Computers, Materials & Continua
DOI:10.32604/cmc.2021.014466
Article

Tech Science Press

Secure Cloud Data Storage System Using Hybrid Paillier–Blowfish Algorithm

Bijeta Seth[1], Surjeet Dalal[1], Dac-Nhuong Le[2,3,*], Vivek Jaglan[4], Neeraj Dahiya[1], Akshat Agrawal[5], Mayank Mohan Sharma[6], Deo Prakash[7] and K. D. Verma[8]

[1]SRM University, Delhi-NCR, Sonipat, Haryana, 131029, India
[2]Institute of Research and Development, Duy Tan University, Danang, 550000, Vietnam
[3]Department of Information Technology, Duy Tan University, Danang 550000, Vietnam
[4]Graphic Era Hill University, Dehradun, Uttarakhand, 248171, India
[5]Amity University, Haryana, 122412, India
[6]Principal Software Lead, Zillow Inc., USA
[7]School of Computer Science & Engineering, Faculty of Engineering, SMVD University, Kakryal, Katra, 182320, India
[8]Material Science Research Laboratory, Department of Physics, S.V. College, Aligarh, 202001, India
[*]Corresponding Author: Dac-Nhuong Le. Email: ledacnhuong@duytan.edu.vn
Received: 22 September 2020; Accepted: 13 November 2020

Abstract: Cloud computing utilizes enormous clusters of serviceable and manageable resources that can be virtually and dynamically reconfigured in order to deliver optimum resource utilization by exploiting the pay-per-use model. However, concerns around security have been an impediment in the extensive adoption of the cloud computing model. In this regard, advancements in cryptography, accelerated by the wide usage of the internet worldwide, has emerged as a key area in addressing some of these security concerns. In this document, a hybrid cryptographic protocol deploying Blowfish and Paillier encryption algorithms has been presented and its strength compared with the existing hybrid Advanced Encryption Standard (AES) and Rivest Shamir Adleman (RSA) techniques. Algorithms for secure data storage protocol in two phases have been presented. The proposed hybrid protocol endeavors to improve the power of cloud storage through a decrease in computation time and ciphertext size. Simulations have been carried out with Oracle Virtual Box and Fog server used on an Ubuntu 16.04 platform. This grouping of asymmetric and homomorphic procedures has demonstrated enhanced security. Compression usage has helped in decreasing the storage space and computation time. Performance analysis in terms of computation overhead and quality of service parameters like loads of parameters with and without attacks, throughput, and stream length for different modes of block cipher mode has been carried out. Security analysis has been carried out by utilizing the Hardening Index as an audit parameter using Lynis 2.7.1. Similarly, for halting the aforementioned approaches and for regulating traffic, firewall protection has been generated in the chosen hybrid algorithms. Finally, enhancements in the performance of the Paillier and Blowfish hybrid scheme with and without compression

compared to the existing schemes using RSA and AES procedures have been demonstrated.

Keywords: Cryptography; blowfish; homomorphic; paillier; cloud computing

1 Introduction

Recent developments in cloud computing have transformed the field of computing as well as its utilization of resources. Cloud computing has seen unprecedented adoption, leading to the widespread distribution of computing power in order to deliver diverse and adaptable services on demand over the internet through the virtualization of hardware and software. However, security remains a crucial impediment to large scale adoption of cloud computing. While cloud storage has revolutionized the way computation and storage is accomplished by the utilization of external storage sources, cementing the advantages of the cloud computing paradigm is not viable till it ensures a safe computing platform, trusted by customers and establishments.

Cryptography is a basic tool used to design and analyze security protocols in a communication system. It is frequently applied to attain key security goals. Therefore, designing a novel and effective cryptographic algorithm is an active area of research. In this document, a host of cryptographic techniques for establishing security and privacy is presented. Along with this, the limitations of traditional cryptographic systems in the cloud and the need for homomorphic cryptosystems and cryptographic mechanisms employing the Paillier Homomorphic cryptosystems are mentioned in Section 1. In Section 2, the motivational scenario and the primary offerings outlined in this paper are mentioned. Section 3 provides a literature survey. The proposed hybrid cryptographic technique is discussed in Section 4. The suggested algorithm for a secure data storage protocol is provided in Section 5. The numerical analysis of the proposed and existing hybrid cryptosystems is discussed in Section 6. Finally, Section 7 discusses the conclusion and the future scope of research.

1.1 Cryptographic Primer

The proliferation of information and the resultant upsurge in the demand for novel storage solutions and improved network bandwidth has propelled a widespread interest in robust and cost-effective cloud storage services. Nevertheless, despite all its striking capabilities, cloud services, despite its innovative nature, continues to be riddled with numerous security concerns, owing specifically to its outsourcing and multi-tenancy features. Security has turned out to be a vital concern in today's era of widespread internet usage. Viable remote storage schemes necessitate the presence of security features that are inevitable in cloud storage, such as, confidentiality, integrity, and availability. Besides these, factors like authentication, authorization, audit are also important (see Fig. 1) [1–3].

Cryptography delivers elementary algorithms, also termed as cryptographic primitives, combined with a symmetric-key or a public-key structure for securing the data such as encryption, hash function, and digital signature schemes. Some terms related to cryptography are shown in Fig. 2 [1].

1.2 The Cryptographic Mechanism in the Clouds

Although traditional cryptographic systems assure delivery of robust levels of security, numerous deficiencies inherent in them diminish the effectiveness of these traditional schemes, specifically, deficiencies that are exacerbated by the enormous volumes of outsourced records.

This greatly effects the availability and security parameters, as well as the performance of cloud delivery systems and services, especially the parameters related to bandwidth, memory, and power management. Consequently, acceptable levels of cryptographic interventions are of utmost significance.

Figure 1: Security factors

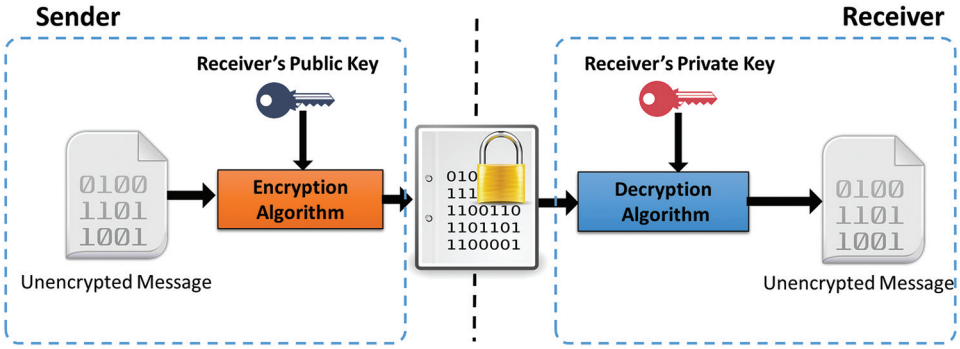

Figure 2: Cryptographic process

The first thing to consider, when employing an unverified security system from a cloud service provider, is the fact that the customer chooses to encipher information that is to be outsourced to distant servers. Here, the huge volumes of data make the practice of traditional asymmetric procedures unsuitable and ineffective. Secondly, the bandwidth ingestion and obtainability requirements of classical algorithms becomes unsuitable in this environment and the efficient sharing of keys for flexible data sharing and the preservation of the confidentiality constraints become paramount. Therefore, the need for innovative interventions in order to overcome the above limitations become necessary.

With regard to cloud computing models, *confidentiality* infers that the customer's sensitive information and computational errands need to be safeguarded from both cloud providers and

illegal consumers. *Integrity* relates to both data integrity as well as to process integrity. The former requires that information must be reliably stored in the cloud without any violation like data loss, etc. Process integrity refers to the computations being performed so that the programs can be implemented without being corrupted by malware, malicious users or any incorrect computation that cannot be easily detected. *Availability* ensures that the data is always physically safeguarded against any attack, etc. *Authentication* implies that unauthorized users cannot access the network. *Authorization* ensures that only authenticated users are allowed to access information. Finally, *audit* enables a systematic evaluation of information security.

Homomorphic cryptography (HC) [4] is a kind of cryptographic scheme whose encryption function is a Homomorphism, and thus conserves group operations accomplished on cipher-texts, guaranteeing confidentiality and protection. In this segment, we first familiarize the reader with the notion of HC and subsequently provide a review of the well-known HC schemes used in cloud storage environments. Next, we discuss Paillier HC. A general privacy-preserving protocol includes two entities, Bob with a secret function f, and Alice with a set of inputs $\{x_1, x_2, x_3, \ldots, x_n\}$ without disclosing her inputs. If Bob's function can be premeditated as a homomorphic function, then Alice can transmit the encoded data to Bob. Bob can then perform the required homomorphic computations, randomize the resultant cipher-text and revert the encrypted result to Alice. Upon decryption, Alice is left with $z = f\{x_1, x_2, \ldots, x_n\}$. A Homomorphic encryption consists of four main functions as shown in Fig. 3 [4].

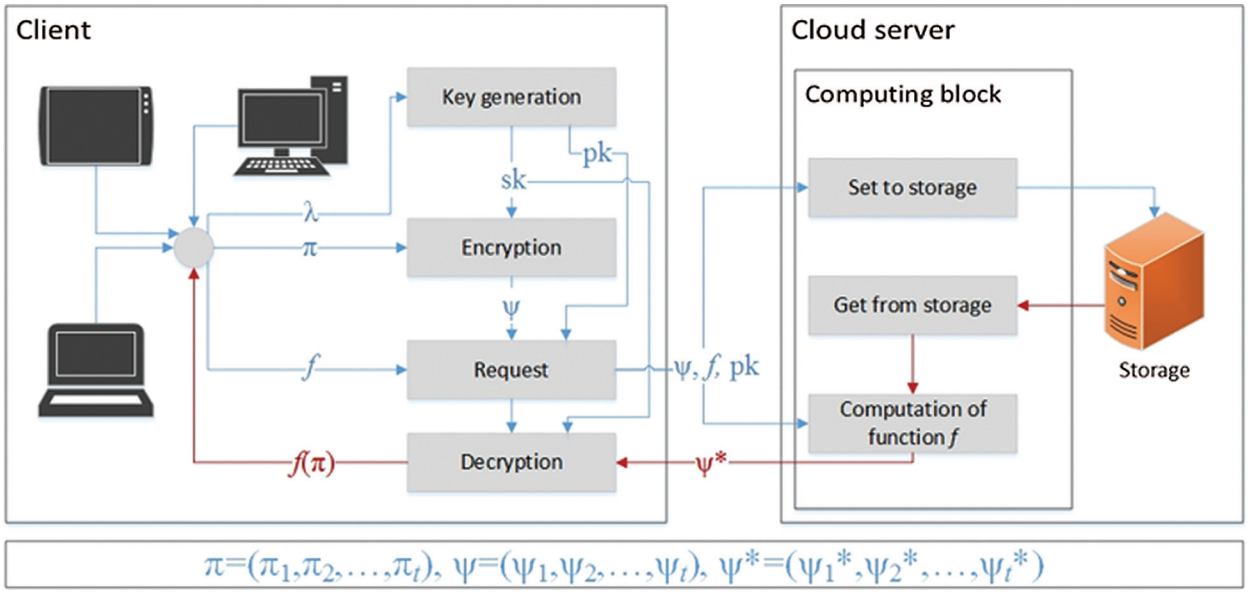

Figure 3: Homomorphic encryption functions

Pascal Paillier developed an additive, probabilistic and non-deterministic asymmetric algorithm intended for communal key cryptography in 1999 [5]. It can compute an encryption of $(m_1 + m_2)$, when only the public key and the two messages m_1 and m_2 are given. It is identical to RSA and uses separate keys for encoding and decoding. It is secure against chosen plaintext attacks and is malleable. It is used in electronic voting, electronic cash, etc.

2 Motivational Background and the Key Offerings of the Paper

The consumers presume that the cloud service suppliers ensure safe transmission of their data that is transmitted from the client's premises to the cloud servers. However, it is imperative that the key safety issues are addressed and carefully handled by the service provider. Therefore, it is important that cloud consumers do the due diligence in choosing suitable cloud service providers to ensure all safety protocols are addressed appropriately.

Following is a summary of key considerations we made in this study:

- The important security factors in terms of confidentiality, integrity, and accessibility of the anticipated scheme with respect to attacks were confirmed.
- The outcomes of the intended method were validated to ensure improved and proficient security measures.
- It was ensured that the proposed hybrid algorithm enhanced the performance of the system as a result of data partitioning.
- It was ensured that the planned algorithm provided a higher hardening index as compared to the existing approach.
- Increased values for throughput and stream length computed in blocks of ciphers were attained.

3 Related Work

As one of the most sought-after technologies of current times, research on cloud computing and especially cloud security is widespread and broad. Taha et al. [6] mentioned that a large amount of data is transmitted through the internet every day. So, the need to transfer data efficiently and effectively is achieved through encryption and decryption techniques. The author discussed several algorithms, namely Triple DES, AES, Blowfish, and Krishna and computed parameters like throughput and time consumed in encrypting data for each algorithm. Wu et al. [7] provided an evaluation of three common encryption techniques for different block sizes of data blocks. Rani et al. [8] underlined that ensuring data security is the most difficult task today in a cloud system. They suggested hybrid encryption, RSA and AES for ensuring trustworthiness, consistency, and efficiency. Waleed et al. [9] mentioned techniques to improve the security of the cloud and the user's privacy. The authors stated that ensuring security and privacy in the cloud system could be a valuable point for cloud database researchers, designers, and vendors. Keerthana et al. [10] utilized IBM Bluemix cloud computing environment to provide the designers with a versatile tool to create their web and portable applications. The detailed architecture of Bluemix and the encryption and decryption algorithms were discussed.

Siregar et al. [11] outlined file security as a technique for providing security parameters through cryptography. The authors compared the Blowfish method with AES and an AES-Blowfish hybrid which provided high throughput. Chaudhary et al. [12] emphasized that security played a vital role in data transmission. The paper dealt with a hybrid approach combining RSA-AES and digital signatures which provided data authentication. 1024 bits public keys of the RSA algorithm were used for verification purposes. Encryption and decryption were performed using hybrid RSA and AES. The result and analysis of AES private key generation and RSA private key and public key generation, Digital signature private and public key generation and a comparison between different cryptographic and hybrid techniques were performed. Jyoti et al. [13] suggested that new threats required strong security tools in order to control and understand the privacy

leaks and to ensure availability and authentication as opposed to attacks. MAES and SHA-512 hybrid approach were employed.

Gajendra et al. [14] emphasized upon the storage and usage of personal data by different users on the cloud that demanded security and protection. The paper used third-party auditors and identity-based encryption to enhance the security of the files. Sharma [15] suggested a system that classified the data according to the security parameters assigned. The basic algorithms of ensemble learning were modified to improve the prediction capability and classification accuracy. Wang et al. [16] proposed that sourcing data led to safety issues. Thus, the truthfulness of sourced data should undergo mandatory checks by clients as a way of safeguarding their data. The computational complexity for key generation, storage, and transfer operations were compared.

Wang et al. [17] suggested that electronic health records benefit from cloud storage. Keyword search on patient health record was employed to enhance data security and the cipher-text generated was stored in the cloud, while proxy re-encryption guaranteed the legitimacy of access and privacy of data. Vyas et al. [18] discussed cloud computing and its deployment models. The cloud storage with its advantages and disadvantages were mentioned. General cloud stowage design and several safety necessities were discussed. The generated meta-data checked the data integrity and was stored at the end of the input file. Metadata was encoded by using AES-256 cryptographic algorithm, and hash of the original file was created using the SHA-256 algorithm.

Thus, in a nutshell, many researchers, as mentioned in the literature stated above, have endeavored relative explorations of encryption procedures for data communication [19–35].

4 The Proposed Hybrid Cryptography Protocol

The anticipated hybrid cryptography protocol intends to construct a competent and safe encryption algorithm constructed on the integration of the encryption procedures to transmit data in cloud-based systems. The planned hybrid system investigates the input and output for a combination of encryption procedures against the existing hybrid algorithm. The projected hybrid approach considers the following combinations of encryption algorithms:

- Amalgamated encoding procedures using RSA and AES techniques.
- The fusion of Paillier and Blowfish encryption processes.

Novelty statement: The purpose of the projected hybrid cryptography algorithm is to define a speedier and safe encryption algorithm in terms of hardening index and safer against malware injection attacks as opposed to previously presented algorithms.

Four setups are expounded upon in the framework:

- Storage of encrypted information over the cloud by means of the RSA-AES hybrid technique.
- Storage of encoded info over the cloud by the RSA-AES hybrid technique without Blocking rules for the attacks.
- Collection of encrypted info over the cloud storage using Paillier–Blowfish Encryption without compression.
- Accumulation of encoded data over the cloud storage using Paillier–Blowfish Encryption with compression and blocking rules to halt attacks using firewall protection.

In the *uploading phase*, before the user stores his data over the cloud, the data is partitioned according to its type every single time. The proposed hybrid cryptographic scheme consists of two phases, one for encryption and the second for decryption. The security of the partitioned

files for each user is ensured using the provided encryption procedure. The partitioned files are encrypted with the hybrid cryptographic algorithm. The various pieces of data are merged at the server and sent to decryption block. Fig. 4 explains the encryption process involved using both hybrid techniques.

In the *decryption process*, the partitioned files are decrypted to provide more protection. The encryption time and decryption times along with the file size and a corresponding hardening index of choosing systems are taken as comparison parameters. Fig. 5 describes the decryption process i.e., the reverse transformation of cipher-text to its original clear text.

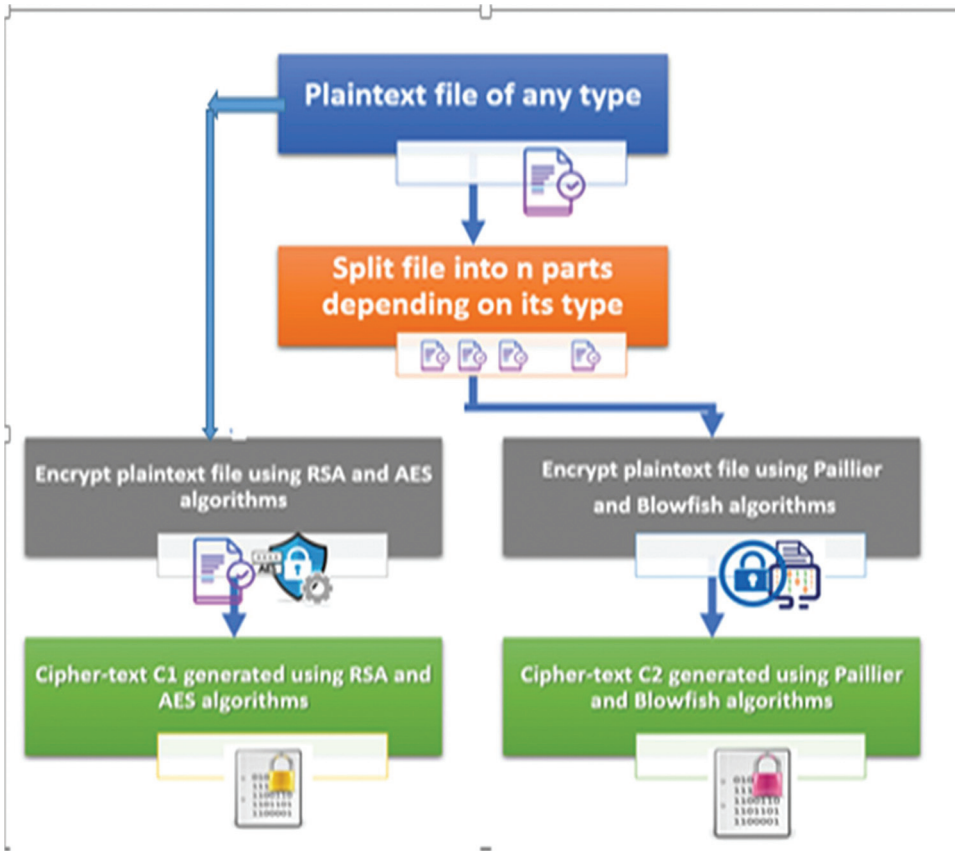

Figure 4: Encryption phase of the hybrid system

Fig. 6 depicts the overall architecture of the system in the case where an *intruder* attacks the server. The *trusted third party* is in interaction with both the client end and the server end. The trusted third-party functionalities include integrity verification which is accomplished by generating the Hashmap of the files and comparing with Hashmap generated at the client end. Any mismatch in both the entries warns the client. User-created firewalls are used to block attacks by employing iptables. Traffic management is done using memory banks.

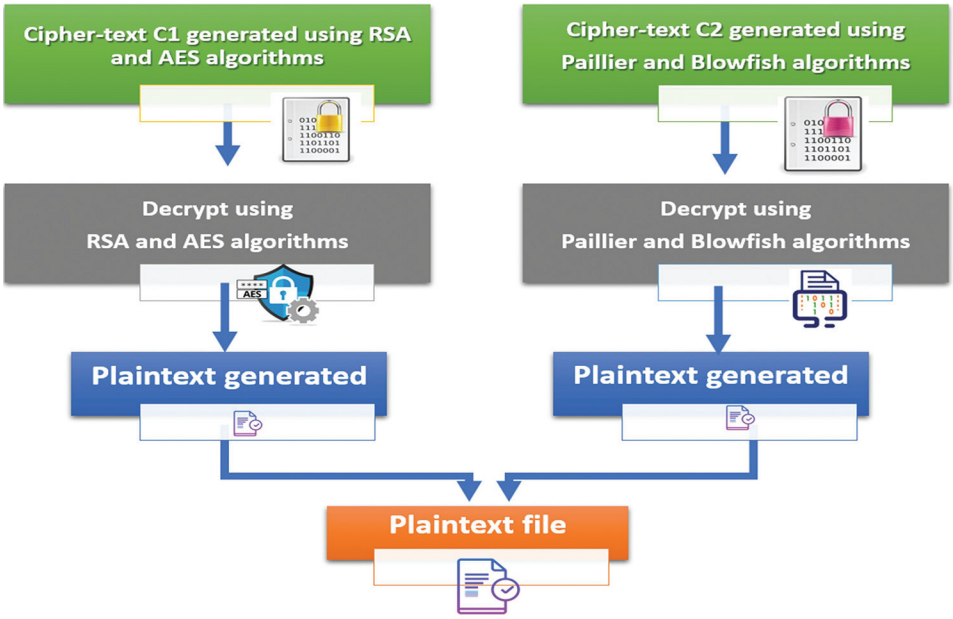

Figure 5: Decryption process of the hybrid system

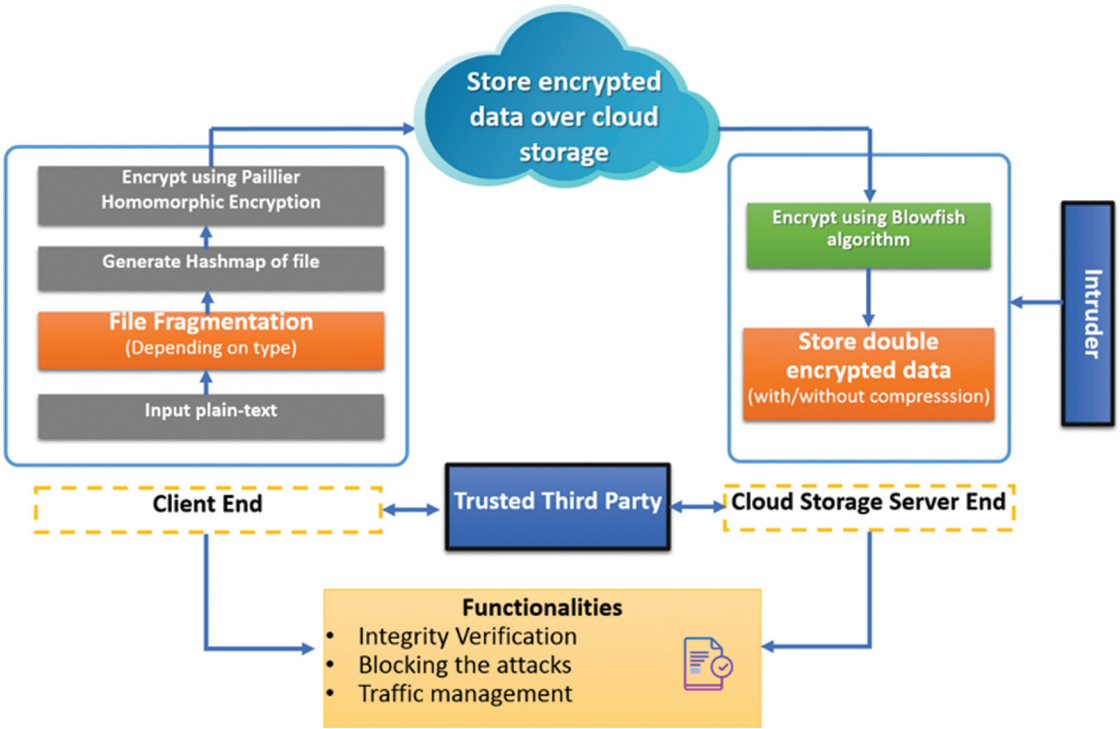

Figure 6: Decryption process of the hybrid system

5 Proposed Algorithm for Secure Data Storage Protocol

The reason for utilizing the symmetric encryption process as a method of recovering the data from the cloud is a consequence of key distribution issues and the utility of homomorphic

encryption is to reduce the overhead of decryption operations. In this section, the algorithms used for uploading and downloading phase are described. Algorithm 1 is employed for uploading of information to diverse cloud-based systems which meet storage capacity constraints. Here, a file is distributed into several sections depending upon its type and a corresponding Hashmap for each chunk is generated. Next, on each chunk, Paillier encryption is performed which encrypts and transform the data into cipher-text and generates its Hashmap again. For each pair of hash and generated cipher-text, these shares are then distributed to diverse cloud stores. At the end of uploading process, each segment is again encrypted using the Blowfish algorithm.

Algorithm 1: Proposed algorithm for uploading phase of cloud storage

Input: A file F taken as a plaintext;
Output: Set of file chunks $FC\left(\sum fc_i = 1, 2, \ldots, n\right)$ accumulated in CSP_i (Honest but Curious) with cloud storages S_i;

BEGIN
  For each (file) **do**
    Generate chunks of files depending on its type;
    For each (file chunk fc_i) **do**
      Generate Hashmap of file chunk;
      Do apply Paillier algorithm;
      Hash each file chunk fc_i;
    endfor
    For (all shares fc_i and signatures of each share) **do**
      Upload each chunk fc_i and hash into destined;
      CSP_i and cloud storages S_i;
      Encrypt each chunk using Blowfish algorithm;
    endfor
  endfor
END

Algorithm 2 depicts the functionality of decryption process involved. For each chunk stored with each CSP, decoding of chunks is accomplished by employing Paillier and Blowfish algorithms. An assessment for metadata and stored Hashmap is done. If matching occurs, all the chunks are joined to reconstruct the original file.

Algorithm 2: Proposed algorithm for downloading operation on cloud storage

Input: *FC* shares stored in CSP_i with cloud storage S_i;
Output: Original file *F* decrypted as plain text;
BEGIN
  For all (*Cloud storage* S_i *and all chunks* fc_i) **do**
    Decrypt the downloaded file using Paillier and Blowfish algorithm;
    Verify metadata comparison;
    if (*match occurs*) **then** Verify the hash of all shares;
    Hash checksums;
    if (*match occurs*) **then** Combine all fc's of *F*;
  endfor
END

Thus, we propose a method for the secure sharing of cloud info and provide a reliability assurance to customers using the aforementioned algorithms.

6 Implementation and Analysis

6.1 Implementation

We implemented a prototype of the model and analyzed the results with Ubuntu 16.04[1] on Oracle VM Virtual-Box 5.1.20[2]. Python[3] 2.7.3 has been used for the implementation of the encryption algorithms. For simulation of cloud security, Fog server has been utilized on account of its ease of access and its policies which are attuned to the infrastructure available at hand. A security audit tool Lynis which is an open source shell script host-based tool for operating systems like Unix, Linux, Solaris 10, MAC, etc. and supports different plugins, compliance checks, and customs checks is employed to calculate the Hardening index which is a unique metric indicative of how well security vulnerabilities have been kept to a minimum in the system. In addition, it provides warnings and suggestions and detailed system logs based on the security tests that have been performed. ArcherySec[4] has been used as a vulnerability assessment and management tool.

Encryption processes are classified into block ciphers and stream ciphers based on the nature of the input data. Block cipher is an encryption procedure that takes fixed size input (generally of the order of bytes) and produces the corresponding enciphered output. The block ciphering mechanism can be implemented in a variety of ways. Now here, we determine to conjoin the diverse styles of functioning of a block cipher and generate new properties, thereby enhancing the security of the core block cipher. The length of the message is generally higher than the block size. Consequently, the long message is sub-divided into a chain of sequential message blocks, and the cipher functions on these blocks one by one. Below we describe the various flavors of block ciphering mechanism that are employed in order to bolster the strength of the core block cipher:

[1] https://ubuntu.com/
[2] https://www.virtualbox.org/
[3] https://www.python.org/
[4] https://www.archerysec.com/

- ***Electronic Code Block (ECB)*** mode encrypts the leading block of plain-text to generate cipher-text and follows the identical process with the similar key and so on.
- ***Cipher-text feedback (CFB)*** encrypts and transmits plain-text immediately one and only at a time with a 'feedback' to encrypt the subsequent plaintext block.
- In ***Cipher block chaining mode (CFC)***, the existing plaintext block is summed to the preceding cipher-text block and the outcome is encoded with the key.
- ***Output Feedback mode (OFB)*** mode comprises nourishing the consecutive output blocks from the fundamental block cipher back to it.
- ***Counter mode (CTR)*** is a counter-based form of CFB sans the feedback. It doesn't propagate transmission errors at all.

6.2 Analysis

The realized protocol is believed to competently deliver security guarantee to clients conditioned on the cloud service providers being stringent in their actions against malicious or illegal users. The truthfulness, obtainability and privacy security constraints in the event of both interior and exterior attacks were validated. Also, detailed general information, vulnerable software packages, and possible configuration issues in both the proposed hybrid approaches were understood using the Lynis security auditing tool. It gives a view of what components in the system pose the greatest security risks and are thereby the high priority targets of hardening projects. We evaluated the performance of different algorithms based on a list of parameters. We have categorized them into two broad groups: Computation overhead parameters and quality of service parameters.

We discuss these parameters in detail below:

— Computation overhead is evaluated in terms of time and space complexity i.e., encryption and decryption time and storage space as measured by file size.
— Quality of service parameters evaluation is accomplished in terms of

1. Throughput comparison of different block cipher modes.
2. Stream length comparison of different block cipher modes.
3. Loads of parameters with and without attack.
4. Average time to attack and without attack.

The *Performance analysis* of the proposed hybrid security algorithm was done from the twin perspectives of numerical and security analysis.

6.2.1 Numerical Analysis

The constraints for assessing the system performance are mentioned in this section. The most significant parameter that is characteristic of the performance of the system is the computation time needed to accomplish the operations under consideration. Compression helps in reduction of space complexity and calculation time. The other parameters that are considered are also defined below.

Cipher-text file size of different file types (text/image/pdf/video/docx) is calculated and listed in Tab. 1. Paillier and Blowfish scheme with compression provides a significant improvement in performance as compared to RSA-AES hybrid scheme in case of text, image, video, docx, and pdf files.

The bar graph for encrypted file size using Paillier and Blowfish (with compression) and RSA + AES is shown in Fig. 7.

Table 1: Encrypted file sizes for Paillier–Blowfish and RSA-AES hybrid approaches

File type	File size (bytes)	Encrypted file size (bytes) RSA + AES	Encrypted file size with compression (bytes) Paillier + Blowfish
Image	23596	24805	23886
txt	7180	7582	7462
Pdf	483694	517919	484110
Video	4254956	4933261	4256534
docx	23679	27565	23958

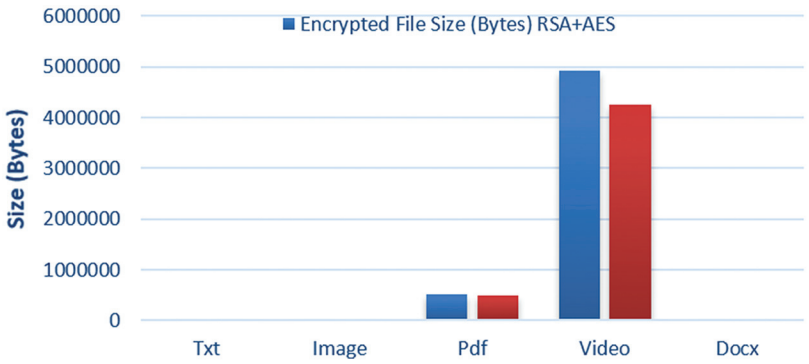

Figure 7: Encrypted file size using Paillier and Blowfish (with compression) and RSA + AES

Tab. 2 depicts the file encryption and Decryption time using Paillier and Blowfish with and without compression and RSA-AES techniques. Encryption time is the time taken for encoding information in such a way that only authorized users can access the information. Decryption time is the time taken to reverse the process of encryption i.e., transformation of the encoded data back into its input form. We have calculated time using formulae tailored to the chosen encryption algorithm.

Table 2: Encryption time and decryption time (s) using Paillier and Blowfish (with and without compression) and RSA + AES

Encryption and decryption time (s) Paillier & Blowfish without compression	Encryption and decryption time (s) Paillier & Blowfish with compression	Encryption and decryption time (s) RSA & AES
613.34	214.052	237.07

Calculation of encryption and decryption time using Paillier four different file types is done using the formula mentioned below:

time ./paillier -e -p password -o OutputFilename InputFilename
time ./paillier -d -p password -o OutputFilename InputFilename

where, InputFilename is the file to be encrypted; OutputFilename is the encrypted file; and NewFilename is the decrypted filename. Different switches used are 'e' for encryption, 'd' for decryption, 'p' for the password, and *'o' for output.

Calculation of encryption and decryption time using Blowfish with compression is done for different file types using the formulae mentioned below:

time ./bcrypt -r Filename

Calculation of encryption and decryption time using Blowfish without compression is done for different file types using the formulae mentioned below:

time ./bcrypt -rc Filename

Fig. 8 is the bar graph representation of Tab. 2 for the parameters of encryption and decryption time.

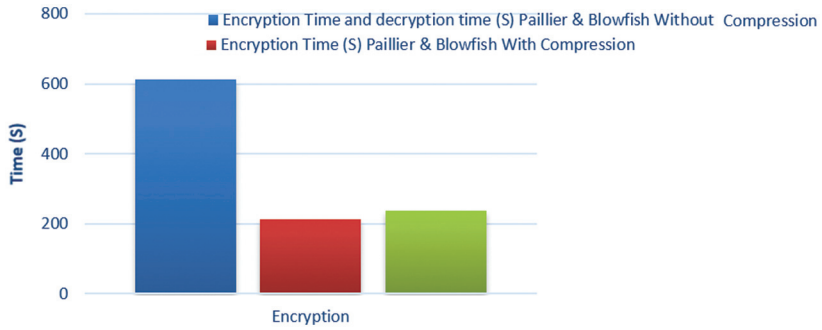

Figure 8: Encryption time and decryption time (s) using Paillier and Blowfish (with and without compression) and RSA + AES

From Figs. 7 and 8, noteworthy reduction in file size and encryption and decryption time for the four scenarios discussed above has been achieved.

We have evaluated *Average Throughput* (AT) in KBps as

$$AT = \frac{\frac{block\ processed}{elapsed\ decipher\ time} + \frac{block\ processed}{elapsed\ encipher\ time}}{2} \tag{1}$$

where, elapsed encipher and decipher time is calculated by total encipher and decipher times divided by number of threads respectively.

Fig. 9 depicts the *Throughput* comparison for different block ciphering modes in which we have kept the key size and initialization vector to be same for all the block cipher modes.

Fig. 10 shows the comparison of average of average throughput for various block ciphering modes.

The *Stream Length (SL)* defines the average length of a stream in each order and is computed by dividing the complete length of all the streams in a specific sequence by the number of streams in that sequence.

$$SL = \frac{\sum_{i=1}^{Total\ vector\ size} (i \times Sizeofeach\ chunk\ of\ stream)}{1\ KB \times 1\ KB} \tag{2}$$

Fig. 11 demonstrates the stream length evaluation of various block ciphering modes.

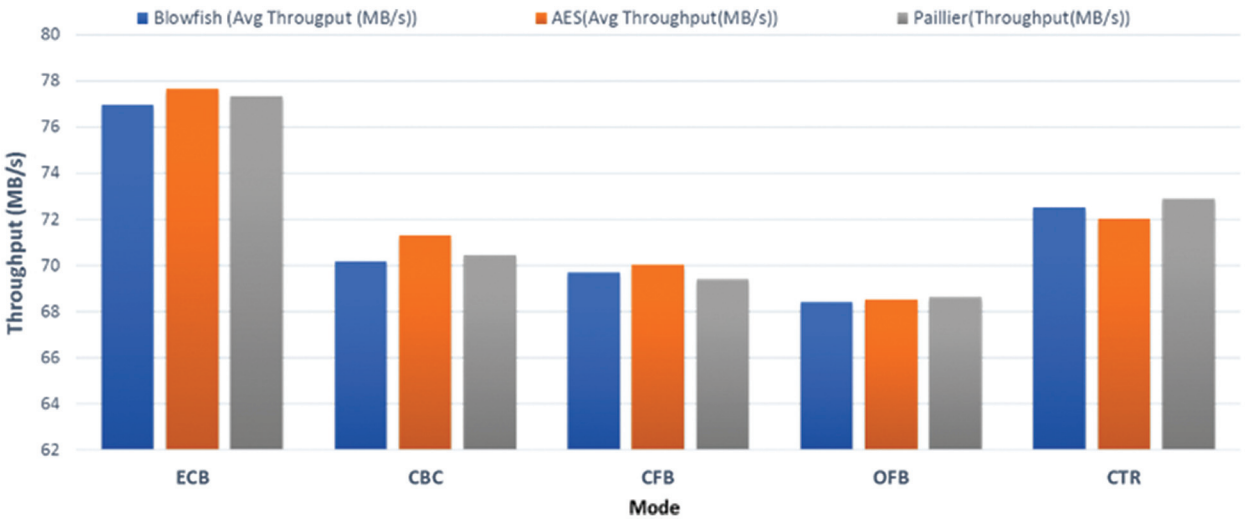

Figure 9: Throughput comparison of various block ciphering modes

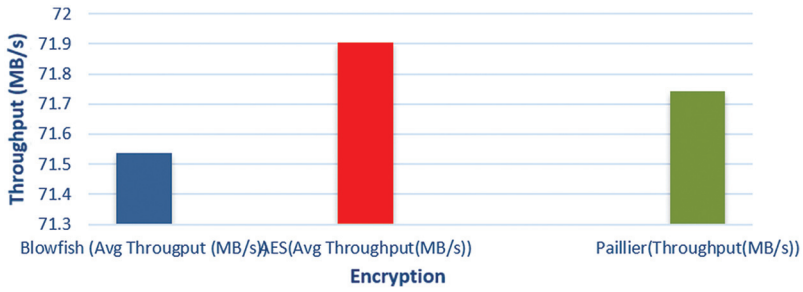

Figure 10: Comparison of average of average throughput for various block ciphering modes

Fig. 12 illustrates the comparison of the average of the average stream length of the various flavors of block ciphering for Blowfish, AES and Paillier schemes.

Loads of parameters signify the number of parameters having the same Hashvalue i.e., same Hashmap values.

Comparison of loads of parameters with and without attack using Paillier–Blowfish and RSA-AES approaches are illustrated in Tab. 3.

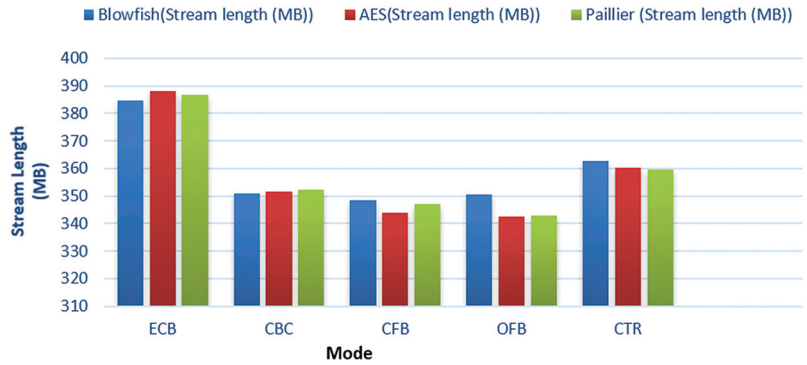

Figure 11: Stream length evaluation of varied manners of block cipher mode

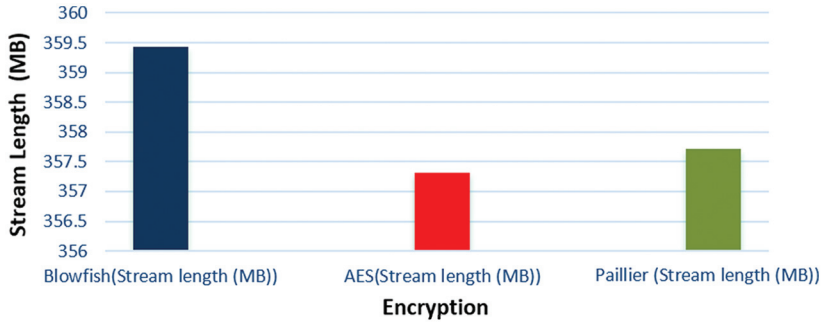

Figure 12: Comparison of average of average stream length of various flavors of block ciphering

Table 3: Comparison of loads of parameters

Load of parameters having the same hash value	Paillier–Blowfish approach with attack time (s)	RSA-AES approach without attack time (s)
100	15.899	13.585
200	10.979	9.942
300	11.121	11.669
10000	11.317	12.091
2000	98.286	11.896

The corresponding graph is shown in Fig. 13. Clearly, augmented time in Paillier–Blowfish hybrid approach is as desired.

Fig. 14 demonstrates the comparison of the *average time for attack and without attack* for both the hybrid architectures under discussion.

$$Attack\ Time = Time\ At\ Which\ Request\ Was\ Sent + Request\ Size \tag{3}$$

Clearly, the increase in time indicates the supplementary time needed to check the introduced attacks on port numbers.

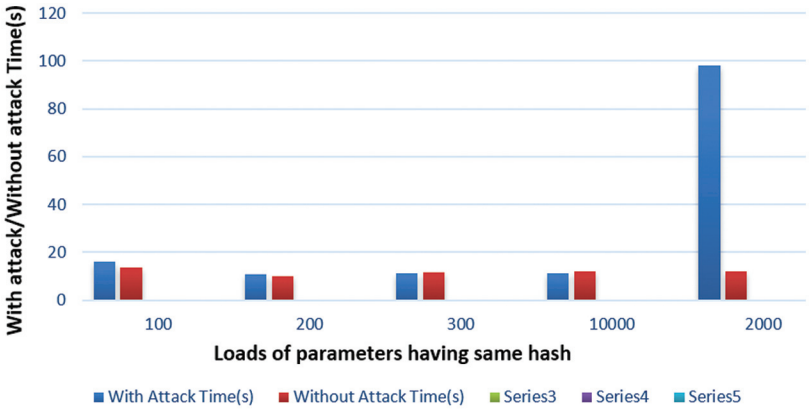

Figure 13: Comparison of loads of parameters with and without attack

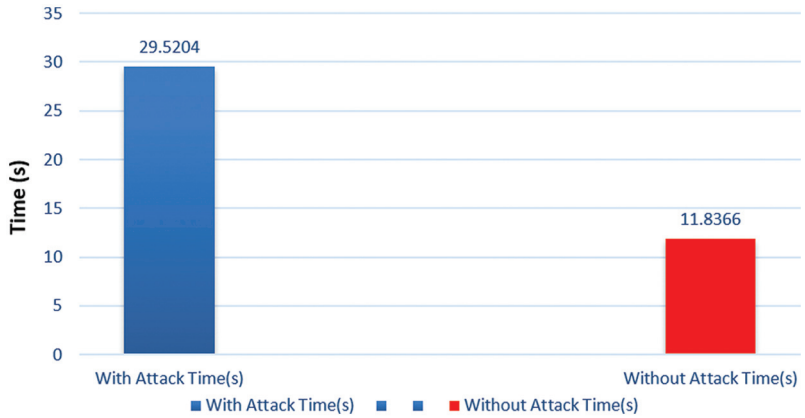

Figure 14: Comparison of the average time to attack and without attack

6.2.2 Security Analysis

The overall system functionality can come to a standstill if the system is under attack. Using iptables in a script, we have *blocked the attacks by creating firewalls* in the proposed system. The script is present in Paillier–Blowfish hybrid approach and is absent in RSA-AES hybrid approach.

An iptables chain rule specification consists of a number of parameters given as options to the iptables command. Each rule in a Firewall comprises of certain conditions and an action to be taken if conditions match entirely. We have utilized the command

iptables [-t table]{-D/-A/-C} chain rule specification

where, iptables is employed to set up, preserve, and examine the tables of IPv4 packet filter rules in the Linux kernel. Every chain is a set of rules applicable to a set of packets and acts as a guide on what to do with the packet; −A appends one or more rules to the culmination of a certain chain; −C checks whether the rule corresponding to the description does occur in the selected chain and doesn't alter the existing iptables configuration as opposed to −D which obliterates one or the other rules from the selected chain.

The following commands are used to block common attacks like SYN and side-channel attack, to drop null packets and to drop incoming packets with fragments.

— For SYN packets checking, the command is
sudo iptables -A INPUT -p tcp!--syn -m state NEW -j DROP
— For checking side-channel packets, the command is
sudo iptables -A INPUT -p tcp- -tcp - flags ALL ALL -j DROP
— For dropping null packets, the command is:
sudo iptables -A INPUT -p tcp- -tcp - flags ALL NONE -j DROP
— To reject receiving packets with chunks
sudo iptables -A INPUT -f -j DROP

For security analysis, we have employed Lynis 2.7.1 which is an open source safety assessment tool for systems executing Unix-derivatives. A report, including the outcomes (cautions and recommendations) and wide-ranging information such as detailed security logs is generated. The Lynis security audit tool can be instantiated using the following command.

sudo ./lynis audit system

The proposed hybrid Paillier–Blowfish with compression approach is compared to RSA-AES approach on the Hardening Index parameter. Hardening index evaluates the number of tests performed on the system. It is used for managing a firewall and aims to provide an easy interface to the user.

Fig. 15 shows the hardening index for both the chosen hybrid approaches.

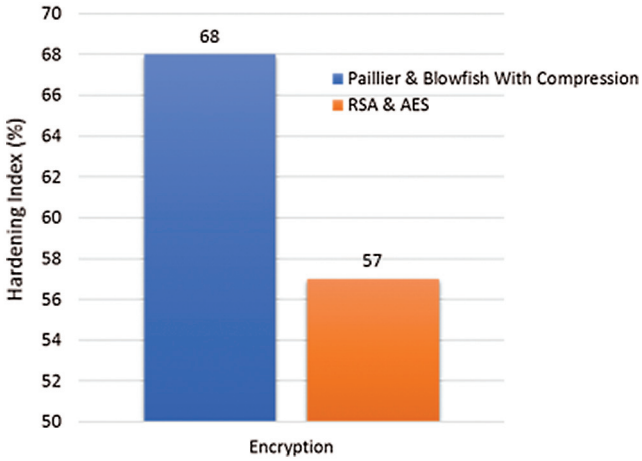

Figure 15: Hardening index using Paillier and Blowfish (with compression) and RSA + AES

These major inferences summarize the main contribution of the research carried out which aimed to analyze cloud security issues and propose a hybrid algorithm to augment the information security triad in a cloud computing environment involving multi-cloud scenarios. It is apparent that the suggested scheme achieves consistent outcomes based on the file type, reduction in computation time, and a decrease in file size. This hybrid technique will amplify the safety of the key and enhance data protection.

7 Conclusion

Cloud computing has become the infrastructural base for future computing paradigms. Yet, the security vulnerabilities in a cloud-based system persist as a vital bottleneck. So, a fusion

of homomorphic and symmetric algorithms has been proposed to deal with cloud data security issues. Multi-cloud systems eliminate the drawbacks of a single cloud system. Security traits like confidentiality, integrity, availability, authorization, and non-repudiation for multi-cloud storage are supported. The main motivation and key contributions of the paper are listed. Architectural details and protocols for the proposed hybrid system is discussed. To create a virtual environment, the simulations have been executed on an Oracle Virtual Box on the Ubuntu platform. A comparative study of the prevailing hybrid scheme against the proposed hybrid system is done in terms of various parameters. For security analysis, attack analysis is used on both the discussed hybrid architectures. The encryption and decryption time of Paillier and Blowfish without compression is approximately 2.5 times higher than RSA-AES.

The encryption and decryption time of Paillier and Blowfish with compression is approximately 10% better than RSA-AES. The cipher-text size of Paillier–Blowfish with compression is significantly better than the RSA-AES combination for all types of files. The Hardening index for the Paillier–Blowfish amalgam system is much higher specifically, 28% better than the RSA-AES combination. In work that will be carried out in the near future, hybrid algorithms will be created from a plethora of algorithms and characterized using results of our work. We also anticipate research efforts that would shed further light on secure encryption schemes for file protection and data preservation with the help of simple scripts and programs.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] D. S. Abd Elminaam, "Improving the security of cloud computing by building new hybrid cryptography algorithms," *International Journal of Electronics and Information Engineering*, vol. 8, no. 1, pp. 40–48, 2018.

[2] Q. Wei, H. Shao and G. Zhang, "Flexible, secure, and reliable data sharing service based on collaboration in multicloud environment," *Wireless Communications and Mobile Computing*, vol. 2018, pp. 1–16, 2018.

[3] D. N. Le, R. Kumar, G. N. Nguyen and J. M. Chatterjee, *Cloud Computing and Virtualization*. USA: John Wiley & Sons, 2018.

[4] M. Tebaa, S. El Hajji and A. El Ghazi, "Homomorphic encryption applied to the cloud computing security," in *Proc. of the Int. Association of Engineers World Congress on Engineering 2012*, London, UK, pp. 4–6, 2012.

[5] A. N. Khan, M. Y. Fan, M. I. Nazeer, R. A. Memon, A. Malik *et al.,* "An efficient separable reversible data hiding using Paillier cryptosystem for preserving privacy in cloud domain," *Electronics*, vol. 8, no. 6, pp. 682, 2019.

[6] A. Taha, D. S. Abd Elminaam and K. M. Hosny, "NHCA: Developing new hybrid cryptography algorithm for cloud computing environment," *International Journal of Advanced Computer Science and Applications*, vol. 8, no. 11, pp. 479–486, 2017.

[7] J. Wu, I. Detchenkov and Y. Cao, "A study on the power consumption of using cryptography algorithms in mobile devices," in *2016 7th IEEE Int. Conf. on Software Engineering and Service Science*, Beijing, China, pp. 957–959, 2016.

[8] B. Rani, "A novice's perception of partial homomorphic encryption schemes," *Indian Journal of Science and Technology*, vol. 9, no. 37, pp. 10–18, 2016.

[9] M. Waleed and L. Chunlin, "User privacy and security in cloud computing," *International Journal of Security and Its Applications*, vol. 10, no. 2, pp. 341–352, 2016.

[10] G. Keerthana, S. Prabu and P. Swarnalatha, "An efficient data security in cloud computing using cryptography," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 6, no. 5, pp. 654–660, 2016.
[11] R. Siregar, "Performance analysis of AES-Blowfish hybrid algorithm for security of patient medical record data," *Journal of Physics Conference Series*, vol. 1007, no. 1, 012018, 2018.
[12] S. Chaudhary and N. K. Joshi, "Secured blended approach for cryptographic algorithm in cloud computing," *International Journal of Pure and Applied Mathematics*, vol. 118, no. 20, pp. 297–304, 2018.
[13] T. Jyoti and G. Pandi, "Achieving cloud security using hybrid cryptography algorithm," *International Journal of Advance Research and Innovative Ideas in Education*, vol. 3, no. 5, pp. 1518–1523, 2017.
[14] B. P. Gajendra and V. K. Singh, "Achieving cloud security using third party auditor, MD5 and identity-based encryption," in *2016 Int. Conf. on Computing, Communication and Automation*, Greater Noida, India, pp. 1304–1309, 2016.
[15] S. Sharma, "Enhance data security in cloud computing using machine learning and hybrid cryptographic techniques," *International Journal of Advanced Research in Computer Science*, vol. 8, no. 9, pp. 393–397, 2017.
[16] Y. Liu, S. Xiao, H. Wang and X. A. Wang, "New provable data transfer from provable data possession and deletion for secure cloud storage," *International Journal of Distributed Sensor Networks*, vol. 15, no. 4, 1550147719842493, 2019.
[17] X. Wang, A. Zhang, X. Xie and X. Ye, "Secure-aware and privacy-preserving electronic health record searching in cloud environment," *International Journal of Communication Systems*, vol. 32, no. 8, pp. e3925, 2019.
[18] J. Vyas and P. Modi, "Providing confidentiality and integrity on data stored in cloud storage by hash and meta-data approach," *International Journal of Advanced Research in Engineering Science Technology*, vol. 4, pp. 38–50, 2017.
[19] D. N. Le, B. Seth and S. Dalal, "A hybrid approach of secret sharing with fragmentation and encryption in cloud environment for securing outsourced medical database: A revolutionary approach," *Journal of Cyber Security and Mobility*, vol. 7, no. 4, pp. 379–408, 2018.
[20] B. Seth and S. Dalal, "Analysis of cryptographic approaches," *International Journal of Recent Research Aspects*, Special Issue: Conscientious and Unimpeachable Technologies 2016, pp. 21–24, 2016.
[21] D. N. Le, V. N. Van and T. T. T. Giang, "A new private security policy approach for DDoS attack defense in NGNs," in *Information Systems Design and Intelligent Applications*, New Delhi, India: Springer, pp. 1–10, 2016.
[22] C. Hazay, G. L. Mikkelsen, T. Rabin, T. Toft and A. A. Nicolosi, "Efficient RSA key generation and threshold Paillier in the two-party setting," *Journal of Cryptology*, vol. 32, no. 2, pp. 265–323, 2019.
[23] M. Suresh and M. Neema, "Hardware implementation of Blowfish algorithm for the secure data transmission in internet of things," *Procedia Technology*, vol. 25, pp. 248–255, 2016.
[24] V. Masthanamma and G. L. Preya, "An efficient data security in cloud computing using the RSA encryption process algorithm," *International Journal of Innovative Research in Science, Engineering and Technology*, vol. 4, no. 3, pp. 1441–1445, 2015.
[25] K. Benzekki, A. El Fergougui and E. A. Elbelrhiti, "A secure cloud computing architecture using homomorphic encryption," *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 2, pp. 293–298, 2016.
[26] Z. Tari, "Security and privacy in cloud computing," in *IEEE Cloud Computing*. vol. 1, RMIT University, pp. 54–57, 2014.
[27] H. Yan, J. Li, J. Han and Y. Zhang, "A novel efficient remote data possession checking protocol in cloud storage," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 1, pp. 78–88, 2016.
[28] M. Ali, S. U. Malik and S. U. Khan, "DaSCE: Data security for cloud environment with semi-trusted third party," *IEEE Transactions on Cloud Computing*, vol. 5, no. 4, pp. 642–655, 2015.
[29] G. S. Aujla, R. Chaudhary, N. Kumar, A. K. Das and J. J. Rodrigues, "SecSVA: Secure storage, verification, and auditing of big data in the cloud environment," *IEEE Communications Magazine*, vol. 56, no. 1, pp. 78–85, 2018.

[30] B. Seth and S. Dalal, "Hybrid homomorphic encryption for secure cloud data storage," *Recent Advances in Computational Intelligence*, vol. 823, pp. 71–92, 2019.

[31] S. Joshi, "An efficient Paillier cryptographic technique for secure data storage on the cloud," in *2020 IEEE 4th Int. Conf. on Intelligent Computing and Control Systems*, Madurai, India, pp. 145–149, 2020.

[32] Y. Kiran Kumar and R. Mahammad Shafi, "An efficient and secure data storage in cloud computing using modified RSA public key cryptosystem," *International Journal of Electrical & Computer Engineering*, vol. 10, pp. 2088–8708, 2020.

[33] V. Sathya, A. Shiny, S. Sajid Hussain and A. Gauda, "Secure data storage in cloud system using modern cryptography," *Journal of Computational and Theoretical Nanoscience*, vol. 17, no. 4, pp. 1590–1594, 2020.

[34] K. Rajkumar and V. Dhanakoti, "Methodological survey to improve the secure data storage in cloud computing," in *IEEE, 2020 Int. Conf. on Emerging Smart Computing and Informatics*, Pune, India, pp. 313–317, 2020.

[35] B. Seth, S. Dalal, V. Jaglan, D. N. Le, S. Mohan *et al.,* "Integrating encryption techniques for secure data storage in the cloud," *Transactions on Emerging Telecommunications Technologies*, pp. e4108, 2020 (Pre-online).