International e-Conference on New Horizons And Multidisciplinary Applications In Science And Technology
In Association withInternational Journal of Scientific Research in Science and Technology
Volume 9 | Issue 6 | Print ISSN: 2395-6011 | Online ISSN: 2395-602X (www.ijsrst.com)

Bagged Decision Tree Algorithm Using Bloom Filters to Reduce the Cloud Storage Capacity

Umesh G Deshmukh[1], Dr. Hemant S Mahalle[2]
[1]PhD Scholar, JJT University, Rajasthan, India
[1]Principle, Shri VR College, Sawana, Maharashtra, India

ABSTRACT

One of the biggest achievements of IT developers around the world is cloud computing. It enables users to access extensible, distributed, virtualized, hardware or software infrastructure across the network. Cloud computing is fraught with security concerns. This research primarily focuses on building an optimal storage approach by reducing redundant data stored in geographically distant cloud data storage servers. According to the current technique, the system reduces security risks in hybrid clouds by protecting data and restricting data access based on space and time. There is an efficient method of storing secret keys in distributed hash tables and destroying them once they have been used to boost data security as a result. In order to achieve data deduplication, in addition to a distributed hash table, Attribute Based Encryption (ABE) is also used. Simulation is done in CloudSim using seven different parameters, namely accuracy, key computation time, kappa statistics, mean absolute error, root mean square error, encryption and decryption time.

Keywords: Cloud Computing, Security, Decision Tree, Bloom Filters, C4.5, Data Deduplication, Decryption Time, Decryption Time, Attribute Based Encryption.

I. INTRODUCTION

Cloud computing is an emerging service-oriented computing platform. It contains various types of services, like Infrastructure as a service (IAAS), platform as a service (PAAS), Data as a service (DAAS) and software as a service (SAAS). It also contains storage for one type of DAAS. The security of the stored data in the cloud is complex due to its global nature. Data deduplication is a specific data compression technique that removes multiple copies of repeated data from storage. Generally, deduplication is categorized into two different types: file-based deduplication and block-based deduplication. Cloud computing majorly focuses on file-based deduplication. As a result of its security features, deduplication protects your data from both insiders and external threats. In existing deduplication, schemas only verify the physical level match with the appropriate server data. But the cloud may contain the same data on different servers. Key sharing for secure content in the cloud generally creates uncommon keys for unique data. When there are a large number of users or documents available, it can complicate storage [1, 2].

Machine Learning (ML) is the process of training a machine to acquire new skills or adapt its present skill set to new settings. ML requires a lot of computational power to train models at times, and not everyone has access to a lot of storage machines. Cloud computing [3] simply refers to computing resources that are accessible over the internet or other comparable networking The merger of machine learning and cloud computing is known as an 'intelligent cloud.' The intelligent cloud gains the ability to carry large amounts of data saved on the cloud in order to do forecasting and analyze scenarios. Integrating ML abilities into enterprise applications has its own limitations. The special skills are required to create, train, and deploy ML nodes and computational special-purpose networks.

Machine learning is currently the focus of research along with cloud security and data deduplication for the security of data in hybrid clouds. Data deduplication precludes data from insider and outsider attacks for security purposes. The machine learning approach 'Enhanced C4.5' is used to classify users to provide different access and roles. Using the same key for all data will reduce the security of the file. Individual key storage for individual files will create a storage complexity. When keys are used after encryption or decryption, the complexity of storing them is increased to an unacceptably high degree. By providing data protection and controlling data access based on space and time, the existing solution reduces security concerns in hybrid cloud computing.

These five sections make up the body of the paper. Section I provides a clear introduction to the convergence of machine learning and cloud computing. Section II depicts a literature survey. In Section III, the complete procedure for the method used (Bagged Decision Tree) in this work. The fourth section explains the outcomes. Section V brings the paper to a close.

II. LITERATURE SURVEY

This section discusses previous studies related to efficient cryptographic algorithms to increase data security by storing secret keys in distributed hash tables.

Li et al. [4] presented Dekey, an efficient and reliable convergent key management solution for safe deduplication. To ensure the semantic security of convergent keys and the secrecy of outsourced data, Dekey leverages deduplication across convergent keys and distributes convergent key shares among many key servers. Implementation of Dekey with Ramp's secret sharing system. Encoding/decoding overhead is minimal in comparison to ordinary upload/download activities.

Li et al. [5] developed distributed deduplication systems to increase data dependability while maintaining the secrecy of users' outsourced data in the absence of an encryption mechanism. To facilitate file-level and fine-grained block-level data deduplication, four architectures were proposed. The tag's uniformity and integrity were ensured. The Ramp secret sharing mechanism was used to create deduplication systems, which revealed that by comparing ordinary upload/download activities to usual network transmission overhead, it incurs negligible encoding/decoding overhead.

"Approved data deduplication" was developed by Li et al. [6] to preserve data security by factoring differences in user privileges into the duplicate check. New deduplication structures supporting approved duplicate checks in hybrid cloud architecture have been developed; file duplicate-check tokens are issued by the private cloud server with the help of a private key. The methods are safe against both insider and outer attacks defined in the suggested security model, according to security analysis. A prototype was suggested, approved duplicate check

technique as a proof of concept and ran testbed experiments on it. We demonstrated that, when compared to convergent encryption and network transfer, our permitted duplicate check technique had a low overhead.

To increase the efficiency of data management, Miao et al. [7] proposed a payment-based incentive mechanism for deduplication systems that is safe. No malicious cloud service provider, according to the security and incentive study, could fool a client with reduced deduplication pricing. In other words, active clients always pay less per bit than inactive clients. The technique could incentivize clients to engage in deduplication.

On the basis of the threshold blind signature, Miao and coworkers [8] created a new multi-server assisted deduplication solution that can effectively resist collusion assaults between the cloud server and several key servers.

Jiang et al. [9] proposed a probabilistic TIMER technique for efficiently detecting malicious cloud server behavior. For example, it relies on cryptographic assumptions and network latency to avoid server collusion. This creates significant incentives for economically reasonable cloud servers to store customer data in their stores.

A secure outsourced ID3 decision tree technique for two hostile model parties was suggested by Li et al. [10]. Cloud servers' data mining techniques and the privacy of users' data will both be protected by the proposed algorithm. Only the result trees are available to the parties; they are not aware of the data mining scheme. Cloud servers can not access any private information about the parties, either. As a result of the protocol, malevolent cloud servers are protected.

Vardharajan et al. [11] focus on the security services that a cloud provider can provide to its customers (tenants) as part of its infrastructure to prevent these risks. There's also an explanation of a customizable security as-a-service model for cloud providers to offer their tenants and the customers of their customers. During the study, the security architecture was implemented, and the security mechanisms and performance evaluation findings were analyzed in great depth.

According to Cao et al. [12], the rule of C4.5 is improved by using the L'Hospital Rule, which simplifies calculation processes and enhances decision-making algorithms. A similar concept is used to calculate the rate of knowledge gain, which greatly improves the method. And the application at the end of the study demonstrates that the modified algorithm is efficient, making it more suitable for the application of vast amounts of data, and its efficiency has been considerably enhanced in accordance with the real application.

Wai et al. [13] proposed the upgraded C4.5 decision tree and Naive Bayesian (NB) classifiers for web page classification. When the class labels are the same in the original C4.5 classification process, the usual entropy measure is inadequate to measure the suitability of nodes. This system can efficiently enable the classification of web pages into each category by utilizing semantic technologies. This system's effectiveness is demonstrated by employing HTML pages from the computer science domain.

Data deduplication is extensively employed in cloud storage systems to remove redundant storage overhead. Jiang et al. [14] suggested two approaches: static and dynamic schemes, the latter of which allows tree reconfiguration by raising the cost of computation. The central concept is to employ an interactive protocol based on static or dynamic decision trees. The benefit is that, by communicating with clients, the server reduces the time complexity of the deduplication equality test from linear time to efficient logarithmic time across all data items in the database. It turns out that our schemes are Path-PRV-CDA2-secure as well as significantly faster than the R-MLE2 scheme for data equality tests when the number of data items is considerable.

To retain the service, Abujassar et al. [15] introduced the novel Cloud Computing Alarm (CCA) mechanism, which allows the manager node to assign jobs to the best and freest available node. The Cloud Computing Alarm (CCA) approach is used to convey all information about the service node, including which one is ready to accept tasks from users. This strategy, according to simulation results, improves QoS, which will increase the number of people who utilize this service. The CCA improved services without affecting network speed by performing each task in less time, according to the data.

For the quantitative assessment and analysis of secSLA-based security levels supplied by CSPs in relation to Cloud Customer security requirements, Luna et al. [16] proposed two evaluation approaches, namely QPT and QHP. These techniques help to improve the security requirements specifications by allowing customers to define and communicate their individual security demands. QPT and QHP are validated using two scenarios and a prototype, based on real-world CSP secSLA data from the Cloud Security Alliance's Security, Trust, and Assurance Registry, in addition to providing recommendations on their standalone and collective use.

There are some promising results in the implementation of cosine similarity in a flat database by Hernandez et al. [17]. As a result of this, a movie that is highly related to another will be recommended. Once these results have been verified, they might be utilized to construct an automated recommendation system that uses customer ratings to provide recommendations for items and services from a company with such a system.

Fog and cloud computing collaboration were used by Alsaffar et al. [18] to build an architecture for IoT service delegation and resource allocation. There are three conditions (service size, completion time, and VM capacity) that determine the algorithm decision rules for managing and delegating user requests in order to balance the burden. Other algorithms are being considered for resource allocation in fog and cloud computing to achieve service level agreements and quality of service, and for optimizing large data distribution. As shown by the simulation results, the suggested technique can efficiently balance workload, improve resource allocation, optimize massive data distribution, and outperform other existing methods.

On the basis of ownership challenge and proxy re-encryption, Yan et al. [19] offer a strategy to deduplicate encrypted data stored in cloud storage. Encrypted cloud storage is widely used to protect the privacy of data holders. In contrast to this, encrypted data presents new obstacles to cloud data deduplication, which is necessary for huge data storage and processing in the cloud. Access control and data deduplication are integrated. Extensive study and computer simulations are used to determine the performance levels. For example, for huge data deduplication in cloud storage, the results reveal that the scheme has greater efficiency and efficacy.

Wang et al. [20] developed and applied a novel weighted local cosine similarity (WLCS) to visual tracking. To begin, the local cosine similarity assesses the similarities between the target template and the candidates and provides some theoretical insights into them. Second, an objective function is established to simulate the discriminative ability of local components, and the objective function is solved using a quadratic programming method to yield the discriminative weights. Finally, within the particle filter architecture, an effective and efficient tracker based on the WLCS approach is built, as well as a simple updating method. Experiment findings on numerous difficult image sequences show that the proposed tracker outperforms other competing approaches.

Because data sharing is unavoidable with deduplication, control over access permissions in encrypted deduplication storage is more crucial than in standard encrypted storage. As a result, data deduplication should be paired with data access control mechanisms for maximum flexibility. Youn et al. [21] proposed a deduplication approach based on CP-ABE to address this issue. The proposed technique provides client-side

deduplication while also offering confidentiality via client-side encryption to avoid users' sensitive data from being exposed on untrusted cloud servers. It also offers a sufficient trade-off between storage space efficiency and security in a cloud environment, making it ideal for the hybrid cloud model, which takes into account both data security and storage efficiency in a business context.

III. PROPOSED BAGGED DECISION TREE ALGORITHM

The suggested bagged DT algorithm is discussed in this section. To effectively process the concept of deduplication in the proposed system, distributed hash tables were introduced to load and store each distributed server's data and their secret keys to encrypt it. A dictionary-like interface is provided by a distributed hash table, but the nodes are scattered over the network. The node assigned to store a specific key is determined by hashing that key. Hence, the effect on the hash-table buckets is now independent of the nodes in a cloud network. According to Figures 1 and 2, the suggested study work and cloud server have a flow chart, respectively.

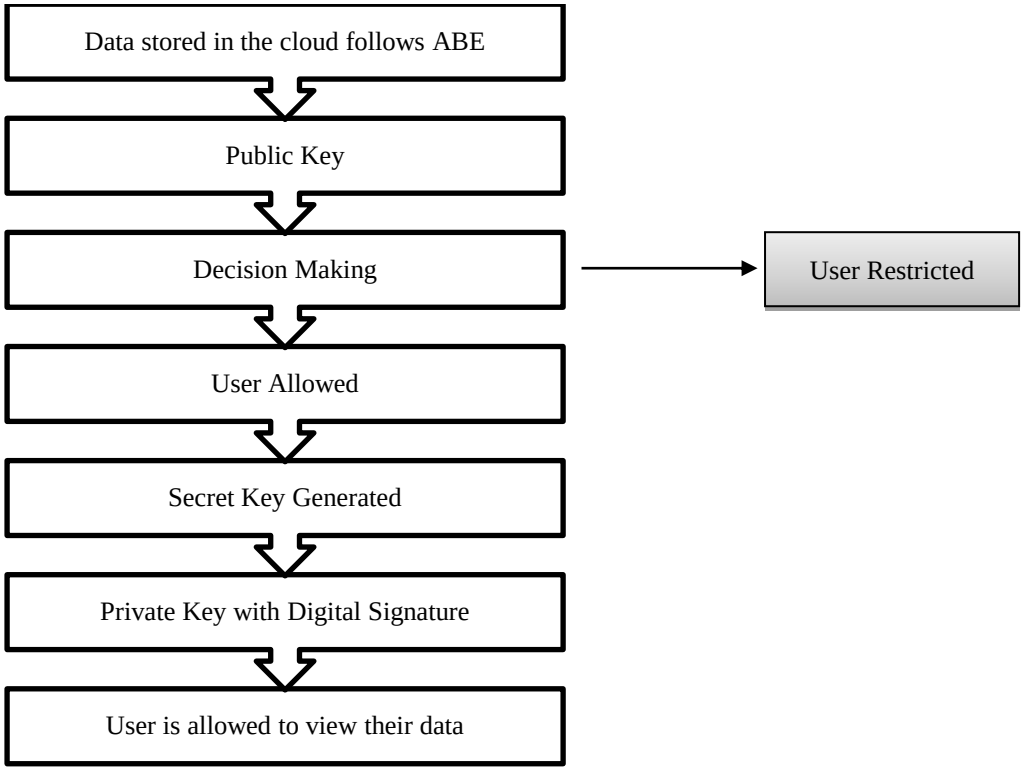

Figure 1. Flowchart of the Proposed Work

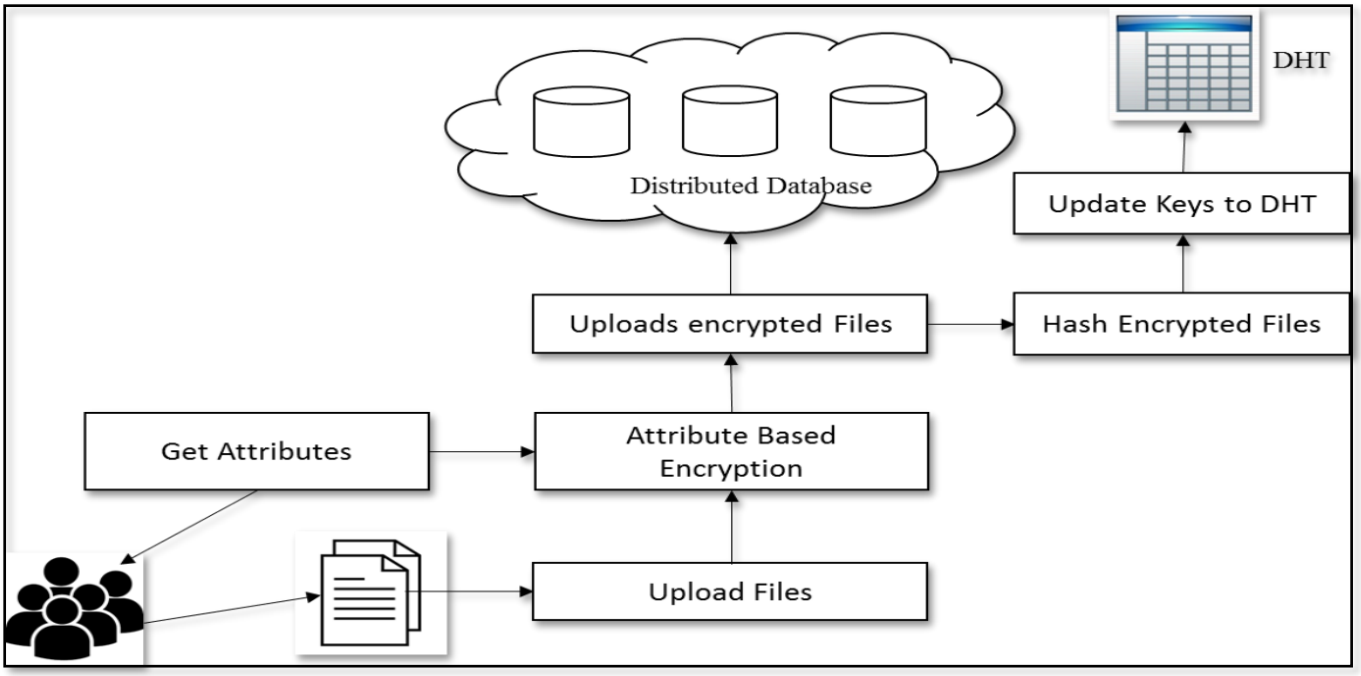

Figure 2. Uploading the File to Cloud Server

A. Data Deduplication

Data deduplication is a computational technique that removes redundant data from a dataset. The method is used to increase storage efficiency and reduce the amount of bytes transported via data networks. It is also referred to as single-instance storage. Extra copies of the same data are eliminated in data deduplication, leaving only one copy to be stored. To confirm that the single instance is truly a single file, data is inspected to find duplicate byte patterns. The duplicates are then replaced with a reference to the stored chunk.

Instead of storing many copies of the data, the de-duplication technique retains only one physical copy. De-duplication can be done at either the file or block level. The file level deduplication technique in this scenario eliminates numerous copies of the file as well as duplicate data blocks that arise in non-identical files. It is also important to note that data deduplication has a huge impact on the protection of sensitive user data, which is secured from both internal and external threats. To achieve this file deduplication, along with the Distributed hash table, Attribute Based Encryption (ABE) is implemented. Figure 3 depicts how data deduplication is divided into five components.

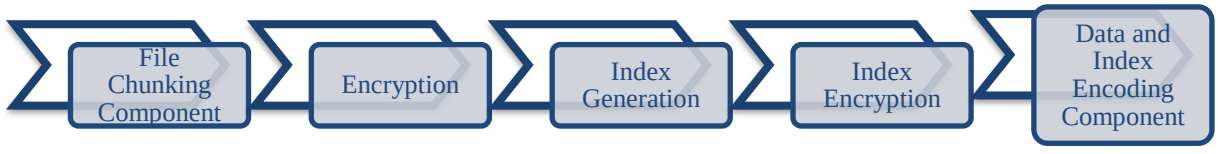

Figure 3. Components of Data Deduplication Framework

B. User Classification

When migrating to the cloud, users are a precious asset and a major issue. The active field of research and development in cloud computing is data privacy and security. User privacy is becoming increasingly important

for many enterprises as they migrate to the cloud. Users might be of numerous sorts, and the level of protection necessary for any data varies as well. In order to classify users based on access and roles, C4.5 machine learning is employed.

The C4.5 algorithm [22] is a user interface module used to identify cloud users prior to handling the access control mechanism for data transfer in cloud networks. It is utilized as a Decision Tree Classifier, which may generate a decision based on a sample of data. A classifier is a part of the code in data mining that takes data for classification and tries to forecast which class the new data belongs to. C4.5 is the successor to the Iterative Dichotomiser (ID3) and eliminates the requirement by dynamically creating a discrete attribute based on numerical factors that separates the continuous attribute value into discrete intervals. C4.5 turns the trained trees (the ID3 algorithm's output) into sets of if-then rules. In order to decide the order in which each rule should be implemented, the precision of each rule is considered next. A rule's precondition can be pruned by removing it if its accuracy improves without it.

C. Bagging

In statistical classification and regression, tagging is a group learning method that increases the stability and accuracy of machine-learning algorithms utilized. It is also known as the bootstrap aggregation. A random data sample is selected in a training set with a replacement to allow individual data points to be selected more than once.

Bagging algorithm creates an ensemble of classifiers or predictors, each of which offers an equally-weighted prediction in the learning scheme. Bagging works as follows when given a set of *p* tuples, P. For iteration i (i = 1, 2,.., n), the original P tuples are replaced with those from the training set, Pi. Bagging is short term for bootstrap aggregation. An example of a bootstrap sample is each of the training data sets. Pi may contain some original P tuples that were not included in P, while others may appear more than once due to sampling with replacement. For every training set, Pi, a classifier model, Ci, is learned. Classifiers return their class predictions to categorize an unclassified tuple (X), which counts as a single vote. When the votes are counted, C allocates X the class that has received the most support. Taking the average of each prediction for a given test tuple can be used to bag continuous variables. The approach is compiled in the bagged classifier with a much higher accuracy than the initial P classifiers. It is not significantly worse and the impacts of noisy data are more robust. The higher precision is due to the reduction in variance of the various classifiers in the composite model. A bagged predictor is always superior to a single P-driven predictor in terms of prediction, according to theorems on the subject.

Input: D, a set of d training tuples; k, the number of models in the ensemble; A learning scheme (e.g., decision tree algorithm, backpropagation, etc.) **Output:** A composite model, M∗.
Method: (1) *for* i = 1 to k do // create k models: (2) create bootstrap sample, Di, by sampling D with replacement; (3) use Di to derive a model, Mi; (4) end *for* **To use the composite model on a tuple, X:**

(1) *if* classification then
(2) let each of the k models classify X and *return* the majority vote;
(3) *if* prediction then
(4) let each of the k models predict a value for X and *return* the average predicted value;

Key sharing for secure content in the cloud generally creates unique keys for unique data. It may create storage complexity at the time of a huge number of users or document availability. To overcome this issue, we created a new shared key model for each unique reference and the shared key can be destroyed. This key will be generated by using the document or file's unique value and server availability value. With the use of an authentication code, it is possible to verify that the sender of an email actually has access to a shared secret key and that no one else could have sent or edited it.

Reducing the storage complexity in terms of destroying the encryption key as well as removing duplicated or replica content by using Blooming Filters, which will efficiently retrieve data stored on a number of geographically separated storage systems. In order to allow membership queries, they are compact data structures for probabilistic representations of a set. When generating a standard Bloom Filter, an array of m bits is used, and it uses k separate hash functions. There are a number of advantages to using Bloom Filters, including cheap storage requirements, rapid membership verification, and no false negatives. False positives are conceivable, but depending on the application needs, their probability can be regulated and greatly reduced.

C. Attribute Based Encryption (ABE)

In order to implement the concept of cryptographic access control, ABE, a cryptographic primitive, was used. Attribute-based encryption often entails neither encrypting the attributes nor encrypting the entire data. Encryption in ABE is simple, secure, and economical compared to other encryption methods presented. Because the encrypted data comprises the attributes in place of the data, the ABE [23] is secure. In the event of a malicious attack, the data is never released. Attribute-based encryption has the drawback of being expensive to decrypt data. The application is protected via attribute-based encryption. When compared to other encryption algorithms, the ABE performs well [24].

Attribute Based Encryption Algorithm

Input: Input Data (M), User Attribute (I_n)
Output: Encrypted Data (Enc)

Method:
n represents number of attributes for particular user
Key Generation:
Step 1: *For* i=1 to n
Step 2: *For* j= 1 to I_1.Size
Step 3: L_x=L_x+I_1(j)
Step 4: End *For*
Step 5: End *For*
Step 6: *For* x=1 to L_x.Size
Step 7: k(x)= L (x)
Step 8: k(x)=k(x)+(L_x(x)&oxff)+0x100
Step 9: End *For*

Encryption Algorithm

Step 1 : Let M be the data to be encrypted
Step 2 : Let Enc be the encrypted Data
Step 3 : Private key P_k ← generate key
Step 4 : Public Key B_K ← generate random prime number
Step 5 : Master Key m_{k1},m_{k2} ← generate random even number less than public key
Step 6 : Secret Key S_K ← power(P_k, m_{k2})
Step 7 : while D != null do
Step 8 : con ← D
Step 9 : for k=1 to con.size()
Step 10 : CT_1=con(char(k))+ S_K
Step 11 : CT_2= m_{k1}*B_K
Step 12 : $Cipher_{CT}$=CT_1*CT_2
Step 13 : End For
Step 14 : Enc ←Enc.append($cipher_{CT}$)
Step 15 : End While

IV. SIMULATION RESULTS

In this section, the results of the experiments are presented graphically, along with a discussion of the findings. The section also discusses the simulation tool and the performance parameters applied. Analysis of the performance of several measures was used to put into practice the proposed framework. The CloudSim simulator with NetBeans was used in this study. As a result of this, CloudSim is able to model and simulate cloud computing data centers and virtualized hosts, as well as allocation policies and network topologies. The NetBeans IDE is an IDE which facilitates the development of desktop, mobile, and online apps. It is free and open source. The IDE runs on Windows, Linux, Mac OS X, and other UNIX systems.

A. Accuracy

Accuracy refers to how close a measured value is to a standard or known value. It is the degree to which a measured or calculated quantity conformity to an actual or true value.

Where TP = True Positive
TN = True Negative
FP = False Positive
FN = False Negative

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$

Table 1. Accuracy Value Table

Existing (C4.5)	Proposed (Bagged DT)
96.5517	97.7273

Table 1 shows that the suggested Bagged Decision Tree Algorithm outperforms the existing C4.5 algorithm. Figure 3 demonstrated the accuracy of the existing and proposed bagged DT algorithm. The existing technique

uses C4.5 only, whereas the proposed scheme uses the concept of bagging. This reduces variance and, as a result, overfitting is reduced. That's why there is a slight increase in accuracy in the proposed scheme than in the existing scheme.

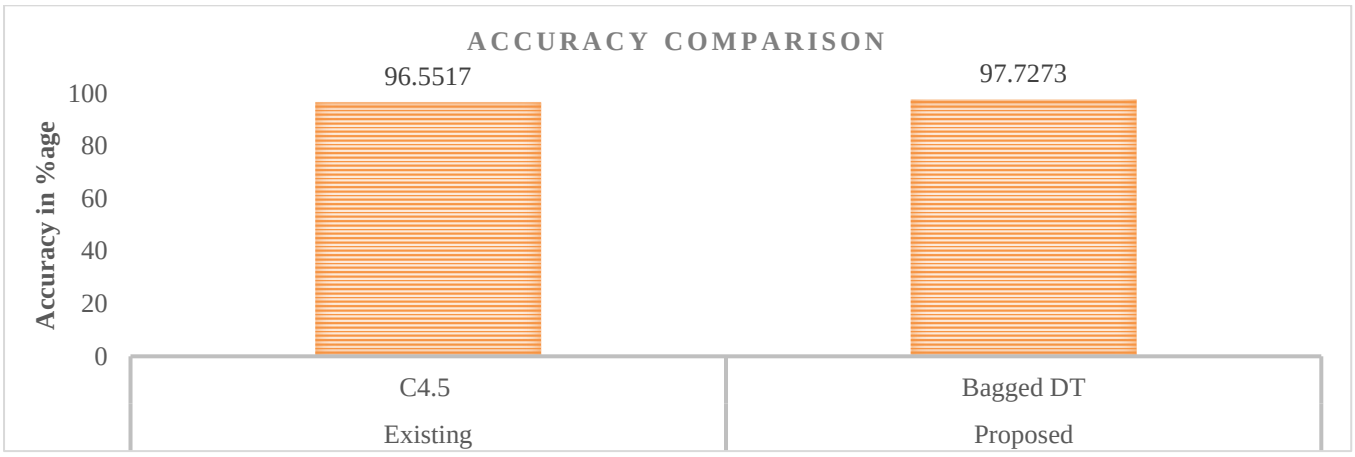

Figure 3. Accuracy

B. Kappa Statistics

Jacob Cohen created the term kappa statistics.Kappa is an inter-rater reliability statistical measure for category variables. When two raters use the same criterion based on a tool to determine whether or not a condition occurs, Kappa is used.

$$K = \frac{p0-pe}{1-pe}$$

where p_o = relative observed agreement among raters

p_e = hypothetical probability of chance agreement.

Table 2. Kappa Statistics Value Table

Existing (C4.5)	Proposed (Bagged DT)
0.9308	0.9545

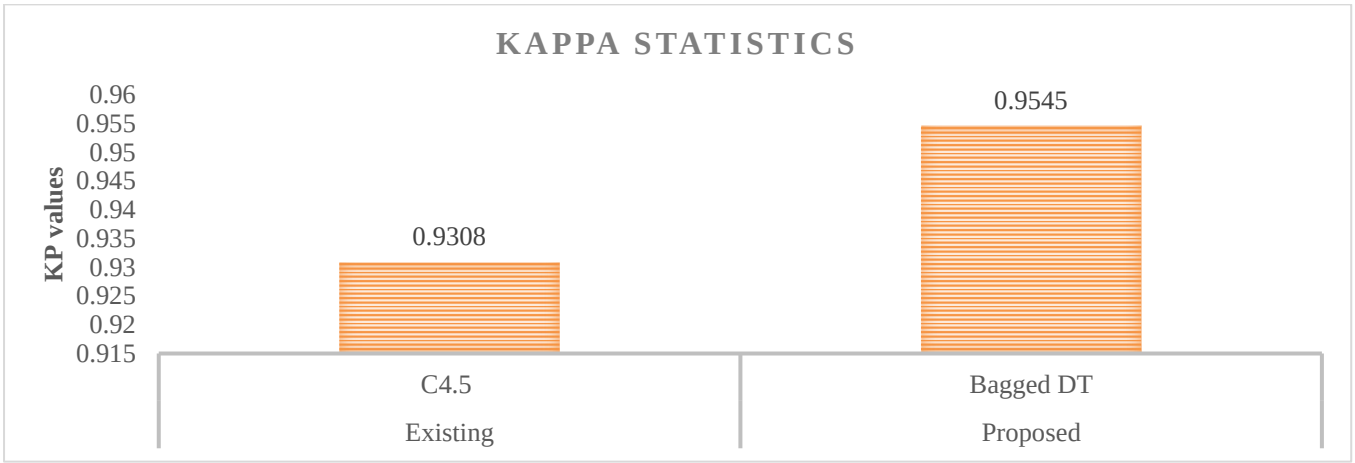

Figure 4. Kappa Statistics

Table 2 represents the kappa statistics values of existing and proposed algorithms. The figure 4 shows the comparison between the values generated after applying kappa statistics. It is clearly shown in figure 4 that the bagged DT algorithm outperforms existing C4.5 algorithm because

C. MAE and RMSE

In order to measure the accuracy of continuous variables, two often used metrics are the mean absolute error (MAE) and the root mean squared error (RMSE). With equal weigh for each difference, MAE calculates the average absolute difference between forecast and actual observation across the test sample.

MAE = $1/n \sum_{i=1}^{n} |xi - x'i|$

RMSE is a quadratic scoring rule that additionally gauges the error's average magnitude. In other words, it is equal to the root of the average squared difference between forecast and observation. The values for MAE and RMSE are shown in table 3 using the following formulas.

$$RMSE= \sqrt{\frac{1}{n}\sum_{i=1}^{n}(xi - x'i)^2}$$

Table 3. Error Rate Value Table

	Existing (C4.5)	Proposed (Bagged DT)
MAE	0.0568	0.0332
RMSE	0.1724	0.1334

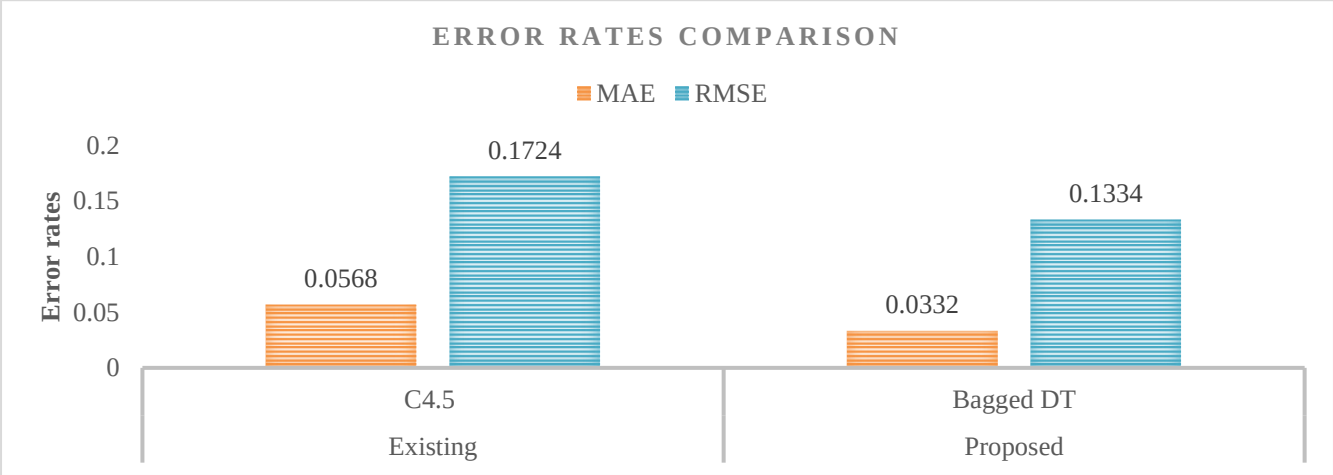

Figure 5. Error Rate

Its performance is displayed in Figure 5 when compared with that of the existing C4.5 method. Both MAE and RMSE are measures of the average model prediction error expressed in units of the variable of interest. In this case, smaller numbers are preferred.

D. Key Computation Time

Key computation time is the amount of time required to halt a computational operation. It is sometimes referred to as running time. Table 4 displays the key time of calculation of the planned and current file size scheme in KB.

Table 4. Values of Key Computation Time

File Size in KB	Existing (C4.5)	Proposed (Bagged DT)
10	0.12	0.08
20	0.16	0.10
30	0.19	0.14
40	0.21	0.17

50	0.25	0.20
60	0.30	0.24
70	0.35	0.31

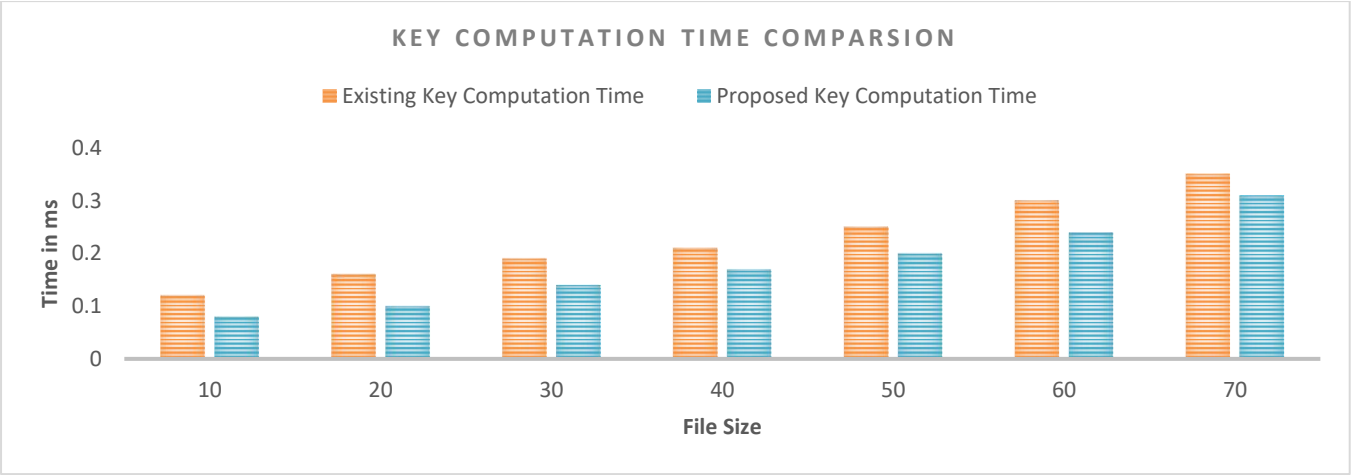

Figure 6. Key Computation Time

Figure 6 represents a key computation time curve, whereas table 4 displays important computation time values for the proposed and current C4.5 algorithms. As the file size grows, so does the time required to compute the key. When compared to the present technique, the suggested technique has a shorter key computation time. For example, when the file size is increased to 70MB, computation time improves by roughly 15%.

E. Encryption Time

The time it takes to encode a cipher text from a plaintext is considered as time an encryption algorithm takes. Table 5 indicates the time of encryption for both the current and the proposed schemes in accordance with the file size in KB.

Table 5. Encryption Time

File Size in KB	Existing (C4.5)	Proposed (Bagged DT)
10	14.6	9.32
20	23.5	14.4
30	30.1	21.3
40	38.6	27.6
50	45.6	39.14
60	78.9	62.96
70	85.7	75.45

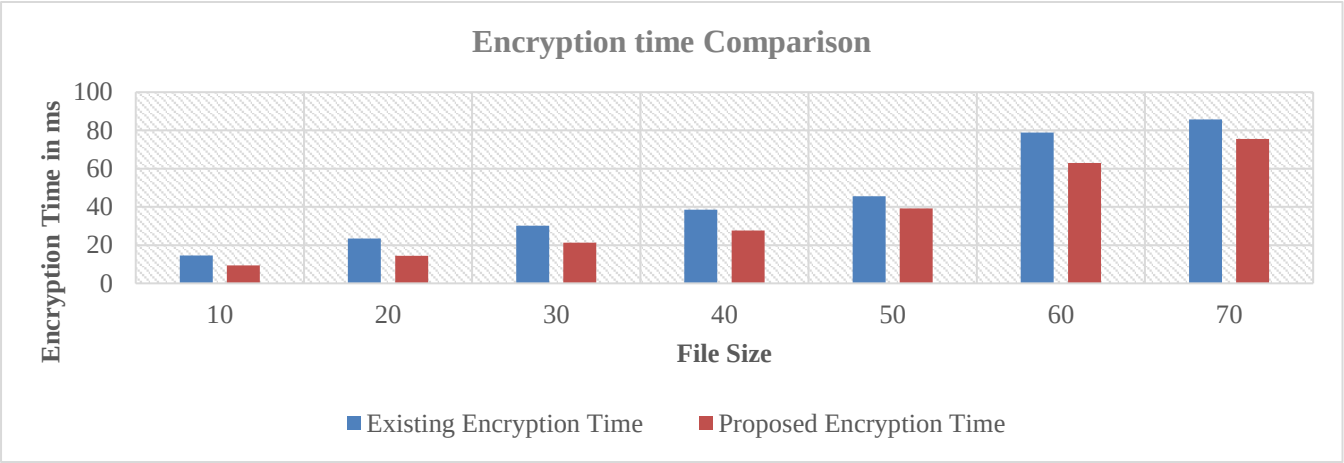

Figure 7. Encryption Time

The graph above depicts the time required to encrypt data based on different file sizes. The time it takes to encrypt a file grows in proportion to its size. When compared to the existing ABE method, the suggested ABE with hash function encryption time is shorter, indicating that the proposed technique performs better in terms of encryption parameters. The graph illustrates that when the file size is increased to 70 MB, the encryption time improves by around 10%.

F. Decryption Time

This is the time taken by a decryption algorithm to work in order to generate plain text from ciphertext. Table 6 presents the decryption time of both the proposed and existing schemes corresponding to the file size in KB.

Table 6. Decryption Time

File Size in KB	Existing (C4.5)	Proposed (Bagged DT)
10	9.26	7.1
20	16.4	12.2
30	22.9	17.3
40	28.1	24.6
50	35.3	30.8
60	49.8	44.9
70	60.4	52.9

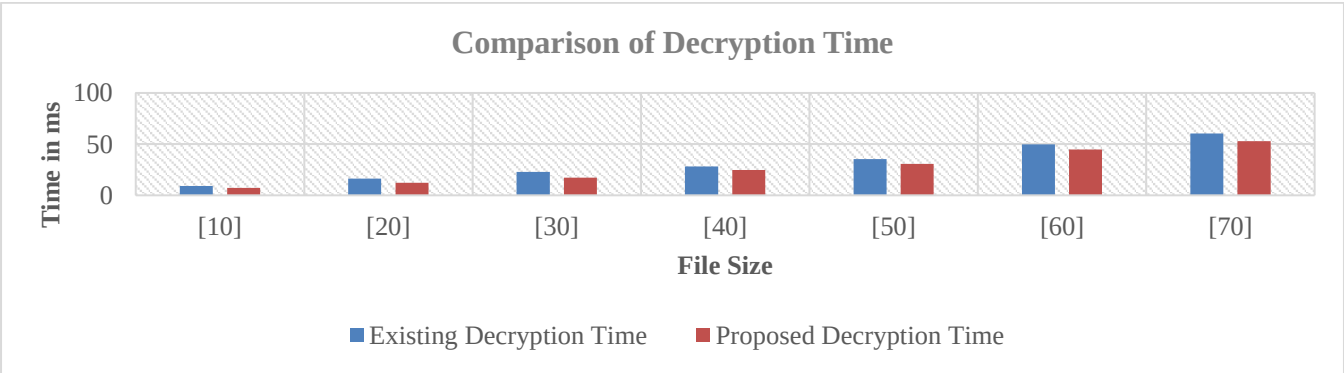

Figure 8. Decryption Time

According to different file sizes, Figure 8 shows how long it takes to decrypt the data comparisons. The time of decryption is growing as the size of the file grows. Compared to the current ABE algorithm, the proposed ABE

using hash-function has shown that the proposed technique is more efficient in terms of decryption parameters. The chart reveals that the decryption time indicated, when the file size grew to 70MB, is around 10%.

V. SUMMARY

In this study, a novel machine-learning application has been built to give security to hybrid cloud networks when storing data and retrieving or accessing data from cloud databases. In order to reduce redundant data storage and retrieval, the Bagged DT deduplication technique has been devised. In addition, the current C4.5 algorithm is used to classify cloud users based on their authenticity. It was necessary to further develop and test the proposed machine learning application in order to establish the level of security of hybrid cloud networks when storing, retrieving, and accessing data from a cloud database. This suggested technique has the greatest potential to mitigate security risks in hybrid clouds. The ABE encryption algorithm is used in conjunction with the bagged decision tree technique. Simulation results of the Bagged DT algorithm are discussed in comparison with the existing C4.5 decision tree algorithm. In addition, the recommended technique has demonstrated less key computation time, encryption and decryption time, a high accuracy rate, and low RMSE and MAE values. More research in this area can be done by employing a strong cryptographic algorithm for secure storage.

VI. REFERENCES

[1] . Bokhari, M. U., Makki, Q., & Tamandani, Y. K., "A Survey on Cloud Computing," Big Data Analytics, pp. 149–164, 2017.

[2] . Shen W., Qin J., & Ma J., "A Lightweight Identity-Based Cloud Storage Auditing Supporting Proxy Update and Workload-Based Payment," Security and Communication Networks, vol. 2019, pp. 1-15, 2019.

[3] . Gondhi, N. K., & Gupta, A., "Survey on Machine Learning based scheduling in Cloud Computing," Proceedings of the 2017 International Conference on Intelligent Systems, Metaheuristics & Swarm Intelligence - ISMSI '17, 2017.

[4] . Li, J., Chen, X., Li, M., Li, J., Lee, P. P. C., & Lou, W., "Secure Deduplication with Efficient and Reliable Convergent Key Management," IEEE Transactions on Parallel and Distributed Systems, vol. 25, no. 6, pp. 1615–1625, 2014.

[5] . Li, J., Chen, X., Huang, X., Tang, S., Xiang, Y., Hassan, M. M., & Alelaiwi, A., "Secure Distributed Deduplication Systems with Improved Reliability," IEEE Transactions on Computers, vol. 64, no. 12, pp. 3569–3579, 2015.

[6] . Li, J., Li, Y. K., Chen, X., Lee, P. P. C., & Lou, W., "A Hybrid Cloud Approach for Secure Authorized Deduplication. IEEE Transactions on Parallel and Distributed Systems, vol. 26, no. 5, pp. 1206–1216, 2015.

[7] . Miao, M., Jiang, T., & You, I., "Payment-based incentive mechanism for secure cloud deduplication," International Journal of Information Management, vol. 35, no. 3, pp. 379–386, 2015.

[8] . Miao, M., Wang, J., Li, H., & Chen, X., "Secure multi-server-aided data deduplication in cloud computing," Pervasive and Mobile Computing, vol. 24, pp. 129–137, 2015.

[9] . Jiang, T., Chen, X., Li, J., Wong, D. S., Ma, J., & Liu, J., "TIMER: Secure and Reliable Cloud Storage against Data Re-outsourcing," Lecture Notes in Computer Science, pp. 346–358, 2014.

[10] . Li, Y., Jiang, Z. L., Wang, X., Fang, J., Zhang, E., & Wang, X., "Securely Outsourcing ID3 Decision Tree in Cloud Computing," Wireless Communications and Mobile Computing, vol. 2018, pp. 1–10, 2018.

[11] . Varadharajan, V., & Tupakula, U., "Security as a Service Model for Cloud Environment," IEEE Transactions on Network and Service Management, vol. 11, no. 1, 60–75, Mar. 2014.

[12] . Cao, R., & Xu, L., "Improved C4.5 Algorithm for the Analysis of Sales, "2009 Sixth Web Information Systems and Applications Conference, Sep. 2009.

[13] . Wai, M., H. P., Tar, P. P., & Thwe, P., "Ontology Based Web Page Classification System by Using Enhanced C4.5 and Naïve Bayesian Classifiers," 2018 International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS), 2018.

[14] . Jiang, T., Chen, X., Wu, Q., Ma, J., Susilo, W., & Lou, W., "Secure and Efficient Cloud Data Deduplication with Randomized Tag," IEEE Transactions on Information Forensics and Security, vol. 12, no. 3, pp. 532–543, Mar. 2017.

[15] . Abujassar R. S., & Jazzar M., "Enhancing the Cloud Computing Performance by Labeling the Free Node Services as Ready-To-Execute Tasks" Journal of Electrical and computer Engineering, vol. 2017, pp. 1-9, March 2017.

[16] . Luna, J., Taha, A., Trapero, R., & Suri, N., "Quantitative Reasoning about Cloud Security Using Service Level Agreements," IEEE Transactions on Cloud Computing, vol. 5, no. 3, pp. 457–471, Sept. 2017.

[17] . Hernandez R., A. F., & Garcia G., N. Y., "Distributed processing using cosine similarity for mapping Big Data in Hadoop," IEEE Latin America Transactions, vol. 14, no. 6, pp. 2857–2861, Jun. 2016.

[18] . Alsaffar A. A., Pham H. P., Hong C. S., Huh E. N., & Aazam M., "An Architecture of IoT Service Delegation and Resource Allocation Based on Collaboration between Fog and Cloud Computing," Mobile Information Systems, vol. 2016, pp. 1-15, Aug. 2016.

[19] . Yan, Z., Ding, W., Yu, X., Zhu, H., & Deng, R. H., "Deduplication on Encrypted Big Data in Cloud," IEEE Transactions on Big Data, vol. 2, no. 2, pp. 138–150, June 2016.

[20] . Wang, D., Huchuan Lu, & Chunjuan Bo., "Visual Tracking via Weighted Local Cosine Similarity," IEEE Transactions on Cybernetics, vol. 45, no. 9, pp. 1838–1850, Sep. 2015.

[21] . Youn T. Y., Jho N. S., Rhee K. H., & Shin S. U., "Authorized Client-Side Deduplication Using CP-ABE in Cloud Storage," Wireless Communications and Mobile Computing, vol. 2019, pp. 1-11, May 2019.

[22] . Sindhu, S., Geetha, S., Subashini, S., Priya, R., & Kannan, A., "An Active Rule approach for Network Intrusion Detection with NeuroC4.5 Algorithm," 2006 Annual IEEE India Conference, 2006.

[23] . Matthew Green, Susan Hohenberger, and Brent Waters, "Outsourcing the decryption of ABE ciphertexts," In Proceedings of the 20th USENIX conference on Security (SEC'11), USENIX Association, Berkeley, CA, USA, pp. 34-34, 2011.

[24] . Kumar, N. S., Lakshmi, G. V. R., & Balamurugan, B., "Enhanced Attribute Based Encryption for Cloud Computing," Procedia Computer Science, vol. 46, pp. 689–696, 2015.