An Integrated Cuckoo AES Key Generationapproach with Bilinear Pairing to Detect Malicious Third-Party Auditors in the Cloud

Akheel Mohammed

Department of Computer Science and Engineering,

Shadan Women's College of Engineering and Technology,

Khairatabad,Hyderabad,Telangana,500004, E-Mail: akheelmd7@gmail.com,

Dr D Vasumathi

Department of Computer Science and Engineering,JNTUH College of Engineering,

Kukatpally,Hyderabad,Telangana,500085,E-Mail: rochan44@gmail.com

Abstract: To handle huge amounts of data, either a single user or organization are depending on the cloud environment but security to their data is the major concern that the cloud service provider has to take care of in the public clouds during the process of outsourcing. So, the cloud environments have deployed a component known as "Third Party Auditor", which verifies the data in the cloud and ensures that the data is not tampered with by the attackers. While interacting with the public cloud, every user has a period known as the "verification period", during which the user can raise the data challenge task to the auditor. In general, the data uploaded by the user are divided into various blocks or segments. Hence, to verify a single user data the auditors have to generate multiple integrity reports and should check their consistency. If TPA detects any inconsistent data, then it has the right to reject the process and notifies the same user but the major problem is TPA works simultaneously with more users because of which the verification process becomes a burden to the TPA. If the system is overloaded with more verification requests then the cost of computing in the cloud increases and the time to check the data integrity also increases which results in compromise by the TPA. So, to address this issue many researchers have proposed integrity verification algorithms but most of them suffer from many design issues like resource allocations, relay time decisions, and time consumption hence the proposed system has implementeda cuckoo algorithm integrated with AES to generate the keys and performs bilinear map functions to repair and audit within an optimal time.

Keywords: Malicious Third-Party Auditor, Integrity Verification, Bilinear Maps, Auditing

1. Introduction:

The job of the TPA is tedious because it has to serve multiple client requests simultaneously and it also makes an agreement with the cloud service provider to guarantee the services at various levels. So, at every epoch, there are huge chances for the TPA [6] to get comprised and this comprises can happen in two ways. They are a. **malicious auditors**, who issue the integrity verifications results without checking the data blocks during the verification (or) challenge phases. b. **misbehaved auditors**, who hide the corrupted data scenario from the users by disapproving the service level agreement. In this scenario, the cloud server also gets compromised either by replacing the public key, which is accessible by both parties during the data exchange or by changing the content of the master key, which is produced by the Key Generation Centre (KGC) [7]. So, the proposed system tries to detect malicious auditors based on the revocation strategies it applies during the resource allocation process.

The primary concern one has to take care of before the auditing is an efficient key generation with the help of a genetic algorithm known as “Cuckoo integrated with AES algorithm” to provide a more dynamic encrypted key[8] for each block associated with the file. This Cuckoo algorithm helps in reducing both encryption and decryption time even when the cloud handles a huge amount of data. This in turn helps the attackers not to easily breach the data of the owners with simple or advanced encryption techniques.

2. Literature Review:

In [1], Yang Xu et al. proposed blockchain-related data auditing mechanism for providing security to the data stored in the devices. This model implements a bilinear cryptosystem by preserving its non-degeneracy and computability properties to verify the duplicate files in the cloud by the Third Party Auditor (TPA). The TPA verifies the data index and audit log using blockchain technology which ensures that the data shred does not tamper and trustworthy. The main problem with the cloud service providers is “untrustworthy”, because of the inefficiency in their deduplication algorithms to handle the attackers when they want to tamper other user’s data. So to handle this situation, this research the TPA takes the help of blockchainto handle every transaction that occurs at every node in the service provider by managing the few parameters in integrity proof checking, deduplication challenge, and auditing the challenge.

In [2], RunhuaXu et al. explored a dynamic protocol of commitment to creating trusted TPA in the cloud environment. A binary list is created to check whether every node is having a unique IP address to prove its trust in the service. By performing this operation the system identifies duplicate nodes and scans them to ensure that they are not BOT attackers. A new list is created by removing the duplicate elements and sent to the service provider, which verifies the protocol log by generating public key and security key sharing with the user. If the user communicates back with the same security key then the data is outsourced confidently in the cloud. These verified users are cross-checked with the TPA using the digital signature verified mechanism of snap freshness.

In [3], Sasha MahdaviHezavehiet al. designed ananomaly framework for identifying the attacks in the cloud by implementing the TPA as controller of the unit to run numerous simulation tests to prevent the denial of service attacks in the cloud, which is caused by the BOTs. This model makes the TPA get a service agreement from the service provider. The notification generator sends a communication message to the provider whenever it identifies that the IT infrastructure resources like CPU, the bandwidth of the network, or any other resources are got flooded with many requests i.e., more than their capability and unable to fulfill them. This is measured by TPA by setting initially some response time, popularly known as RTSLA for every request based on their type and it also includes the excess time as the threshold value. Now, the TPA waits for the node to send the acknowledgment until its estimated time if it is not able to get it then the notification detector comes into the picture for taking care of the malicious attackers through the gateway controlling unit. This framework can also handle the filtering attacks i.e.classifications of users based on their activities and computing the various parameters like alarm, mean time between the request and response, computational latency, and several requests handled per unit time.

In [4], Folasade Mercy Okikiolaet al,experimented with the health care domain to prevent the health records from getting tampered with by the attackers. This system implements logging and watermark extraction techniques to perform audit trails by the TPA. The novel concept in this algorithm is it can even handle the attacker within the system whereas the previous research works try to find the outside attacker through a biometric authentication system. The watermarking module helps the system to find the no interestedregions, which is a complement component to the general procedures, and these parts are divided into the least significant bits and the features are extracted from it. The extracted bits are encrypted using the SHA algorithm are transmitted successfully to the patients as well as doctors and logs are created to visualize the activities performed by each authenticated user for the further verification process. It also implements the session management scheme for every user to provide security by providing a predefined time for every activity they do.

In [5], Yuan Zhang et al. defines a novel integrity mechanism for verifying the auditors in the cloud. It implements a certificate-less transmission mechanism using blockchain technology to control the malicious auditors. In this model, the TPA and the user agrees on the verification period. The security for the data is provided by the symmetric key generation component, which generates bilinear map keys to exchange the data between the auditor and the validity of the key is decided by the blockchain technology, which converts the normal data block into Ethereum block for doing the further process. During the hash function construction process, if any of the blockchain cryptosystems feels it as an inaccurate data challenge then it rejects the user and TPA to access the data. It also takes care of the security analysis by performing the chosen- message attack, in which the search query encryption is taken care of by the public keys and a forgery generates output as true if it finds the valid digital signature approved by the service provider otherwise it returns false.

3. Proposed Methodology:

The proposed methodology has been executed in the simulator known as “CLOUDSIM”, which is a robust and fastest environment to work and utilize the resources of cloud computing. In this environment, there are two important classes, which are represented as shown in figure 1.’

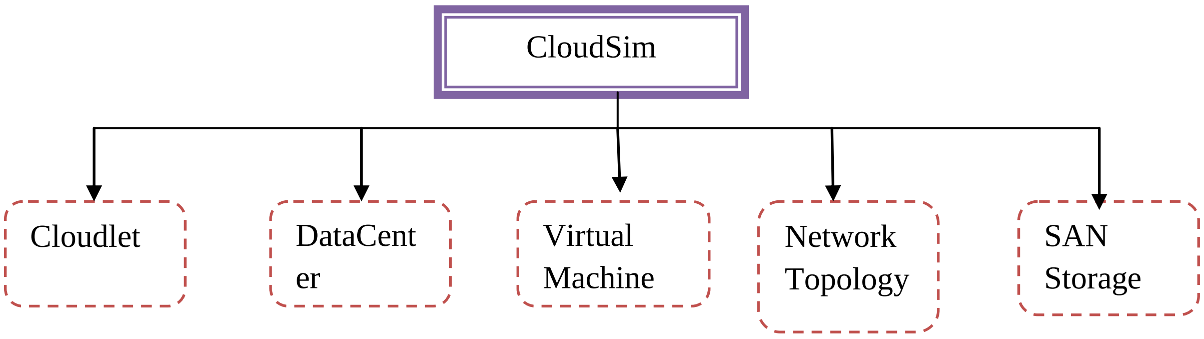

Figure 1: Components of Cloudsim

The description of the components are shown in table 1

S.No	Component Name	Description
1	Cloudlet	This class defines the attributes related to file and operations related to task management
2	Data Center	This provides core hardware infrastructure services to the cloud service provider. It is responsible for memory-related and network-related measurements
3	Virtual Machine	It is initiated by the cloud host to interact with different resources of the cloud to complete the task
4	Network Topology	It holds the essential information related to network topology
5	SAN Storage	It's a combination of multiple storage devices like hard disk

The major objective of this system is to detect the malicious TPA's using the bilinear map function. In this mechanism, to reduce the burden on the single TPA, the system has proposed a group auditing mechanism that dynamically distributes the user information among different auditors. This overall process is divided into two steps as shown in figure 2.

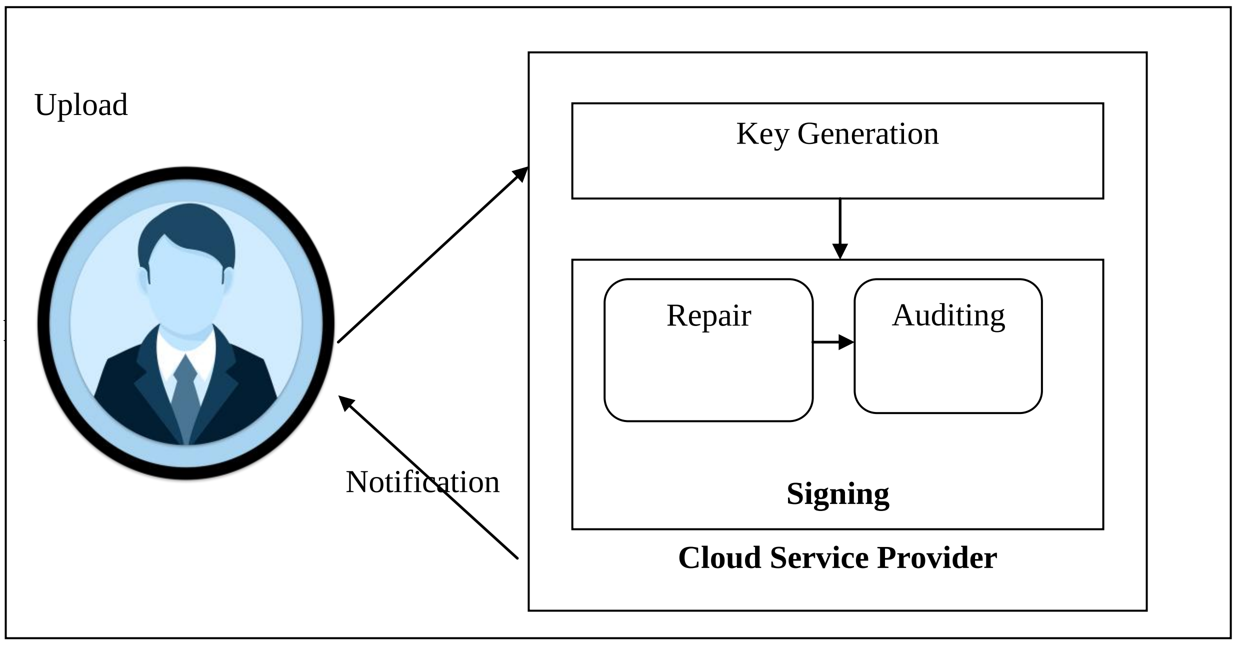

Figure 2: Architecture of Auditing Process in Cloud

3.1. Key Generation:

When the data is outsourced in the cloud, the major concern of any data owner is its security; the proposed algorithm provides security by generating a secret key by integrating the cuckoo search, which is a popular genetic algorithm with the AES encryption algorithm. The power of cuckoo search lies in its fitness function to generate the key, which is k-bit hexadecimal. Initially, the file uploaded by the data owner is divided into blocks and for each block, an associated key is generated and is shared with TPA's for further auditing. In the proposed algorithm, the fitness function decides the number of rounds and the size of the key depending on the file size.

Algorithm for Cuckoo Search AES:

```
Input: A plain text data file uploaded by the user

Output: Encrypted data file each having its own secret associated with it

Begin:

1. Initialize the population with round keys of AES

S(k)←{S(k1), S(k2),……….,S(kn)}

2. key←S(k1)

3. g_value[]←random()

3. for i in len(S(k)) do:

   a. cuckoo_secret_key←levyflight_population(S(k))

b. j←nest(cuckoo_secret_key)

c. if (j>key):

j←update_solution()

d. generate probabilities of each key

e. if(g_value[i]>g_value[i-1])

update the key value by finding the difference  between two values, which are generated
              randomly.

f. if(fitness(S(k[i])>fitneness(S[k[i+1]))):

S(k[i+1])=S(k[i+1])

else

S(k[i+1])=S(k[i])
```

End

Many cloud servers like Google, Microsoft Azure, and others use the AES algorithm for providing data integrity to the user information. The reason behind the combination of the AES algorithm with cuckoo is to define an objective function that minimizes the number of data blocks and keys to encrypt the data. Cuckoo search is a simple and popular Meta-heuristic algorithm that can efficiently work with the dynamically generated population adapts to the blocks.

3.2. Signing:

In general, the data can be verified either by the data owner or by a third-party auditor due to overhead problems caused by the misbehaved owners; the cloud system has moved the process of verification from owners to TPA's. Either due to overloaded or misbehaviour, TPA verification is also not up to the mark, so the proposed system instead of single auditing, performs group auditing if it finds malicious auditor by using bilinear map function during the group verification mechanism. The bilinear map function, which is a cyclic abelian group to define a function that pairs the multiple elements with the help of homomorphic substitution and later it performs the aggregation operation. In this mechanism, initially, parameters are generated which are associated with cyclic groups namely {G1,G2,G3,….Gn} which is a famous mechanism for forming a round key agreement (or) tata pairing that generates the encrypted key and decrypted key using the equations (1) & (2)

$$div\left({}^{f}_{e}X\right) = n(X) - n(infinity) \text{ - (1)}$$

Where,

n(X) represents the zero-order of n at X point

Equation 1 performs suborder groups by defining nth order derivative and generates the encrypted keys as

$$E(X,Y) = f_X^{(D(Y))^{Y^{k-1/n}}} \text{ - (2)}$$

Where,

R ∉{ϕ, X, -Y, X-Y}

D(Y) is computed as (Y+R)-R

This bilinear pairing helps us to identify the malicious attackers which are having the same identity block. The algorithm for the identification of malicious attackers is:

Pseudocode for bilinear pairing:

```
1. G_binary←binary_res(G1,G2,……,Gn)

2. Choose a point
```

```
3. Initialize generating parameters, P←X, Q←1
4. for i← 0 to n do:
a. define the tagent and vertical line through 2*P
b. compute P using cuckoo search fitness function
c. if n(i)==1 then
i. define lines through P+Q
    ii. P←P+Q
5. Return E(X,Y)
```

Results and Discussion:

Many research works are carried out to generate the keys efficient manner but still the time to generate them is a crucial factor. The proposed algorithm reduces the time to encrypt the key as shown in table 2 for different sizes of data.

Table 2: Time to Encrypt the file blocks

Input Size in MB	RSA	AES	Cuckoo+AES
100	0.635	0.61	0.584
150	0.685	0.643	0.602
200	0.713	0.699	0.681
250	0.775	0.742	0.71
300	0.895	0.878	0.85

The performance of the encryption time is graphically represented in Figure 3, which clearly states that among all the algorithms the proposed algorithm has the least time to encrypt even in the case of different file size bocks.

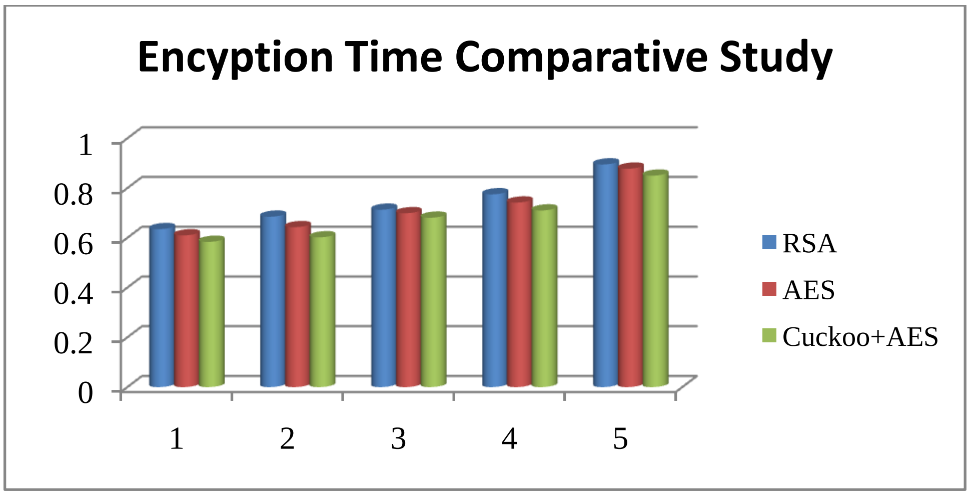

Figure 3: Encryption Time

The proposed algorithm reduces the time to decrypt the key as shown in table 3 for different sizes of data.

Table 3: Time to decrypt the file blocks

Input Size in MB	**RSA**	**AES**	**Cuckoo+AES**
100	0.815	0.793	0.629
150	0.887	0.809	0.653
200	0.891	0.823	0.701
250	0.902	0.874	0.724
300	0.912	0.883	0.744

The performance of the decryption time is graphically represented in Figure 4, which clearly states that among all the algorithms the proposed algorithm has the least time to encrypt even in the case of different file size bocks.

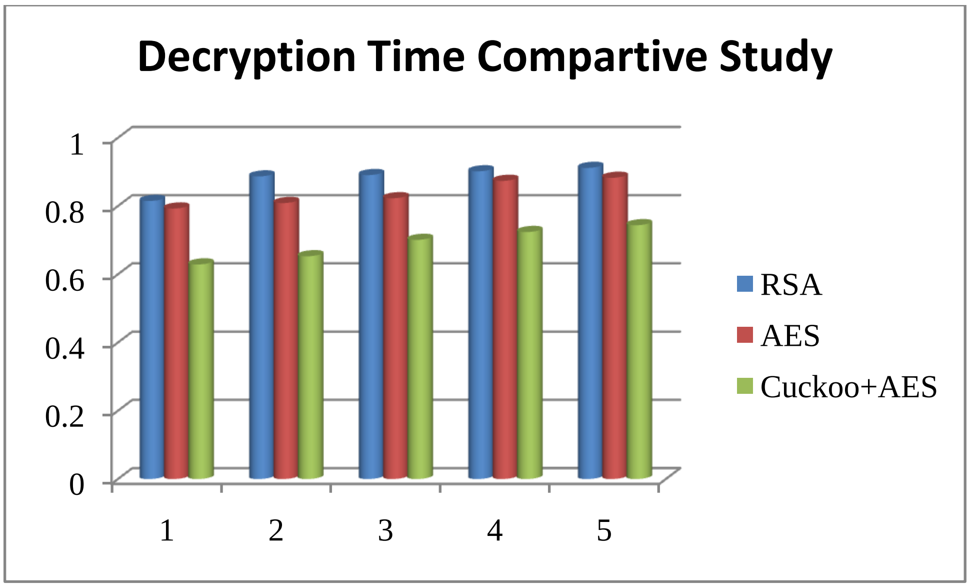

Figure 4: Decryption Time

The Cuckoo AES Algorithm helps in finding the block size dynamically with variable length. The execution time for RSA and Cuckoo AES is shown in table 4.

Table 3: Comparative Study on Execution Time

Input size in MB	**RSA Algorithm**	**Cuckoo AES**	
	Fixed Length (Time to execute)	Data Block Size+ Variable Length	Time to Execute
100	0.52	4 blocks+8 bit	0.11
150	0.56	4 blocks+8 bit	0.13
200	0.60	6 blocks+10 bit	0.16
250	0.63	6 blocks+10 bit	0.17
300	0.66	8 blocks+12 bit	0.20

In table 5, the proposed system compares the amount of storage and execution required by different servers and proves that the proposed system consumes less memory than the existing models

Table 5: Memory Utilization Analysis

S.No	Algorithm	Memory Utilization in MB
1	RSA	160
2	AES	135
3	CUCKOO+AES	108

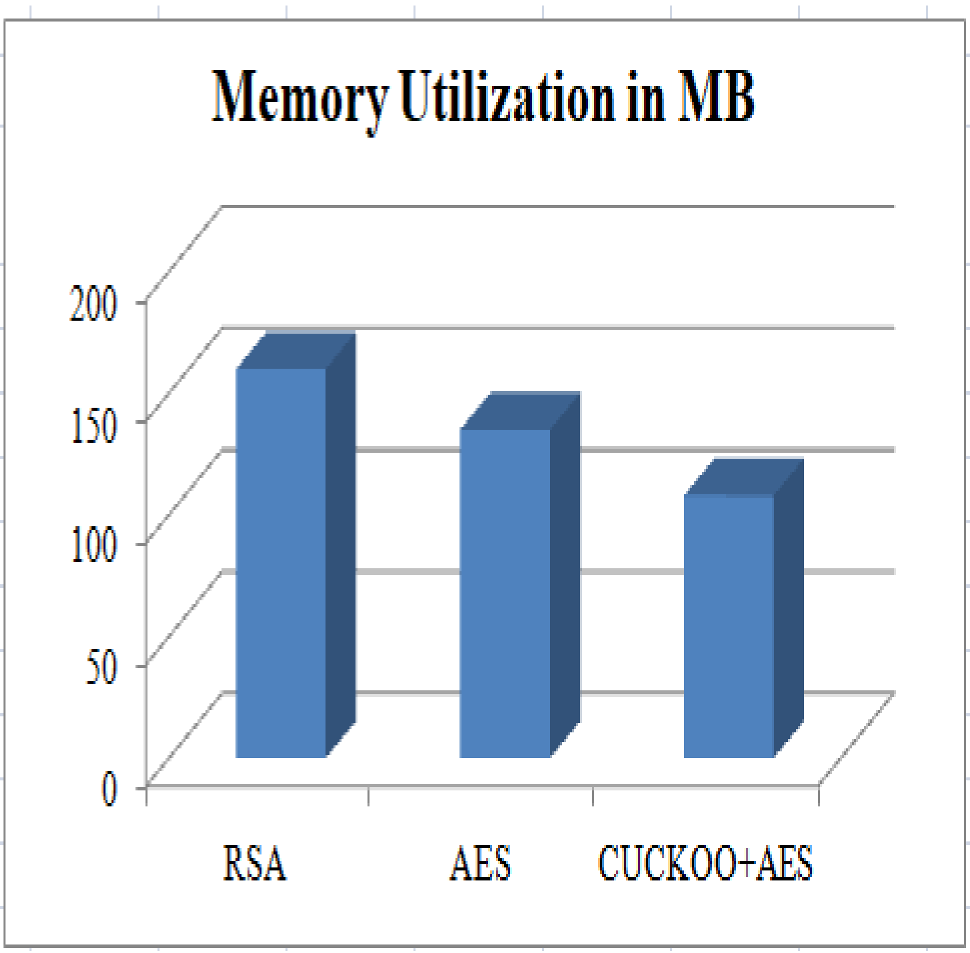

Figure 5: Memory Utilization by Different Algorithms

In Figure 5, X-axis represents the previous algorithms RSA and AES, the proposed algorithm (CUCKOO+AES) and Y-axis represents the amount of memory utilized in MB, to execute a data block.

The results are executed using the simulator known as CloudSim, which is better explained as: There have been various advancements in the Cloud computing sector due to its demand. Many drawbacks have been overcoming gradually to maintain its scalability and meet the standards of the present-day requirements. One such innovation is CloudSim,which is a simulating tool for accessing the cloud's services and infrastructure. This isdesigned by the CLOUDS Lab organization as anopen-sourceplatform and is entirely written in JAVA. This is used to design and simulate a cloud - based applications as a technique to assess an assumption for the reproduction of experiments and findings before the implementation of applications.For instance, if we want to implement an application or a webpage on the cloud, evaluate the facilities and loads our products that can be used to counteract bottlenecks and adjust their performance prior to the stake of use, then such assessments could merely be done using numerous scalable and cost - effective classes offered by the CloudSim package by programming the computation of the environment

Table 6: Execution Environment Parameters

Conditions	**Statistics with traditional approaches**	**Statistics with CUCKOO+AES**
Number of Chassis Switches in L4	1980	1980
Packet Size	1260 KB	1000KB
Line cards at L4	1630	1630
Ports at L4	72	72
Number of racks at L4	16	16
Number of Chassis Switches at L3	432	432
Line Cards at L3	164	164
Ports at L3	48	48
Number of racks at L3	128	128
Used virtual machines	1800	1361
Number of Servers	64	58
Maximum number of Cloud Service Users	18000	15642
Hosts in each rack	132	121
Each Host supports	16 processors	4 processors
Memory with each processor	256 GB	64 GB
Storage Memory	512 GB	128 GB
Virtual Disk Memory	430 GB	430 GB
Bandwidth for L4	256 GB/Sec	128 GB/Sec
Bandwidth for L3	128 GB/Sec	56 GB/Sec
Bandwidth for L2	64 GB/Sec	32 GB/Sec
Bandwidth for L1	16 GB/Sec	8 GB/Sec
Queue delay	0.005 Seconds	0.000 seconds
Burst time	0.0056 Seconds	0.0018 seconds
Idle time	0.0032 Seconds	0.00005 seconds

One more efficiency parameter, to prove the accurate system, table 7 compares the previous developed modules

Table 7: Accuracy Comparison

Algorithm/Metric	Accuracy	Execution Time	Response Time
APCLET	87.81	309817	100233
BSSCLET	80.22	357721	129034
LKCLET	86.59	305174	108931
Proposed	88.45	294189	99254

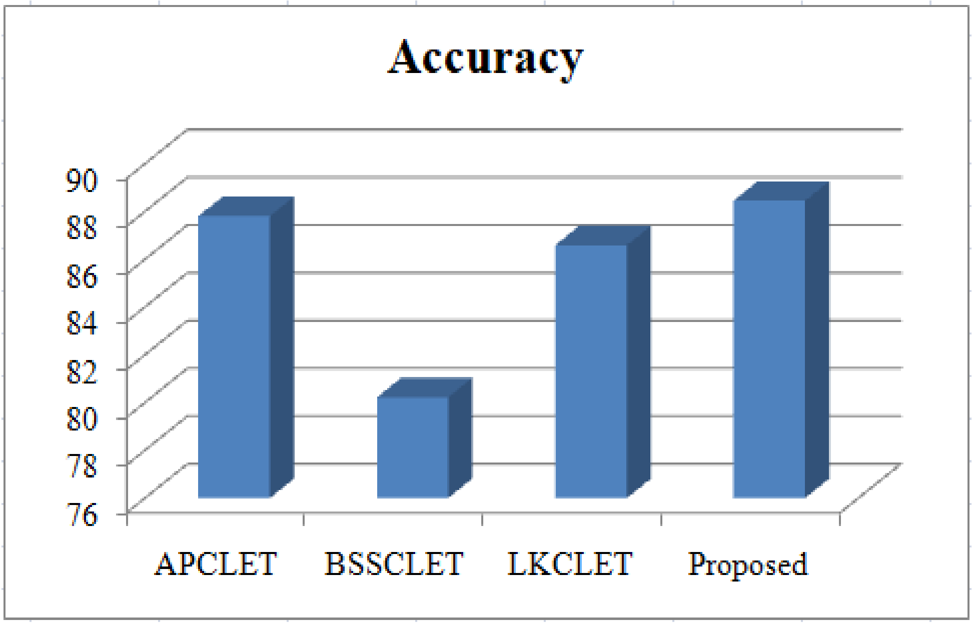

Figure 7: Analysis on Comparison

The table 8 compares the communication blocks cost against the number of malicious attackers based on their locations.

Table 8: Communication Cost versus Malicious Attackers

S.No	Methodology	Communication Block Cost	Number of Malicious Attackers	Dishonest Rate	Storage Required
1	PPP	900	2	5%	800 KB
2	CUCKOO+AES	883	2	4%	750 KB

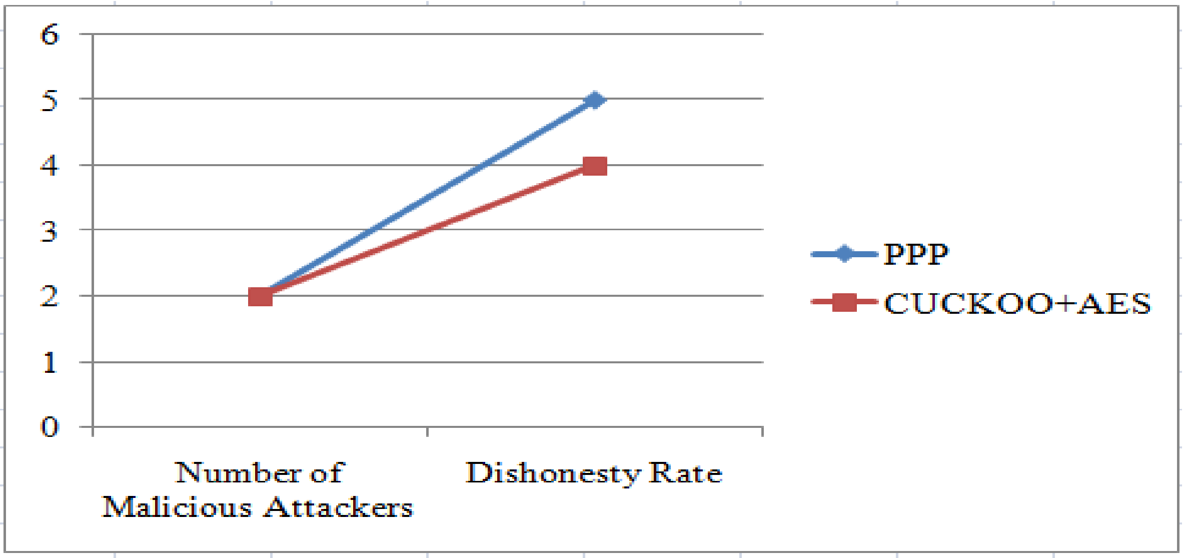

Figure 8: Analysis of various parameters in terms of PPP and Cuckoo+AES Algorithm

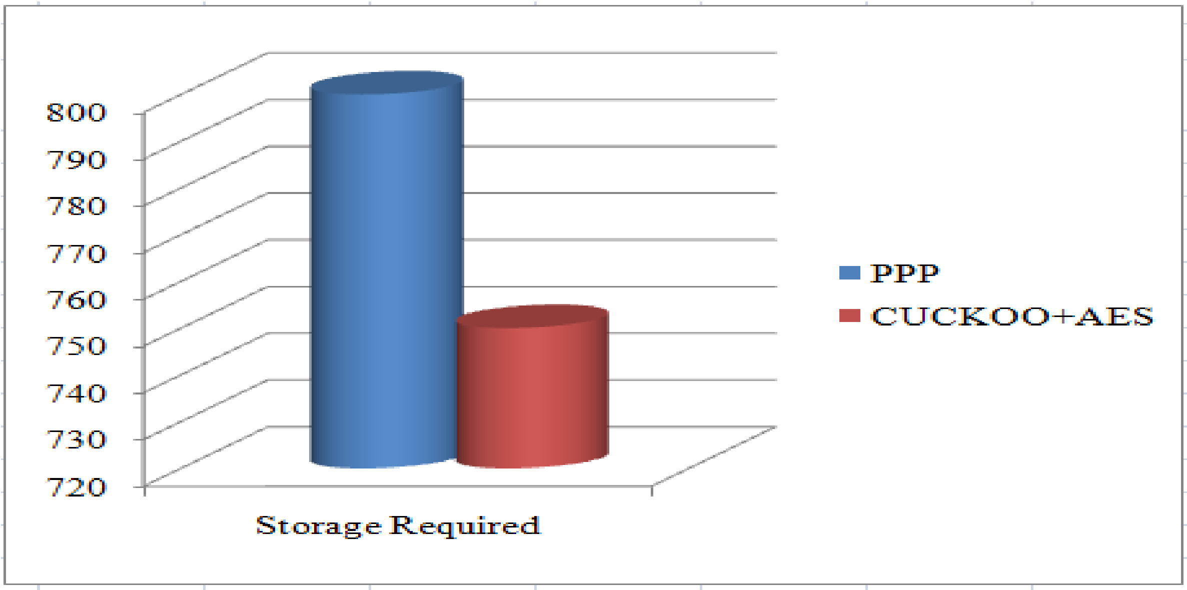

Figure 9: Analysis of Storage parameters in terms of PPP and Cuckoo+AESAlgorithm

Figure 8 and 9, shows the visualization of attackers, dishonesty rate and storage required. It clearly shows that even though attacker's rate remains same but dishonesty rate and storage requirements are reduced, proving that proposed algorithm not efficient in terms of storage but also dishonesty rates are better than the previous privacy preserving protocol.

Conclusion:

The proposed algorithm implements a cuckoo AES algorithm for key generation and efficiently reduces the encryption and decryption time. It also dynamically decides the number of blocks based on the file size and it also dynamically decides the length of the key. When compared with previous mechanisms it provides the repairing mechanism when it identifies the malicious Third-Party Auditors by defining the bilinear pairing algorithm to generate a round key, which ensures a trusted integrity check. This process of auditing helps the data owners to have a secure data outsourcing mechanism to publish their data across the globe. In future work, the system can concentrate certificate less management by the TPA, for providing the data integrity efficiently by reducing the overhead and cost expensiveness involved by the certification authority to issue a certificate every time whenever wants to verify the data.

References:

[1] Xu, Y., Zhang, C., Wang, G., Qin, Z., &Zeng, Q. (2020). A Blockchain-enabled Deduplicatable Data Auditing Mechanism for Network Storage Services. IEEE Transactions on Emerging Topics in Computing, 1–1. doi:10.1109/tetc.2020.3005610

[2] Xu, R., & Joshi, J. (2020). Trustworthy and Transparent Third-party Authority. ACM Transactions on Internet Technology, 20(4), 1–23. https://doi.org/10.1145/3386262

[3] MahdaviHezavehi, S., &Rahmani, R. (2020).An anomaly-based framework for mitigating effects of DDoS attacks using a third party auditor in cloud computing environments.Cluster Computing.doi:10.1007/s10586-019-03031-y

[4] Okikiola, F. M., Mustapha, A. M., Akinsola, A. F., &Sokunbi, M. A. (2020). A New Framework for Detecting Insider Attacks in Cloud-Based E-Health Care System.2020 International Conference in Mathematics, Computer Engineering and Computer Science (ICMCECS). doi:10.1109/icmcecs47690.2020.240889

[5] Zhang, Y., Xu, C., Lin, X., &Shen, X. S. (2019). Blockchain-Based Public Integrity Verification for Cloud Storage against Procrastinating Auditors.IEEE Transactions on Cloud Computing, 1–1. doi:10.1109/tcc.2019.2908400

[6] shree, S. R., ChilambuChelvan, A., & Rajesh, M. (2020). *Optimization of Secret Key using cuckoo Search Algorithm for ensuring data integrity in TPA.2020 International Conference on Computer Communication and Informatics (ICCCI).* doi:10.1109/iccci48352.2020.9104102

[7] Kalluri, R. K., & Guru, C. V. (2021). An effective analytics of third party auditing and Trust architectures for integrity in cloud environment. Materials Today: Proceedings. https://doi.org/10.1016/j.matpr.2021.03.312

[8] P. Huang, K. Fan, H. Yang, K. Zhang, H. Li and Y. Yang, "A Collaborative Auditing Blockchain for Trustworthy Data Integrity in Cloud Storage System," in IEEE Access, vol. 8, pp. 94780-94794, 2020, doi: 10.1109/ACCESS.2020.2993606.

[9] Alrabea, A. (2020). A modified Boneh-Lynn-Shacham signing dynamic auditing in cloud computing. Journal of King Saud University - Computer and Information Sciences.https://doi.org/10.1016/j.jksuci.2020.06.001

[10] A. Kumar, "A Novel Privacy Preserving HMAC Algorithm Based on Homomorphic Encryption and Auditing for Cloud," *2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, 2020, pp. 198-202, doi: 10.1109/I-SMAC49090.2020.9243340.