ISSN 1330-3651 (Print), ISSN 1848-6339 (Online) https://doi.org/10.17559/TV-20240121001277
Original scientific paper

An Adaptive Defense Model for Cloud Computing Data Security

Xing YANG*, Ke XIANG, JiLan HUANG

Abstract: Due to the widespread application of cloud computing data, this paper proposes a cloud computing data defense model integrating Random Binary Extension Codes (RBEC) encoding and an adaptive Dynamic Heterogeneous Redundancy (DHR) technique. The core novelties lie in utilizing RBEC to strengthen data redundancy and incorporating reputation calculation and heterogeneity optimization in the DHR model to enhance adaptivity. Experimental analyses demonstrate the RBEC encoding achieves over 96% transformation success rate and 94s average time with optimized parameters. Comparisons of execution entity credibility, error counts and defense capability further validate the effectiveness of the adaptive optimizations in improving model security and attack resilience. This study provides an efficient mimetic defense methodology for robust cloud data protection.

Keywords: binary random extension code; dynamic heterogeneous redundancy; file encryption system; key update; moving target defense

1 INTRODUCTION

With the rapid development of network technology, Moving Target Defense (MTD) technology divides the system attack surface into data, software, network, and platform layers [1, 2]. Among these, data information is a crucial resource stored in cloud computing, and its security is the primary application of defense technology. Furthermore, given the ever-evolving means and techniques of network attacks, there is a profound exploration of data encryption technology to enhance defense capabilities against network attacks [3-5]. Current defense technologies not only detect and capture known attack vulnerabilities but also involve moving target defense and mimetic security defense against unknown random attack behaviors. To address the security risks of file data, leveraging encryption strategies and systems for secure data storage is required [6, 7]. This research combines Random Binary Extension Codes (RBEC), Moving Target Defense, and encryption systems to construct an adaptive Dynamic Heterogeneous Redundancy (DHR) mimetic defense model. The defense model for cloud computing data security not only enhances data redundancy features but also increases the difficulty of data attacks and promotes technological progress in data security defense capabilities. The research has four parts. The first part provides an exposition of current research findings. The second part involves an analysis of file encryption systems and data storage patterns to understand the characteristics of attack behavior and to design deceptive defense technologies that can increase the difficulty of attacks. In the third part, an optimization analysis of the Adaptive DHR defense model is performed by combining credibility calculations and heterogeneity transformations. The final part summarizes and narrates the overall research results.

2 LITERATURE REVIEW

The widespread application of cloud computing has brought focused attention to data security and storage. With the rapid development of cloud computing platforms, defense technologies for data security have become a major hot-spot in network security. Analyzing file encryption systems and data storage methods is crucial due to the access and information update requirements for data files. This analysis aims to understand the characteristics of attack behaviors and design deceptive defense technologies [8, 9]. Domestic and international scholarshave conducted significant research on data security and storage technologies. Pawar A B and others addressed issues related to data security and privacy protection in cloud computing. They used dynamic stitching, elliptic curve cryptography, and data encryption standards to enhance data security. Additionally, they employed privacy protection assessment methods to verify and analyze datasets, thereby improving operational efficiency and data access security [10]. Singh R and colleagues addressed cloud computing data security mechanism issues by proposing the use of clustering algorithms and advanced encryption standard algorithms to enhance data sensitivity. They combined these with a method for automatically generating keys to improve system encryption efficiency and data security privacy [11]. Wang P. and team tackled data security model issues by proposing an optimization of system operation and data security through the integration of the Internet of Things (IoT) and blockchain technologies. This approach aimed to improve the effectiveness of the model and enhance the security of data storage [12]. Kefeng F and others focused on cloud service data storage issues, suggesting the use of blockchain technology and audit frameworks to enhance the stability of audit data. They also utilized a hash number model to optimize the security and efficiency of data storage systems. Many studies have utilized blockchain technology and the IoT domain to construct relevant security models [13]. Regarding specific data encryption technologies, considerable progress has been made. Chaudhari A and team addressed issues related to encryption schemes by proposing a homomorphic encryption scheme based on elliptic curves to protect data privacy and enhance the operational efficiency and security defense capabilities of encryption technology [14]. Ziar R A and colleagues addressed key encryption technology issues by proposing the integration of blockchain technology to build smart contracts, thereby enhancing the security performance of user access information [15]. Chen M addressed data encryption processing issues by proposing the use of data encryption standard algorithms. They optimized the algorithm to improve computational

efficiency and increase the difficulty of attacks, demonstrating its high-security defense capability [16]. In the realm of research applications for network risk models and data defense technologies, Wang S et al. addressed issues related to network risk assessment techniques. They proposed a network security risk assessment method based on the fuzzy theory of attack-defense trees and a probability-based network security risk assessment technique. This was further combined with industrial control systems to enhance the scientific validity and reliability of the model, thus providing protective techniques for network security [17]. Regarding data privacy concerns, Ding Y et al. proposed the use of DL models and MIA defense technology to improve the accuracy of attack type classification, thereby enhancing the security protection of data privacy [18]. In summary, despite numerous optimizations and designs by scholars globally for data encryption technologies and algorithm models, there is still a lack of in-depth analysis concerning data redundancy features and attack surface transformations. Therefore, the research that combines RBEC's adaptive DHR mimetic defense model is deemed superior and feasible.

3 CLOUD COMPUTING DATA ENCRYPTION STORAGE MODES AND DEFENSE TECHNOLOGIES

The security of cloud computing data storage in static network systems makes it susceptible to deeper network attacks. MTD technology and mimetic secure defense technology serve as proactive network security defenses. They are combined with encrypted file systems and dynamic data transformation to optimize the DHR mimetic defense model and enhance its performance.

3.1 Design of File Encryption System and DHR Model Design

The security issues of data storage in cloud computing demand key management methods. The characteristics of dynamic data are considered by applying stream encryption and symmetric encryption. The key updating strategy ensures that file keys are dynamic, thereby improving the data security of the network system [19, 20]. The data dynamic encryption includes a Private Key Generator (PKG), user groups, file storage servers, and key management servers. The encryption method employs stream encryption and symmetric encryption, namely Advanced Encryption Standard (AES), for double-layered file encryption, as illustrated in Fig. 1.

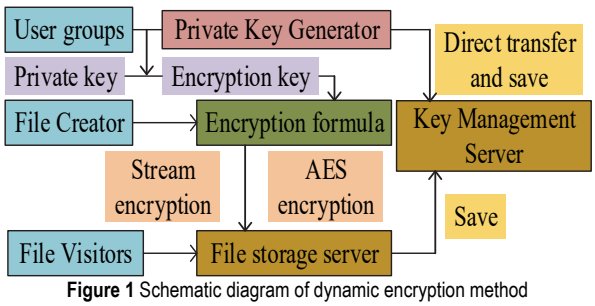


Figure 1 Schematic diagram of dynamic encryption method

Fig. 1 depicts the process wherein encrypted files are stored on file storage servers upon creation. The PKG collaborates with user groups to generate encryption keys and user private keys. Users can then access files directly through the key management server. The combination of stream encryption and the AES algorithm provides dual protection in the dynamic encryption scheme. Additionally, the key updating strategy adapts to changes in the network system, with different user operations leading to key and encryption method transformations. The irregularity in the key updates of the cloud computing data storage system enhances its security. The dynamic encryption method is designed based on MTD technology, and MTD technology primarily defends against attacks by dynamically transforming the attack surface. Besides, the data storage system dynamically encrypts under different conditions, thereby improving system security. The success rate of an attacker targeting a specific resource in the system is represented by Eq. (1).

$$P = \frac{n}{N} \tag{1}$$

In Eq. (1), N represents the total number of attacks, and n denotes the number of successful attacks. Due to the static nature and stability of cloud computing data storage systems, there is an increased risk of attacks. Therefore, by employing data storage redundancy methods, data storage nodes undergo random transformations to enhance the dynamism and uncertainty of cloud computing data storage systems. This ensures the integrity and security of data. The use of RBEC for data encoding is explored, as illustrated in Fig. 2.

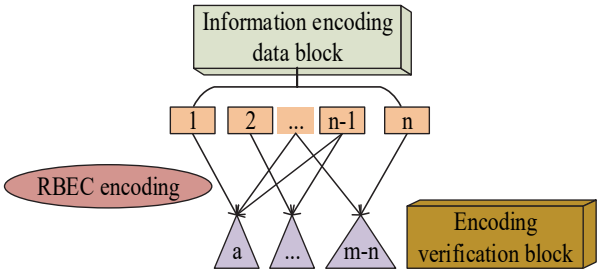


Figure 2 GF (2) finite field data encoding process

From Fig. 2, it is evident that RBEC coding is operated based on Galois finite field encoding. It utilizes simple XOR operations, thereby enhancing computational efficiency. The coding matrix of RBEC contains four parameters: m, n, t, and k, as indicated by Eq. (2).

$$m = n + t + k \tag{2}$$

In Eq. (2), m represents the length of the encoding vector, n is the length of the information, t is the number of redundant bits to ensure the probable completeness of data, k is the maximum number of discardable data blocks, and (m, n) represents the rows and columns of the encoding matrix. The analysis of the dynamic storage mechanism for data redundancy involves four main parts: storage system model, data upload and initialization, data download, and node data recovery. The structure of the storage system model is depicted in Fig. 3.

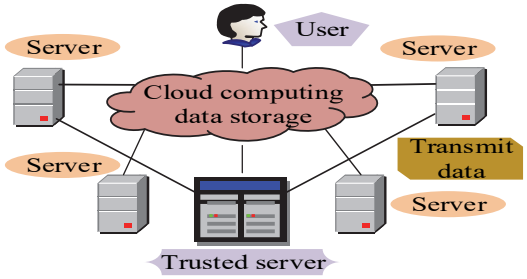


Figure 3 Schematic diagram of cloud node data storage model structure

From Fig. 3, it is observed that users upload data to a trusted server through cloud computing, with all data transmissions being protocol encrypted to ensure the security of information during transmission. The trusted server generates encoding vectors during data initialization and randomly generates parameter vectors during encoding transformations. Other cloud node servers handle data redundancy processing and vector operations. The feasibility of data upload, download, and recovery is ensured by initializing data with RBEC parameters (n, t, k). A matrix consisting of only "0" and "1" is generated, which is combined with the identity matrix, as shown in Eq. (3).

$$H_{(t+k)\cdot n}=\begin{bmatrix} h_{1,1} & h_{1,2} & \cdots & h_{1,n} \\ h_{2,1} & h_{2,2} & \cdots & h_{2,n} \\ \cdots & \cdots & \cdots & \cdots \\ h_{t+k,1} & h_{t+k,2} & \cdots & h_{t+k,n} \end{bmatrix} \quad (3)$$

$$D_{n\cdot n}=\begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

In Eq. (3), $H_{(t+k)\cdot n}$ is a matrix consisting only of the elements "0" and "1," and $D_{n\cdot n}$ is the identity matrix. The encoding matrix is designed to provide sufficient redundancy to the encoded data blocks, ensuring that the vectors within the randomly generated encoding matrix are linearly independent. This guarantees the security of the data storage system. Additionally, based on the defined form of the attack surface, dynamically changing the attack surface increases the difficulty of attacks, forming effective defense methods. However, dynamic changes in the attack surface require handling the range of transformations, unpredictability, and the interval of transformation times. With this foundation, the DHR structure, with its dynamic and random characteristics, enhances attack difficulty in network systems through defense technologies. The specific structure is illustrated in Fig. 4.

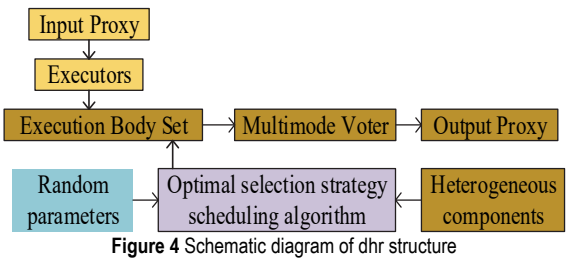


Figure 4 Schematic diagram of dhr structure

From Fig. 4, it is evident that the DHR structure primarily includes an execution body set, a multi-mode voter, an output agent, a dynamic scheduling algorithm, and heterogeneous components. The execution body set is composed of an input agent and multiple execution bodies, while the dynamic scheduling algorithm and the set of heterogeneous components are crucial for dynamically selecting the online execution set. The input agent distributes the system input to various execution bodies, and, after processing through the multi-mode voter, the system's output agent is determined.

3.2 Optimization Design and Security Algorithm of the Adaptive DHRDefense Model

Based on the attack surface transformation and key update strategy of server nodes, the DHR model is enhanced by incorporating reputation feedback and optimizing the execution body set selection algorithm. This leads to the formation of an Adaptive DHR model, which comprises a selector and a cloud storage server pool. The cloud computing data storage server pool includes servers with various heterogeneous types, and the selector continuously controls the execution body set based on the reputation of server execution bodies and the differences between heterogeneous types. The optimization of the Adaptive DHR mimetic defense model involves adding a reputation feedback template, as shown in Fig. 5.

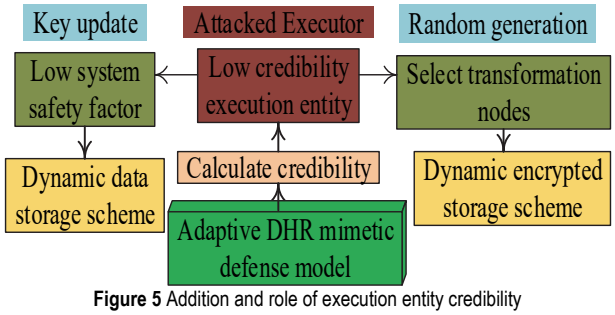


Figure 5 Addition and role of execution entity credibility

From Fig. 5, it is observed that when an execution body is subjected to an attack, its reputation within the execution body set is assessed, leading to appropriate handling of the execution body to address the shortcomings of dynamic encryption mechanisms and data storage modes. When the reputation calculation indicates a low reputation for an execution body, the key update strategy can modify the system's security coefficient detection. If the model's heterogeneous body yields a low reputation, indicating a low-security coefficient, the system automatically updates the encryption key for the file data. When a heterogeneous body acts as an execution body, its heterogeneous body set possesses a reputation index, as shown in Eq. (4).

$$R_p(s)=\frac{\sum_{s\in T_p(s)} t\left(\left|v(s)-y_p(s)\right|\le d_y\right)}{\left|T_p(s)\right|} \quad (4)$$

In Eq. (4), $R_p(s)$ represents the credibility of agent s at time p, $T_p(s)$ is the set of heterogeneous entities p acting as agents at s, $v(s)$ is the decision result of the voting

mechanism, and $y_p(s)$ is the processing result of the agent when $p \in A$, $p = 1, 2, ..., m$. d_y is the maximum difference in processing results between non-attacking agents in the selector and the voting mechanism, which is determined by the data and its complexity. Additionally, the credibility update strategy remains unchanged for agents at different times, as shown in Eq. (5).

$$R_a(s)=\begin{cases}\dfrac{\left|T_p(s-1)\right|\times R_a(s-1)+t\left(\left|v(s)-y_p(s)\right|\le d_y\right)}{\left|T_p(s-1)\right|+1} & a\in x(s)\\ R_a(s-1) & a\notin x(s)\end{cases} \tag{5}$$

In Eq. (5), $R_a(0)$ is the initial credibility of the agent, with $s = 1, 2, ...$, $R(s)$ is the credibility of all heterogeneous entities in the heterogeneous component set, i.e., $R(s) = [R_1(s), R_2(s), ..., R_n(s)]$. The functional expression for the voting mechanism used in agent processing is shown in Eq. (6).

$$v(i)=\begin{cases}0 & i\neq 0\\ 1 & i=0\end{cases} \tag{6}$$

In Eq. (6), $v(i)$ represents the function definition of the voting mechanism. When the difference between the agent's computation result at a given time and the voting mechanism's processing result exceeds d_y, the agent p is susceptible to an attack. Its credibility decreases, and subsequently, the selector clears it and chooses a new agent. However, when the difference does not exceed d_y, the agent's credibility remains unchanged or increases. Due to differences between agents, when the disparity is significant, the system's voting mechanism increases the likelihood of attacks or vulnerability underreporting, thus reducing system security. Therefore, the study focuses on quantifying the differences between agents, as specified in Eq. (7).

$$C_{ij} = F\cdot W^T \quad \begin{cases} F=[f_0,f_1,f_2,f_3]\\ W=[w_0, w_1, w_2, w_3]\end{cases} \tag{7}$$

In Eq. (7), C_{ij} represents the heterogeneity between entities, F is the matrix denoting the differences between the agent software stacks at f_i and f_j layers, and W is the weighted coefficient of differences between various layers of the software stack. Additionally, the difference arises from data errors. When the credibility feedback template sends agents with low credibility to the selector, the selector marks them as attacked agents, clears them, and initializes the credibility of the newly chosen agents in the heterogeneous component server pool. As the DHR model operates with multiple agents simultaneously and emphasizes heterogeneity between agents, the study calculates the difference between unselected heterogeneous entities and selected agents, as shown in Eq. (8).

$$O_i^A = \alpha_\eta \eta_i^A - \alpha_\delta \delta_i^A \tag{8}$$

In Eq. (8), α_η and α_δ are weighting coefficients for the mean and variance, respectively. i and A denote the sets of heterogeneous entities and agents, η_i^A represents the average difference between heterogeneous entities and the agent set, and δ_i^A is the variance of differences between heterogeneous entities and the agent set. Based on the mean and variance differences, the algorithm selects the heterogeneous entity with the highest O_i^A value, where a larger mean difference and smaller variance enhance the correct detection rate by the voting mechanism for the agents. Subsequently, following the credibility update strategy, the calculation formula for the reselection of agent sets is shown in Eq. (9).

$$Q=\sum_{i,j\in A(s)}\left|C_{ij}\right|+\sum_{i\in A(s)}R_a(s) \tag{9}$$

In Eq. (9), the Q-value is the maximum among the selected agents, $\sum_{i,j\in A(s)}\left|C_{ij}\right|$ represents the diversity among various agents, and $\sum_{i\in A(s)}R_a(s)$ is the sum of reputations among the different agents. Therefore, the decision and handling by the selector for the agent set can alter the results of the voting mechanism and reputation feedback, ensuring the data security of the agent set. The security of the adaptive DHR model includes the success attack rate, dynamic transition frequency, and the same error output rate. Dynamic transition frequency occurs in a state without attacks, with random changes in the system, as shown in Eq. (10).

$$\Delta T = T \quad \Delta T=\{\Delta t_1, \Delta t_2, ..., \Delta t_k\} \tag{10}$$

In Eq. (10), $\Delta T = T$ represents the dynamic transitions within one cycle, and $\{\Delta t_1, \Delta t_2, ..., \Delta t_k\}$ is the time interval for each transition. The same error output rate is shown in Eq. (11).

$$\varphi_l(u)=\frac{z'}{z} \tag{11}$$

In Eq. (11), u represents a vulnerability in the attacked agent set, l is the number of the same error results, z is the total number of attacks, and z' is the number of occurrences of the same error output. Specifically, as shown in Eq. (12).

$$\varphi_l(u)=\frac{\sum_{1\le i_1\le i_2\le i_n\le i_m}t(i_1, i_2, ..., i_n, u)}{C_m^n} \tag{12}$$

In Eq. (12), C_m^n is the selected agent set from the heterogeneous component set, $\{Y_1, Y_2, ..., Y_m\}$ is the heterogeneous component set, and $\{A_1, A_2, ..., A_n\}$ is the agent set. Next, a security analysis of the defense process of the adaptive DHR model is performed, first calculating the success rate of a single attack based on the given parameters, as shown in Eq. (13).

$$P_s = \sum_{i=1}^{n} \alpha_{ui} \cdot p_{ui} \cdot \varphi_l(u_i) \cdot I(u_i) \quad (13)$$

In Eq. (13), α_{ui} is the weight of attack vulnerabilities, p_{ui} is the success rate of attack vulnerabilities, $\varphi_l(u_i)$ is the same error output rate, and $I(u_i)$ is the success rate of attack vulnerabilities during dynamic transitions. Then Eq. (14) is obtained.

$$P_s = \frac{\sum_{i=1}^{N}\left[\alpha_{ui} \cdot p_{ui} \cdot I(ui) \cdot \sum_{1 \le i_1 \le i_2 \le i_n \le i_m} t(i_1, i_2, ..., i_n, u)\right]}{C_m^n} \quad (14)$$

In Eq. (14), $\{Y_1, Y_2, ..., Y_m\}$ is the heterogeneous component set, $\{A_1, A_2, ..., A_n\}$ is the agent set, and C_m^n is the selected agent set from the heterogeneous component set. When $I(u_i)$ does not exist in the adaptive DHR defense model, the success value of an attack vulnerability within a certain time is either 0 or 1. Therefore, in the absence of the requirement for attack timing, the success rate of a single attack vulnerability is shown in Eq. (15).

$$P_u = p_u \cdot \varphi_l(u) \cdot I(u) \quad (15)$$

In Eq. (15), u is a vulnerability in the attacked agent set, l is the number of the same error results, and $\varphi_l(u)$ is the same error output rate. Based on the formula relationships and system structure analysis, the specific defense process of the model is shown in Fig. 6.

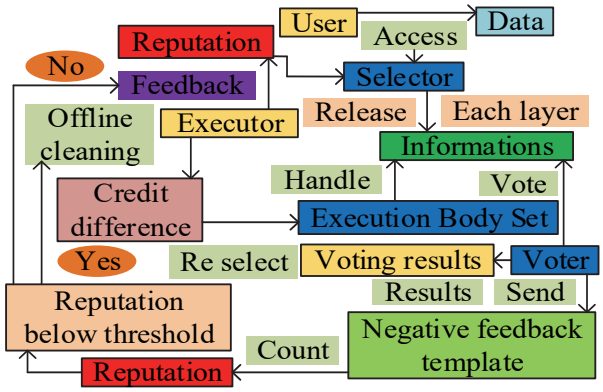


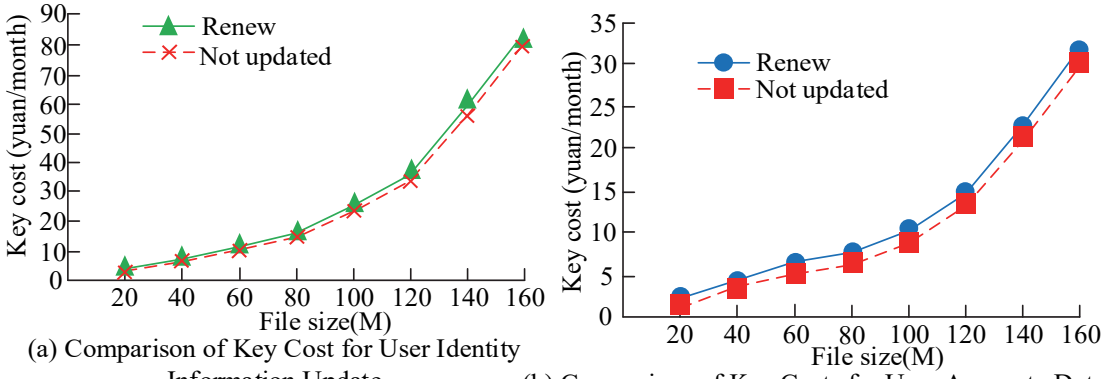
Figure 6 Flow diagram of adaptive DHRDefense model

From Fig. 6, it can be seen that the heterogeneity is increased within the agent set, and the dynamic transition frequency is reduced. Agent algorithms are used to reduce the success rate of attack vulnerabilities and increase the diversity within the agent set, thereby the defense and security of the cloud computing data storage system can be improved.

4 RESULTS AND DISCUSSION

4.1 Experimental Analysis of the Adaptive DHR Data Defense Model

Based on the characteristics of the encrypted file system and data dynamic transformation, and in conjunction with MTD technology and RBEC algorithm, an experimental analysis of the adaptive DHR mimetic defense model was conducted. The study compared the file encryption system based on key update strategies and performed cost comparisons for different file sizes, including 20 M, 40 M, 60 M, 80 M, 100 M, 120 M, 140 M, and 160 M, as shown in Fig. 7.



(a) Comparison of Key Cost for User Identity Information Update

(b) Comparison of Key Costs for User Access to Data

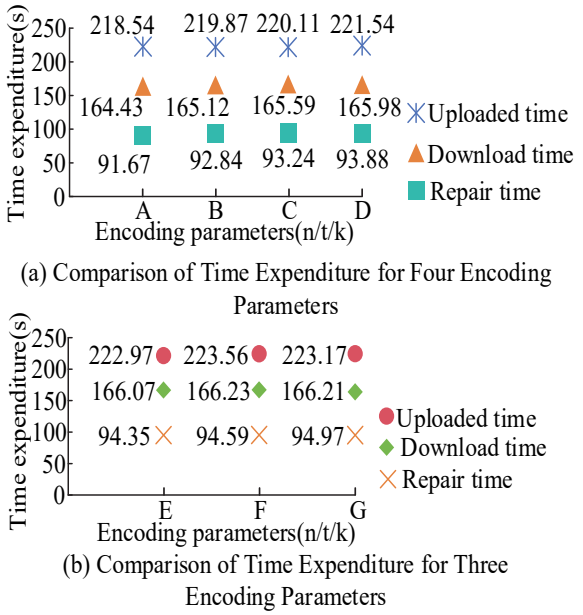
Figure 7 Comparison of key update costs for user information and access data

In Fig. 7a, it was observed that the update of keys resulted in minimal impact on user information changes and data access costs. This implies that user data operations constitute a small proportion of the system's key updates and data storage. In Fig. 7b, the excess expenditure of dynamic encrypted file storage and key update strategy remained within reasonable limits, indicating the security and defensive performance of the key update strategy and dynamic encryption system. Subsequently, a feasibility and accuracy analysis of dynamic redundant storage of data was conducted. It utilizes RBEC parameters: n, t, k, and their relationships, which are represented by serials A-G as shown in Tab. 1.

Table 1 Results of 7 sets of data combinations for RBEC encoding parameters

SerialNumber	n	t	k
A	10	20	10
B	20	5	10
C	20	10	10
D	20	20	10
E	40	20	5
F	40	20	10
G	40	20	20

From Tab. 1, relationships and conditions for RBEC parameters were derived. Here, n represents the data block information code and takes values of 10, 20, and 40. t is the parameter ensuring the full rank of data within the encoding matrix and takes values of 5, 10, and 20. k represents the maximum disposable data value, taking values of 5, 10, and 20. t values are required to be less than k. These three parameters were combined to ensure a high level of data recovery. Seven sets of data combinations were then compared for time expenditure, and 1 GB files were uploaded, downloaded, and node-repaired 100 times under a 100 Mbps broadband connection, as shown in Fig. 8.



(a) Comparison of Time Expenditure for Four Encoding Parameters

(b) Comparison of Time Expenditure for Three Encoding Parameters

Figure 8 Comparison of time expenditure for different encoding parameters

Fig. 8a reveals that an increase in any of the three parameters resulted in an increase in upload, download, and repair times, with a relatively minor impact on system computational efficiency. In Fig. 8b, when encoding parameters were 40, 20, and 10, both upload and download times were highest at 223.56 s and 166.23 s, respectively, while the maximum time expenditure for node repair was with parameters 40, 20, and 20, at 94.97 s. Subsequently, the time expenditure for the conversion of the seven data sets was compared, and the results are illustrated in Fig. 9.

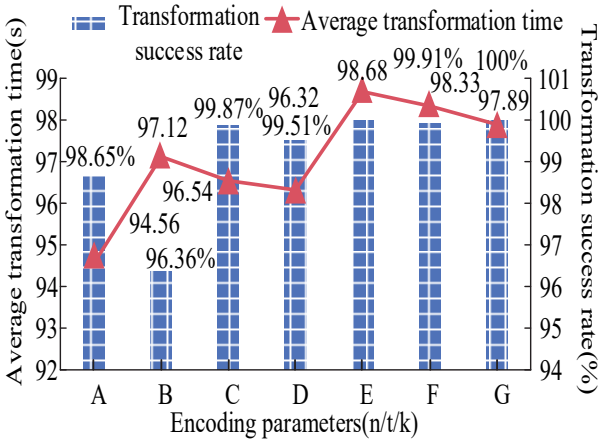
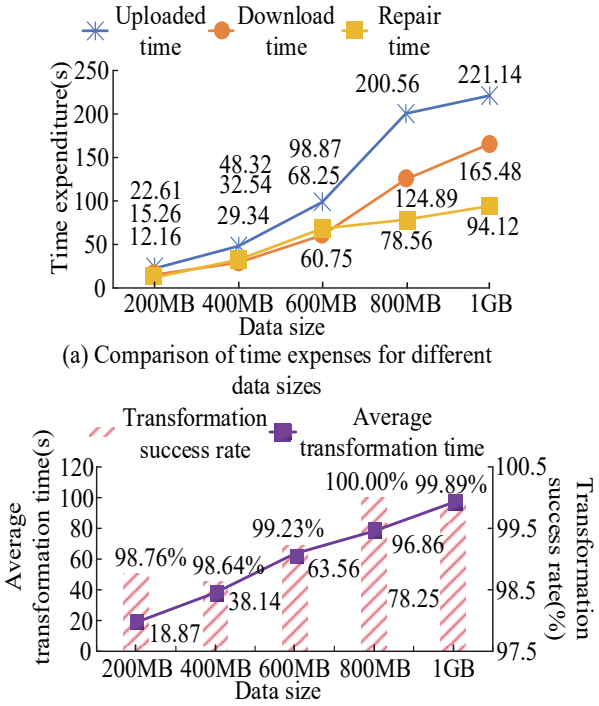


Figure 9 Comparison of dynamic conversion time expenditure for different encoding parameters

Fig. 9 indicates that the conversion rates of the seven data sets were favorable, with success rates consistently exceeding 96%, and an average conversion time of over 94 s. The highest success rates were observed when the encoding parameters were 40, 20, 5, and 40, 20, 20. As the t value increased, the success rate also rose, and smaller n values resulted in lower time expenditures. Therefore, the system primarily focused on selecting reasonable values for n and t to enhance performance. Subsequently, the performance analysis of dynamic conversion with varying data sizes was conducted, using fixed RBEC parameters ($n = 20$, $t = 10$, $k = 10$), and data sizes of 200 MB, 400 MB, 600 MB, 800 MB, and 1 GB, as depicted in Fig. 10.



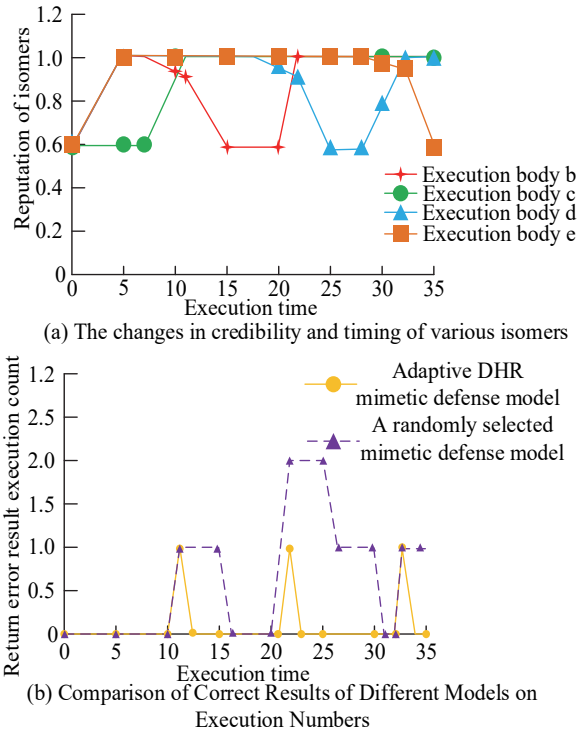
(a) Comparison of time expenses for different data sizes

(b) Comparison of transformation costs for different data sizes

Figure 10 Comparison of time and transformation costs for different data sizes

In Fig. 10a, it was observed that the increasing amount of data led to a continuous increase in the corresponding expenses. Specifically, the expenses for 1 GB upload time, download time, and node repair time were 221.14 s, 165.48 s, and 94.12 s respectively. In Fig. 10b, the performance of the system was evaluated based on the data transformation. The data transformation for 400 MB achieved a success rate of 98.64%, with an average transformation time of 38.14 s. This range of values ensured optimal utilization of data resources and met the performance requirements of the system. The structure and operational flow of the adaptive DHR mimicry defense model were analyzed. The selector and its execution body sets were computationally determined using formulas. Weight values were selected to represent the differences in execution bodies, simplifying the experimental process of the defense model. The weight values for cloud computing database software were set at 0.2, the server weight was set at 0.3, and the operating system weight was set at 0.5. Consequently, a simulation experiment was conducted to analyze the reputation of the

heterogeneous execution body set (execution bodies a-e). The results are shown in Fig. 11.



(a) The changes in credibility and timing of various isomers

(b) Comparison of Correct Results of Different Models on Execution Numbers

Figure 11 Comparison of reputation and correct results of different isomers and models

From Fig. 11a, it was found that none of the execution bodies were attacked during the time interval of 0-10. At the moment when the execution time was 11, the reputation of execution body b dropped to 0.9 and a clearing process was initiated. This process was completed at time 15, with the reputation dropping to 0.5. At execution time 22, execution body d was attacked and cleared, while at execution time 33, execution body e was attacked until time 35, with its reputation also dropping to 0.5. Fig. 11b illustrates the results of the selector's clearing process for different mimicry models. It can be observed that when there were more heterogeneous execution bodies, a lower number of bodies were attacked, indicating a stronger defense capability. Therefore, the adaptive DHR mimicry defense model demonstrated superior security and defense performance, as it selected a higher number of correct execution bodies. Finally, a phased comparison was made on the additional overhead of cloud computing data storage systems. As the system was divided into initial, operational, and static stages, the additional overhead resources of the system included the Central Processing Unit (CPU), Input/Output (I/O), and network transmission. Therefore, the proportion of expenses for the three stages is shown in Tab. 2.

Table 2 Additional overhead results for cloud computing data storage systems

Overhead/Stage	CPU	I/O	Network Transmission	Other
Initial Stage	46%	12%	32%	10%
Operate Phase	5%	61%	31%	3%
Static Phase	31%	7%	51%	11%

According to Tab. 2, the proportion of additional system overhead varied at different stages, with the highest proportion of CPU computational overhead being 46% in the initial stage. The proportion of data storage I/O rapidly increased to 61% during the runtime phase, while in the final static phase, the system's network transmission and communication costs were the main expenses, while the overall cost of other resources decreased. Therefore, the overall operational performance and power consumption of cloud computing data storage systems varied at different stages, and different parameter trade-offs were adopted to achieve the best balance between performance and consumption, thereby ensuring the security of stored data.

4.2 Discussion

Cloud computing enables ubiquitous data access, but it also poses serious security risks to sensitive information. The study proposes a binary random encoding method to encode data to resist attacks. An optimized adaptive heterogeneous redundancy model is designed to improve the adaptability of data protection, thereby increasing the performance of cloud computing data defense. The research combines the robustness of data level encoding with the self-adaptation of system heterogeneity perception, and creates an intelligent pseudo-security mechanism that provides technical reference and feasibility for the security defense analysis of future cloud computing platforms.

5 CONCLUSION

This study addresses the security concerns of cloud computing data storage systems. A systematic analysis and data transformation were conducted on the encrypted file system, key update strategies, and RBEC. Subsequently, a DHR defense model was constructed and subjected to simulation experiments. The experimental outcomes indicated that files ranging from 20 M to 160 M had minimal time overheads for key updates, user information modifications, and data access, thereby not compromising the security of the system data. Consequently, parameters such as n, t, k, and their relationships in RBEC were selected and encoded parameter combinations were established. The minimal times recorded for file upload, download, and node repair were 218.54 s, 164.43 s, and 91.67 s, respectively. The most efficient encoding parameters were 40, 2, and 10, with corresponding times of 223.56 s, 166.23 s, and 94.59 s. Additionally, the successful conversion rate for all data encoding exceeded 96%. Further comparisons were made regarding time overheads for different files, revealing upload times of 221.14 s, download times of 165.48 s, and node repair times of 94.12 s for a 1 GB file. For 400 MB data, the transformation success rate stood at 98.64%, with an average transformation time of 38.1 s. Lastly, comparing the correct execution numbers of different model selectors on isomers, it was shown that the cloud computing data defense model integrating binary random extension codes and adaptive DHR models had superiority in terms of conversion accuracy and time cost, and adaptive credibility calculation and entity selection enhance attack resilience. The technical novelty of this study lies in the synergistic combination of data-level encoding robustness and systemic heterogeneity-aware adaptivity to create an

intelligent mimetic security mechanism. With further extensions to the technique's scalability and generalizability, this pioneering methodology promises to provide an integral security platform for next-generation cloud ecosystems.

6 REFERENCES

[1] Li, P., Lipeng, W., Chunyu, Z., & Jie, C. (2023). Review of text and data mining with AI technology in scientific journals. *Chinese Journal of Scientific and Technical Periodicals, 34*(8), 1007-1013.

[2] Chinthamu, N. & Karukuri, M. (2023). Data Science and Applications. *Journal of Data Science and Intelligent Systems, 1*(2), 83-91. https://doi.org/10.47852/bonviewJDSIS3202837

[3] Lu, Z. & Mohamed, H. (2021). A complex encryption system design implemented by AES. *Journal of Information Security, 12*(2), 177-187. https://doi.org/10.4236/jis.2021.122009

[4] Gnanamalar, A. J., Bhavani, R., Arulini, A. S., & Veerraju, M. S. (2023). CNN-SVM Based Fault Detection, Classification and Location of Multi-terminal VSC-HVDC System. *Journal of Electrical Engineering & Technology, 18*(4), 3335-3347. https://doi.org/10.1007/s42835-023-01391-5

[5] Ciclosi, F. & Massacci, F. (2022). The data protection officer: A ubiquitous role that no one really knows. *IEEE Security & Privacy, 21*(1), 66-77. https://doi.org/10.1109/MSEC.2022.3222115

[6] Yiğit, B., Gür, G., Alagöz, F., & Tellenbach, B. (2023). Network fingerprinting via timing attacks and defense in software defined networks. *Computer Networks, 232*(8), 109850-109859. https://doi.org/10.1016/j.comnet.2023.109850

[7] Hallaji, E., Razavi-Far, R., Saif, M., & Herrera-Viedma, E. (2023). Label noise analysis meets adversarial training: A defense against label poisoning in federated learning. *Knowledge-Based Systems, 266*(4), 1-10. https://doi.org/10.1016/j.knosys.2023.110384

[8] Ali, U., Sahni, S. A. R., & Khan, O. (2023). Characterization of timing-based software side-channel attacks and mitigations on network-on-chip hardware. *ACM Journal on Emerging Technologies in Computing Systems, 19*(3), 1-23. https://doi.org/10.1145/3585519

[9] Elmasry, W., Akbulut, A., & Zaim, A. H. (2021). A design of an integrated cloud-based intrusion detection system with third party cloud service. *Open Computer Science, 11*(1), 365-379. https://doi.org/10.1515/comp-2020-0214

[10] Pawar, A. B., Ghumbre, S. U., & Jogdand, R. M. (2023). Privacy preserving model-based authentication and data security in cloud computing. *International Journal of Pervasive Computing and Communications, 19*(2), 173-190. https://doi.org/10.1108/IJPCC-11-2020-0193

[11] Singh, R. & Pateriya, R. (2021). An efficient data security mechanism based on data groups in cloud computing environment. *Solid State Technology, 64*(2), 131-141.

[12] Wang, P. & Susilo, W. (2021). Data Security Storage Model of the Internet of Things Based on Blockchain. *Computer Systems Science & Engineering, 36*(1).213-224. https://doi.org/10.32604/csse.2021.014541

[13] Kefeng, F., Fei, L., Haiyang, Y., & Zhen, Y. (2021). A Blockchain-Based Flexible Data Auditing Scheme for the Cloud Service. *Chinese Journal of Electronics, 30*(6), 1159-1166. https://doi.org/10.1049/cje.2021.08.011

[14] Chaudhari, A. & Bansode, R. (2021). Survey on securing IoT data using homomorphic encryption scheme. *International Journal of Engineering, 10*(4), 76-81. https://doi.org/10.35940/ijeat.D2333.0410421

[15] Ziar, R. A., Irfanullah, S., Khan, W. U., & Salam, A. (2021). Privacy preservation for on-chain data in the permissionless blockchain using symmetric key encryption and smart contract. *Mehran University Research Journal Of Engineering & Technology, 40*(2), 305-313. https://doi.org/10.22581/muet1982.2102.05

[16] Chen, M. (2021). Accounting data encryption processing based on data encryption standard algorithm. *Complexity, 2021*(5), 1-12. https://doi.org/10.1155/2021/7212688

[17] Wang, S., Ding, L., Sui, H., & Gu, Z. (2021). Cybersecurity risk assessment method of ICS based on attack-defense tree model. *Journal of Intelligent & Fuzzy Systems, 40*(6), 10475-10488. https://doi.org/10.3233/JIFS-201126

[18] Ding, Y., Huang, P., Liang, H., Yuan, F., & Wang, H. (2023). Output regeneration defense against membership inference attacks for protecting data privacy. *International Journal of Web Information Systems, 19*(2), 61-79. https://doi.org/10.1108/IJWIS-03-2023-0050

[19] Gupta, R., Dharadhar, S., & Churi, P. (2021). CloudJS: novel cloud-based design framework for text-file encryption. *World Journal of Engineering, 20*(3), 472-485. https://doi.org/10.1108/WJE-04-2021-0234

[20] Udayakumar, P. & Rajagopalan, N. (2022). (Retracted) Blockchain enabled secure image transmission and diagnosis scheme in medical cyber-physical systems. *Journal of Electronic Imaging, 31*(6), 062002-062018. https://doi.org/10.1117/1.JEI.31.6.062002

Contact information:

Xing YANG
(Corresponding author)
Geely University of China,
Chengdu, Sichuan, 610000, China
E-mail: 18980091368@163.com

Ke XIANG
Sichuan Post and Telecommunication College,
Chengdu, Sichuan, 610000, China
E-mail: xiangke@sptc.edu.cn

JiLan HUANG
Geely University of China,
Chengdu, Sichuan, 610000, China
E-mail: hlhuanglan@163.com