SvedbergOpen
DISSEMINATION OF KNOWLEDGE
International Journal of Artificial Intelligence and Machine Learning
Publisher's Home Page: https://www.svedbergopen.com/
Research Paper Open Access

A Novel Hybrid Cryptographic Model For Cloud Data Protection

Dr. Dhaval J Varia[1], Dr. Mital Nikunj Panchal[2], Patel Hirenkumar Ramanbhai[3], Pratikkumar A Barot[4], Dr. Prashant Modi[5]

[1]Associate Professor, IT Department Shantilal shah engineering College Bhavnagar dhavalkvaria@gmail.com
[2]Associate Professor, IT Department Shantilal shah engineering College Bhavnagar panchal.mital@gmail.com
[3]Assistant Professor, IT Department Government Engineering College Modasa hiren.patel@gecmodasa.ac.in
[4]Associate Professor, computer engineering Governmentengineering College, Patan pratikabarot@gmail.com
[5]Assistant Professor, Department of Information Technology, Government Engineering College, Modasa, India prashant7modi7it@gmail.com

Abstract— Cloud computing has make an emerging technology for delivering on-demand, virtualized computing resources over the Internet. Because cloud services fundamentally rely on shared, outsourced infrastructure, they introduce serious concerns regarding data confidentiality, integrity, privacy and authentication. A huge range of cryptographic mechanisms have been proposed to address these concerns, yet each individual algorithm, whether symmetric or asymmetric that carries inherent limitations in terms of security strength, computational cost to modern attacks such as cryptanalytic. This research go through existing experiments so find out the use of hybrid cryptography as a means of strengthening data security on the cloud. There are basically three different setups are constructed and tested to see which one worked best. First, a concurrent approach that chops a file into chunks and encrypts them simultaneously is using AES, DES, and RC4. Second, a layered method that runs the same three algorithms one after another on the whole files. And third, a hybrid system that wraps AES symmetric encryption inside RSA asymmetric protection. All three were coded in Node.js with the CryptoJS library and put through their paces on files from 1 MB up to 30 MB. The results after experiments show that the parallel hybrid algorithm achieves encryption and decryption times comparable to, and for larger files marginally better than, standalone AES, while simultaneously distributing trust across three distinct cryptographic primitives. The layered and RSA-assisted hybrid cryptosystem approaches, in contrast, it also provide additional layers of security at a measurable cost in turnaround time. The comparison we have here helps us figure out which hybrid encryption strategy is best for storing cloud data when we have to balance security and performance. We need to find the hybrid encryption strategy for cloud data storage. This is due to the inherent conflict between maximizing security and maintaining speed.

Keywords— Cloud computing; data security; hybrid cryptography; AES; DES; RC4; RSA; layered encryption; hybrid cryptosystem; encryption performance.

I. INTRODUCTION

Cloud computing are the way people getting computing services over the Internet Nowadays. Cloud computing is about the programs you can use on the web and the computers and software that make those programs work which are usually in big data hubs. The National Institute of Standards and Technology says that cloud computing is a way to get to a group of computing things like computers and storage over the Internet when you need them. You can get these things quickly. When you are done, give them back, without having to do a lot of work to manage them. Cloud computing is mainly differentiated into three parts: (1) Software as a Service, (2) Platform as a Service and (3) Infrastructure as a Service. These cloud services have some things in common: You can get those resources when you need them from a pool of resources. However, the resources are available anywhere on the Internet. Cloud computing services like Software as a Service and Platform as a Service and Infrastructure as a Service are, about making it easy to get what you need.

Despite its advantages, cloud computing introduces significant security concerns that stem directly from its defining characteristics. Outsourcing involves the fact that the owner of the data loses the physical control over the data as soon as it is outsourced to a third-party service provider. The principle of multi-tenancy involves the sharing of the underlying physical infrastructure between several non- trusted clients, making the system vulnerable to threats and attacks like data

leakage and computation breaches. Apart from that, here also describe that due to the huge volume of data and computations involved in cloud computing, the majority of the existing computational security mechanisms become inefficient[2], [3].
Cryptography means the conversion of data into unreadable forms that will recoverable only by an authorized person or to those who are authorized to access, is the principal tool used to protect data confidentiality on the cloud [7]. Cryptographic algorithms are broadly classified as symmetric, where the same secret key is used for both encryption and decryption (e.g., DES, AES, RC4, Blowfish), and asymmetric, where one public key is used for encryption and another private key for decryption (e.g., RSA, Diffie–Hellman, El-Gamal, DSA). Symmetric algorithms are computationally efficient but require secure key distribution, whereas asymmetric algorithms simplify key management at the cost of higher computational overhead.
Hybrid cryptography combines two or more than two encryption techniques to obtain the combined benefits of the constituent algorithms, typically the computational efficiency of symmetric encryption together with the key-management convenience of asymmetric encryption, or the added resilience of multiple independent symmetric primitives. Since every individual cipher is, in principle, vulnerable to cryptanalysis given sufficient computational power, combining algorithms increases the effort required for a successful attack without a proportional increase in processing cost. This motivates the central problem addressed in this paper: given that outsourced cloud storage is inherently exposed to security and privacy risks, how can multiple well-established encryption primitives — specifically AES, DES, RC4 and RSA — be combined into a hybrid scheme that improves resistance to attack while remaining practical in terms of encryption and decryption time for files of realistic size [10].

II. RELATED WORK

A number of previous research works inform the design adopted in this study. Xiao and Xiao[1] present a thorough treatment of security and privacy issues in cloud computing, they list some difficulties such as data segregation, data leakage prevention, identity and access management, and physical security and reporting that data segregation/protection and data-leak prevention are rated as critical by the large majority of practitioners surveyed. According to them the most important aspects existing defense strategies — including virtualization-based isolation, data-centre-level controls and MapReduce-based processing controls — each address only part of the problem, leaving open the need for more holistic, cryptography-based protection of the data itself rather than only the infrastructure around it [1].
Another research propose SeDaSC (Secure Data Sharing in Clouds), introduces a methodology that encrypts a user's file conventionally and then generated a result as a symmetric key into two shares — one retained by the user and one stored, itself encrypted under the user's public key, on a dedicated cryptographic server. Their evaluation, conducted on files from 0.1 MB to 500 MB, shows that key-computation time grows only marginally with file size while encryption time grows more steeply, and highlights that reliance on an external key-holding server introduces its own trust and threat-model considerations. This observation suggests that splitting responsibility across independent components can improve manageability but must be weighed against the new attack surface it creates that can directly motivates the file-splitting and cryptosystem designs explored in this paper[2].
The existing research also present a comparative study of RC4, DES, 3DES and AES with respect to CPU usage, encryption/decryption time, throughput and behavior across varying file and packet sizes[7][9]. Their findings, along with related comparative studies of AES versus RC4 and of DES, AES and Blowfish, consistently report that AES offers the strongest security margin among the symmetric candidates while remaining competitive in speed, that RC4 is fast but sensitive to key-scheduling weaknesses if not implemented carefully, that single DES is no longer considered secure against brute-force attack given modern computational power, and that Triple DES, while more secure than DES, is significantly slower and remains vulnerable to meet-in-the-middle style attacks unless a sufficiently long effective key is used. These characterizations of the individual algorithms directly inform the choice of AES, DES and RC4 as complementary building blocks for the hybrid schemes developed in Section III, since each algorithm compensates for a different weakness of the others when combined[7][8].
Taken together, the literature indicates two complementary directions for improving cloud data security: (a) redesigning the trust model around key storage and sharing, as in SeDaSC, and (b) combining multiple encryption primitives to mitigate the specific weaknesses of any single algorithm, as suggested by the comparative cipher studies. The work presented here follows the second direction, while additionally incorporating a hybrid-cryptosystem variant that also addresses secure key exchange in the spirit of the first.

III. PROPOSED METHODOLOGY

This paper designs and empirically compares three hybrid encryption strategies for securing files prior to upload on a cloud server: a parallel hybrid algorithm, a layered (cascade) algorithm, and an RSA-assisted hybrid cryptosystem. All three strategies are built from the same four underlying primitives — AES, DES, RC4 and RSA — so that the comparison isolates the effect of combination strategy rather than the effect of algorithm choice.

A. System Requirements

The prototype system was implemented using JavaScript on the Node.js runtime, the CryptoJS cryptographic library for AES, DES, RC4, PBKDF2 and PKCS7 primitives, and the Node.js File System (fs) module for file input/output. The hardware baseline used for evaluation consisted of a machine with 1 GB or more of RAM and a CPU clocked at 1.7 GHz or above, running Windows 8 or later.

B. Parallel Hybrid Algorithm

In the parallel hybrid approach, an input file is the first partitioned into three approximately equal parts using a file-splitting module. Each part is then encrypted concurrently using a different algorithm — the first part with AES, the second with DES, and the third with RC4 — so that no single algorithm processes the complete file and no single cryptanalytic attack on one primitive can compromise the entire dataset. The three encrypted parts are concatenated and uploaded to the cloud server as a single artefact. Decryption reverses the process: the encrypted file is downloaded, split back into its three constituent parts using a delimiter retained from the encryption step, each part is decrypted in parallel using the algorithm originally applied to it, and the decrypted parts are merged to reconstruct the original file.

C. Layered (Cascade) Encryption

The layered approach applies the same three algorithms to the entire file but sequentially rather than in parallel: the file is first encrypted with DES, the resulting cipher text is re-encrypted with RC4, and that result is encrypted a third time with AES to produce the final cipher text uploaded to the cloud. Decryption is performed in the reverse order — first AES, then RC4, then DES — to recover the original message. Because every byte of the file passes through all three ciphers in sequence, this approach is expected to provide the strongest layered security guarantee among the three schemes, at the cost of substantially higher processing time, since the full dataset is processed three times rather than once.

D. Hybrid Cryptosystem (RSA + AES)

The hybrid cryptosystem combines the key-management convenience of asymmetric cryptography with the computational efficiency of symmetric cryptography. A random symmetric secret key is generated and used to encrypt the file data using AES. The secret key itself is then encrypted using the recipient's RSA public key, so that only the holder of the corresponding RSA private key can recover it. Both the AES-encrypted file and the RSA-encrypted key are stored on the cloud. During decryption, the encrypted key is first recovered using the RSA private key, and the recovered symmetric key is then used to decrypt the AES-encrypted file. This design avoids the need to pre-share a symmetric key between sender and receiver while retaining the speed advantage of symmetric encryption for the bulk of the data.

IV. ALGORITHMIC FOUNDATIONS

This section summarises the four cryptographic primitives combined in the proposed hybrid schemes.

A. Advanced Encryption Standard (AES)

AES, also known as Rijndael, is a symmetric block cipher standardised by NIST in 2001[7]. It operates on 128-bit blocks with key sizes of 128, 192 or 256 bits, using 10, 12 or 14 rounds respectively, arranged as a substitution–permutation network. Each round applies four transformations — SubBytes (a non-linear byte substitution), ShiftRows (a cyclic shifting of rows), MixColumns (a linear mixing of column data) and AddRoundKey (an XOR with a round-specific sub-key derived through key expansion). AES is not based on the Feistel structure and is widely regarded as providing an excellent balance of security and throughput among symmetric ciphers.

B. Data Encryption Standard (DES)

DES is a Feistel-structure block cipher operating on 64-bit blocks with a nominal 64-bit key, of which 8 bits serve as parity/check bits, leaving an effective 56-bit key. Plaintext is first passed through an initial permutation, processed through 16 rounds each using a 48-bit round key derived from the round-key generator, and passed through a final permutation to yield the cipher text. While no attack meaningfully more efficient than brute-force key search is known against DES, its 56-bit effective key length is now considered insufficient against modern computational capability, which motivated its extension to Triple DES using a 168-bit (three-key) or 112-bit (two-key) configuration [9].

C. Rivest Cipher 4 (RC4)

RC4 is a symmetric stream cipher that generates a pseudorandom key stream through a two-stage process — a Key Scheduling Algorithm (KSA) that initialises an internal 256-byte state array using the supplied key, and a Pseudo-Random Generation Algorithm (PRGA) that repeatedly permutes the state to emit key-stream bytes. The key stream is combined with the plaintext using a bit-wise XOR to produce the ciphertext. RC4 supports key sizes from 40 to 2048 bits and is computationally lightweight, but is known to be vulnerable to bit-flipping and related-key attacks if the key-scheduling stage is not implemented and seeded correctly [8].

D. Rivest–Shamir–Adleman (RSA)

RSA is a public-key cryptosystem whose security rests on the practical difficulty of factoring the product of two large distinct prime numbers [7]. Key generation proceeds by selecting two large primes, computing their product and Euler's totient function of that product, and deriving a public exponent and a corresponding private exponent satisfying the modular inverse relationship required for correct decryption. Typical key sizes range from 1024 to 4096 bits. Because RSA

operations are considerably more computationally expensive than symmetric ciphers of comparable security, RSA is used in the proposed system only to protect the short symmetric key rather than the bulk file data.

V. IMPLEMENTATION

The three schemes were implemented as Node.js modules built around the CryptoJS library. File splitting was performed with a dedicated split-file module that divides an input file into three approximately equal parts and provides the corresponding merge functionality. For the parallel hybrid algorithm, the three file parts were encrypted concurrently using the run-parallel control-flow module, with AES, RC4 and DES bound respectively to the first, second and third parts; decryption mirrored this structure, decrypting each part with its corresponding algorithm before concatenating the results. For the layered algorithm, encryption and decryption were implemented as strictly sequential operations using the run-series control-flow module, applying DES → AES → RC4 during encryption and the reverse order during decryption. For the hybrid cryptosystem, an RSA module was used to encrypt and decrypt the session password (secret key) using the recipient's public/private key pair, while AES was used to encrypt and decrypt the file data itself under that recovered key. A front-end web interface was additionally built to allow a user to log in or register, choose one of the three encryption approaches, submit data for encryption, store the resulting ciphertext, and later select stored ciphertext for decryption back to plaintext.

VI. RESULTS AND PERFORMANCE ANALYSIS

The three hybrid schemes were evaluated against standalone AES encryption on files ranging from 1 MB to 30 MB, and against each other, to assess the trade-off between security strength and processing overhead.

A. Cipher Characteristics

TABLE I. COMPARISON OF THE UNDERLYING CIPHERS

Factor	DES	AES	RC4
Originator	IBM, 1975	Daemen & Rijmen, 2001	Rivest, 1987
Key length	56 bits	128/192/256 bits	40–2048 bits
Rounds	16	10/12/14	1
Block size	64 bits	128 bits	Stream (no fixed block)
Speed	Slow	Fast	Fast
Security	Weak by modern standards	Excellent	Adequate

B. AES vs. Parallel Hybrid Algorithm

Table II reports the measured encryption and decryption times, in seconds, for standalone AES and for the proposed parallel hybrid algorithm (AES + DES + RC4) across file sizes from 1 MB to 30 MB.

TABLE II. AES VS. PARALLEL HYBRID ALGORITHM (SECONDS)

File Size	AES Enc.	Hybrid Enc.	AES Dec.	Hybrid Dec.
1 MB	0.617	0.732	0.589	0.685
5 MB	2.344	2.476	2.099	2.134
10 MB	3.257	3.221	3.643	3.532
20 MB	6.829	6.575	6.625	6.363
30 MB	13.110	12.818	11.345	10.828

For small files (1–5 MB), the parallel hybrid algorithm is marginally slower than standalone AES, since the overhead of splitting the file and coordinating three concurrent encryption tasks is not yet offset by the reduced per-algorithm workload.

From 10 MB onward, however, the hybrid algorithm consistently outperforms standalone AES for both encryption and decryption, since each of the three algorithms processes only one-third of the file concurrently. Overall, the parallel hybrid algorithm requires approximately 10–15% less time than single-algorithm AES encryption for larger files, while distributing the data across three independent cryptographic primitives rather than relying on a single algorithm for the entire file.

C. Comparison of Hybrid, Layered and Cryptosystem Approaches

Table III and Table IV compare the encryption and decryption times, respectively, of the parallel hybrid algorithm, the RSA-assisted hybrid cryptosystem, and the layered (cascade) algorithm.

TABLE III. ENCRYPTION TIME COMPARISON (SECONDS)

File Size	Hybrid	Cryptosystem	Layered
1 MB	0.732	0.889	2.032
5 MB	2.476	2.499	7.676
10 MB	3.221	3.543	15.121
20 MB	6.575	7.125	40.575
30 MB	12.818	14.245	57.118

TABLE IV. DECRYPTION TIME COMPARISON (SECONDS)

File Size	Hybrid	Cryptosystem	Layered
1 MB	0.685	0.889	1.832
5 MB	2.134	2.299	7.576
10 MB	3.532	3.943	15.121
20 MB	6.363	6.825	32.375
30 MB	10.828	12.145	52.318

The results show a clear ordering of processing cost: the parallel hybrid algorithm is consistently the fastest of the three schemes because its three constituent ciphers operate concurrently on disjoint segments of the file. The hybrid cryptosystem is only marginally slower than the parallel hybrid algorithm, since the additional RSA operation is applied only to the short symmetric key rather than to the file data, and the bulk of the file is still encrypted once with AES. The layered algorithm is markedly slower than the other two — by roughly a factor of four to five at 30 MB — because the entire file is processed sequentially by all three algorithms in full, tripling the total amount of data that must be transformed. This confirms that combining multiple ciphers by parallel partitioning or by protecting only the key material achieves a substantially better security–performance trade-off than combining ciphers by full sequential cascading.

D. Discussion

Across all three designs, the use of multiple independent cryptographic primitives means that an attacker must successfully cryptanalysis more than one algorithm — or, in the parallel case, must obtain and correctly reassemble all three differently-encrypted file segments — to fully recover the plaintext, which is not possible when a single algorithm secures the entire file. The parallel hybrid algorithm offers the most favorable balance for practical cloud deployment, since it matches or improves upon the throughput of standalone AES while still combining three distinct ciphers. The hybrid cryptosystem is attractive when secure key exchange between mutually distrusting parties is itself a requirement, since it removes the need to pre-share a symmetric key. The layered algorithm, while considerably slower, provides the deepest defense-in-depth, since every bit of the file is transformed by all three algorithms, and may be appropriate for smaller, higher-sensitivity files where processing time is a secondary concern.

VII. CONCLUSION AND FUTURE SCOPE

This paper examined three strategies for combining AES, DES, RC4 and RSA into hybrid encryption schemes for securing data stored on cloud infrastructure, and evaluated them empirically on files from 1 MB to 30 MB. The parallel hybrid algorithm, which partitions a file and encrypts each partition concurrently with a different cipher, was found to be the most time-efficient of the three schemes and, for files of 10 MB or larger, outperformed standalone AES encryption while additionally distributing cryptographic risk across three independent primitives. The hybrid cryptosystem, which protects a symmetric key with RSA and the file data with AES, achieved comparable performance while additionally solving the problem of secure key exchange. The layered cascade algorithm provided the strongest sequential security guarantee at a substantially higher computational cost. These findings suggest that, for general-purpose cloud storage where both security

and throughput matter, a parallel hybrid or RSA-assisted hybrid cryptosystem approach is preferable, whereas layered cascading remains suited to smaller or particularly sensitive files.

Future work will extend the set of underlying primitives to include Blowfish and additional public-key schemes, evaluate the hybrid algorithms against a broader range of cryptanalytic attacks and real-time cloud deployment conditions, and further optimise the parallel and cryptosystem-based approaches to provide enhanced, low-latency security for client data stored and shared on commercial cloud platforms.

REFERENCES

[1] M. Ali, R. Dhamotharan, E. Khan, S. U. Khan, A. V. Vasilakos, K. Li, and A. Y. Zomaya, "SeDaSC: Secure data sharing in clouds," IEEE Syst. J., vol. 11, no. 2, pp. 395–404, Jun. 2017.

[2] Z. Xiao and Y. Xiao, "Security and privacy in cloud computing," IEEE Commun. Surveys Tuts., vol. 15, no. 2, pp. 843–859, 2nd Quart., 2013.

[3] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: State-of-the-art and research challenges," J. Internet Services Appl., vol. 1, no. 1, pp. 7–18, 2010.

[4] M. Armbrust et al., "A view of cloud computing," Commun. ACM, vol. 53, no. 4, pp. 50–58, Apr. 2010.

[5] M. Batra, R. Kumar, and N. Ohri, "Secure file storage in cloud computing using hybrid encryption algorithm," Int. J. Comput. Eng. Appl., vol. 19, no. 6, Jun. 2018.

[6] M. Tahaseen, S. Nawaz, and A. Kumar, "Data storage on cloud using hybrid encryption with one time password," IOSR J. Comput. Eng., vol. 19, no. 4, pp. 1–5, Aug. 2017.

[7] G. Singh and Supriya, "A study of encryption algorithms (RSA, DES, 3DES and AES) for information security," Int. J. Comput. Appl., vol. 67, no. 19, Apr. 2013.

[8] N. Singhal and J. P. S. Raina, "Comparative analysis of AES and RC4 algorithms for better utilization," Int. J. Comput. Trends Technol., Aug. 2011.

[9] J. Thakur and N. Kumar, "DES, AES and Blowfish: Symmetric key cryptography algorithms simulation based performance analysis," Int. J. Emerg. Technol. Adv. Eng., vol. 1, no. 2, Dec. 2011.

[10] T. Jyoti and R. Sharma, "Achieving cloud security using hybrid cryptography algorithm," Int. J. Adv. Res. Innov. Ideas Educ., vol. 3, no. 5, Jun. 2017.

[11] S. Seo, M. Nabeel, X. Ding, and E. Bertino, "An efficient certificateless encryption for secure data sharing in public clouds," IEEE Trans. Knowl. Data Eng., vol. 26, no. 9, pp. 2107–2119, Sep. 2013.

[12] D. Chen et al., "Fast and scalable multi-way analysis of massive neural data," IEEE Trans. Comput., vol. 64, no. 3, pp. 707–719, Mar. 2015.

[13] L. Xu, X. Wu, and X. Zhang, "CL-PRE: A certificateless proxy re-encryption scheme for secure data sharing with public cloud," in Proc. 7th ACM Symp. Inf., Comput. Commun. Security, 2012, pp. 87–88.

[14] A. N. Khan, M. L. M. Kiah, S. U. Khan, and S. A. Madani, "Towards secure mobile cloud computing: A survey," Future Gener. Comput. Syst., vol. 29, no. 5, pp. 1278–1299, Jul. 2013.

[15] Cloud Security Alliance, "Security guidelines for critical areas of focus in cloud computing v3.0," 2011.

[16] Vaza, R.N., Prajapati, R., Rathod, D. et al. Developing a novel methodology for virtual machine introspection to classify unknown malware functions. Peer-to-Peer Netw. Appl. 15, 793–810 (2022). https://doi.org/10.1007/s12083-021-01281-5

[17]Prajapati, Dr Ramesh T., Dr Dushyantsinh Rathod, and Mr Tejas Kadiya. "A New Algorithm for Load balancing and Fault Tolerance Mechanism in Computational Grid." International Journal of Advanced in Management, Technology and Engineering Sciences 7: 1-9.

[18]Mr.Ramesh Prajapati,Dr.Samrat Khanna," Grid Computing with different Fault Tolerant Mechanisms", IJSRD,March-2014Mrs.PriyaPatel,Mr.Ramesh Prajapati,Dr. Samrat Khanna," To Improve The Performance Of Computational Grid Using Fault Tolerant And Dynamic Load Balancing Algorithm For Grid Environment", IJTRE ,May-2016

[19]Ramkumar, M., Basker, N., Pradeep, D., Prajapati, Ramesh, Yuvaraj, N., Arshath Raja, R., Suresh, C., Vignesh, Rahul, Barakkath Nisha, U., Srihari, K., Alene, Assefa, [Retracted] Healthcare Biclustering-Based Prediction on Gene Expression Dataset, BioMed Research International, 2022, 2263194, 7 pages, 2022. https://doi.org/10.1155/2022/2263194

[20]Dr. R. T. Prajapati, D. N. Patel, D. D. P. Patel, D. M. Patel, D. R. Bhavsar, and P. P. Panchal, "Develop Whale Scan System for Marine life Protection using Convolutional Neural Network", IJASIS, pp. 84–91, Aug. 2025, doi: 10.29284/wy3qeg08

[21]Solanki, S., Prajapati, R. (2024). Wheat Seed Classification Using Gurobi Optimized Piecewise Linear Approximation-Based SVM. In: Pandit, M., Gaur, M.K., Kumar, S. (eds) Artificial Intelligence and Sustainable Computing. ICSISCET 2023. Algorithms for Intelligent Systems. Springer, Singapore. https://doi.org/10.1007/978-981-97-0327-2_30