Security and High Performance Data Classification for Cloud Storage

Bharath Kumar Rama[1], [2]Dr. S. Thaiyalnayaki

[1]Research Scholar, Dept. of CSE , Bharath Institute of Higher Education and Research, Chennai, Tamil Nadu, India
[2]Associate Professor, CSE, Bharath Institute of Higher Education and Research, Chennai, Tamil Nadu, India
Corresponding Author Email:bharathrama1010@gmail.com

Abstract –Cloud storage is crucial for modern data management, but security remains a challenge due to the increasing number of cyber threats and data breaches. By dividing information into three different classes—public, sensitive, and confidential—this article suggests a strong security mechanism for cloud storage. Public data undergoes basic integrity checks, sensitive data undergoes moderate encryption, and secret material undergoes strong encryption and stringent integrity controls. These individualized approaches are based on the method's assessment of the sensitivity of the various data classes. For better data security in cloud storage, this research suggests a novel design for format-preserving encryption based on random Sboxes substitution-permutation networks. This research introduces a novel categorization approach that classify data into appropriate encryption systems based on their class, thereby merging the fields of machine learning and multi-criteria decision-making. According to the simulation findings, our solution guarantees data security and integrity while being more efficient, accurate, and requiring less processing time.

Index Terms –Cloud Storage, Classification,Encryption, Tokenization, Decryption

1. Introduction

As businesses increasingly rely on cloud-based data and applications, safeguarding this information becomes crucial. Scalability, adaptability, and financial savings are just a few of the many advantages of cloud storage [1]. However, the security risks associated with these advantages are severe, including data breaches, unauthorised access, and data loss. To counter these threats, a compact security system that can keep private data safe is required. Data classification is one approach; it entails sorting data into sensitivity levels and then implementing separate security measures for each level. Without secure cloud storage, we compromise data availability, secrecy, and integrity [2]. Data breaches can cause reputational damage, legal consequences, and financial losses. There was a 10% rise from the previous year, with an average cost of $4.24 million, as reported by IBM in 2021 [3]. Additionally, organisations are required by regulations like HIPAA and the General Data Protection Regulation (GDPR) to adhere to tight criteria when it comes to protecting sensitive data. Serious penalties, including legal action, may ensue from maliciousnoncompliance to these rules. Data classification involves sorting information into different groups according to certain criteria, such as how sensitive, valuable, or vital it is [4]. Using this technique, organisations can prioritise their security efforts by identifying their most important and sensitive data. There are three main layers to data classification. (i) Openly available, non-sensitive information that may be given without fear of consequences. (ii) Private, internally held data that does not need extensive protection measures. It is typically only accessible to internal staff. (iii) Data that is considered confidential consists of very sensitive data that, if leaked, might result in serious consequences for the company or its employees. Included in this group are financial records, confidential company information, and personally identifiable information (PII). Advanced data-finding and classification techniques can automate this procedure to improve efficiency and accuracy.

These technologies scan and classify data according to predetermined criteria using machine learning algorithms [5]. To meet legal standards and account for data sensitivity, organisations set differentclassification levels. Organisations may customise the classification levels to meet their own requirements; nevertheless, public, internal, and confidential are common ones. There have to be specified criteria and corresponding security controls for each level of classification. For example, whereas internal data may only require basic access control, confidential data may require encryption, access restrictions, and continuous monitoring [6]. To make sure that only authorised individuals can access sensitive information, organisations may set up access restrictions based on the levels of classification. RBAC, or role-based access control, is a popular technique for controlling who can do what in an organisation. Using multi-factor authentication (MFA), users are required to confirm their identity using various means before they can access sensitive data, adding an additional layer of protection. They can encrypt data in transit or at rest to render it unintelligible if intercepted [7].In order to identify and react to any security problems, it is necessary to continuously monitor and audit data access and use. In addition to assisting with compliance reporting by giving proof of safety measures and data handling practices, automated monitoring technologies may provide real-time warnings and comprehensive audit trails, allowing organisations to quickly detect and mitigate risks [8]. Data classification is an ongoing process. Data sensitivity, legal requirements, and company activities are dynamic; therefore, it's important to evaluate and update this on a regular basis. Organisations should regularly review and update data classification policies as required. This keeps security measures up-to-date and in line with what the organisation requires and must comply with [9].Homomorphic encryption (HE) is a method that encrypts

data while allowing for homomorphic operations like adding and multiplying. When it comes to homomorphic operations, partially homomorphic encryption often only meets one of the requirements for addition and multiplication. Therefore, when a homomorphic encryption strategy encrypts the controller, it eliminates the need to first decode the controller's input data, thereby resolving the issue with the previously discussed conventional encryption method. Figure 2 shows that the controller doesn't need the private key to decode the encrypted data; it may assess it directly. To address the issue that partly homomorphic encryption does not enable additive homomorphism, they modified RSA encryption [10] and ElGamal encryption [11]. As a result, they encrypted the linear controller using the modified technique. Paillier encryption is an example of partial homomorphic encryption that only allows for homomorphic addition [12]. In [13], the authors propose a Paillier-based encrypted static feedback controller method. This has been the basis for a number of investigations into NCS cyber-security [14–16]. When it comes to adding and multiplying encrypted data, completely homomorphic encryption is superior to partly homomorphic encryption.

2. Background and related work

Many papers discuss the use of classification to secure cloud data. The suggested methods fall into two main classifications: intelligent or automated classification, which utilises an algorithm, and manual classification, where the user defines the classification.

In [17], the authors proposed cloud data centre security requirements. When compared to other suitable algorithms, Blowfish takes the least amount of time to encrypt and decode files, and it has the highest throughput. When it comes to securing sensitive data, the concept of disseminating and mixing works. With the use of a hybrid approach, users may have more confidence in cloud providers because their remote servers are secure when connected to the cloud with a hybrid approach. They overcome the most fundamental barrier to isolating sensitive data from access control when it comes to data privacy and security issues. Cryptography is the process of transforming material from its original form into an unintelligible code. Both public-key and private-key encryption fall under the umbrella of cryptography. In this case, they used SSL, DES, RC, and AES 128-algorithm encryption. They use keys to transform data into an unintelligible format. Consequently, only authorised individuals can access the stored information on the cloud server. Everyone can see the ciphertext data. They used Cloud, Advanced Encryption Standard (AES), Secure Sockets Layer (RSA), Blowfish algorithms, and encryption and decryption methods.

In [18], the authors specifically looked at the problem of privacy-preserving k-nearest neighbour (kNN) classification. In this scenario, a data owner (DO) sends their encrypted cloud database to a query user (QU), who then sends an encrypted query point to a cloud server (CS) and asks for the kNN classification labels without letting CS know about the privacy of either side. Previous safe kNN query techniques have not found any viable real-world applications due to their excessive computation costs or inability to fully provide the necessary security features. In order to address this issue more effectively, they suggested a new, efficient kNN classification technique that uses Paillier and ElGamal cryptosystems to create a semantically safe hybrid encrypted cloud database. In addition to concealing data access patterns from CS, the suggested methodology safeguards database security and query privacy. They conduct thorough tests to assess the protocol's efficacy and officially examine its security. Even though it achieves the same security and privacy features as the state-of-the-art protocol, the experimental findings show that the protocol's computing cost is about two orders of magnitude lower.

In [19], the authors used a spiderman correlation to compare machine learning algorithms trained on encrypted datasets. They encrypt the dataset using homomorphic encryption (HE)-based Rivest Shamir Adleman (RSA) and the Paillier algorithm. These techniques encrypt data on cloud servers. They encrypt data using an in-house key generator, which generates keys for the HE-based RSA technique. When compared to how long it takes to encrypt data and build a machine learning model with Azure auto-machine learning, the HE-based RSA technique comes out on top.

In [20], the authors developed a model that can dynamically categorise data according to security levels, ranging from low to high. The policies and classification process categorized the data into five tiers of security. The purpose of classification is to apply a sophisticated security method to data with a higher security level than data with a lower security level. Additionally, it could help keep the system usable by separating illegal data from legal data. Finally, multiple trials have tested the suggested methods. A lot of research has been done to test and compare different feature selection and classification algorithms for filtering legal documents. These include decision tree classifiers, SVM, NB, KNN, and meta-classifier arrangements. Compared to their performance while using the Chi-Square feature selection approach, results demonstrate that all classifiers perform better when using the information-gain feature selection methods. When compared to other individual classifiers, Support Vector Machine (SVM) performed the best. Nevertheless, out of all the classification methods, the suggested meta-classifier technique produces the most favorable outcomes.

In [21], the authors zeroed in on multimodal biometrics to generate a non-intrusive key that cryptography algorithms may use to encrypt and decode files sent to the cloud from clients. The authors train convolutional neural networks to extract features from biometric data, such as fingerprints and faces, to identify patterns for classification. They hashed each multimodal biometric to 512 bits, producing an output label with multi-label classification qualities. Combining these bits yields 1024 bits, sufficient for both encryption and decryption

using RSA cryptography. They used the false acceptance rate and the false rejection rate to look at the system as a whole. As a result, on average, fingerprints show 2.71% FAR, while faces show 2.13% FAR.

3. Proposed modelling

Scalability, adaptability, and reduced overhead are just a few of the many advantages that have contributed to cloud storage solutions' meteoric rise in popularity. Nevertheless, it raised major security concerns. One successful method for improving cloud storage security is data classification, which classifies data according to its sensitivity and then subjects it to appropriate security measures. This proposed system integrates data categorization with strong security methods to safeguard cloud-stored data. The primary goals of data classification are to establish the necessary level of data security and distinguish sensitive data from non-sensitive data. Figure 1 illustrates the suggested approach.

3.1 Data security necessities specification

This specifies the necessary safeguards to keep the organization's (or a particular system's) data safe. It guarantees the availability, integrity, and secrecy of data; it acts as a guide for establishing security protocols.

3.2 Data Classification

Organisations establish and classify data based on factors like sensitivity, significance, and regulatory or legal mandates. Assuring the security and privacy of an organization's data is an essential first step. Using the suggested strategy, we divided the data into three categories, each with its own sensitivity level, and presented them in this study. Data security is an issue with cloud computing, despite the fact that it provides a scalable and easy means to store and handle data. In order to navigate the cloud effectively, one must be aware of the differences between public, sensitive, and secret data.

Public data: Like publicly shared videos, images, and files, this information is not private or significant to the data owner, so everyone may view it. For public data, the best option is a public cloud storage service, such as Amazon S3. In the event that we need to host a website, a product brochure, or a publicly accessible dataset, they are straightforward to use, scalable, and inexpensive.

Sensitive data: We restrict access to sensitive data, such as images, videos, and files, to authorized entities only, ensuring that only the original owners or authorized individuals have access.

Confidential data: Data owners hold personal and significant information, such as financial transactions, health records, and company records, which are examples of such data. This class clearly defines and restricts access to data, preventing even cloud providers from accessing it unless authorized by the rightful owners.

For classification purposes, machine learning employs the Random Forest approach. Classification challenges are ideal for the strong machine learning approach known as random forests. They have the potential to be an invaluable asset in cloud computing for automatically determining the sensitivity level of data and implementing the necessary security measures. In order to provide more reliable and accurate prediction outcomes, random forests construct an ensemble of decision trees during the training phase. Each decision tree receives data separately for evaluation. The trees then cast votes to determine the final assessment findings. It is important to mention that the privacy-preserving evaluation system covered in this study relies on plurality voting. Each model's output serves as a vote, and the prediction relies on the model with the most votes. If there is a tie, we break the results randomly. With their higher noise tolerance and ability to circumvent decision trees' performance limits, random forests are a preferable option. Additionally, they are capable of running simultaneously.

3.3 Application of the security mechanisms

We solved the cloud storage security problem by implementing a suite of security mechanisms to protect data stored in the cloud. We have implemented measures like data encryption, authentication, and integrity check methods to ensure the security of our cloud-based system. A key component of information security is authentication, which restricts access to systems and data to only authorised users. Single authentication, also known as single-factor authentication (SFA), refers to the use of just one factor or technique to confirm a user's identity. Alternatively, multi-factor authentication (MFA), which includes three-factor authentication (3FA), uses many separate credentials for validation.

Single authentication: This includes three distinct types of authentication factors: an object we possess (like a smart card or token), an object we know (like a password), and an object we are (like a fingerprint or other biometric attribute). This method provides a step-by-step guide for users to create three-factor authentication, backed by pseudocode for security.

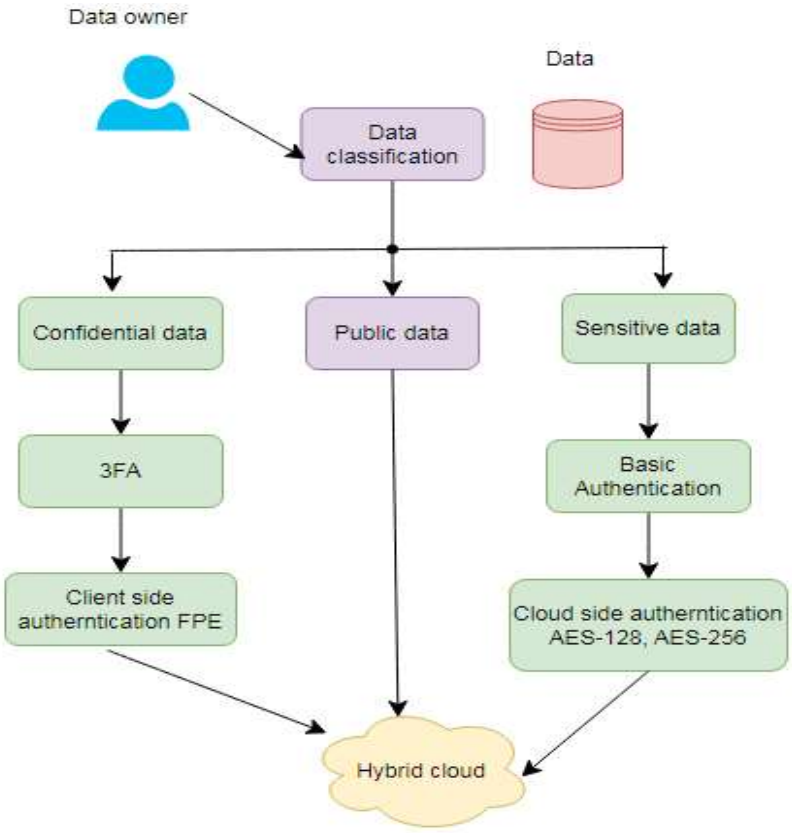


Figure1: Proposed framework

Three-Factor Authentication (3FA) involves three distinct types of authentication factors: (e.g., a password), something we have (e.g., a token or smart card), and something we are (e.g., a biometric trait like a fingerprint). The following algorithm outlines the process of implementing 3FA, accompanied by pseudocode to illustrate each step.

1) Collect the three authentication factors from the user.
2) Independently validate each factor.
3) Combine the results of the three validations to determine the overall authentication result.
4) Grant or deny access based on the aggregated result.

Baseline and Proposed encryption techniques

Advanced Encryption Standard (AES) 128-bit:

The AES 128-bit algorithm is widely used for safeguarding sensitive data. In its role as a symmetric block cypher, it encrypts and decrypts data in fixed-size blocks, with 128 bits being the current value, using a single shared secret key. Carefully moving the data blocks through a series of rounds (10 for 128-bit keys) is part of the encryption process. This key, which has 2^128 possible combinations (about 340 undecillion), is a very strong defence against brute-force attacks, in which hackers try each key option one at a time. These rounds employ both substitution and permutation procedures. While permutation rearranges the bits inside a block, substitution swaps out certain bits for others in accordance with a predetermined plan. Without the right key, this intricate ballet of replacements and permutations essentially scrambles the data.

Advanced Encryption Standard (AES) 256-bit:

AES-256 is a superior data encryption standard that offers exceptional protection for critical data. With a remarkably large number of potential permutations (2^256, surpassing 1.1 x 10^77), this version of the Advanced Encryption Standard uses a 256-bit secret key. Given that brute-forcing such an encryption is almost unfeasible with present technology, the astronomical security provided by such a long key is understandable. Similar to its 128-bit equivalent, AES-256's fundamental operation is identical. By using the same key for both encryption and decryption, it undergoes a sequence of manipulations that break data into 128-bit blocks, allowing it to function as a symmetric block cypher. If we compare AES-256 with its 128-bit counterpart, we see that the latter uses a more complex pattern of replacements and permutations inside each round—14 rounds for 256-bit keys compared to 10. The data is already highly resistant to decryption efforts, and the increased complexity only exacerbates its existing issues.

Proposed Format-preserving encryption (FPE):

In format-preserving encryption (FPE), ciphertext and plaintext remain the same. We can demonstrate the

mathematical basis of FPE using the Feistel network, a popular method for building such encryption systems. A common FPE strategy based on the Feistel network involves the mathematics described above in a high-level overview. A Feistel network divides the input data in half and then processes each half through a series of encryption operations. Each round combines the two halves using an F-round function and certain XOR operations. Considering a partially divided input block x (L and R),

Split: $x=L0\|R0$

Rounds: For n rounds, where $i=1,2,\ldots,n$:

$1Li=Ri-1$

$Ri=Li-1\oplus F(Ki,Ri-1)$

After n rounds, the ciphertext is $Ln\|Rn$

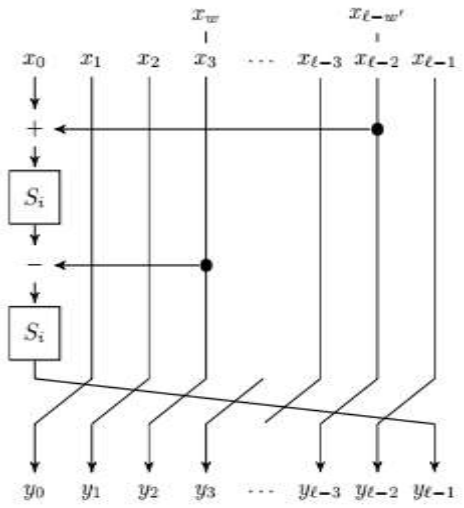
Where F is a round function that usually makes use of cryptographic primitives like hash functions or block cyphers, and Ki is the round key for round i.

This study proposes and evaluates a novel format-preserving protection-building approach. We refer to our format-preserving addition substitution transformation method as FAST. Both FPE and tokenization modes are compatible with FAST. The tokenization mode distinguishes itself from the FPE mode by using a key for domain separation and pregenerated random Sboxes as a stateless table secret, respectively. Several domains may share the former key. Compared to a particular domain-specific key, the stateless table secret sees far greater use. As a result, we consider a security model that functions similarly to FPE security, but with the stateless table secret exposed to the opponent. We create pseudorandom Sboxes using the key in FPE mode. $Y_b= \{0, 1,..., b-1\}$ is the alphabet that we use to define the group structure modulo an addition, without limiting its relevance. The "input" has a place in Y_b^l. A Sbox is an Y_b permutation. A collection of Rboxes consists of m Sboxes, represented as a tuple $R = (R_0, \ldots, R_{h-1})$.

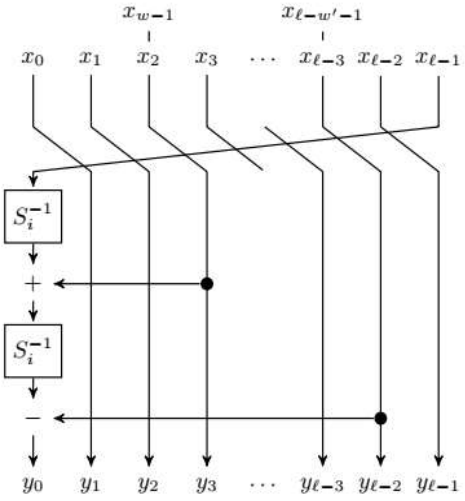
The following is the definition of the permutation $F_S[i]$ of Y_b^e, in $\{0, \ldots, m-1\}$, In Y_b^e, we have for any $(w_0, \ldots, w_{e-1})$as represented in Figure 2.

$$E_s[i](x) = \begin{cases} (w, \ldots, w_{l-1}, R_i(R_i(w_0 + w_{l-x'}))) & if x = 0 \\ (w_1, \ldots, w_{l-1}, R_i(R_i(w_0 + w_{l-x'}) - w_x)) & if x > 0 \end{cases}$$

$$D_S[i](x) = \begin{cases} (S_i^{-1}(S_i^{-1}(x_{l-1})) - x_{l-w'-1}, x_0, \ldots, x_{l-2}) if w = 0 \\ (S_i^{-1}(S_i^{-1}(x_{l-1}) + x_{w-1}) - x_{l-w'-1}, x_0, \ldots, x_{l-2}) \; if w > 0 \end{cases}$$



(a)



(b)

Figure 2. Shifting operation of branches: (a) Forward(b) Backward

Figure 3 shows the result of a three-layer circuit that does not use the circular shift of registers. Each layer involves one register that underwent a change x' layers ago, and another that will undergo another change x layers later.

Consider"sequence" $i_0, \ldots, i_{h-1}$ of n"indices", we describe our m-layer core "encryption/decryption" functions

$$CEnc_S[i_0, \ldots, i_{n-1}] = E_S[i_{n-1}] \circ \ldots \circ E_S[i_0]$$

$$CDec_S[i_0, \ldots, i_{n-1}] = D_S[i_0] \circ \ldots \circ E_S[i_{n-1}]$$

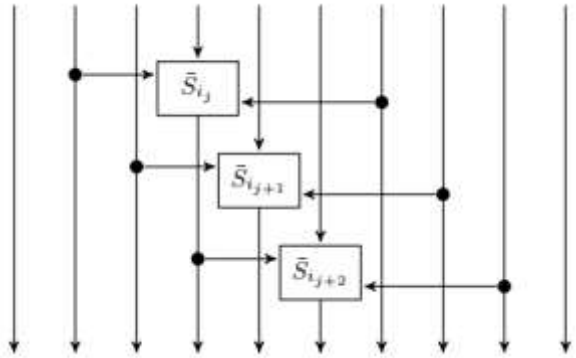


Figure3. Sequential Layers using$w = 3$ and $w' = 2$

Figure 4 shows the n-layer encryption system with the parameters $\ell = 4$, $h = 16$, $x = 2$, and $x' = 1$. Every h layer is considered to be h/ℓ rounds of ℓ layers as we need n to be a "multiple of ℓ".

Generate Rbox index sequences. Using a pseudorandom generator, we may create a sequence SEQ = $[i_0, i_1, i_2 \ldots, i_{n-1}]$ with n indices in $\{0, 1, \ldots, l-1\}$, given an E1-bit key ISEQ:

SEQ=$PRNG_{1,m,n}(I_{SEQ})$

CEncS $[i_0, \ldots, i_{n-1}](x_0, \ldots, x_{\ell-1})$

1: for j = 0 to n − 1 do

2: z ←R_{i_j}

3: for k = 1 to $\ell - 1$ do

4: $x_{k-1} \leftarrow x_k$

5: end for

6: $x_{\ell-1} \leftarrow$ y

7: end for

8: return $(x_0, \ldots, x_{\ell-1})$

CDec$S[i_0, \ldots, i_{n-1}](y_0, \ldots, y_{\ell-1})$

9: for j = n − 1 down to 0 do

10: z ← yℓ−1

11: for k = ℓ − 1 down to 1 do

12: ye ← ye−1

13: end for

14: $y_0 \leftarrow R_{i_j}^{-1}$

15: end for

16: return $(y_0, \ldots, y_{\ell-1})$

17: $I=I_S \leftarrow PRF^{L2}$ (I, $instance_1$, $cste_2$)

18: $R \leftarrow PRNG_{2,b,l}(I_S)$

19: return R

Setup$_2$ (I, $instance_1$, $instance_2$,tweak)

20: $(b, l) \leftarrow$ instance1

21: (ℓ, h, x, x') ← instance2

22: $I_{SEQ} \leftarrow PRF^{L1}$

23: SEQ ← $PRNG_{1,m,n}$(ISEQ)

24: return SEQ

Rbox Generation. Given an E_2-bit key I_S,

$$R \leftarrow PRNG_{2,b,l}(I_R)$$

A fixed pool R of Rboxes serves as the stateless table secret in the tokenization method.

$$instance_1 = (b, l)$$

$$instance_2 = (l, h, x, x')$$

$$I_{SEQ} = PRF^{L1}(I, instance_1, instance_2, cste_1, tweak)$$

$$ct = CEn_S[PRNG_{1,l,h}(I_{SEQ})](pt)$$

$$pt = CDec_S[PRNG_{1,l,h}(I_{SEQ})](ct)$$

E_{SEQ} is a key that generates SEQ and contains an E_1 bit. A constant representing the label "tokenization" and the size E_1, cste1, has no value. As a pseudorandom function, PRFΛ accepts an input of arbitrary length and returns a λ-bit value using an E-bit key. When $\lambda = E_1$, a key ISEQ for $PRNG_1$ is obtained. There is no predetermined PRF. For the most part, we stick with AES-CMAC.

Format-Preserving Encryption (FPE)

The FPE utilizes a pool of derived Sboxes, known as *R*. We may create the following given an E-bit key I, a tweak, a format defined by a and ℓ, parameters l and h, the chosen cypher suite algo = ($PRNG_1$, $PRNG_2$, PRF), and an input $pt \in Y_b^e$.

$$instance_1 = (b, l)$$

$$instance_2 = (e, h, x, x')$$

$$I_{SEQ} = PRF^{I1}(I, instance_1, instance_2, cste_2, tweak)$$

$$I_{SEQ} = PRF^{E2}(I, instance_1 cste_3)$$

$$R = PRNG_{2,b,l}(L_S)$$

$$ct = CEnc_S[PRNG_{1,l,h}(I_{SEQ})](pt)$$

$$pt = CDec_S PRNG_{1,l,h}(I_{SEQ})](ct)$$

Tokenization is similar to instances 1, 2, and PRF. We use an E_1-bit key named ISEQ to create SEQ. To create *S*, one uses an E2-bit key called IR. Changing either *m* or *a* necessitates recalculating pool S, a laborious process. Nevertheless, recalculating R shouldn't be necessary when ℓ is changed. This is the rationale for dividing instances 1 and 2.

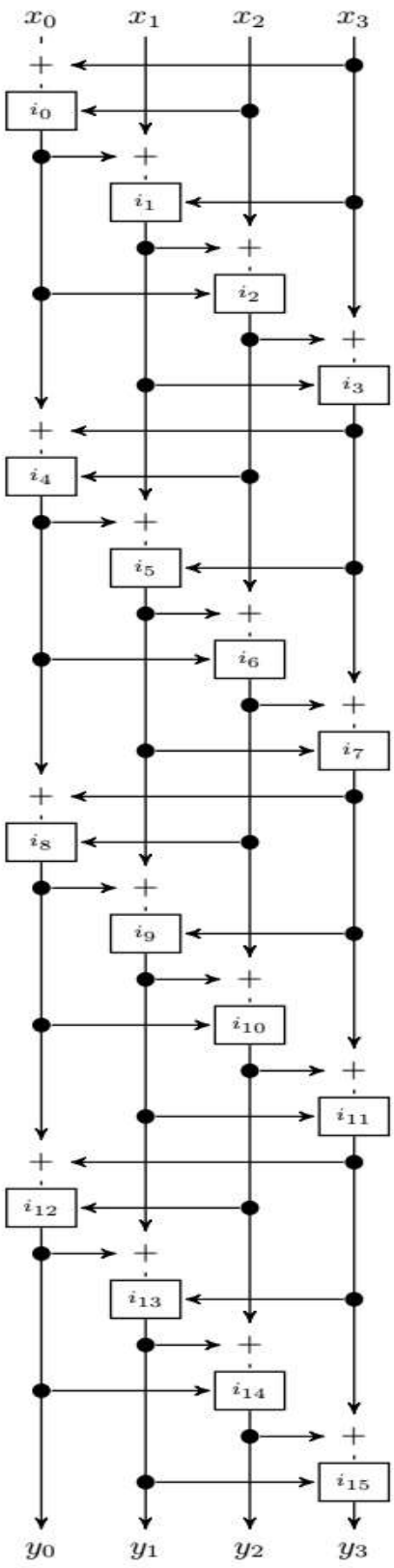


Figure 4. The n-layer encryption system

4. Results and discussions

This section discusses the steps and techniques for coding experimental algorithms in Python 3.9.3, using a system with 16 GB RAM, an i5 Intel Core i7 CPU, and Windows 10, emphasizing the importance of protecting sensitive information. For this reason, robust encryption techniques and thorough integrity checks are required to prevent data breaches and illegal access. Ensuring the highest level of data safety is crucial, even if it means sacrificing performance. Regular audits and modifications to encryption procedures are vital for adhering to regulatory obligations. Tables 1 and 2 display the results for 5 MB files both before and after categorization. Using three different methods—AES 128 bits, AES 256 bits, and FEP—to achieve a 5 MB file size is evident.

Table 1. Performance comparison of execution time before classification for 5MB file size

Sensitivity level	AES 128 bits	AES 256 bits	FEP
Public	42.32	38.35	37.36
Sensitive	45.36	42.3	35.46
Confidential	48.77	45.21	38.77

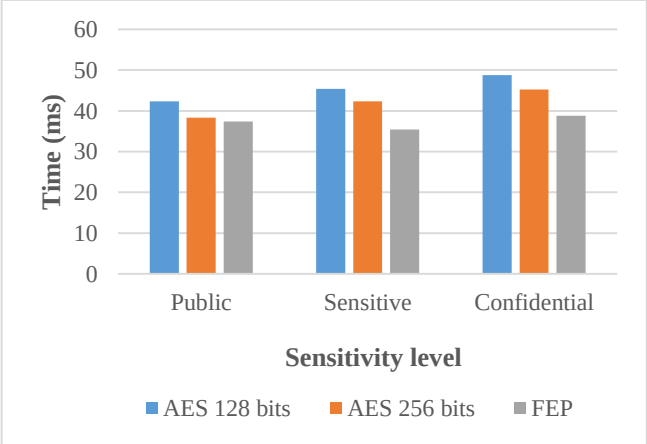


Figure 5. Execution results of AES 128 bits vs AES 256 bits vs. FPE before classification

Table 2. Performance comparison of execution time after classification for 5MB file size

Sensitivity level	AES 128 bits	AES 256 bits	FEP
Public	39.24	37.21	35.14
Sensitive	43.21	41.21	33.21
Confidential	45.21	42.12	33.21

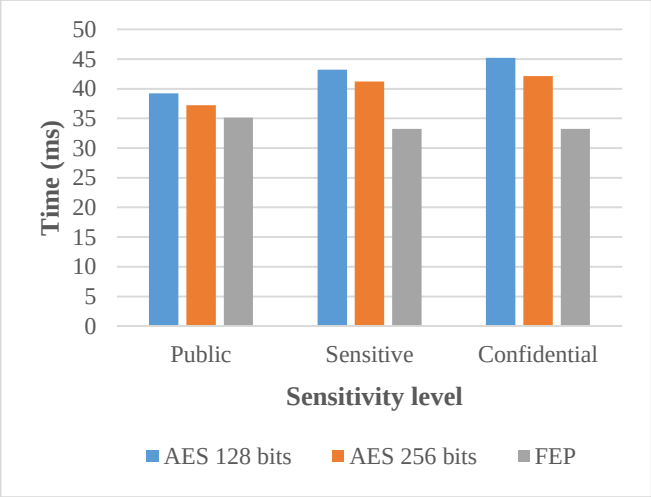


Figure 6. Execution results of AES 128 bits vs AES 256 bits vs FPE after classification

Figure 7 shows that prior to classification, the FPE approach outperforms AES 128-bit and AES 256-bit in terms of encryption time, with times of 11 ms for public, 27 ms for sensitive, and 33 ms for secret data. In a similar vein, when compared to AES 128-bit and AES 256-bit, the FPE approach gets almost the best decryption time: 9 ms for the public, 45 ms for the sensitive, and 52 ms for the secret.

Figure 8 shows that after categorization, the FPE approach outperforms AES 128-bit and AES 256-bit in terms of encryption time, with 9 ms for public, 23 ms for sensitive, and 22 ms for secret data. The FPE approach outperforms AES 128-bit and AES 256-bit in terms of decryption time, achieving 7 ms for public, 35 ms for sensitive, and 38 ms for private data.

Table 3. Performance comparison of encryption and decryption before classification for 5MB file size

Sensitivity level	AES 128 bits		AES 256 bits		FPE	
	Encryption (ms)	Decryption (ms)	Encryption (ms)	Decryption (ms)	Encryption (ms)	Decryption (ms)
Public	21	53	34	62	11	9
Sensitive	38	68	53	86	27	45
Confidential	52	86	64	85	33	52

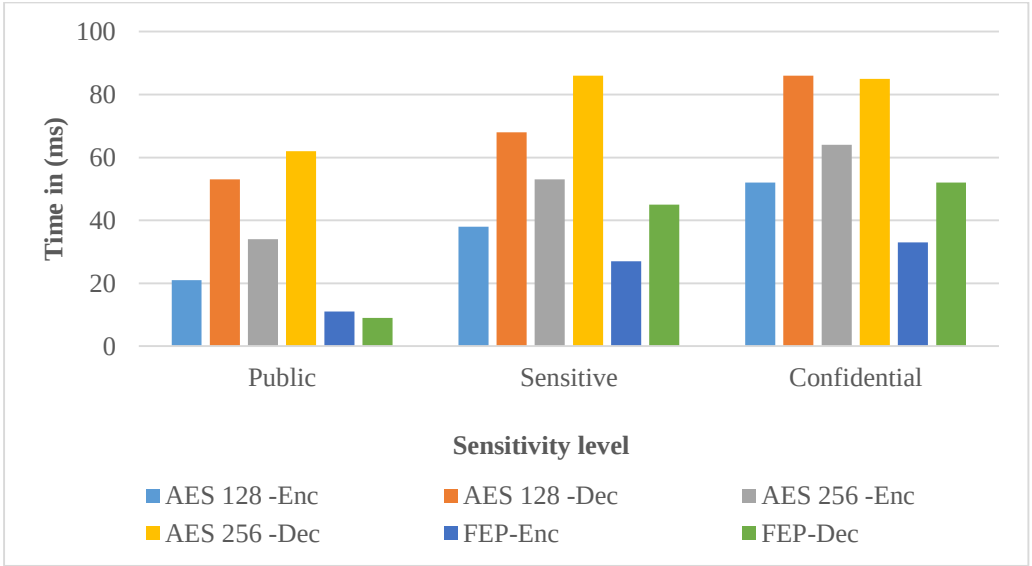


Figure 7. Encryption and decryption results of AES 128 bits vs AES 256 bits vs FPE before classification

Table 4. Performance comparison of encryption and decryption after classification for 5MB file size

Sensitivity level	AES 128 bits		AES 256 bits		FPE	
	Encryption (ms)	Decryption (ms)	Encryption (ms)	Decryption (ms)	Encryption (ms)	Decryption (ms)
Public	18	41	30	52	9	7
Sensitive	32	56	50	72	23	35
Confidential	48	75	58	75	22	38

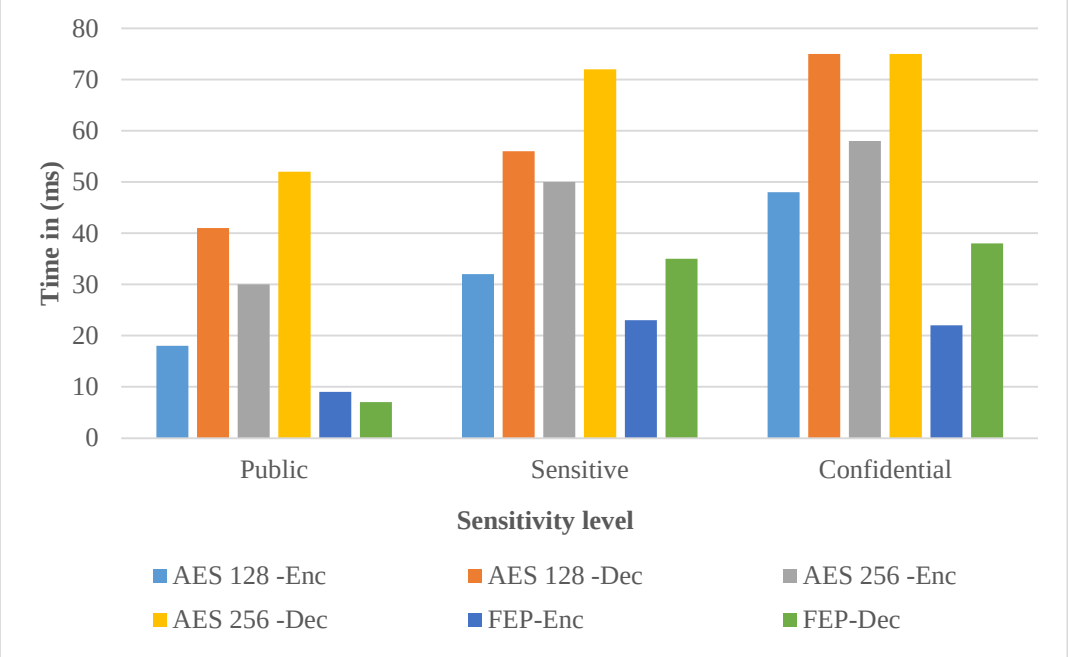


Figure 8. Encryption and decryption results of AES 128 bits vsAES 256 bits vs FPE after classification

5. Conclusion

The proposed technique classifies data into public, sensitive, and confidential categories to ensure the implementation of the right security measures for cloud-stored data. This approach improves the overall security of cloud storage solutions while balancing performance, regulatory compliance, and resource utilisation. It also optimizes operational efficiency. Using a classification model to group the data according to security needs was the main emphasis of the work. To classify data according to data security needs, our approach enhances the Random Forest classifier with the help of goal-programming decision-making. The optimal time for public, sensitive, and secret encryption is 9 ms, 23 ms, and 22 ms, respectively. We must use such techniques to ensure data security and integrity and to mitigate risks.

References

[1] Okike, Benedict & Omali, Theresa. (2023). Leveraging cloud computing for enhanced library services and efficient information delivery in the digital age. 19. 2023.

[2] Fadli, Ouijdane & Younes, Balboul & Fattah, Mohammed & Mazer, Said & Elbekkali, Moulhime. (2024). Security of IoT-Cloud Systems Based Machine Learning. 10.1007/978-3-031-48573-2_64.

[3] IBM. (2021). Cost of a data breach report 2021. Retrieved from https://www.ibm.com/security/data-breach.

[4] Kumar, K. & Reddy, P. & Gowri, D. & Lakshmi, P. &Priya, K. & Sainadh, A. & Kalyani, A.. (2023). Text Classification on Twitter Data Using Machine Learning Algorithm. International Journal for Research in Applied Science and Engineering Technology. 11. 631-635. 10.22214/ijraset.2023.57419.

[5] Amma N G, Dr & Madhavaraj, R.. (2023). Malware Analysis Using Machine Learning Tools and Techniques in IT Industry. 10.1007/978-981-99-2115-7_8.

[6] Luqman, Saqib. (2018). Securing Cloud Storage: Encryption, Access Controls, and Data Loss Prevention.

[7] Jain, S., & Bhardwaj, S. (2016). Enhancing data security in cloud computing using 256 bit AES encryption. International Journal of Computer Applications, 149(10), 9-13. https://doi.org/10.5120/ijca2016911575.

[8] Chakraborty, S., Ramachandran, A., & Rao, B. K. (2018). Cloud computing security challenges & solutions - A survey. Procedia Computer Science, 132, 1415-1423. https://doi.org/10.1016/j.procs.2018.05.281

[9] Fernandez, E. B., Mujica, S., Larrondo, E., &Juárez, M. (2019). Data classification and security in cloud computing. Journal of Cloud Computing, 8(1), 1-14. https://doi.org/10.1186/s13677-019-0131-0.

[10] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key," Communications of the ACM, vol. 21, no. 2, pp. 120–126, 1978.
[11] T. E. Gamal, "A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms," Proceedings of CRYPTO, vol. 196, pp. 10–18, 1984.
[12] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," Advances in Cryptology-Eurocrypt'99, vol. 1592, pp. 223–238, 1999.
[13] F. Farokhi, I. Shames, and N. Batterham, "Secure and private control using semi-homomorphic encryption," Control Engineering Practice, vol. 67, pp. 13–20, 2017.
[14] Y. Lin, F. Farokhi, and I. Shames, "Secure control of nonlinear systems using semi- homomorphic encryption," in Proceedings of the IEEE Conference on Decision and Control, pp. 5002–5007, Miami, FL, USA, December 2018.
[15] C. Murguia, F. Farokhi, and I. Shames, "Secure and private implementation of dynamic controllers Using semi-homomorphic encryption," IEEE Transactions on Automatic Control, vol. 65, no. 9, pp. 3950–3957, 2020.
[16] J. Tran, F. Farokhi, and M. Cantoni, "Implementing homomorphic encryption based secure feedback control," Control Engineering Practice, vol. 97, pp. 1043501–10435012, 2020.
[17] Dalave, Chetan &Lodh, Anushka&Dalave, Tushar. (2022). Secure the File Storage on Cloud Computing Using Hybrid Cryptography Algorithm. International Journal for Research in Applied Science and Engineering Technology. 10. 672-676. 10.22214/ijraset.2022.41332.
[18] Wu, Wei & Liu, Jian&Rong, Hong & Wang, Huimei& Xian, Ming. (2018). Efficient k-Nearest Neighbor Classification Over Semantically Secure Hybrid Encrypted Cloud Database. IEEE Access. 6. 1-1. 10.1109/ACCESS.2018.2859758.
[19] Kiratsata, Harsh & Panchal, Mahesh. (2021). A Comparative Analysis of Machine Learning Models developed from Homomorphic Encryption based RSA and Paillier algorithm. 10.1109/ICICCS51141.2021.9432348.
[20] Shakir, Mohanaad& ABUBAKAR, ASMIDAR &Yusoff, Yunus&Ashfaque, Mohammed & Al-Emran, Mostafa. (2016). Model of security level classification for data in hybrid cloud computing. Journal of Theoretical and Applied Information Technology. 94.
[21] Mammo, Andualem&Zemene, Solomon. (2023). Multimodal Bio Cryptography for Securing Cloud Computing with CNN. 10.59122/134F5QW.