reuter Database

Algorithm	Fitness Value Iterations
	1
CPGEN	9.543
GWO	9.234
GA	9.132
Genetic-GWO	8

Figure 6.11: Fitness Value Analysis in Terms of Reuter Database

Privacy and Utility Analysis

The overall privacy and utility of the proposed PBDPA model were assessed comparatively. It is important to note that for these metrics, lower values indicate a better balance between privacy preservation (less compromise) and utility (less information loss), consistent with the optimization goals of minimizing GILoss and CAvg. The Genetic-GWO algorithm consistently demonstrates superior performance in achieving this optimal balance across both the 20 Newsgroups and Reuter databases.[1]

10.48047/jocaaa.2024.33.08.185

For the 20 Newsgroups database, Genetic-GWO shows lower values in both Privacy Analysis (Table 6.7, Figure 6.12a) and Utility Analysis (Table 6.8, Figure 6.12b) compared to other algorithms. For instance, at k=1 anonymization, Genetic-GWO's privacy value is 0.4098 and utility value is 0.3496, indicating a better trade-off.[1] Similar results are observed for the Reuter database, with Genetic-GWO consistently yielding lower values for privacy (Table 6.9, Figure 6.13a) and utility (Table 6.10, Figure 6.13b).[1] These results provide direct comparative evidence of PBDPA's ability to achieve a superior balance between privacy and utility for sensitive data publishing, validating the effectiveness of the Genetic-GWO approach across different datasets.

Table 6.7: Privacy Analysis in Terms of 20 News Group Database

Algorithms	K-Anonymization				
	k=1	k=2	k=3	k=4	k=5
CPGEN	0.576844	0.589447	0.532869	0.598188	0.50611
GWO	0.542084	0.589124	0.455492	0.5832	0.42315
GA	0.428126	0.509046	0.455109	0.516557	0.35163
Genetic-GWO	0.409829	0.319296	0.386565	0.388599	0.34617

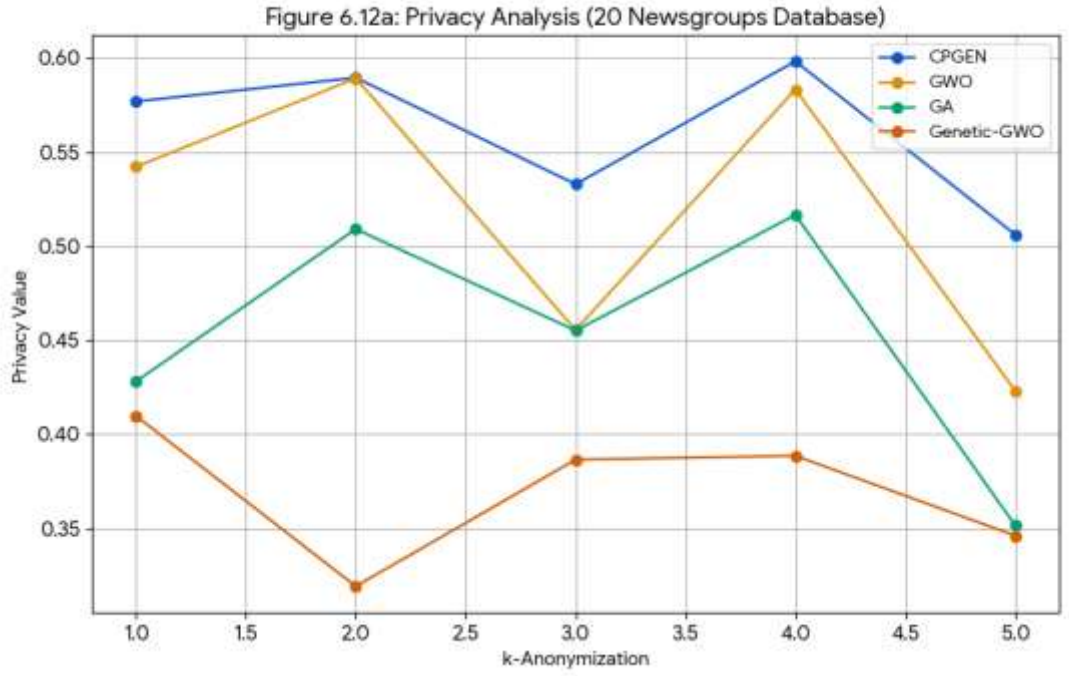


Figure 6.12a: Comparative Analysis of the Proposed data Publishing Phase with 20 Newsgroups Database a) Privacy

5. Conclusion and Future Enhancements

10.48047/jocaaa.2024.33.08.185

5.1 Conclusion

The escalating challenges of data security and privacy in cloud computing necessitate comprehensive and innovative solutions. This research has presented a novel, integrated cloud security model designed to address these critical concerns through three core components: the Amino Hydropathy Index Encryption Algorithm (AHIEA), the Modified Chebyshev polynomial Based Access Control (MCBAC) scheme, and the Privacy Based Data Publishing Algorithm (PBDPA).

The AHIEA significantly enhances data security by employing a chaos-based key generation mechanism, leveraging the unpredictable nature of multiple chaotic maps. This approach provides dynamic keys for encryption, improving the performance of the AES algorithm and ensuring robust protection for data both in transit and at rest. Empirical evaluations demonstrate that AHIEA achieves superior key generation, encryption, and decryption times, along with higher throughput, compared to conventional DES and AES algorithms, thereby increasing user control over their data and reducing computational overhead.

The MCBAC scheme establishes a secure and fine-grained access control framework. By integrating Chebyshev polynomials, hash functions, and session keys within a multi-layered authentication process, it provides robust verification against various adversary attacks. The rigorous six-level authentication process ensures that only legitimate users gain access to cloud data, effectively mitigating threats such as password guessing, brute force, and dictionary attacks. Comparative analyses confirm MCBAC's superior precision, recall, and detection rates, highlighting its effectiveness in securing transactional communications and access to sensitive information.

Furthermore, the PBDPA addresses the critical need for privacy-preserving data publishing. This algorithm utilizes ASCII code conversion and k-anonymization, optimized by a hybrid Genetic Grey Wolf Optimization (Genetic-GWO) algorithm, to balance data privacy and utility. This ensures that sensitive information, particularly in contexts like medical research, can be shared responsibly without compromising individual identities. The PBDPA's ability to minimize Generalized Information Loss (GILoss) and Average Equivalence Value (CAvg) demonstrates its effectiveness in achieving an optimal balance, providing a reliable solution for promoting data-driven insights while adhering to strict privacy requirements.

REFERENCES

1. **Abbasinezhad-Mood D.** and **Nikooghadam M.**, (2018), Efficient anonymous password-authenticated key exchange protocol to read isolated smart meters by utilization of extended Chebyshev chaotic maps, *IEEE Transactions on Industrial Informatics*, **14**(11), 4815–4828.

10.48047/jocaaa.2024.33.08.185

2. **Abdullah A. M.**, (2017), Advanced encryption standard (aes) algorithm to encrypt and decrypt data, *Cryptography and Network Security*, **16**.

3. **Al-Anzi F. S.**, **Yadav S. K.**, and **Soni J.**, Cloud computing: Security model comprising governance, risk management and compliance, *In International Conference on Data Mining and Intelligent Computing (ICDMIC)*. IEEE, 2014, 1–6.

4. **Aldossary S.**, **Allen W.**, *et al.*,, (2016), Data security, privacy, availability and integrity in cloud computing: issues and current solutions, *International Journal of Advanced Computer Science and Applications*, **7**(4), 485–498.

5. **Alliance C.**, (2011), Security guidance for critical areas of focus in cloud computing v3. 0, *Cloud Security Alliance*, **15**.

6. **Alqahtani H.**, **Sarker I. H.**, **Kalim A.**, **Hossain S. M. M.**, **Ikhlaq S.**, and **Hossain S.**, Cyber Intrusion Detection Using Machine Learning Classification Techniques, *In International Conference on Computing Science, Communication and Security*. Springer, 2020, 121–131.

7. **Alvarez G.** and **Li S.**, (2006), Some basic cryptographic requirements for chaos-based cryptosystems, *International journal of bifurcation and chaos*, **16**(08), 2129–2151.

8. **Amazon A.** (2015), Amazon Web Services Overview of Security Processes.

9. **Appari A.** and **Johnson M. E.**, (2010), Information security and privacy in healthcare: current state of research, *International journal of Internet and enterprise management*, **6**(4), 279–314.

10. **Ardagna D.**, **Casale G.**, **Ciavotta M.**, **Pérez J. F.**, and **Wang W.**, (2014), Quality-of-service in cloud computing: modeling techniques and their applications, *Journal of Internet Services and Applications*, **5**(1), 11.

11. **Arroyo D.**, **Diaz J.**, and **Rodriguez F.**, (2013), Cryptanalysis of a one round chaos-based substitution permutation network, *Signal Processing*, **93**(5), 1358–1364.

10.48047/jocaaa.2024.33.08.185

12. **Baek J.**, **Vu Q. H.**, **Liu J. K.**, **Huang X.**, and **Xiang Y.**, (2014), A secure cloud computing based framework for big data information management of smart grid, *IEEE transactions on cloud computing*, **3**(2), 233–244.

13. **Bakhache B.**, **Ahmad K.**, and **El Assad S.**, (2011), A new chaotic encryption algorithm to enhance the security of ZigBee and Wi-Fi networks, *International Journal of Intelligent Computing Research*, **2**(1/2/3/4), pp–219.

14. **Balasubramaniam S.** and **Kavitha V.**, (2014), A survey on data encryption techniques in cloud computing, *Asian Journal of Information Technology*, **13**(9), 494–505.

15. **Balu A.** and **Kuppusamy K.**, (2014), An expressive and provably secure ciphertext-policy attribute-based encryption, *Information Sciences*, **276**, 354–362.

16. **Bayardo R. J.** and **Agrawal R.**, Data privacy through optimal k-anonymization, *In 21st International conference on data engineering (ICDE'05)*. IEEE, 2005, 217–228.

17. **Bertino E.** and **Ferrari E.**, Big data security and privacy, *In A Comprehensive Guide Through the Italian Database Research Over the Last 25 Years*, Springer, 2018, 425–439.

18. **Bertino E.**, **Lin D.**, and **Jiang W.**, A survey of quantification of privacy preserving data mining algorithms, *In Privacy-preserving data mining*, Springer, 2008, 183–205.

19. **Bertino E.** and **Sandhu R.**, (2005), Database security-concepts, approaches, and challenges, *IEEE Transactions on Dependable and secure computing*, (1), 2–19.

20. **Bethencourt J.**, **Sahai A.**, and **Waters B.**, Ciphertext-policy attribute-based encryption, *In IEEE symposium on security and privacy (SP'07)*. IEEE, 2007, 321–334.

21. **Bisong A.**, **Rahman M.**, *et al.,*, (2011), An overview of the security concerns in enterprise cloud computing, *arXiv preprint arXiv:1101.5613*.

22. **Buyya R.**, **Yeo C. S.**, **Venugopal S.**, **Broberg J.**, and **Brandic I.**, (2009), Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th

10.48047/jocaaa.2024.33.08.185

utility, *Future Generation computer systems*, **25**(6), 599–616.

23. **Chard K.**, **Bubendorfer K.**, **Caton S.**, and **Rana O. F.**, (2011), Social cloud computing: A vision for socially motivated resource sharing, *IEEE Transactions on Services Computing*, **5**(4), 551–563.

24. **Chatterjee R.**, **Roy S.**, and **Scholar U.**, (2017), Cryptography in cloud computing: a basic approach to ensure security in cloud, *International Journal of Engineering Science*, **11818**.

25. **Che J.**, **Duan Y.**, **Zhang T.**, and **Fan J.**, (2011), Study on the security models and strategies of cloud computing, *Procedia Engineering*, **23**, 586–593.

26. **Chen D.** and **Zhao H.**, Data security and privacy protection issues in cloud computing, *In International Conference on Computer Science and Electronics Engineering*, volume 1. IEEE, 2012, 647–651.

27. **Chen T.-S.**, **Liu C.-H.**, **Chen T.-L.**, **Chen C.-S.**, **Bau J.-G.**, and **Lin T.-C.**, (2012), Secure dynamic access control scheme of PHR in cloud computing, *Journal of medical systems*, **36**(6), 4005–4020.

28. **Choi C.**, **Choi J.**, and **Kim P.**, (2014), Ontology-based access control model for security policy reasoning in cloud computing, *The Journal of Supercomputing*, **67**(3), 711–722.

29. **Chu C.-K.**, **Chow S. S.**, **Tzeng W.-G.**, **Zhou J.**, and **Deng R. H.**, (2013), Key-aggregate cryptosystem for scalable data sharing in cloud storage, *IEEE transactions on parallel and distributed systems*, **25**(2), 468–477.

30. **Cui B.**, **Liu Z.**, and **Wang L.**, (2015), Key-aggregate searchable encryption (KASE) for group data sharing via cloud storage, *IEEE Transactions on computers*, **65**(8), 2374–2385.

31. **Da Costa G.**, **De Assuncao M. D.**, **Gelas J.-P.**, **Georgiou Y.**, **Lefèvre L.**, **Orgerie A.-C.**, **Pierson J.-M.**, **Richard O.**, and **Sayah A.**, Multi-facet approach to reduce energy consumption in clouds and grids: the GREEN-NET framework, *In Proceedings of the 1st international conference on energy-efficient computing and networking*. ACM, 2010, 95–104.

10.48047/jocaaa.2024.33.08.185

32. **di Vimercati S. D. C.**, **Foresti S.**, **Livraga G.**, and **Samarati P.**, Practical techniques building on encryption for protecting and managing data in the cloud, *In The New Codebreakers*, Springer, 2016, 205–239.

33. **Dillon T.**, **Wu C.**, and **Chang E.**, Cloud computing: issues and challenges, *In 24th IEEE international conference on advanced information networking and applications*. Ieee, 2010, 27–33.