International Journal of Scientific Research in Science and Technology
Print ISSN: 2395-6011 | Online ISSN: 2395-602X (www.ijsrst.com)

Lightweight Optimal Technique for Auditable Secure Cloud Using Hybrid Artificial Intelligence

Ranadeep Reddy Palle

Article Info
Page Number : 972-988

Publication Issue :
Volume IIII, Issue I
Jan-Feb-2014

Article History
Accepted : 01 Jan 2014
Published : 30 Jan 2014

Abstract : In the era of big data, the exponential growth of data poses significant challenges for storage, leading many entities to migrate their data to cloud storage services. While cloud storage offers numerous advantages, it also introduces substantial risks, including the potential loss or unauthorized modification of data by service providers. The extensive data gathered in the cloud, originating from various datasets and storage devices necessitates a thorough analysis of storage performance. Each data instance is defined by specific features, while devices are characterized by their hardware or software components. General restrictions for data allocation and device capacity are also considered. The computation of structural constraints is based on the interactions between cloud-based devices and data instances. In order to address these issues, we introduce a hybrid artificial intelligence approach that is lightweight and ideal for constraint optimization. It focuses on auditable secure cloud storage with dynamic data. Our approach begins with the development of the enhanced electric fish optimization (EEFO) algorithm for constraints optimization to ensure the integrity of data stored in the cloud. To accommodate dynamic data operations, including block modification, insertion, and deletion, we employ the triple tree-seed algorithm (TTSA) to record the location of each data operation within the system. The proposed model's performance is validated, and results are systematically analyzed, compared against existing approaches, demonstrating its effectiveness in appropriately managing cloud data.

Keywords: Cloud Security, Cloud Computing, Constraints Optimization, Data Dynamics, Hybrid, Artificial Intelligent

I. INTRODUCTION

Cloud computing [1] has become an integral part of modern technology, utilized globally for both internal and external purposes. It leverages technological advancements to provide cost-effective and scalable computing and storage solutions. The data audit function within cloud storage is crucial for implementing an efficient storage audit protocol [2][3]. The audit process verifies whether users, either individually or

collectively, can access or manage the stored data, playing a pivotal role in maintaining data integrity [4]. The classification of Aggregate management can be done into two classes: private inquiry and general condition data. private inquiry will enable a data owner to assess data integrity. General condition data, which is considered confidential and subject to verification [5]. Third-party auditing, akin to customer representation, serves as a mechanism for external verification. Cloud storage, as a fundamental service in cloud computing [6], offers a scalable, cost-effective, and location-independent platform for managing user data. Its advantages, including accessibility, reliability, and cost-effectiveness, have lead to widespread adoption. Organizations are increasingly opting for cloud storage outsourcing to optimize costs and focus on core business functions. Availability and reliability are paramount, allowing users to access cloud data anytime and from anywhere over the Internet. Confidentiality, security, availability, data placement, and secure transmission are some of the difficulties associated with cloud data security [7]. Protecting against threats, data loss, business interruptions, external attacks, and tenant-related difficulties are all part of addressing these challenges. One of the main priorities for cloud computing companies is guaranteeing the quality and integrity of data. From the user's point of view, data confidentiality is essential, especially in light of the cloud's storage of private key information [8]. Strategies for identity and access control are essential for maintaining the confidentiality of data. As a result of the improvements in security and dependability brought about by cloud computing initiatives, it is crucial to take user security, reliability, and integrity into account.

Our contributions. A lightweight optimal technique is proposed for constraints optimization, focusing on auditable secure cloud storage with dynamic data, using hybrid artificial intelligence. The key contributions of proposed work are summarized as follows.

1. An enhanced electric fish optimization (EEFO) is designed for constraints optimization, ensuring the integrity of data stored in the cloud.
2. Another significant contribution is the incorporation of the triple tree-seed algorithm (TTSA) to handle dynamic data operations. Dynamic data operations include block modification, insertion, and deletion. TTSA plays a crucial role in recording the location of each data operation within the system. This is particularly valuable in ensuring that the system can effectively manage changes to the data, maintaining integrity and security.
3. The proposed model's performance is systematically validated. This involves rigorous testing and analysis to assess its effectiveness in managing cloud data. The results of these validations are then compared against existing approaches.

This is how the remainder of the paper is structured. The evaluation of recent studies on secure cloud storage is included in Section 2. The problem methodology and system design of the suggested work are provided in Section 3. Section 4 provides a summary of the proposed work's comprehensive working methodology. Section 5 discusses the findings and a comparative study of the safe cloud storage solutions that have been suggested and those that are currently in use. Section 6 brings the paper to a close.

2. Related works

In this section, we provides a comprehensive review of recent research works focusing on secure cloud storage and addressing the challenges associated with data dynamics. The literature review covers various aspects, methodologies, and innovations related to securing data in the cloud environment.

Liu et al. [11] introduced the cloud computing background key exchange (CCBKE) scheme, specifically tailored for security-aware scheduling within the realm of cloud computing service providers. This scheme utilizes an adopted internet key exchange (IKE) scheme. Pervez et al. [12] introduced the self-healing attribute-based privacy-aware data sharing in Cloud (SAPDS) system. This system delegates the key distribution and management processes to a cloud server while preserving the confidentiality of sensitive information. Wang et al. [13] introduced an innovative mechanism for flexible distributed storage integrity auditing, leveraging homomorphic tokens and distributed erasure-coded data. This design enables users to audit cloud storage with minimal communication and computation costs. Sun et al. [14] introduced a secure storage model known as peer to cloud and peer (P2CP), utilizing the cloud storage system as the primary storage backbone. Qin-long et al. [15] introduced a secure and privacy-preserving digital rights management (DRM) scheme utilizing homomorphic encryption in cloud computing. DRM framework enables content providers to outsource encrypted contents to a centralized content server and allows users to consume contents with licenses issued by a license server.

Han et al. [16] introduced an identity-based data storage scheme suitable for cloud computing scenarios, supporting both intra-domain and inter-domain queries. The access key is tied to the requester's identity and the requested ciphertext, and the owner can compute it independently without the assistance of the private key generator (PKG). Yeh et al. [17] proposed the use of a red–black tree to address efficiency issues related to updating data in cloud environments. The rbTree-Doc framework aims to reduce the amount of data requiring encryption by specifically analyzing edited content in collaborative services. Cheng et al. [18] introduced an efficient and secure revocation scheme that operates without assistance. The scheme involves dividing the original data into multiple slices, which are then uploaded to cloud storage. Pervez et al. [19] introduced an oblivious term matching (OTM) mechanism, allowing authorized subscribers to create search queries with varying selection criteria. Itani et al. [20] introduced a service security framework called SNUAGE, for constructing secure and scalable multi-layered services within the realm of cloud computing.

3. Problem methodology and system model

3.1 Research gaps

Secure cloud storage is paramount in the realm of cloud computing, offering a multitude of benefits to users and organizations. One of the fundamental aspects is data protection, wherein robust security measures are implemented to safeguard sensitive information, ensuring its confidentiality and integrity. The remote accessibility characteristic of cloud storage allows users to access their data from anywhere, underpinned by a secure storage infrastructure. Scalability is another key advantage, enabling businesses and individuals to seamlessly accommodate growing storage needs. Moreover, cloud storage offers cost-effective solutions, allowing users to pay for the storage they require without hefty upfront investments. Wang et al. [21] delved into the challenge of simultaneously ensuring public auditability and accommodating data dynamics in remote data integrity checks within cloud computing. From literature review [11]-[21], we address the following research gaps, the critical issue of ensuring data integrity in the face of constant changes, be it through modifications, insertions, or deletions. Maintaining the accuracy and consistency of data becomes a complex undertaking in the dynamic environment of the cloud. Traditional audit mechanisms may struggle to keep pace with the rapid transformations in data, potentially compromising the overall security and trustworthiness of stored information. Addressing these multifaceted challenges necessitates innovative

solutions that carefully consider the unique characteristics of dynamic data within the dynamic environment of cloud computing, with a focus on efficiency, security, and privacy.

- Design and implement advanced data integrity verification mechanisms capable of ensuring the accuracy and consistency of dynamically changing data in the cloud.
- Develop audit mechanisms verify the integrity of dynamic data, accommodating modification, insertion, and deletions while maintaining comprehensive and secure audit trail.
- Identify optimization strategies to minimize the performance overheads associated with security measures for dynamic data. Balance robust security practices with acceptable performance levels to ensure efficient data operations.

3.2 System architecture

Secure cloud data storage has emerged as an efficient solution for data management, involving the remote management and protection of data through third-party servers. This process ensures data security by leveraging the capabilities of the cloud, employing a framework that considers four key entities: the data owner, the data user, the cloud user, and the third-party server. The data owner is responsible for managing and storing data across various virtual machines (VMs), while the data user has the flexibility to select specific machines for data retrieval. Concurrently, third-party servers play a crucial role in regularly verifying data integrity. The architecture of the proposed lightweight optimal technique for secure cloud storage, depicted in Fig. 1, aims to rectify both general and structural constraints associated with data storage. In this framework, the storage network encounters several constraints due to the diverse components and VMs. The system architecture takes these components into account, introducing a reliable model to address various constraints. Each piece of cloud data is characterized by individual traits, further classified by hardware and software components. Consequently, a new objective function is derived to tackle both structural and general constraints, defining the interactions and instances within the cloud data storage. To mitigate constraint challenges, the proposed framework introduces the enhanced electric fish optimization (EEFO) algorithm. For data dynamics like block modification, insertion, and deletion, the triple tree-seed algorithm (TTSA) is employed to record the location of each operation within the system. In the evaluation stage, the performance of the proposed framework is measured using metrics, and simulation results demonstrate its efficacy in appropriately storing cloud data with involved components. The extensive findings validate the effectiveness of the proposed approach in overcoming constraints and ensuring secure cloud data storage.

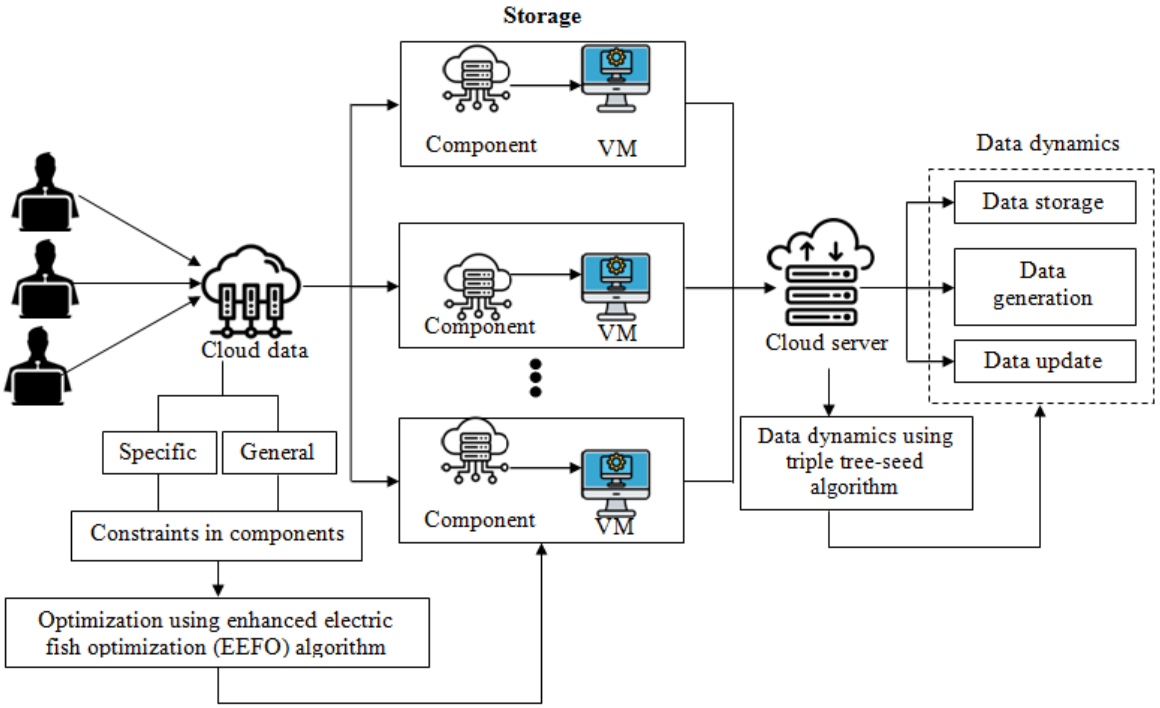

Fig. 1 System architecture of proposed work

4. Proposed methodology

In this section, we briefly discuss the process of proposed lightweight optimal technique for secure cloud storage with dynamic data in cloud computing environment.

4.1 Component constraints optimization

The optimization of component constraints in our framework is orchestrated through the development and application of the enhanced electric fish optimization (EEFO) algorithm. This algorithm's application within our framework extends beyond mere optimization, encompassing the assurance of data integrity. First, the search space's electric fish population (P) is dispersed at random while accounting for the geographical limits.

$$p_{hg} = p_{Ming} + \phi(p_{Max\,g} - p_{Ming}) \tag{1}$$

where size$|B|$ $(h = 1, 2, \ldots, |B|)$ in the c-dimensional search space represents the p_{hg} indicate the position of the h-th discrete in the population. p_{Ming} and $p_{Max\,g}$ lower and upper bounds of the dimension $g \mid g \in 1, 2, \ldots c$, respectively. The value of an individual's frequency (F_h^s) is derived from its fitness value:

$$F_h^s = F_{Min} + \left(\frac{Fit_{worst}^s - Fit_h^s}{Fit_{worst}^s - Fit_{best}^s} \right) (F_{Max} - F_{Min}) \tag{2}$$

Where Fit_{worst}^s and Fit_{best}^s are the lowest and highest fitness values from members of the present population, respectively. The individual's amplitude does not change much because it is dependent upon the weight of the individual's prior amplitudes. The amplitude of the h-th individual can the described as:

$$M_h^s = \alpha M_h^{s-1} + (1-\alpha) F_h^s \tag{3}$$

where $\alpha \mid \alpha \in [0,1]$ indicate the constant that establishes the preceding amplitude value's magnitude. According to EEFO, initial frequency F_h is used to define initial amplitude value. the h-th individual amplitude (R_h) is used to find its active range.

$$R_h = (p_{Maxg} - p_{Ming})M_h \tag{4}$$

To find neighbors in the sensitivity/efficiency range ($T \mid T \subset B$), we measure the distance between the h-th individual and the rest of the population. The distance between individuals H and K is determined by calculating the Cartesian distance:

$$c_{hK} = \| p_h - p_k \| = \sqrt{\sum_{g=1}^{c} (p_{hg} - p_{Kg})^2} \tag{5}$$

The active sensing of least neighbor can be determined by the equation mention below:

$$p_{hg}^{cand} = p_{hg} + \varphi(p_{Kg} - p_{hg}) \tag{6}$$

where K represents an individual randomly selected from the set of neighbors of the H-th individual

$$p_{hg}^{cand} = p_{hg} + \varphi R_h \tag{7}$$

A $\varphi \in [-1, 1]$ random number generated from a uniform distribution p_{hg}^{cand} representing the candidate location of the h-th person.

$$x_K = \frac{M_K / c_{hK}}{\sum_{g \in B_M} M_g / c_{hg}} \tag{8}$$

Here K individuals are selected from the NA using different strategies such as roulette wheel selection, compute as follows.

$$p_{Rg} = \frac{\sum_{K=1}^{k} M_K p_{Kg}}{\sum_{K=1}^{k} M_K} \tag{9}$$

$$p_{hg}^{New} = p_{hg} + \varphi(p_{Rg} - p_{hg}) \tag{10}$$

Unlike searching in active electro location, multiple parameters can be manipulated, meaning individuals explore the search space more quickly.

$$p_{hg}^{cand} = \begin{cases} p_{hg}^{New} & rand_g(0,1) > F_h \\ p_{hg} & else \end{cases} \tag{11}$$

Changing the h-th person's parameter is the final step in passive electrolocation, which increases the likelihood that the behavior will change.

$$p_{hg}^{cand} = p_{Ming} + \varphi(p_{Maxg} - p_{Ming}) \tag{12}$$

$$Rand(0,1) \le Rand(0,1) \tag{13}$$

where $Rand(0,1)$ is a number that is chosen at random from a uniform distribution. The g-th parameter of the h-th individual is shifted to the limits of the limited space if its value surpasses the search space's boundaries:

$$p_{hg}^{cand} = \begin{cases} p_{Min\,g} & p_{hg}^{cand} < p_{Min\,g} \\ p_{hg}^{cand} & p_{Max\,g} > p_{hg}^{cand} > p_{Min\,g} \\ p_{Max\,g} & p_{hg}^{cand} > p_{Min\,g} \end{cases} \tag{14}$$

4.2 Data dynamics

In the realm of data dynamics, we address operations such as block modification, insertion, and deletion through the utilization of the triple tree-seed algorithm (TTSA). The four stages of the TTSA optimization mechanism are as follows. initialization of the tree, which is produced based

$$t_{h,g} = l_{g,Min} + R_{h,g} \times \left(I_{g,Max} - l_{g,Max} \right) \tag{15}$$

where $l_{g,Min}$ search space's lower bound, higher one is indicated by $I_{g,Max}$ and in each dimension and location generates random, which is indicated by $R_{h,g}$ and the radon number ranges between in the range of [0, 1]. Seed's generation is given by:

$$s_{h,g} = t_{h,g} + \alpha_{h,g} \times \left(N_g - t_{R,g} \right) \tag{16}$$

$$s_{h,g} = t_{h,g} + \alpha_{h,g} \times \left(t_{h,g} - t_{R,g} \right) \tag{17}$$

where $s_{h,g}$ is the h-th seed formed in the h-th tree which has g-th dimension, $t_{h,g}$ is the h-th tree having g-th dimension, N_g is the finest tree's location on the g-*t*h dimension in the entire trial, $t_{R,g}$ randomly chosen tree of g-th dimension in the whole population, α is the scaling factor which has a range of [–1, 1] and h and R indices different trees are updated in other trials.

$$Max\ fes = C \times 10{,}000 \tag{18}$$

$$fes = fes + ns \tag{19}$$

where 'ns' indicates seeds generated in the tree. The initial random candidate solutions produced by TTSA must fluctuate outward. It uses the sine and cosine function mathematical model to get close to the best result. The technique emphasizes exploration and exploitation in the search space of several optimization phases by integrating many random variables and adaptive variables. The TTSA changed its position in both phases as follows:

$$P_h^{T+1} = P_h^T + R_1 \times \sin(R_2) \times \left| R_3 X_h^T - P_h^T \right|, R_4 < 0.5 \tag{20}$$

$$P_h^{T+1} = P_h^T + R_1 \times \cos(R_2) \times \left| R_3 X_h^T - P_h^T \right|, R_4 \geq 0.5 \tag{21}$$

where P_h^T indicates the position of the present solution in h-th dimension at the T-th iteration, X_h^T is the position of the destination point in h-th dimension, R_1 , R_2 , R_3 , and R_4 are random numbers, R_4 is the range in [0, 1]. There are four main parameters in SCA: h-th dimension, R_1 , R_2 , R_3 , and R_4 . R_1 dictates the next position area, R_2 shows the distance that the movement should travel to reach the target or outward, and R_3 adds a random weight to the destination to stochastically increase ($R_3 > 1$) or decrease ($R_3 < 1$) the destination's influence on the distance defined. The following function adaptively modifies the sine and cosine range to strike a balance between exploration and exploitation.

$$R_1 = m - s\frac{m}{t} \tag{22}$$

where T is the current iteration, t is the maximum number of iterations and m is a constant. In the basic TTSA, the current best tree is regarded as the candidate tree. Sine function inspired by SCA is added to increase diversity. After the above improvements, two tree migration equations are added in this phase.

$$t_{h,g} = \left(t_{xn,1} + t_{xn,2} + t_{xn,3}\right)/3 + \left(N_g - t_{R,g}\right) * \sin\left(2 * xh * Rand\right) \tag{23}$$

$$t_{h,g} = \left(t_{xn,1} + t_{xn,2} + t_{xn,3}\right)/3 + \left(N_g - t_{R,g}\right) * \left(Rand - 0.5\right) * 2 \tag{24}$$

where $t_{h,g}$ is the i-th tree of the j-th dimension, $t_{xn,1}$, $t_{xn,2}$, and $t_{xn,3}$ represents the initial three trees which is produced before generating the seed, $\left(t_{xn,1} + t_{xn,2} + t_{xn,3}\right)/3$indicate the initial tress gravity center in the present run, N_g is the j-th dimension's best tree, $t_{R,g}$ is the R-th tree of the g-th dimension chosen at random from the population.

5. Results and Discussion

In this section, we present the outcomes of our proposed secure cloud storage model alongside a comparative analysis involving various existing models. The simulations for our proposed model were conducted using MATLAB 2020a using a machine that included a 7200 RPM Western Digital 250 GB Serial ATA drive with an 8 MB buffer, a 2.4 GHz Intel Core 2 processor, and 768 MB of RAM. Our analysis includes a focus on convergence, a comparative assessment, and statistical analysis to establish the effectiveness of our proposed model. During the evaluation process, key performance parameters like makespan, processing time, and active servers were taken into account. The results obtained from the proposed EEFO-TTSA model are juxtaposed with those from existing secure cloud storage models, including rbTree-Doc [27], CP-ABE [28], OTM [29], SNUAGE [30], and Merkle Hash Tree [31]. This comparative analysis aims to provide insights into the relative strengths and weaknesses of the proposed model in relation to established approaches, offering comprehensive understanding of its performance across diverse parameters.

5.1 Simulation setup

Table 1 delineates the parameters employed in the simulation setup to evaluate the proposed secure cloud storage model. The configuration encompasses specifications for the data center, host, cloudlets, and VMs, offering a comprehensive view of the experimental setting. Concerning Data Center Parameters, the simulation involves two data centers, each equipped with a single host possessing 2GB of RAM, a storage capacity of 1TB, and a bandwidth of 10Gbps. In the realm of Cloudlets Parameters, the experimental conditions encompass a varied number of cloudlets ranging from 100 to 1000, with cloudlet lengths spanning from 100 to 1000 Million Instructions (MIs). File sizes for these cloudlets vary between 200 and 400 MB, with an output size set at 300. The VM parameters include a range of VM IDs from 1 to 20, monitored using the Xen system. This diverse setup facilitates a thorough assessment of the proposed secure cloud storage model's performance across different scenarios, accommodating various cloudlet characteristics and VM configurations. Such variability ensures robust evaluation of the model's efficiency

under diverse workloads and conditions. The configuration of the proposed secure cloud storage model is characterized by four distinct setups that account for the components and virtual machines (VMs). Table 2 provides a detailed overview of the configuration settings devised for the storage of cloud data.

Table 1 Simulation setup

Data center	Parameter	Values
Host	Number of data centre	2
	Number of host	1
	Host Ram	2GB
	Storage	1TB
	Bandwidth	10Gbps
Cloudlets	Number of cloudlets	100-1000
	Lengths	100-1000 MIs
	File size	200 to 400 MB
	Output size	300
Virtual machine	VM ID	1 to 20
	VMs monitor	Xen

Table 2 Cloud data storage configuration setting

Configuration Case	No. of Components	No. of VMs
1	6	5
2	12	10
3	18	15
4	24	20

5.2 Comparative analysis

The convergence analysis results for the proposed and existing secure cloud storage models across different configuration cases (1, 2, 3, 4) are presented in Table 3. In Configuration-1, the computational costs of secure cloud storage models such as rbTree-Doc, CP-ABE, OTM, SNUAGE, and Merkle Hash Tree, along with EEFO-TTSA, consistently increased as the number of components grew from 5 to 25. The cost functions for rbTree-Doc, CP-ABE, OTM, SNUAGE, and Merkle Hash Tree rose by 11.05%, 13.1%, 16.27%, 23.08%, and 35.88%, respectively. These results indicate a proportional increase in computational expenses with the growth in the number of components. However, the proposed EEFO-TTSA model demonstrated an impressive 81.99% decrease in computational costs, shown its efficiency in managing larger component numbers. This significant reduction in costs positions the EEFO-TTSA model as a more effective and resource-efficient solution compared to existing models in Configuration-1.

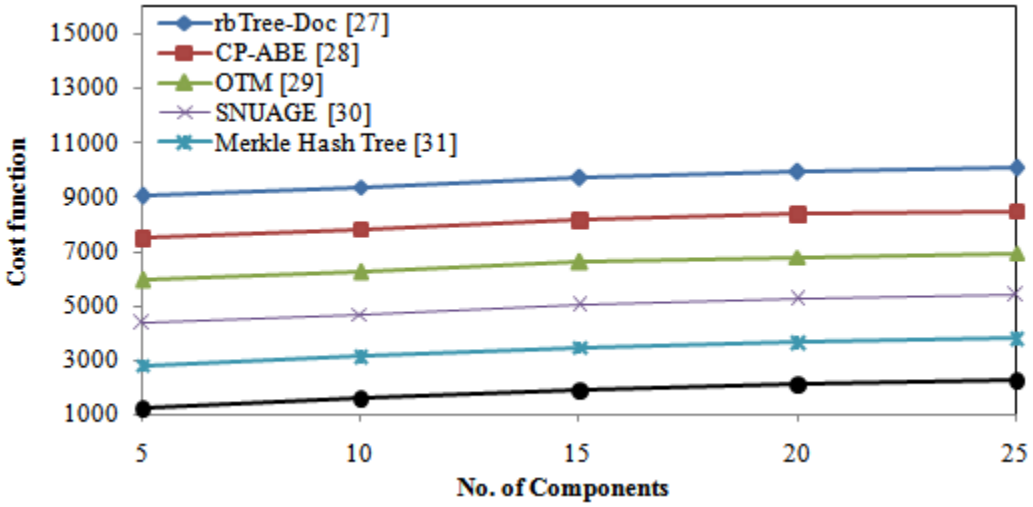

Fig. 2 Results comparison of cost function for Configuration-1

In Configuration-2, the computational costs of secure cloud storage models, including rbTree-Doc, CP-ABE, OTM, SNUAGE, Merkle Hash Tree, and EEFO-TTSA, were examined as the number of components varied from 5 to 25. For rbTree-Doc, CP-ABE, OTM, SNUAGE, and Merkle Hash Tree, the cost functions increased by 9.92%, 8.51%, 11.16%, 10.87%, and 13.53%, respectively. These findings indicate a consistent rise in computational expenses corresponding to the increment in the number of components. Notably, the proposed EEFO-TTSA model exhibited a substantial 63.89% reduction in computational costs, underscoring its efficiency in managing larger component numbers. This remarkable decrease in costs positions the EEFO-TTSA model as a promising and resource-efficient solution compared to existing models in Configuration-2.

Table 3 Convergence analysis of proposed and existing secure cloud storage models with different configuration cases

Secure cloud storage models	Number of Components									
	5	10	15	20	25	5	10	15	20	25
	Configuration-1					Configuration-2				
rbTree-Doc [27]	9050	9378	9710	9920	10051	8396	8724	9056	9266	9397
CP-ABE [28]	7487	7815	8147	8357	8488	6833	7161	7493	7703	7834
OTM [29]	5924	6252	6584	6794	6925	5270	5598	5930	6140	6271
SNUAGE [30]	4361	4689	5021	5231	5362	3707	4035	4367	4577	4708
Merkle Hash Tree [31]	2798	3126	3458	3668	3799	2144	2472	2804	3014	3145
EEFO-TTSA	1235	1563	1895	2105	2236	581	909	1241	1451	1582
	Configuration-3					Configuration-4				
rbTree-Doc [27]	8694	9022	9354	9564	9695	8655	8983	9315	9525	9656
CP-ABE [28]	7131	7459	7791	8001	8132	7092	7420	7752	7962	8093
OTM [29]	5568	5896	6228	6438	6569	5529	5857	6189	6399	6530
SNUAGE [30]	4005	4333	4665	4875	5006	3966	4294	4626	4836	4967
Merkle Hash Tree [31]	2442	2770	3102	3312	3443	2403	2731	3063	3273	3404
EEFO-TTSA	879	1207	1539	1749	1880	840	1168	1500	1710	1841

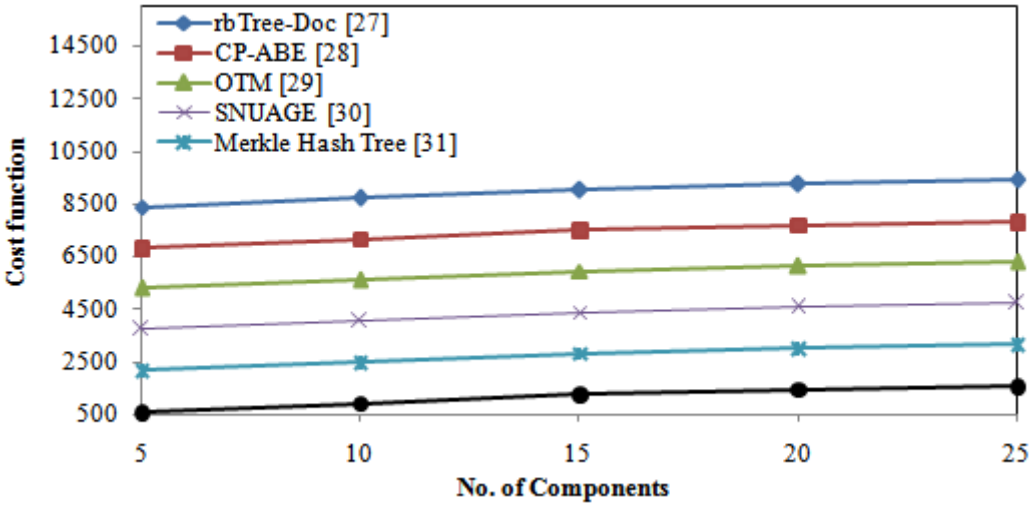

Fig. 3 Results comparison of cost function for Configuration-2

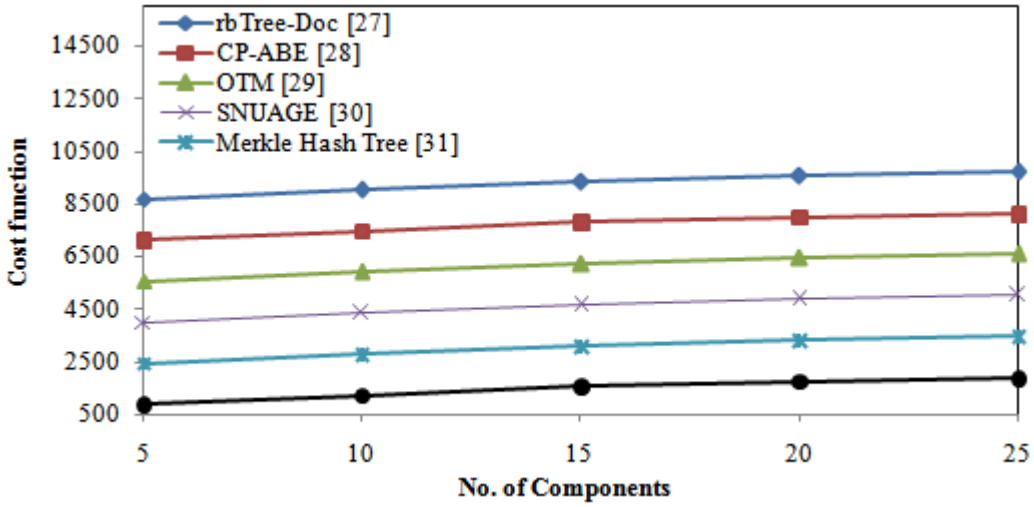

Fig. 4 Results comparison of cost function for Configuration-3

Examining configuration-3, the computational costs of various secure cloud storage models, including rbTree-Doc, CP-ABE, OTM, SNUAGE, Merkle Hash Tree, and EEFO-TTSA, were analyzed as the number of components ranged from 5 to 25. For rbTree-Doc, CP-ABE, OTM, SNUAGE, and Merkle Hash Tree, the cost functions increased by 9.68%, 8.48%, 11.69%, 8.45%, and 13.96%, respectively. The results highlight consistent upward trend in computational expenses corresponding to the increase in the number of components. Significantly, the proposed EEFO-TTSA model demonstrated a 53.67% reduction in computational costs, emphasizing its efficiency in handling larger component numbers. This noteworthy reduction in costs establishes the EEFO-TTSA model as a compelling and resource-efficient solution compared to existing models in Configuration-3.

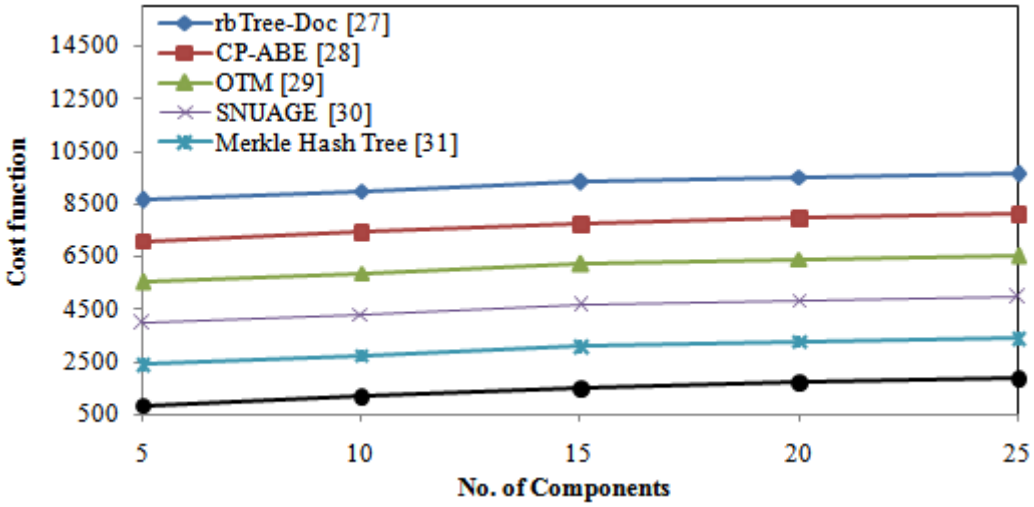

Fig. 5 Results comparison of cost function for Configuration-4

Configuration-4 entails an examination of the computational costs associated with various secure cloud storage models as the number of components expands from 5 to 25. Specifically, rbTree-Doc, CP-ABE, OTM, SNUAGE, Merkle Hash Tree, and EEFO-TTSA were evaluated. For rbTree-Doc, CP-ABE, OTM, SNUAGE, and Merkle Hash Tree, the cost functions demonstrated increases of

9.67%, 8.45%, 11.68%, 8.46%, and 13.94%, respectively. These findings underscore consistent rise in computational expenses corresponding to the escalating number of components. Notably, the proposed EEFO-TTSA model showcased an impressive 53.87% reduction in computational costs, underscoring its efficacy in handling a larger number of components. This significant reduction positions the EEFO-TTSA model as a robust and resource-efficient solution in Configuration-4 compared to existing models

Table 4 presents the number of active servers, computational time, and makespan comparison of proposed and existing secure cloud storage models with different configuration cases. Fig. 6 shows the number of active servers for rbTree-Doc increased by 10%, CP-ABE by 10.3%, OTM by 12%, SNUAGE by 200%, Merkle Hash Tree by 100%, and EEFO-TTSA by 100% when compared to Configuration Case 1. In Configuration Case 2, the number of active servers for rbTree-Doc increased by 40%, CP-ABE by 40%, OTM by 16.67%, SNUAGE by 16.67%, Merkle Hash Tree by 20%, and EEFO-TTSA by 100% when compared to Configuration Case 1. In Configuration Case 3, the number of active servers for rbTree-Doc increased by 14.29%, CP-ABE by 14.29%, OTM by 14.29%, SNUAGE by 14.29%, Merkle Hash Tree by 16.67%, and EEFO-TTSA by 50% when compared to Configuration Case 2. In Configuration Case 4, the number of active servers for rbTree-Doc increased by 11.11%, CP-ABE by 11.11%, OTM by 12.5%, SNUAGE by 28.57%, Merkle Hash Tree by 28.57%, and EEFO-TTSA by 33.33% when compared to Configuration Case 3. Fig. 7 shows the computational time for secure cloud storage models exhibited a noticeable decrease as the configuration number increased. Specifically, rbTree-Doc decreased by approximately 8.78%, CP-ABE by 8.87%, OTM by 8.98%, SNUAGE by 9.08%, Merkle Hash Tree by 8.96%, and EEFO-TTSA by 9.33% compared to Configuration Case 1. Moving to Configuration Case 2, there was a further decrease in computational time for all models compared to Configuration Case 1. The reductions were approximately 9.44% for rbTree-Doc, 9.54% for CP-ABE, 9.65% for OTM, 9.76% for SNUAGE, 9.74% for Merkle Hash Tree, and 9.99% for EEFO-TTSA.

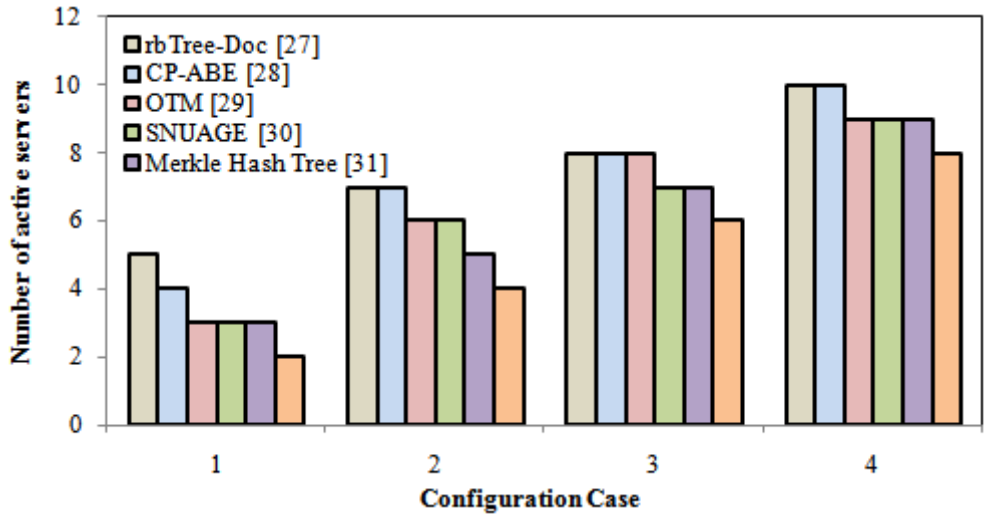

Fig. 6 Number of active server's comparison of proposed and existing secure cloud storage models

In Configuration Case 3, the trend of decreasing computational time persisted. The reductions compared to Configuration Case 2 were approximately 9.55% for rbTree-Doc, 9.65% for CP-ABE, 9.76% for OTM, 9.87% for SNUAGE, 9.85% for Merkle Hash Tree, and 10.1% for EEFO-TTSA. Finally, in Configuration Case 4, the computational time continued to decrease, showcasing the efficiency of the models. The reductions compared to Configuration Case 3 were approximately 9.60% for rbTree-Doc, 9.70% for CP-ABE, 9.81% for OTM, 9.92% for SNUAGE, 9.90% for Merkle Hash Tree, and 10.15% for EEFO-TTSA.

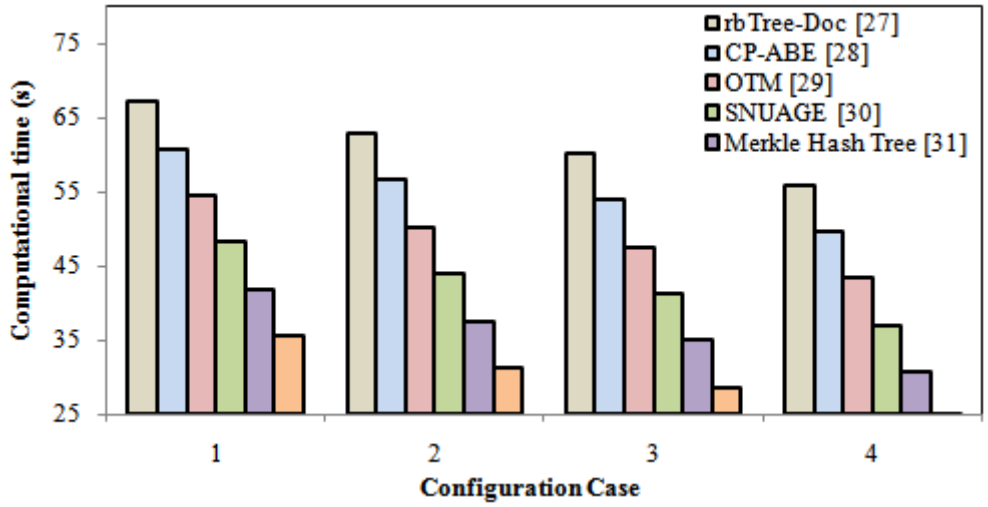

Fig. 7 Computational time comparison of proposed and existing secure cloud storage models

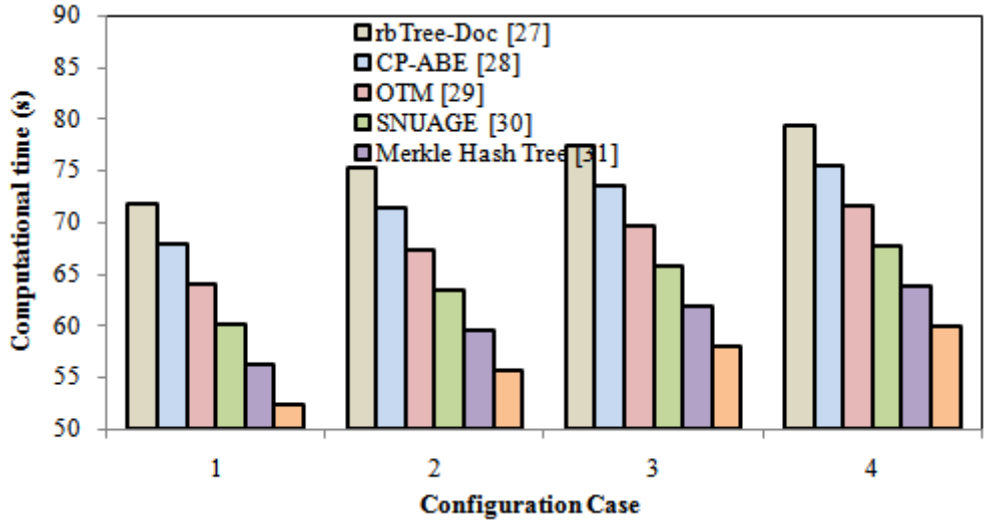

Fig. 8 Makespan comparison of proposed and existing secure cloud storage models

In Configuration Case 1, the makespan for secure cloud storage models showed an incremental trend as the configuration number increased. Specifically, rbTree-Doc increased by approximately 5.49%, CP-ABE by 5.54%, OTM by 5.67%, SNUAGE by 5.80%, Merkle Hash Tree by 5.78%, and EEFO-TTSA by 6.23% compared to Configuration Case 1. Transitioning to Configuration Case 2, there was a further increase in makespan for all models compared to Configuration Case 1. The increments were approximately 5.93% for rbTree-Doc, 5.98% for CP-ABE, 6.11% for OTM, 6.24% for SNUAGE, 6.22% for Merkle Hash Tree, and 6.69% for EEFO-TTSA. In Configuration Case 3, the trend of increasing makespan persisted. The increments compared to Configuration Case 2

were approximately 5.89% for rbTree-Doc, 5.94% for CP-ABE, 6.07% for OTM, 6.20% for SNUAGE, 6.18% for Merkle Hash Tree, and 6.64% for EEFO-TTSA. Finally, in Configuration Case 4, the makespan continued to increase, indicating the impact of the configurations on the overall execution time. The increments compared to Configuration Case 3 were approximately 5.87% for rbTree-Doc, 5.92% for CP-ABE, 6.05% for OTM, 6.18% for SNUAGE, 6.16% for Merkle Hash Tree, and 6.62% for EEFO-TTSA.

Table 4 Number of active servers, computational time, and makespan comparison of proposed and existing secure cloud storage models with different configuration cases

Secure cloud storage models	Configuration Case											
	1	2	3	4	1	2	3	4	1	2	3	4
	Number of active servers				Computational time (s)				Makespan			
rbTree-Doc [27]	5	7	8	10	67.188	62.883	60.187	55.980	71.838	75.171	77.371	79.340
CP-ABE [28]	4	7	8	10	60.863	56.558	53.862	49.655	67.943	71.276	73.476	75.445
OTM [29]	3	6	8	9	54.538	50.233	47.537	43.330	64.048	67.381	69.581	71.550
SNUAGE [30]	3	6	7	9	48.213	43.908	41.212	37.005	60.153	63.486	65.686	67.655
Merkle Hash Tree [31]	3	5	7	9	41.888	37.583	34.887	30.680	56.258	59.591	61.791	63.760
EEFO-TTSA	2	4	6	8	35.563	31.258	28.562	24.355	52.363	55.696	57.896	59.865

6. Conclusion

Our proposed lightweight optimal technique focuses on constraints optimization for auditable secure cloud storage with dynamic data, leveraging hybrid artificial intelligence. The development of the enhanced electric fish optimization (EEFO) algorithm plays a crucial role in ensuring the integrity of data stored in the cloud. Additionally, to accommodate dynamic data operations such as block modification, insertion, and deletion, we introduce the triple tree-seed algorithm (TTSA), which records the location of each data operation within the system. The simulation results shed light on the significant impact of chosen configurations on the overall execution time of secure cloud storage models. The observed increase in makespan across different configurations underscores the influence of specific conditions and allocated resources on the efficiency and performance of these models. This emphasizes the importance of thoughtful configuration analysis and optimization to minimize execution time, ensuring that secure cloud storage models operate with maximum efficiency and effectiveness. These findings highlight the need for a comprehensive understanding of system configurations to enhance the performance of secure cloud storage models in real-world cloud computing environments. By addressing these considerations, we contribute to the ongoing efforts to optimize and advance the field of auditable secure cloud storage, particularly in the context of dynamic data and evolving hybrid artificial intelligence techniques.

References

1. Ebejer, J.P., Fulle, S., Morris, G.M. and Finn, P.W., 2013. The emerging role of cloud computing in molecular modelling. Journal of Molecular Graphics and Modelling, 44, pp.177-187.
2. Abbadi, I.M. and Martin, A., 2011. Trust in the Cloud. information security technical report, 16(3-4), pp.108-114.
3. Apostu, A., Rednic, E. and Puican, F., 2012. Modeling cloud architecture in banking systems. Procedia economics and finance, 3, pp.543-548.
4. Xu, X., 2012. From cloud computing to cloud manufacturing. Robotics and computer-integrated manufacturing, 28(1), pp.75-86.
5. Jorissen, K., Vila, F.D. and Rehr, J.J., 2012. A high performance scientific cloud computing environment for materials simulations. Computer Physics Communications, 183(9), pp.1911-1919.
6. Chonka, A., Xiang, Y., Zhou, W. and Bonti, A., 2011. Cloud security defence to protect cloud computing against HTTP-DoS and XML-DoS attacks. Journal of Network and Computer Applications, 34(4), pp.1097-1107.
7. Fu, Y., Wang, Y. and Biersack, E., 2013. A general scalable and accurate decentralized level monitoring method for large-scale dynamic service provision in hybrid clouds. Future Generation Computer Systems, 29(5), pp.1235-1253.
8. Das, S.K., Kant, K. and Zhang, N., 2012. Handbook on securing cyber-physical critical infrastructure. Elsevier.
9. McClintock, W.E., Rusch, D.W., Thomas, G.E., Merkel, A.W., Lankton, M.R., Drake, V.A., Bailey, S.M. and Russell III, J.M., 2009. The cloud imaging and particle size experiment on the Aeronomy of Ice in the mesosphere mission: Instrument concept, design, calibration, and on-orbit performance. Journal of Atmospheric and Solar-Terrestrial Physics, 71(3-4), pp.340-355.
10. Fernandez, E.B., La Red, D.L. and Peláez, J.I., 2013. A conceptual approach to secure electronic elections based on patterns. Government Information Quarterly, 30(1), pp.64-73.
11. Liu, C., Zhang, X., Yang, C. and Chen, J., 2013. CCBKE—Session key negotiation for fast and secure scheduling of scientific applications in cloud computing. Future Generation Computer Systems, 29(5), pp.1300-1308.
12. Pervez, Z., Khattak, A.M., Lee, S. and Lee, Y.K., 2012. SAPDS: self-healing attribute-based privacy aware data sharing in cloud. The Journal of Supercomputing, 62, pp.431-460.
13. Wang, C., Wang, Q., Ren, K., Cao, N. and Lou, W., 2011. Toward secure and dependable storage services in cloud computing. IEEE transactions on Services Computing, 5(2), pp.220-232.
14. Sun, Z. and Shen, J., 2013. A high performance peer to cloud and peer model augmented with hierarchical secure communications. Journal of Systems and Software, 86(7), pp.1790-1796.
15. Huang, Q.L., YANG, Y.X., FU, J.Y. and NIU, X.X., 2013. Secure and privacy-preserving DRM scheme using homomorphic encryption in cloud computing. The Journal of China Universities of Posts and Telecommunications, 20(6), pp.88-95.
16. Han, J., Susilo, W. and Mu, Y., 2013. Identity-based data storage in cloud computing. Future Generation Computer Systems, 29(3), pp.673-681.
17. Yeh, S.C., Su, M.Y., Chen, H.H. and Lin, C.Y., 2013. An efficient and secure approach for a cloud collaborative editing. Journal of Network and Computer Applications, 36(6), pp.1632-1641.

18. Cheng, Y., Wang, Z.Y., Ma, J., Wu, J.J., Mei, S.Z. and Ren, J.C., 2013. Efficient revocation in ciphertext-policy attribute-based encryption based cryptographic cloud storage. Journal of Zhejiang University SCIENCE C, 14(2), pp.85-97.
19. Pervez, Z., Awan, A.A., Khattak, A.M., Lee, S. and Huh, E.N., 2013. Privacy-aware searching with oblivious term matching for cloud storage. The Journal of Supercomputing, 63, pp.538-560.
20. Itani, W., Kayssi, A. and Chehab, A., 2013. SNUAGE: an efficient platform-as-a-service security framework for the cloud. Cluster computing, 16, pp.707-724.
21. Wang, Q., Wang, C., Ren, K., Lou, W. and Li, J., 2010. Enabling public auditability and data dynamics for storage security in cloud computing. IEEE transactions on parallel and distributed systems, 22(5), pp.847-859.