# Randomized Block Size (RBS) Model for Secure Data Storage in Distributed Server

**Keshav Sinha[1*], Partha Paul[2], and Amritanjali[1]**
[1] Birla Institute of Technology, Mesra, Ranchi
Jharkhand - India
[e-mail: keshav.sinha@yandex.com, amritanjali@bitmesra.ac.in]
[2] Sarala Birla University, Ranchi
Jharkhand - India
[e-mail: p_india@rediffmail.com]
*Corresponding author: Keshav Sinha

*Received October 3, 2020; revised August 28, 2021; accepted October 18, 2021; published December 31, 2021*

## Abstract

Today distributed data storage service are being widely used. However lack of proper means of security makes the user data vulnerable. In this work, we propose a Randomized Block Size (RBS) model for secure data storage in distributed environments. The model work with multifold block sizes encrypted with the Chinese Remainder Theorem-based RSA (C-RSA) technique for end-to-end security of multimedia data. The proposed RBS model has a key generation phase (KGP) for constructing asymmetric keys, and a rand generation phase (RGP) for applying optimal asymmetric encryption padding (OAEP) to the original message. The experimental results obtained with text and image files show that the post encryption file size is not much affected, and data is efficiently encrypted while storing at the distributed storage server (DSS). The parameters such as ciphertext size, encryption time, and throughput have been considered for performance evaluation, whereas statistical analysis like similarity measurement, correlation coefficient, histogram, and entropy analysis uses to check image pixels deviation. The number of pixels change rate (NPCR) and unified averaged changed intensity (UACI) were used to check the strength of the proposed encryption technique. The proposed model is robust with high resilience against eavesdropping, insider attack, and chosen-plaintext attack.

***Keywords:*** Distributed Storage Server (DSS), Key Management Server (KMS), Chinese Reminder Theorem (CRT), Key Generation Phase (KGP), Rand Generation Phase (RGP)

A preliminary version of this paper appeared in NCCS 2018, Nov 03-04, Jharkhand, India. This version includes a concrete analysis and supporting implementation results on Image and Text encryption. The result is analyzed based on the RBS model for secure multimedia data stored in the distributed server.

**http://doi.org/10.3837/tiis.2021.12.014** **ISSN : 1976-7277**

# 1. Introduction

In a fast-growing and multitasking world, data has become an energy drink for every user. Every industrialist is generating a multi-billion ($) business using data analytics. It means that the transmission of any personal data requires security. However, in the current scenario, people are facing three different types of threats (i) Computer threats (protecting the data on the computer), (ii) Network threats (data protection during the transmission phase), and (iii) Content threats (protection of intellectual property and provide trustworthiness of the data). This paper refers to offline/online data, primarily e-content (multimedia data) for study.

## 1.1. Cryptographic Approaches

The distributed environment is a giant pool of data, where every user is interlinked, and anyone can download or share their e-content. The challenging part for the content provider is to identify the authenticity of the receiver. There are two different approaches present to overcome the problem of data security; (i) Digital Right Management (DRM) plays an essential role in the protection of the e-content, and (ii) Cryptography technique has been used to secure both physical and digital content [1]. The notion of cryptography has based on symmetric (both parties use the same key) and asymmetric (pair of keys used by both parties) encryption keys that provide strong privacy against any attack. The cryptographic algorithm also takes input in the stream (bit by bit) or block (a chunk of bits) for encryption [2]. The classical notion of privacy is achieved by using semantic security that provides safety under attack, but it also inherits the properties of randomized encryption schemes such as indexing of size and searching each block from a large storage system [3]. The randomized encryption schemes mainly depend on the entropy of plaintext, which is detrimental for lightweight application devices and recovered the drawback of semantic security. If the encryption algorithm is deterministic, file size indexing and searching processes are fast for large datasets [4]. In randomized encryption, the plaintext ‘m’ is divided into consecutive small blocks $m_i = (m_1 \| m_2 \| \ldots \| m_n)$, where each block is encrypted independently. The incremental cryptography uses to encrypt large plaintext, where it takes the document ‘M’ for encryption; next time a single bit flip has performed to create a new version of the document ‘M’, it should rapidly encrypt the document in polynomial-time of log |M| [5]. The large amount of data encrypting with the deterministic algorithm is very much problematic. This incremental scheme feature is widely employed in the digital signature, hashing, and encryption techniques [6].

## 1.2. Open Issues of Distributed Resources and Data Center

Resource organization in a distributed environment is one of the challenging tasks. The unpredictable failure of distributed server cause problem of availability [13]. Here, we have discussed some of the problems such as (i) single point accessibility of resources makes an easy target for the adversary, (ii) the unauthorized access to distributed storage exploit the confidentiality of shared data, (iii) flooding network traffic with unwanted packets to take down the server, and (iv) tracking network packets to perform the unwanted operation. In general, the distributed environment has the following challenges such as (i) Unauthorized access to distributed storage can quickly gain control of the data and exploit the integrity and privacy, and (ii) the adversary uses data manipulation attacks to modify the stored data, collect the user information without authorization, and later use it for earning [36].

The motivation of this work is to provide end-to-end security for multimedia data in a distributed environment and resolve the problems of classical encryption algorithms such as high memory usage, CPU utilization, and time complexity during encryption. The significant contributions of this work can be summed as follows:

a. A Randomized Block Size (RBS) model has been proposed to provide high-level data security while sharing files in distributed environment.
b. The main idea uses the owner-defined block size attribute to provide accuracy and consistency during decryption.
c. The optimal asymmetric encryption padding is used to prevent known-plaintext attacks.
d. The Chinese Remainder Theorem-based RSA (C-RSA) technique reduces the encryption/decryption time and improves data security.
e. The distributed storage server (DSS) does not store any original and encrypted files to avoid data leakage.
f. The confidentiality of data achieves by sharing the encrypted data with authorized users.

The paper's content is organized as follows. Section 1 introduces the cryptographic approaches and open issues of distributed environments. Section 2 presents the distributed storage and cryptography work background, and Section 3 presents the proposed system model. Section 4 discusses the experimental results and analysis. Finally, Section 5 concludes the entire work.

## 2. Related Work

The concept of distributed server has undergone nearly 50 years of development and innovation, and it has become the preferred storage destination for users due to its low cost and scalability. The continuous data storage requirements demand the increased storage node in the distributed cluster. The storage nodes mainly rely on the network to exchange their data, but it does not have any defensive ability to secure against network attacks. Various researchers present distributed architecture that uses trusted third parties (TTP) for storage and provide security against network attacks [11]. Another problem is unauthorized access to the distributed servers and manipulation of the stored data. The traditional distributed storage architecture mainly focused on three things (i) while using the asymmetric key, the server has only a public key and is not aware of the private key of each entity, (ii) the server limits the users by authenticating with TTP, and (iii) the server uses the multiple authentication policy schemes for client [10]. The multimedia security divides into three different types (i) to accomplish a high level of confidentiality of content using proper authentication, (ii) cryptographic algorithm does not affect the quality of data, and (iii) reduce the size of the file after encryption. This section presents some recent work in secure distributed storage and cryptographic technique, which establish the theoretical foundation of our research work. The study has connoted two aspects: the current security issue in distributed storage and the primary active technique to provide data security.

Various distributed server architectures have been proposed to provide proper authentication to users and confidentiality to store data. Here we present some of the works related to the proposed model, such as Tomar and Dhar [8], which have presented a mutual authentication key establishment scheme in a multi-control server environment. Their proposed protocol provides security against DoS and Replay attacks. Ren et al. [9] emulated the identity-based proxy aggregate signature scheme to overcome the problem of transmission and storage space. Their work reduces the communication cost and increases the storage

performance by 20 percent. According to Kaaniche [13], providing cost-efficient storage and transmission architecture is a very challenging task. Data loss and privacy are two critical issues that clients face in the cloud storage environment. A Security-Aware Efficient Distributed Storage model has been proposed by Y. Li et al. [14]. Their model divides the data and stores it in a distributed server to avoid direct data interaction with the service provider. Ferretti et al. [15] proposed the Bloom filter to detect unauthorized modification on outsourced data. Their proposed filter provides confidentiality and integrity to data during transmission. Users are continuously facing the problem of copyright and privacy protection of data in distributed servers. H. Li et al. [16] proposed the Secure Media Cloud, where edge servers have been used to store the video file. The secure-ABAC access control protocol has been used to authorize the users for sharing the data. The Re-encryption and Fingerprinting scheme has been proposed by L. Xiong et al. [17]. The work precisely aimed at controlling the piracy of e-content. It also maintains the privacy and copyright of the content in a semi-trusted environment.

Distributed storage is also concerned about key and storage size, reliability, accessibility, and confidentiality. Laia et al. [18] presented Blum-Blum-Shub (BBS) as an external key generator in the Data Encryption Standard (DES) algorithm. The approach relies on block-based encryption, which requires 56 bits of key size and is vulnerable to security threats. The external generator increases the security, and it does not create any burden for determining the keys of the algorithm. Kumar and Rana [22] proposed the modified AES algorithm for data security. This work mainly focused on enhancing the algorithm's performance by adding one more round in the AES structure. The modified AES compares with traditional algorithms (DES, 3DES, and AES), and it concludes that adding one more round creates a huge difference and provides more security to data during the attacks. Kandar et al. [39] proposed the bit shifting scheme for image encryption. Here the image is partitioned into small blocks, the random permutations on row and column pixels are performed to encrypt the image. Dahua & Kuo [30] proposed the random rotation and partition technique. Here the author uses the randomized rotation of each block to achieve security against ciphertext-only attacks.

In a client-server environment, multiple user authentication uses the key sharing scheme where both client and server have their private key for authentication, and it does not share with the TTP to protect against traditional attacks [7]. W. N. A. Ruzai et al. [19] proposed the continued fraction-based RSA cryptosystem, which uses mathematical problems like factorizing large prime numbers for key generation. This approach uses the modified Euler quotient for essential relation and performs a continuous midpoint subdivision strategy for key distribution. However, every algorithm has some drawbacks, and Fujisaki et al. [20] have presented the two pitfalls of RSA cryptosystem (i) deterministic and (ii) multiplicative property. The Optimal Asymmetric Encryption Padding (OAEP) technique has been used to overcome the RSA cryptosystem's pitfall by adding redundancy and randomness to a message before encryption [21]. The systems like cloud-based distributed storage primarily depend on replication and cryptographic technique to provide data security. The cryptographic techniques require tremendous computational power while performing encryption in the cloud environment. Paul et al. [12] introduced a public key cryptographic technique for video file encryption in a cloud environment. The work aims to store data securely in the cloud server and reduce computational time. Enayatifar et al. [31] proposed a synchronous permutation and diffusion technique to encrypt a grayscale image. The permutation uses the deoxyribonucleic acid (DNA) based chaotic map to permute the image pixel. Nematzadeh et al. [32] combine the concept of Binary Search Tree (BST) and Deoxyribonucleic Acid (DNA) for image encryption. However, the approach requires extra space, which becomes the limitation of the

algorithm. The comparison of various classical algorithms such as (Blowfish, DES, 3DES, AES, and RSA) has been performed by Patil et al. [33]. The main contribution is algorithm analyzes in terms of speed, data size, and security. Sohal & Sharma [34] present the BDNA structure-based symmetric key technique for text file encryption in the cloud environment. The scheme is used the encoding technique at a bit level.

Data security has also been achieved by XORing the pseudo-random bit with the original message, which creates redundancy and is unreadable to the attackers. J. von Neumann [23] had first introduced the mathematical concept of the Middle-Square Method (MSM) in 1946. The technique generates a little keyspace random bit, which is not suitable for any cryptographic applications. In 1958, W. E. Thomson and A. Rotenberg [24] proposed the Linear Congruential Generator (LCG), which uses a discontinuous linear equation for sequence generation. According to the NIST randomness test, the generated sequence has sizeable keyspace and is cryptographically secure, suitable for key generation and random padding. The application of chaos theory is another approach for key generation and encryption. The method uses the logic gates concept, which is morphed into each other to create chaotic behavior. The generic chaotic system is in the form of the Logistic map and Tent map [25]. Kaur et al. [26] proposed a 2D image encryption technique using a multiparameter fractional Fourier transform-based chaotic system. The method generates an ample keyspace of sequence and robust against various attacks, and resolves the transmission and storage of encrypted data. Parida et al. [27] proposed the Elliptic Curve Diffie-Hellman key exchange technique for the shared session key. Here authors use the 3D and 4D Arnold cat maps for image data security. Their method creates a high entropy encrypted image suitable for storage and has high resilience against various attacks. Pourasad et al. [28] use the random chaos sequences for securing the image file. The proposed technique has high accuracy, and it is a suitable choice for real-time image encryption. A Chen's chaotic system based on Runge-Kutta mathematical scheme for image encryption has been proposed by Hamza [29]. The approach uses the minimum input of (64 × 64) bits block size for image encryption. The system produced high entropy encrypted data for transmission.

In summary, the recent work for data security in the distributed environment follows mainly two approaches. The first approach is using authentication and authorization mechanism to control the unwanted behavior of the client. The other approach is to protect data by using encryption, symmetric or asymmetric. The performance of the traditional cryptographic technique is limited by the size of the key and data block, insufficient to meet the current demands of applications requiring the storage of large files on distributed servers.

## 3. Proposed Randomized Block Size Model

The proposed system consists of two components: (i) Randomized Block Size (RBS) model for multimedia security, consisting of various access control modules to check authenticity and confidentiality, and (ii) Mathematical representation of the RBS model and Chinese Remainder Theorem-based RSA (C-RSA) encryption technique.

### 3.1 Model Framework

The proposed Randomized Block Size (RBS) model encrypts large data files ensuring min-entropy storage in a distributed environment. According to Mironov et al. [35], the incremental scheme updates the ciphertext adds small change, enabling less time complexity during encryption. In light of this observation, randomized block size encryption is appropriate for large data volumes. **Fig. 1** represents the framework for the RBS model to encrypt the

multimedia data in a client-server environment.

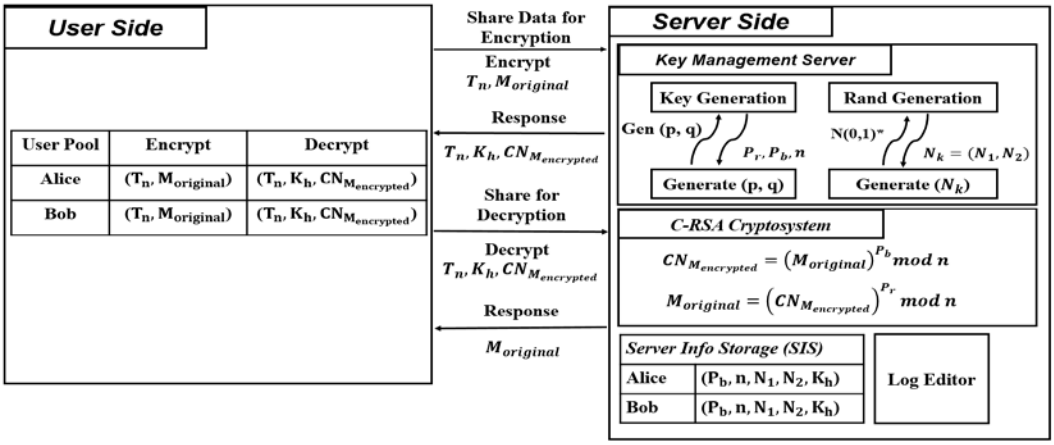



**Fig. 1.** The Randomized Block Size (RBS) model for encryption

**Table 1** presents various parameters uses in the RBS model for encryption and decryption.

**Table 1.** The parameter uses in RBS Model

| Symbols | Description |
|---|---|
| $T_n$ | Number of Threads |
| $M_{original}$ | Original File |
| $P_r$ | Private Key |
| $P_b$ | Public Key |
| n | Multiplicative Moduli |
| $K_h$ | Hash Function |
| $N_1$ | Rand(•) function generates fixed random bits |
| $N_2$ | Rep(•) function generates repeated random bits |
| $CN_{M_{Encrypted}}$ | Encrypted Message |

The encryption uses the improved version of an asymmetric cryptosystem, which combines the Chinese Remainder Theorem (CRT) and RSA, also known as C-RSA, for encryption and decryption. The file encryption uses the owner-defined block size attribute. The RBS model divides the entire original data into equally shared small blocks. Each block is padded with random data before encryption. The encryption key is a hash value, and file encryption and decryption services have been performed on demand. The server does not store a copy of the original or the encrypted file. The RBS model consists of five different components, and their functions are as follows:

- **User Side:** The user's side's basic operation is to upload and retrieve multimedia data from the Distributed Storage Server (DSS). The access controls for users are File_Info (), Share_Type (), Encrypt (), and Decrypt ().
- **Distributed Storage Server (DSS):** The users will share the data in online/offline mode in DSS. Proper authentication is required for all users to access the shared data. The attributes are User_ip_address (), File_Type (), Key_Generation (), Rand_Generation (), and Time_Stamp ().
- **Key Management Server (KMS):** It is a unique implementation in the proposed model, which provides the environment for key generation ($K_i$) and Rand generation ($N_k$). The

key generation module will generate a Public Key ($P_b$), Private Key ($P_r$), and Moduli (n). It also acts as a key splitter, where the public key ($K_{Public}$ ($X_i$)) is shared only with the authenticated users, and the private key ($K_{Private}$ ($Y_i$)) is used for the decryption of data. The attribute for the KMS is File_Type (), $K_{Private}$ ($Y_i$), $K_{Public}$ ($X_i$), and moduli (n).

- **Rand Generation:** The KMS module generates the random bit [$N_k$ = (0, 1)*] for padding with the original file. It mainly consists of two different functions: (i) Rand(•) function, which generates one time fixed random bit ($N_1$) for padding with different original message, and (ii) Rep(•) function generates repeated random bit ($N_2$) for padding with the same original message. The attributes are File_Type (), Rand (•), and Rep (•).
- **Server Info Storage (SIS):** The other feature of the proposed model is maintaining every user's encryption and decryption credentials. The SIS update at every 60ms to store the new request information. The primary feature of this block is to crosscheck the encryption process at every ($Server_{Time}$ (t - 60)).

## 3.2 Functional Description

This section presents the working of the proposed model. The pre-requisite is (i) generation of one-way hash function, (ii) generation of the random number, and (iii) Key generation.

### 3.2.1 One-way Hash Function

The deterministic hash function is represented as ({h (a)} → B), which means that the fixed-length data [a ϵ $(0, 1)^*$] is generated and send to the receiver. Some of the discussions are as follows (i) user has private key [h ($P_r$ || n)], where [h ($P_r$) = h (n)] and [$P_r \neq n$], it is very difficult for an adversary to calculate the input value. (ii) The [h (•)] is the one-way hash function and it is infeasible to extract the input [b = h (a)], and, (iii) the value lies in the interval of [0, $2^n$ - 1] for [B = h (a)].

### 3.2.2 Random Generation Phase (RGP)

The production of a random bit using the Linear Congruential Generator (LCG). The Rand(•) function has a probabilistic nature that generates fixed random bit [$N_k \rightarrow N_1$], and Rep(•) function generates repeated random string [$N_k \rightarrow N_2$]. The generated random bits [$N_1$, $N_2$] are padded with the original message ($M_{original}$) at ΔT time intervals.

### 3.2.3 Key Generation Phase (KGP)

The Key Generation Phase (KGP) comes under the KMS, where the server generates ($P_b$, $P_r$, n) to perform the encryption process.

#### 3.2.3.1 Encryption Process

- Step 1: The ($U_K$) users select two prime numbers, 'p' and 'q'.
- Step 2: Calculate the moduli 'n' (where n = p × q).
- Step 3: Calculate Φ (n) = (p - 1) (q - 1) and λ (n) = LCM (n) = LCM (Φ (n)), where, 'λ' is the Totient function.
- Step 4: Select (e), where the value 'e' lies in-between (e, 1 < e < Φ (n)), such that (e, Φ (n) = 1).
- Step 5: Compute [(Public Key ($P_b$) × Private Key ($P_r$)) = 1 mod Φ (n)].

#### 3.2.3.2 C-RSA: CRT + RSA based Decryption Technique

Before moving further, we explain the mathematical background of CRT-based RSA

decryption. Let us consider $(n_1, n_2, n_3, \ldots, n_k)$ be the pairwise co-prime numbers and greater than '1', and $(a_1, a_2, a_3, \ldots, a_k)$ are positive integer value, which represents using Eq. (1).

$$a^{(\Phi(n))} = 1 \bmod n \tag{1}$$

The Chinese remainder theorem state that the standard solution to the above equation is of the form ($X = X_o + kN$) for some integer 'k' and $N = (n_1 \cdot n_2 \cdot n_3 \cdot \ldots \cdot n_r)$. The solution obtains through solving the Eq. (2).

$$X \equiv \sum_{i=0}^{n} a_i \overline{X_i} N_i \equiv 1 \bmod N \tag{2}$$

Here $\overline{X_i}$ is the modular inverse of '$X_i$' has calculated using Eq. (3).

$$\overline{X_i} N_i \equiv 1 \bmod N \tag{3}$$

Then [$N_i = (N/n_i)$] and it is reduces using the Eq. (4).

$$X = X_0 \bmod N \tag{4}$$

Where [$X_0 = a_i \overline{X_i} N_i$] it completes the proof of CRT. The RSA encryption works on Euler's Totient function denoted as $\Phi(n)$. It gives the number of integers less than 'n' that are co-prime to 'n'. Eq. (5) represents the solution if 'n' is prime.

$$\Phi(n) = n\left(1 - \frac{1}{n}\right) = n - 1 \tag{5}$$

If 'n' is prime, then [$C^{\Phi(n)} \equiv 1 \bmod n$], and GCD(c, n) =1 & n = prime, therefore $C^{(n-1)} \equiv 1 \bmod n$**.** We use the CRT and Totient function for decryption, which is represented using Eq. (6).

$$\begin{cases} D_p = P_r \bmod (p-1) \\ D_q = P_r \bmod (q-1) \end{cases} \tag{6}$$

Here 'p' and 'q' are relatively prime's, and $P_r$ = decryption key. To reduce the power by using the Eq. (7).

$$\begin{cases} X_p = (\text{Encrypted Message})^{D_p} \bmod p \\ X_q = (\text{Encrypted Message})^{D_q} \bmod q \end{cases} \tag{7}$$

The unique solution of the C-RSA algorithm is given by using Eq. (8).

$$M = [(X_p \times q \times q^{-1}) + (X_q \times p \times p^{-1})] \bmod n \tag{8}$$

Here $p^{-1}$ = (p mod q) and $q^{-1}$ = (q mod p) are modulo inverse. This process shows the theoretical steps involve in moduli reduction, and it is 4x times faster in computing than the basic RSA encryption system.

### 3.3 Encryption with Randomized Block Size (RBS)

The working principle of the RBS model is first to take the original data, then has padded the random bits with the original data, then calculate the number of threads and block size, apply the base64 conversion for bit-level encryption, then the message is divide into small sub-blocks, where each block encrypts with the C-RSA cryptosystem. Finally, each block is XOR and 1-bit right shift, and the encrypted data has sent to the end-user. The use of randomized block size for encryption provides the first layer of security. The second layer of security achieves by using random number padding followed by encryption. The mathematical formulation provides a brief insight into the RBS model.

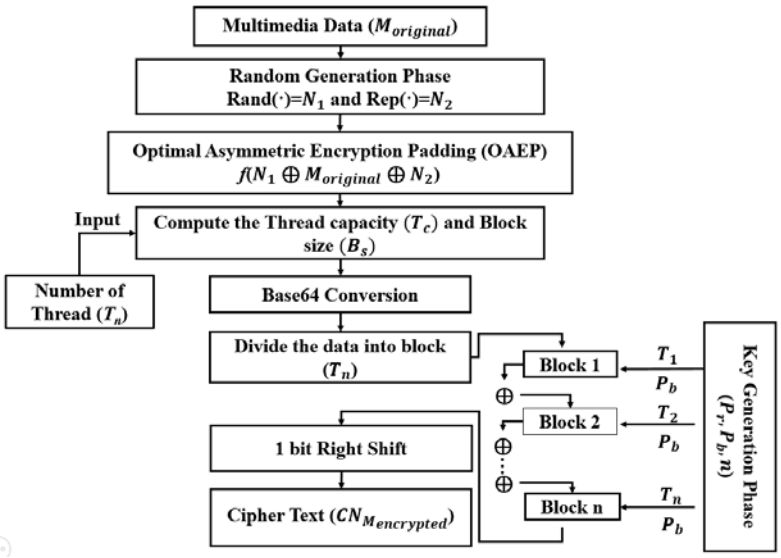



**Fig. 2.** The encryption process for multimedia data

**Fig. 2** represents the encryption process of multimedia data. For encryption, every user requires '$T_n$' and '$M_{original}$'. The pre-requisite to encrypt the data are:

**Step 1:** The server generates the Rand (N1) and Rep (N2) for every $k^{th}$ user.
**Step 2:** The user shares the '$M_{original}$' with DSS.
**Step 3:** The server calculates the following values ($P_b$, $P_r$, n, and $T_n$) for every user.

### 3.3.1 Encryption Process

**Step 1:** The $K^{th}$ users send the '$T_n$' and '$M_{original}$' to the server.
**Step 2:** The server generates the encryption key and random padding using the KMS component. The encryption key is generated in the Key Generation Phase (KGP), and the Rand Generation Phase (RGP) generates random padding. The Optimal Asymmetric Encryption Padding (OAEP) technique uses RGP to create randomness in the original message. The Fixed random padding is applied using Eq. (9).

$$\text{Fixed random } (N_1) = \sum_{i=1}^{k} N_{1_i} \tag{9}$$

Where 'i' is a data request by $k^{th}$ user ($1 \leq i \leq k$). The Repeated random padding is applied using Eq. (10).

$$\text{Repeated random } (N_2) = \sum_{i=1}^{k} \sum_{j=1}^{k} N_{2_{ij}} \tag{10}$$

Here, 'k' is the number of different user messages, and 'm' is the number of requests for that message. The modified message represents by using Eq. (11).

$$N_{M\,original} = (N_1 \oplus M_{original}\,(i) \oplus N_2) \tag{11}$$

A server has to perform Fixed and Repeated padding whenever users request the same data. Here '$N_{1_i}$' denote the value of fixed padding for the $i^{th}$ data request, '$N_{2_{ij}}$' denote the value of repeated padding for $i^{th}$ data at $j^{th}$ request. Then, for each data, we generate the different but fixed value of '$N_1$', but the values of 'N2' change for each different data request. According

to this concept, every request has Rand (•) and Rep (•) function for padding, which is described in **Table 2**.

**Table 2.** Matrix Representation of OAEP

| $N_{1_i}$ | | $M_{original}$ (i) | | $N_{2_{ij}}$ | | | |
|---|---|---|---|---|---|---|---|
| | | | | User 1 | User 2 | ⋯ | User 'm' |
| $N_{1_1}$ | ⊕ | $M_{original}$ (1) | ⊕ | $N_{2_{(1,1)}}$ | $N_{2_{(1,2)}}$ | ⋯ | $N_{2_{(1,m)}}$ |
| $N_{1_2}$ | ⊕ | $M_{original}$ (2) | ⊕ | $N_{2_{(2,1)}}$ | $N_{2_{(2,2)}}$ | ⋯ | $N_{2_{(2,m)}}$ |
| ⋮ | ⊕ | ⋮ | ⊕ | ⋮ | ⋮ | ⋱ | ⋮ |
| $N_{1_k}$ | ⊕ | $M_{original}$ (k) | ⊕ | $N_{2_{(k,1)}}$ | $N_{2_{(k,2)}}$ | ⋯ | $N_{2_{(k,m)}}$ |

**Step 3:** The server has to compute the Thread capacity ($T_c$) and size of block ($B_s$) by using Eq. (12).

$$T_c = \frac{N_{M_{original}(i)}}{T_n} = B_s \tag{12}$$

Here $N_{M_{original}(i)}$ = $k^{th}$ users uploaded the $i^{th}$ data size to the server for encryption.

**Step 4:** Server calculate the Number of Blocks ($B_n$) by using Eq. (13).

$$B_n = \frac{N_{M_{original}(i)}}{B_s} \tag{13}$$

In short, the number of blocks generated is equal to the number of threads because all the threads are synchronously working on the same message together. It means that ($B_s = T_c$) and ($B_n = T_n$).

**Step 5:** Now server converts each block of data into Base64. The conversion of data for $i^{th}$ data block represents by using Eq. (14).

$$\sum_{i=1}^{n} B_{N_{M_{original}(i)}} = \text{Base64.subset}\left(\frac{N_{M_{original}(i)}}{B_n}\right) \tag{14}$$

**Step 6:** The server uses synchronous threading to execute each Base64 block. Eq. (15) represents the encryption process.

$$CN_{M_{encrypted}} = \left(\left[T_{n_1}\left(\left(B_{N_{M_{original(1)}}}\right)^{P_b} \bmod n\right)\right]\right) \oplus \cdots \oplus \left(\left[T_{n_n}\left(\left(B_{N_{M_{original\,(n)}}}\right)^{P_b} \bmod n\right)\right]\right) \tag{15}$$

Here $T_{n(i)} = i^{th}$ thread working on the different data block, where ($i \in [1, n]$), and $B_{N_{M_{original}(i)}} = i^{th}$ data block.

**Step 7:** The 1-bit right shift (>>) is performed on encrypted data, and the final output is store on the Distributed Storage Server.

$$DSS \leftarrow [CN_{M_{encrypted}} = (1\text{bit right shift} >> CN_{M_{encrypted}})]$$

**Step 8:** The server stores every user credential such as [$P_b$, n, $N_1$, $N_2$, and $K_h$] in Server Info Storage (SIS). The private hash key ($K_h$), $T_n$, and encrypted file are sent to the user side.

$$\text{User side} \leftarrow [CN_{M_{encrypted}}, K_h, T_n]$$

### 3.3.2 Decryption Process

Every traditional cryptography algorithm follows the reverse process of encryption, and it performs on the server-side. The steps for decryption is as follows:

**Step 1:** The user upload ciphertext, number of thread and hash private key to the server ($CN_{M_{encrypted}}$, $K_h$, $T_n$).

**Step 2:** The server checks the corresponding information based on hash value ($K_h \rightarrow P_b$, n, $N_1$, $N_2$, $K_h$)

**Step 3:** The server retrieves the private key from hash value [$P_r \leftarrow (n \oplus K_h)$]

**Step 4:** Calculate the Capacity of the block using Eq. (16).

$$C_{c_{CN_{M_{encrypted}}}} = \frac{CN_{M_{encrypted}}}{T_n} \tag{16}$$

**Step 5:** The server calculates the number of blocks using Eq. (17).

$$C_n = \frac{CN_{M_{encrypted}}}{C_{c_{CN_{M_{encrypted}}}}} = T_n \tag{17}$$

**Step 6:** The 1-bit left shift is performed on the encrypted data.

$CN_{M_{encrypted}} \leftarrow (1bit\ left\ shift << CN_{M_{encrypted}})$

**Step 7:** Calculate the size of the blocks ($C_s$) by using Eq. (18).

$$\sum_{i=1}^{n} C_{s_i} = \left(\frac{CN_{M_{encrypted}}}{C_n}\right) \tag{18}$$

**Step 8:** The decryption process of message ($DN_{M_{encrypted}}$) performs by using Eq. (19).

$$DN_{M_{encrypted}} = \left(\left[T_{n_1}\left(\left(C_{s_1}\right)^{P_r} \bmod n\right)\right] \oplus \ldots \oplus \left[T_{n_n}\left(\left(C_{s_n}\right)^{P_r} \bmod n\right)\right]\right) \tag{19}$$

The decomposition of private key and moduli computes using Eq. (6), Eq. (7), and Eq. (8).

**Step 9:** The server performs the base64 conversion using Eq. (20).

$$N_{M_{encrypted}} = B64.subset\left(\frac{DN_{M_{encrypted}}}{C_n}\right) \tag{20}$$

**Step 10:** The original data extract by removing the fixed and repeated padding from the encrypted message.

$$M_{original} \leftarrow (N_1 \oplus N_{M_{encrypted}} \oplus N_2)$$

The work of Log editor is to store the user credential periodically at every 60 sec. It completes the whole working principle of the RBS model.

### 3.3.3 RBS Model Security Advantages

Let us see how the block size affects the data during encryption. We know that ($T_n = B_n$), i.e., Number of thread ($T_n$) ∝ Number of blocks (Bn). During the decryption phase, if the number of threads is changed, the number of blocks is also gets changed. Due to the difference in block size, the resulting decrypted data does not match the original data. It presents the first security layer where only the authorized users have the actual value of ($T_n$) to get the original data. The confidentiality of data in the proposed model has been achieved by using the cryptographic technique. The combination of CRT and RSA is used to provide the end-to-end security of data. Let us see the scenario, how the stored data is safe against attack. Here, Server Info Storage (SIS) is used to store the user's credentials such as [$P_b$, n, $N_1$, $N_2$, and $K_h$], where '$P_b$' and 'n' are large prime numbers that are used for the encryption, $N_1$ and $N_2$ are used for padding with the original file, and the security of data relies on '$K_h$' which is in the form of the hashed value (h ($P_r$ || n)). The adversary cannot decode the information present in '$K_h$' at a limited time, whereas DSS does not store any original or encrypted data. It means that the adversary

only gets the public key and random padding values, which is insufficient to decrypt the message. The theoretical mechanism shows that the adversary cannot perform any attack in the given time. The second layer of security of the RBS model has been achieved by protecting the store data.

## 4. Experimental Results and Analysis

This section provides the experimental result analysis. The experiment simulation requires Oracle-11g for the DSS, where every user uploads their data for encryption. The server provides a GUI interface to upload/download multimedia data files. The functional components of the systems are implemented using Eclipse V 3.5.1, with JDK 8 on the Windows 8.1 platform, having hardware specification of Intel Core i5 CPU with frequency 1.70GHz of the clock with 4GB RAM x 64-bit processor. The paper uses Matlab R2013a for result comparison. The pseudo-random number is generated using online Jsbin simulators for the fast generation of random bits.

### 4.1 Text file Encryption

The text file (.txt) encryption is performed on the randomly generated text from the website (https://www.lipsum.com/). The input file of sizes (5, 10, 15, 20, 25, and 30) kb, having word counts (758, 1516, 2274, 3032, 3790, and 4548) and character counts of (5097, 10194, 15291, 20388, 25485, and 30582). The first step is block size indexing and distribution of each block to separate threads. For proper file size indexing, the following two factors need to be considered (i) the size of data and (ii) negative values. The file size comes in both integer and float formats and can be of odd or even size. The fixed padding is used for the odd size file, and the indexing normally works for even file size. The second issue of file indexing is the negative values of pixels. The default implementation of the Java Runtime Environment (JRE) and Java cryptography extension (JCE) handle the error of negative value. After indexing of file size, the C-RSA technique is applied to each block.

#### 4.1.1 Encryption Efficiency

**Table 3.** The comparison of different ciphertext with given plaintext size

| Plain Text Size (KB) | Ciphertext Size (kb) | | | | | | |
|---|---|---|---|---|---|---|---|
| | **Blowfish** [34] | **AES** [34] | **DES** [34] | **DNA** [34] | **BDNA** [34] | **K-RSA** [12] | **Proposed C-RSA** |
| 5 | 6.76 | 6.86 | 6.86 | 6.67 | 5.83 | 5.21 | 5.25 |
| 10 | 13.52 | 13.70 | 13.70 | 13.33 | 11.66 | 11.66 | 11.70 |
| 15 | 20.27 | 20.55 | 20.55 | 20.0 | 17.5 | 15.43 | 15.53 |
| 20 | 27.03 | 27.39 | 27.39 | 26.67 | 23.34 | 20.89 | 20.59 |
| 25 | 33.78 | 34.25 | 34.25 | 33.33 | 29.17 | 25.79 | 25.59 |
| 30 | 40.5 | 41.09 | 41.06 | 40.0 | 35.0 | 30.51 | 30.28 |

**Table 3** presents the comparison between plaintext and ciphertext sizes (kb). Here the results of the C-RSA technique have been compared with the various traditional encryption algorithms. It is observed that the results of the proposed techniques are better than the methods (Blowfish, AES, DES, DNA, and BDNA) and very close to the K-RSA [34, 12]. It shows that the proposed technique requires less space for storing encrypted text files.

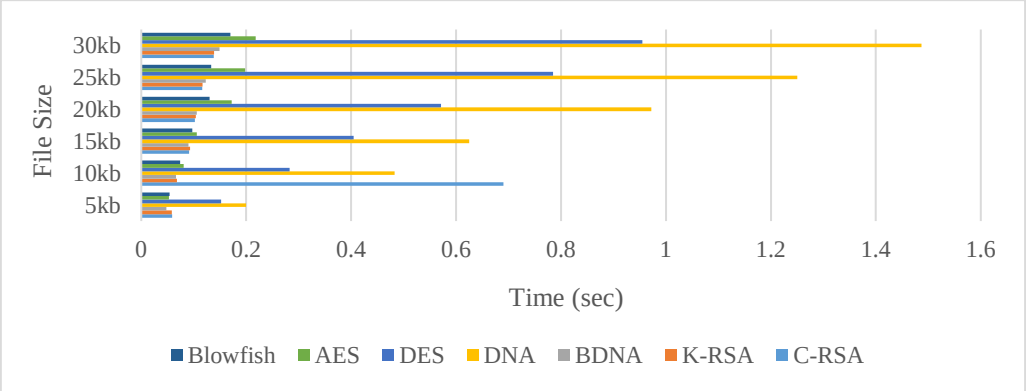



**Fig. 3.** The comparison of encryption time for the various algorithm

**Fig. 3** presents the evaluation of the cryptographic algorithm based on encryption time. Here, the proposed C-RSA technique value is acceptable and matches Blowfish, AES, DES, DNA, and BDNA [34], and is comparable with the K-RSA [12].

### 4.1.2 Throughput

The throughput uses to measure the performance of the algorithm during the time of transmission of data. Eq. (21) presents the throughput calculation where maximizing the throughput provides better performance during transmission.

$$\text{Throughput} = (\text{Plaintext}/\text{Encryption Time}) \tag{21}$$

**Table 4.** Comparison of Throughput

| Algorithm | BDNA [34] | Blowfish [34] | AES [34] | DES [34] | BDNA [34] | K-RSA [12] | Proposed C-RSA |
|---|---|---|---|---|---|---|---|
| **Throughput (kb/sec)** | 180.4 | 159.6 | 126.5 | 33.32 | 20.92 | 181.89 | 183.34 |

**Table 4** presents the comparison of throughput. Here RBS model and C-RSA technique are used for data encryption. The results of the C-RSA technique provide better results than Blowfish, AES, DES, DNA, BDNA, and K-RSA. It shows that the proposed technique requires less data rate during transmission.

## 4.2 Image File Encryption

In this section, the C-RSA technique uses for image (.jpg) encryption. The benchmark image is download from the website (http://www.imageprocessingplace.com /root_files_V3/image_databases.htm). Here, the five grayscale images (Lena, House, Cameraman, Jet Pane, and Boat) are considered for encryption.

### 4.2.1 Encryption Efficiency

**Table 5.** The comparison of various ciphertext sizes after encryption

| File Name | Size (kb) | Encryption Technique | | | | |
|---|---|---|---|---|---|---|
| | | Blowfish | AES | XOR | RSA | C-RSA |
| Lena | 116 | 272 | 273 | 109 | 346 | 118 |
| House | 45 | 106 | 106 | 120 | 181 | 130 |
| Cameraman | 112 | 263 | 264 | 118 | 295 | 128 |
| Jet Plane | 76 | 178 | 179 | 135 | 183 | 147 |
| Boat | 112 | 265 | 266 | 105 | 307 | 114 |

**Table 5** present the comparisons of original and encrypted image sizes. The C-RSA results produce significantly better results than blowfish, AES, and RSA and comparable results with XOR [33]. Finally, it concludes that the C-RSA technique produces the encrypted file requires less memory space at the time of storage.

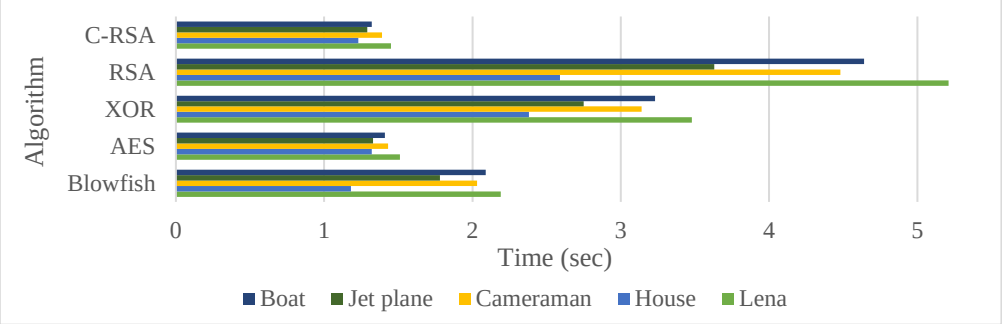



**Fig. 4.** The comparison of decryption time

**Fig. 4** presents the comparison of traditional and proposed encryption techniques based on the decryption time. It observes that a small file size takes more decryption time than a large file. The traditional RSA technique requires more time for decryption [33], whereas AES and C-RSA took almost similar decryption time. Finally, it concludes that the proposed technique has suitable for decrypting large file sizes in a distributed environment.

### 4.2.2 Statistical Analysis

This section presents a statistical analysis of the encrypted image file. We study the similarity analysis using Peak Signal to Noise Ratio (PSNR) computation between the encrypted and original images. The pixel change measures by calculating two parameters, the unified average changing intensity (UACI) and the number of pixels change rate (NPCR). The deviation between the two pixels is measured using correlation coefficient analysis. The histogram analysis uses to visualize the deviation of the original and encrypted image.

#### 4.2.2.1 Similarity Measurement

The PSNR is a ratio between the maximum possible pixels and noise value that affects the image's quality. It uses to measure the similarity between the original and encryption image represented by Eq. (22).

$$\text{PSNR} = 20 \log_{10} \left( \frac{\text{MAX}_f}{\sqrt{\text{MSE}}} \right) \tag{22}$$

Here Mean Squared Error (MSE) has been calculated using Eq. (23).

$$\text{MSE} = \frac{1}{m \times n} \sum_0^{m-1} \sum_0^{n-1} \| Original(i,j) - Encrypt(i,j) \|^2 \tag{23}$$

Here Original (i, j) represents the original image, Encrypt (i, j) represents the encrypted image, m = numbers of rows, i = index of that row, n = number of columns, j = index of that column, and $\text{MAX}_f$ = maximum value of the image pixel.

**Table 6.** Comparison of PSNR values for different techniques

| References | Image | PSNR |
|---|---|---|
| Proposed C-RSA | Cameraman (256 × 256) | 8.2135 |
| | House (256 × 256) | 8.3655 |
| | Jet Plane (256 × 256) | 8.5247 |
| | Lena (256 × 256) | 7.6522 |
| | Boat (256 × 256) | 7.9854 |
| [26] | Cameraman (256 × 256) | 8.6375 |
| | Lena (256 × 256) | 7.7857 |
| [27] Scheme 1 | Lena (256 × 256) | 27.897 |
| [27] Scheme 2 | Lena (256 × 256) | 27.894 |
| [28] | Lena (256 × 256) | 42.612 |
| | Boat (256 × 256) | 38.223 |

**Table 6** compares PSNR values for different techniques when the PSNR values are smaller than the more significant difference between the two images. Here C-RSA technique provides better results than any other methods in [26, 27, and 28].

#### 4.2.2.2 Correlation Coefficient (CF) Analysis

It uses to detect the deviation between two adjacent pixels. In this paper, the results are prepared based on the evaluation of 62,500 adjacent pixels pairs. The correlation concept using the mean and variance of adjacent pixels calculates using Eq. (24).

$$\text{Cov}_{x,y} = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n} (x_i - \bar{x})^2 \sum_{i=1}^{n} (y_i - \bar{y})^2}} \tag{24}$$

Here $\bar{x} = \left(\frac{1}{n}\right) \sum_{i=1}^{n} x_i$, and $\bar{y} = \left(\frac{1}{n}\right) \sum_{i=1}^{n} y_i$, (x) and (y) are two adjacent pixels of horizontal, vertical, and diagonal direction, and 'n' is the total number of pixels. The correlation coefficient of adjacent pixels is maximum (nearer to 1) for the original image, whereas minimum (nearer to 0) for the encrypted image. Furthermore, the results are prepared based on three different directions horizontal, vertical, and diagonal.

**Table 7.** The comparison of the correlation coefficient values

| Algorithm | Images | Direction | Plane | Encrypted |
|---|---|---|---|---|
| | Cameraman (256 × 256) | Horizontal | 0.9688 | -0.0027 |
| | | Vertical | 0.9725 | -0.0018 |
| | | Diagonal | 0.9997 | -0.0004 |
| | House (256 × 256) | Horizontal | 0.9954 | -0.0019 |
| | | Vertical | 0.9932 | -0.0039 |
| | | Diagonal | 1.0000 | -0.0045 |
| | Jet Plane (256 × 256) | Horizontal | 0.9664 | -0.0013 |

| | | | | |
|---|---|---|---|---|
| | | Vertical | 0.9661 | -0.0018 |
| | | Diagonal | 1.0000 | -0.0006 |
| | Lena (256 × 256) | Horizontal | 0.9419 | -0.0031 |
| | | Vertical | 0.9615 | -0.0032 |
| | | Diagonal | 0.9993 | -0.0037 |
| | Boat (256 × 256) | Horizontal | 0.9962 | -0.0018 |
| Proposed C-RSA | | Vertical | 0.9721 | -0.0021 |
| | | Diagonal | 0.9215 | -0.0007 |
| | Lena (256 × 256) | Horizontal | 0.9409 | -0.0006 |
| | | Vertical | 0.9692 | -0.0057 |
| [26] | | Diagonal | 0.9143 | 0.0009 |
| | Cameraman (256 × 256) | Horizontal | 0.9333 | -0.0017 |
| | | Vertical | 0.9565 | -0.0035 |
| | | Diagonal | 0.9059 | 0.0099 |
| | | Horizontal | 0.9718 | 0.0019 |
| [27] Scheme 1 | Lena (256 × 256) | Vertical | 0.9849 | 0.0017 |
| | | Diagonal | 0.9592 | 0.0011 |
| | | Horizontal | 0.9718 | 0.0012 |
| [27] Scheme 2 | Lena (256 × 256) | Vertical | 0.9849 | 0.0016 |
| | | Diagonal | 0.9592 | -0.0007 |
| | | Horizontal | 0.9962 | 0.0117 |
| Chaos [32] | Boat (256 × 256) | Vertical | 0.9721 | 0.0193 |
| | | Diagonal | 0.9215 | 0.0081 |
| | | Horizontal | 0.9962 | 0.0074 |
| Chaos-DNA [32] | Boat (256 × 256) | Vertical | 0.9721 | 0.0073 |
| | | Diagonal | 0.9215 | 0.0049 |
| | | Horizontal | 0.9688 | 0.0024 |
| | Cameraman (256 × 256) | Vertical | 0.9725 | 0.0042 |
| | | Diagonal | 0.9997 | 0.0019 |
| | | Horizontal | 0.9954 | 0.0053 |
| | House (256 × 256) | Vertical | 0.9932 | 0.0091 |
| | | Diagonal | 1.0000 | 0.0032 |
| | | Horizontal | 0.9664 | 0.0055 |
| DNA-BST [32] | Jet Plane (256 × 256) | Vertical | 0.9661 | 0.0086 |
| | | Diagonal | 1.0000 | 0.0092 |
| | | Horizontal | 0.9419 | 0.0015 |
| | Lena (256 × 256) | Vertical | 0.9615 | 0.0019 |
| | | Diagonal | 0.9993 | 0.0012 |
| | | Horizontal | 0.9962 | 0.0084 |
| | Boat (256 × 256) | Vertical | 0.9721 | 0.0084 |
| | | Diagonal | 0.9215 | 0.0012 |

**Table 7** presents the comparison of correlation coefficient values. The experimental results show that the plane image values are nearer to '1', whereas the ciphered images values are nearer to '0' for all the images. The proposed C-RSA technique results for Lena image are better than [27, 32] and comparable with the method [26]. For the Boat image, the C-RSA has performed better than Chaos, Chaos-DNA, and DNA-BST [32].

#### 4.2.2.3 Histogram Analysis

**Table 8.** The Histogram Analysis between original and encrypted image

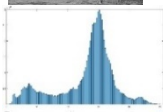
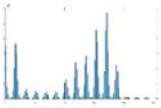
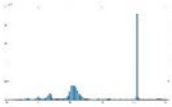
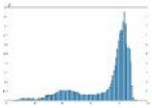
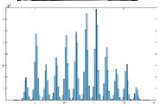
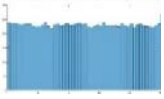
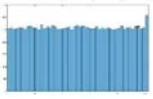
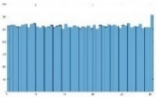
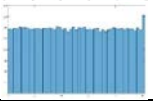
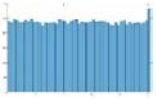
| Type | Boat | Cameraman | House | Jet Plane | Lena |
|---|---|---|---|---|---|
| Original File |  |  |  |  |  |
| Histogram Original File |  |  |  |  |  |
| Histogram Encrypted File |  |  |  |  |  |

**Table 8** presents a comparison of the original and encrypted files. As we see, the original file pixels are not distributed uniformly, whereas, for encrypted files, the frequencies of pixels have been evenly distributing across the histogram graph plot. It indicates no correlation between adjacent pixels after encryption and reveals the excellent quality of cipher images.

#### 4.2.2.4 Differential Attack on Image

The differential attack is a type of attack where adversaries try to tamper with the particular state of the original message. The work of the cryptanalyst is to detect such attacks. The detection uses two different types of mathematical models. The first one detects the absolute Number of Pixels Change Rate (NPCR), and the second detects the Unified Average Change Intensity (UACI) of paired ciphertext. Eq. (25) is used to check the percentage change in pixel, and Eq. (26) is used to calculate the average change of intensity of the pixel.

$$\text{Percentage pixel (NPCR)} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{k} P(i,j)}{n \times k} \times 100 \tag{25}$$

$$\text{Percentage intensity (UACI)} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{k} |C_1(i,j) - C_2(i,j)|}{255 \times n \times k} \times 100 \tag{26}$$

Here $M_{original}$ is encrypted with C-RSA and produced ($CN_{M_{encrypted}} = C1$) whereas adversary tamper the original file and change the 1 bit to create ($M_{original}{}'$), now the message is encrypted with C-RSA having the same initial key to produce ($CN_{M_{encrypted}}{}' = C_2$). If the pixel value (C1 = C2) then P (i, j) = 0, and (C1 ≠ C2) then P (i, j) = 1.

**Table 9.** The comparison of NPCR and UACI values

| Technique | Image | NPCR | UACI |
|---|---|---|---|
| Proposed C-RSA | Cameraman (256 × 256) | 0.9963 | 0.3338 |
| | House (256 × 256) | 0.9960 | 0.3346 |
| | Jet Plane (256 × 256) | 0.9958 | 0.3338 |
| | Lena (256 × 256) | 0.9957 | 0.3326 |
| [26] | Boat (256 × 256) | 0.9958 | 0.3338 |
| | Lena (256 × 256) | 0.9960 | 0.3463 |
| [27] Scheme 1 | Cameraman (256 × 256) | 0.9965 | 0.3453 |
| [27] Scheme 2 | Lena (256 × 256) | 0.9961 | 0.3334 |
| Chaos [32] | Lena (256 × 256) | 0.9963 | 0.3334 |
| Chaos-DNA [32] | Boat (256 × 256) | 0.9925 | 0.3319 |
| | | 7.9948 | 0.3342 |

| | | | |
|---|---|---|---|
| | Cameraman (256 × 256) | 0.9952 | 0.3333 |
| | House (256 × 256) | 0.9929 | 0.3328 |
| DNA-BST [32] | Jet Plane (256 × 256) | 0.9937 | 0.3349 |
| | Lena (256 × 256) | 0.9939 | 0.3340 |
| | Boat (256 × 256) | 0.9967 | 0.3363 |
| [40] | Lena (256 × 256) | 0.9961 | 0.3353 |
| | Cameraman (256 × 256) | 0.9960 | 0.3353 |
| [41] | Lena (256 × 256) | 0.9962 | 0.3346 |
| [42] | Lena (256 × 256) | 0.9961 | 0.3346 |

**Table 9** represents the comparison of NPCR and UACI for the grayscale images. Here, it is observed that all the NPCR values are nearer to 1, and UACI values are nearer to 0.33 for the images considered and compared to the methods in [26, 27, 32, 40, 41, and 42]. For the Boot image, the C-RSA technique result has comparable with the Chaos-DNA [32].

#### 4.2.2.5 Entropy analysis

It is used to measure the degree of uncertainty present in the data. Theoretically, the entropy of a grayscale image is equal to ($2^8 \approx 256$) when all the pixels distribute uniformly. If the encrypted value is close to the 8Sh (Shannon), it is highly robust against attacks. The mathematical formulation for entropy 'E' has represented using Eq. (27).

$$E = - \sum_{i=0}^{m} Q_i \log_2(Q_i) \tag{27}$$

Here '$Q_i$' is the probability of occurrence of gray pixel value 'i'. The probability of each pixel is 1/256 and lies in the range of [0, 255]. To create maximum obscuring in the image, we need to maximize the entropy value.

**Table 10.** The comparisons of entropy value

| Technique | Image | Entropy Value |
|---|---|---|
| | Cameraman (256 × 256) | 7.9976 |
| | House (256 × 256) | 7.9981 |
| Proposed C-RSA | Jet Plane (256 × 256) | 7.9982 |
| | Lena (256 × 256) | 7.9978 |
| | Boat (256 × 256) | 7.9972 |
| [26] | Lena (256 × 256) | 7.9938 |
| | Cameraman (256 × 256) | 7.9846 |
| [27] Scheme 1 | Lena (256 × 256) | 7.993 |
| [27] Scheme 2 | Lena (256 × 256) | 7.993 |
| Chaos [32] | Boat (256 × 256) | 7.9921 |
| Chaos-DNA [32] | Boat (256 × 256) | 7.9932 |
| DNA-BST [32] | Boat (256 × 256) | 7.9932 |
| [40] | Lena (256 × 256) | 7.9973 |
| | Cameraman (256 × 256) | 7.9973 |
| [41] | Lena (256 × 256) | 7.9992 |
| [42] | Lena (256 × 256) | 7.9993 |

**Table 10** presents the entropy values comparison for various traditional and proposed encryption techniques. It is observed that the result proposed by the C-RSA technique is satisfactory and better than those in [26, 27, and 32] for the cameraman, Lena, and Boat images. For Lena image, the result of the C-RSA technique is comparable with the methods in [41, 42].

### 4.3 Security Analysis

The section provides the security analysis of the RBS model based on the various attacks. One of the advantages of the RBS model is KMS which uses the key splitting technique, where asymmetric keys are store and share separately for every individual user. The security analysis on stored data, where adversary (A) performs three types of attacks (i) Eavesdropping attack, (ii) Insider Attack, and (iii) Chosen plaintext attack.

#### 4.3.2 Eavesdropping Attack

In this, adversary (A) captures the data packets and tries to reconstruct the message. The decryption of encrypted messages required a private key. In the RBS model, the private key store in the form of hash $h(P_r \oplus n)$, which is unknown to the adversary. To retrieve the data by calculating the parameters like:

a. Adversary (A) has to calculate the p, q, $N_1$, $N_2$, $P_r$, n, $C_n$, and $C_{c_{CN_{M_{encrypted}}}}$ to decrypt the encrypted message. The value of '$P_b$' and 'n' is to be kept public, but without the value of 'p' and 'q', the adversary will not be able to prime factorized 'n' in a limited time. The Rand Generation can only be calculated correctly by using initial seed values. It is not possible to calculate all the combinations in a limited period.
b. Moreover, the adversary needs some knowledge about user-defined ($T_n$). In order to successfully retrieve the original data. However, $T_n$ is known only to authentic users.

It concludes that the RBS model resists the eavesdropping attack, and the user's credentials are stored securely in Distributed Storage Server (DSS).

#### 4.3.3 Insider Attack

It is performed by a user who has access to the secret information about the KMS and data stored in the DSS. The RBS model has a component Server Info Storage (SIS) responsible for storing the user info. In the SIS, private key and multiplicative moduli (n) are stored in the hash form. Here 'n' is the product of two large prime numbers, which prevents its factorizing in a limited time. Thus without having all the parameters for decryption, the malicious insider attack is not possible.

#### 4.3.4 Chosen-Plaintext Attack (CPA)

In a chosen-plaintext attack, the adversary generates the random plaintext to obtain the corresponding ciphertext. In [37], the author demonstrates the security proof of RSA for ciphertext. However, certain pitfalls of RSA will violate the security paradigm and make the RSA vulnerable. In 2002 [21], the notion of the RSA-OAEP scheme has presented to create verifiable security for ciphertext. The proposed RBS model uses random padding with the original message, where the message divides into sub-blocks. Then, the k-bits padding message is selected and encrypted with the C-RSA technique to provide security for extensive data. To perform CPA, the adversary generates ($M_{original}{}'$) file and encrypt with the publicly shared key ($P_b$) and (n). Now to decrypt the ($M_{encrypted}{}'$) the adversary required ($P_r$) and (n) to exploit the security of original data. The security of the original message is associated with {plaintext = key, encryption, decryption} and it is said to be secure from CPA if and only if it

cannot be decrypted in probabilistic polynomial time, which is represented by Eq. (28).

$$\text{Plaintext}\,[\text{Asymmetric\_Key}_{e,A}^{CPA}] \leq 1 \bmod n \tag{28}$$

For backtracking of (n = 1024 bit) prime multiplicative with the standard computer is very tedious work, and it will take almost ($2^{1024} = 1.79 \times 10^{308}$) combination, or it takes 13 billion years, only quantum computers will calculate the key in a limited year [38].

## 5. Conclusion

The distributed environment is one of the vulnerable places where all the users share their data. In this paper, we have proposed a Randomized Block Size (RBS) model based on the concept of Optimal Asymmetric Encryption Padding (OAEP) and asymmetric encryption technique (C-RSA) for encrypting the text and image files. The proposition of a multifold block size model increases the efficacy and performance of encryption in a distributed environment. The unique implementation of a key management server (KMS) enables the proper authentication and authorization of the users. The combination of OAEP with the original file removes the drawbacks of asymmetric encryption. The Chinese remainder theorem with RSA provides high security and reduces the file size and decryption time. The result indicates that C-RSA provides satisfactory results in terms of security. After different statistical and security analyses, the strength of C-RSA has been concluded. The precise analyses of the RBS model provide better data security for storage and transmission in a distributed environment. The future aspect of this study is to provide end-to-end security of multimedia. Organizations working in social networking, electronic health care, and the military can deploy the proposed RBS model to create a secure multimedia data storage and transmission environment.

## References

[1] S. Rana and D. Mishra, “An authenticated access control framework for digital right management system,” *Multimed. Tools and Appl.*, vol. 80, no. 16, pp. 25255–25270, Apr. 2021. Article (CrossRef Link)

[2] S. Li and P. Liu, “Detection and Forensics of Encryption Behavior of Storage File and Network Transmission Data,” *IEEE Access*, vol. 8, pp. 145833–145842, Aug. 2020. Article (CrossRef Link)

[3] M. Bellare, A. Boldyreva, and A. O’Neill, “Deterministic and Efficiently Searchable Encryption,” in Advances in Cryptology - CRYPTO 2007, 1st ed., vol. 4622, A. Menezes, Ed. New York, NY, USA: Springer, 2007, pp. 535–552.

[4] M. Bellare and D. Micciancio, “A New Paradigm for Collision-Free Hashing: Incrementality at Reduced Cost,” in Advances in Cryptology — EUROCRYPT ’97, 1st ed., vol.1233, W. Fumy, Ed. New York, NY, USA: Springer, 1997, pp. 163–192.

[5] M. Bellare, O. Goldreich, and S. Goldwasser, “Incremental Cryptography: The Case of Hashing and Signing,” in Advances in Cryptology — CRYPTO ’94, 1st ed., vol.839, Y. G. Desmedt, Ed. New York, NY, USA: Springer, 1994, pp. 216–233.

[6] E. Buonanno, J. Katz, and M. Yung, “Incremental Unforgeable Encryption,” in Fast Software Encryption, 1st ed., vol. 2355, M. Matsui, Ed. New York, NY, USA: Springer, 2002, pp. 109–124.

[7] A. Braeken, “Public key versus symmetric key cryptography in client–server authentication protocols,” *Int. J. Inf. Secur.*, vol. 21, pp. 1–12, Mar. 2021. Article (CrossRef Link)

[8] A. Tomar and J. Dhar, “An ECC Based Secure Authentication and Key Exchange Scheme in Multi-server Environment,” *Wireless Pers. Commun.*, vol. 107, no. 1, pp. 351–372, Mar. 2019. Article (CrossRef Link)

[9] Y. Ren, Y. Leng, J. Qi, P. K. Sharma, J. Wang, Z. Almakhadmeh, and A. Tolba, "Multiple cloud storage mechanism based on blockchain in smart homes," *Future Gener. Comput. Syst.*, vol. 115, pp. 304–313, Feb. 2021. Article (CrossRef Link)

[10] A. Majumdar, A. Biswas, K. L. Baishnab,.and S. K. Sood, "DNA Based Cloud Storage Security Framework Using Fuzzy Decision Making Technique," *KSII Transactions on Internet and Information Systems*, vol. 13, no. 7, pp. 3794-3820, Jul. 2019. Article (CrossRef Link)

[11] H. Li, J. Hu, H. Ma, and T. Huang, "The architecture of distributed storage system under mimic defense theory," in *Proc. of 2017 IEEE International Conference on Big Data (Big Data)*, Boston, MA, USA, pp. 2658-2663, 2017. Article (CrossRef Link)

[12] K. Sinha, A. Priya, and P. Paul, "K-RSA: Secure data storage technique for multimedia in cloud data server," *J. Intell. Fuzzy Syst.*, vol. 39, no. 3, pp. 3297–3314, Oct. 2020. Article (CrossRef Link)

[13] N. Kaaniche and M. Laurent, "Data security and privacy preservation in cloud storage environments based on cryptographic mechanisms," *Comput. Commun.*, vol. 111, pp. 120–141, Oct. 2017. Article (CrossRef Link)

[14] Y. Li, K. Gai, L. Qiu, M. Qiu, and H. Zhao, "Intelligent cryptography approach for secure distributed big data storage in cloud computing," *Information Sciences*, vol. 387, pp. 103–115, May 2017. Article (CrossRef Link)

[15] L. Ferretti, M. Marchetti, M. Andreolini, and M. Colajanni, "A symmetric cryptographic scheme for data integrity verification in cloud databases," *Information Sciences*, vol. 422, pp. 497–515, Jan. 2018. Article (CrossRef Link)

[16] H. Li, C. Yang, and J. Liu, "A novel security media cloud framework," *Comput. Electr. Eng.*, vol. 74, pp. 605–615, Mar. 2019. Article (CrossRef Link)

[17] L. Xiong, Z. Xia, X. Chen, and H. J. Shim, "Secure multimedia distribution in cloud computing using re-encryption and fingerprinting," *Multimed. Tools. Appl.*, vol. 78, no. 21, pp. 30297–30313, Jan. 2019. Article (CrossRef Link)

[18] O. Laia, E. M. Zamzami, and Sutarman, "Analysis of Combination Algorithm Data Encryption Standard (DES) and Blum-Blum-Shub (BBS)," *J. Phys. Conf. Ser.*, vol. 1898, no. 1, p. 012017, Jun. 2021. Article (CrossRef Link)

[19] W. N. A. Ruzai, M. R. K. Ariffin, M. A. Asbullah, Z. Mahad, and A. Nawawi, "On the Improvement Attack upon Some Variants of RSA Cryptosystem via the Continued Fractions Method," *IEEE Access*, vol. 8, pp. 80997–81006, 2020. Article (CrossRef Link)

[20] E. Fujisaki, T. Okamoto, D. Pointcheval, and J. Stern, "RSA-OAEP Is Secure under the RSA Assumption," *Journal of Cryptology*, vol. 17, no. 2, pp. 81–104, 2004. Article (CrossRef Link)

[21] N. Cao, A. O'Neill, and M. Zaheri, "Toward RSA-OAEP without Random Oracles," in Public-Key Cryptography – PKC 2020, 1st ed., vol. 12110, A. Kiayias, Ed. New York, NY, USA: Springer, 2020, pp. 279–308.

[22] P. Kumar and S. B. Rana, "Development of modified AES algorithm for data security," *Optik*, vol. 127, no. 4, pp. 2341–2345, Feb. 2016. Article (CrossRef Link)

[23] G. E. Forsythe, "von Neumann's comparison method for random sampling from the normal and other distributions," *Math. Comput.*, vol. 26, no. 120, pp. 817–826, 1972. Article (CrossRef Link)

[24] P. L'Ecuyer, "Tables of linear congruential generators of different sizes and good lattice structure," *Math. Comput.*, vol. 68, no. 225, pp. 249–260, Jan. 1999. Article (CrossRef Link)

[25] T. Munakata, S. Sinha, and W. L. Ditto, "Chaos computing: implementation of fundamental logical gates by chaotic elements," *IEEE Trans. Circuits Sys. I, Fundam. Theory Appl.*, vol. 49, no. 11, pp. 1629–1633, Nov. 2002. Article (CrossRef Link)

[26] G. Kaur, R. Agarwal, and V. Patidar, "Chaos based multiple order optical transform for 2D image encryption," *Eng. Sci. Technol. an Int. J.*, vol. 23, no. 5, pp. 998–1014, Oct. 2020. Article (CrossRef Link)

[27] P. Parida, C. Pradhan, X.-Z. Gao, D. S. Roy, and R. K. Barik, "Image Encryption and Authentication With Elliptic Curve Cryptography and Multidimensional Chaotic Maps," *IEEE Access*, vol. 9, pp. 76191–76204, 2021. Article (CrossRef Link)

[28] Y. Pourasad, R. Ranjbarzadeh, and A. Mardani, "A New Algorithm for Digital Image Encryption Based on Chaos Theory," *Entropy*, vol. 23, no. 3, p. 341, Mar. 2021. Article (CrossRef Link)
[29] R. Hamza, "A novel pseudo random sequence generator for image-cryptographic applications," *J. Inf. Secur. Appl.*, vol. 35, pp. 119–127, Aug. 2017. Article (CrossRef Link)
[30] Dahua Xie and C.-C. J. Kuo, "Multimedia Data Encryption via Random Rotation in Partitioned Bit Streams," in *Proc. of IEEE International Symposium on Circuits and Systems (ISCAS)*, Kobe Japan, pp. 5533-5536, 2005.
[31] R. Enayatifar, A. H. Abdullah, I. F. Isnin, A. Altameem, and M. Lee, "Image encryption using a synchronous permutation-diffusion technique," *Opt. Lasers Eng.*, vol. 90, pp. 146–154, Mar. 2017. Article (CrossRef Link)
[32] H. Nematzadeh, R. Enayatifar, M. Yadollahi, M. Lee, and G. Jeong, "Binary search tree image encryption with DNA," *Optik*, vol. 202, p. 163505, Feb. 2020. Article (CrossRef Link)
[33] P. Patil, P. Narayankar, Narayan D.G., and Meena S.M., "A Comprehensive Evaluation of Cryptographic Algorithms: DES, 3DES, AES, RSA and Blowfish," *Procedia Comput. Sci.*, vol. 78, pp. 617–624, 2016. Article (CrossRef Link)
[34] M. Sohal and S. Sharma, "BDNA-A DNA inspired symmetric key cryptographic technique to secure cloud computing," *J. King Saud Univ. - Comput. Inf. Sci.*, Sep. 2018. Article (CrossRef Link)
[35] I. Mironov, O. Pandey, O. Reingold, and G. Segev, "Incremental Deterministic Public-Key Encryption," *Journal of Cryptology*, vol. 31, no. 1, pp. 134–161, 2018. Article (CrossRef Link)
[36] N. Kaaniche and M. Laurent, "Data security and privacy preservation in cloud storage environments based on cryptographic mechanisms," *Comput. Commun.*, vol. 111, pp. 120–141, Oct. 2017. Article (CrossRef Link)
[37] H. Jeyaprakash, K. Kartheeban, A. K. Sahu, and B. Chokkalingam, "Data Hiding Using PVD and Improving Security Using RSA," *Appl. Secur. Res.*, pp. 1–8, Jul. 2021. Article (CrossRef Link)
[38] Y.-Q. Zhang and X.-Y. Wang, "Analysis and improvement of a chaos-based symmetric image encryption scheme using a bit-level permutation," *Nonlinear Dyn.*, vol. 77, no. 3, pp. 687–698, Mar. 2014. Article (CrossRef Link)
[39] S. Kandar, D. Chaudhuri, A. Bhattacharjee, and B. C. Dhara, "Image encryption using sequence generated by cyclic group," *J. Inf. Secur. Appl.*, vol. 44, pp. 117–129, Feb. 2019. Article (CrossRef Link)
[40] L. Xu, X. Gou, Z. Li, and J. Li, "A novel chaotic image encryption algorithm using block scrambling and dynamic index based diffusion," *Opt. Lasers Eng.*, vol. 91, pp. 41–52, Apr. 2017. Article (CrossRef Link)
[41] Z. Liu, T. Xia, and J. Wang, "Image encryption technique based on new two-dimensional fractional-order discrete chaotic map and Menezes–Vanstone elliptic curve cryptosystem," *Chin. Phys. B*, vol. 27, no. 3, p. 030502, Mar. 2018. Article (CrossRef Link)
[42] Y. Luo, X. Ouyang, J. Liu, and L. Cao, "An Image Encryption Method Based on Elliptic Curve Elgamal Encryption and Chaotic Systems," *IEEE Access*, vol. 7, pp. 38507–38522, Mar. 2019. Article (CrossRef Link)

**Keshav Sinha** received B.E. degree in Computer Science and Engineering from SCSVMV University Kanchipuram, India, in 2013, and an M.E. degree in Software Engineering from the Birla Institute of Technology, Mesra, India, in 2016. As a research scholar, he is currently doing a Ph.D. from BIT, Mesra, in cryptography and network security. As a research scholar, he published several papers in various conferences, journals, and eBooks. His current research interest includes Soft Computing, and Cryptography & Network Security, which provides flexibility in the computer science society.

**Partha Paul** received the B.E (CS) and M.E (CS) degrees from Moscow State University, Russia, in 1998 & 1999, respectively. He did his Ph.D. degree from Birla Institute of Technology, Mesra, Ranchi, India, in 2014. He is currently an Associate Professor in the Department of Computer Science & Engineering, Sarala Birla University, Ranchi, India. He has authored or co-authored more than 40 Papers published in various International Journals and Conference Proceedings. His research interests include Cryptography and Network Security, Artificial Intelligence, Cloud Computing, and Traffic Grooming in Optical WDM.

**Amritanjali** received B.E. degree in Computer Science in 2000, M.E. degree in Software Engineering in 2005 and Ph.D. in Engineering in 2014 from the Birla Institute of Technology, Mesra, Ranchi. After completing her B.E., she worked as Software Engineer at Computer Associates-TCG S/W Pvt. Ltd. where she was responsible for software development and maintenance in various industrial projects. In 2006, she joined the Birla Institute of Technology, Mesra, as Assistant Professor in Computer Science and Engineering. Her research interest is in Wireless Networks, Parallel Computing, Computational Biology, and Machine Learning.