Webology (ISSN: 1735-188X)
Volume 18, Number 5, 2021

Fast Syndrome-Cryptographic Hash Storage Based Tanimoto Index Margin Relaxing Support Vector Regressive Data Auditing With Iot

[1]S.Sivakamasundari , [2]Dr.K.Dharmarajan

[1]Research Scholar, School of Computing Sciences, VISTAS, Chennai, India

[1]Assistant Professor New Prince Shri Bhavani Arts and Science College

[2]Associate Professor, Dept. of Information Technology, VISTAS

Abstract

Data auditing in the cloud server has more significance than any other data protection mechanism to ensure the integrity of the user data. When users store their data, integrity is a major concern for data owners due to the lack of direct control. However, the existing remote data auditing schemes for big data platforms are difficult to provide a higher integrity rate. In order to improve the integrity verification of data on cloud storage, A Fast Syndrome-Cryptographic Hash Storage based Tanimoto index Margin Relaxing Support Vector Regressive Data Auditing (FSCHS-TIMRSVRDA) technique is introduced. The FSCHS-TIMRSVRDA technique has three steps, namely the registration phase, generation phase, and verification phase. In the registration phase, the user registers his detail to a cloud server for storing their data collected from IoT. IoT collects the information from an entity and sends the information to the cloud server. In the generation phase, the FSCHS-TIMRSVRDA technique uses the Fast Syndrome-Cryptographic Hash function to generate the hash value for the cloud user data and send it to the cloud server for dynamic storage. Whenever the cloud user needs to audit the data, the audit request is sent to the third-party auditor (TPA). The TPA receives the audit request and the data from the cloud server for data auditing. TPA generates the hash value for user data. Finally, TPA verifies the generated hash value of data from CS with the hash value of data stored in TPA by using Tanimoto index Margin Infused Relaxing Support Vector Regression. In this way, data auditing is performed in a cloud environment. Experimental evaluation is carried out on factors such as space complexity, data integrity, and data auditing time with respect to a number of cloud user data. Results show that the proposed FSCHS-TIMRSVRDA technique can efficiently enhance the data integrity rate and reduce the space complexity as well as data auditing time.

Keywords—component, formatting, style, styling, insert (key words)

1. Introduction

Due to the rapid development of communications and networks, cloud storage is a method to store the data online, which permits the data owner (DO) to efficiently access data at any time and any place. With the extensive application of cloud storage, ensuring the integrity of user outsourced data receives more and more consideration. Currently, data auditing is an important method to verify the integrity of the data stored on the cloud server. Existing data auditing schemes have done many techniques in terms of functionality, implementation, and security.

The privacy-preserving cloud storage auditing (PP-CSA) method was introduced in [1] for data sharing, where only legal users access the data using Diffie–Hellman protocol. However, a higher integrity rate was not achieved with minimum time. In order to verify the integrity of data, a dynamic auditing scheme was developed in [2] for big data storage. But it failed to use more effective data auditing schemes for multiple users.

Identity-based remote data integrity checking scheme was introduced in [3] to decrease the system complexity. However, the performance of time consumption for integrity verification was not focused on. An effective sampling verification algorithm was developed in [4] for dynamic auditing the data and helping the users to update data efficiently. But the complexity analysis was not focused on the verification process.

A Secure Auditable Cloud Storage (SecACS) system was introduced in [5] for considering the data dynamics with minimum computation time using a lightweight cryptographic process. But the higher integrity rate was not achieved when considering more cloud user data. An efficient public auditing scheme was introduced in [6] for cloud data to guarantee public auditing and preserve data privacy. However, the designed scheme failed to support the auditing for a multi-cloud environment.

An efficient approach was designed in [7] for data integrity auditing to decrease the complexity of the auditing protocol. But the approach reduces the storage cost but the higher integrity rate was not achieved. An online/offline remote data auditing (OORDA) framework was introduced in [8] for the auditing process to guarantee the integrity verification of cloud data. But the framework was not efficient to allow dynamic data auditing for cloud storage.

A user behavior prediction-based data integrity auditing system was designed in [9] for secure cloud storage and minimizing the auditing overhead. Data Integrity Auditing based on minimizing any Trust on Third Parties (DIA-MTTP) was developed in [10] to minimize the storage overhead. But it failed to consider the multiple users for accessing and updating the data on the cloud.

1.1 Main contribution

In order to address the above issues, an efficient technique called FSCHS-TIMRSVRDA is introduced. Our main contributions are as follows:

(1) An efficient data auditing technique called FSCHS-TIMRSVRDA is introduced based on three different processes namely registration phase, generation phase, and verification phase.
(2) To improve the dynamic data storage and minimize the space complexity, the Fast Syndrome-Cryptographic Hash function is introduced in the FSCHS-TIMRSVRDA technique to generate the hash value of user data. Then the data is stored in the cloud server with minimum storage space.

(3) To increase data integrity on cloud storage, FSCHS-TIMRSVRDA uses the Tanimoto index Margin Infused Relaxing Support Vector Regression. Whenever the user wants to audit the data, the audit request is sent to the third-party auditor (TPA). The TPA performs hash verification to validate the integrity based on the Tanimoto index. The Margin Infused Relaxing Support Vector Regression uses the hyperplane and marginal hyperplanes for verifying the hash value based on the Tanimoto index similarity measure. If the two hash value gets matched, achieves higher data integrity.

(4) To minimize the data auditing time, the TPA performs the auditing for only the registered cloud user. Then it uses the Tanimoto index similarity for hash verification. This helps to minimize time consumption.

(5) Finally, extensive experiments are conducted to evaluate the performance of our FSCHS-TIMRSVRDA technique along with the conventional auditing scheme based on various performance metrics.

1.2 Organization of paper

The remainder of this research is organized as follows. A brief review of related works concerning secure data auditing is presented in Section 2. Our network system model and the proposed novel FSCHS-TIMRSVRDA are described in Section 3. In Section 4, the experimental setting is provided, and in Section 5, the discussion of different parameters is described in detail. Finally, the paper is concluded in Section 6.

2. Related works

A public auditing-based framework was developed in [11] for secure data auditing on cloud storage. However, the designed framework failed to improve the security and efficiency of the integrity verification system. An integrity verification approach was introduced in [12] for securing cloud storage based on Ternary Hash Tree (THT) and Replica-based Ternary Hash Tree (R-THT) to perform data auditing. But the approach failed to verify the data integrity over the shared data across the user group.

A certificate less multi-replica and multi-cloud data public audit system were developed in [13] based on blockchain technology. But the designed system failed to achieve higher efficiency of computation cost but the complexity was not minimized. A novel identity-based data auditing (IBDA) system was developed in [14] for cloud storage. However, the designed system failed to improve the data integrity.

A decentralized arbitrable remote data auditing method was introduced in [15] for network storage service based on blockchain techniques. A compressive secure cloud storage protocol was designed in [16] for verifying the integrity and minimizing the storage costs. However, it failed to improve the effectiveness of the proposed protocol. A cloud audit method was introduced in [17] for multi-copy dynamic data integrity based on a red-black tree. But the performance of complexity was not minimized.

A stable and efficient audit method was designed in [18] based on the trust architecture that guarantees the safety and confidentiality of records. A data integrity auditing without private key storage method was introduced in [19] for secure cloud storage. But the performance of auditing time

was not minimized. A dynamic outsourced auditing system was developed in [20] for dynamic updates of outsourced data.

3. Methodology

With the extensive application of cloud storage, guarantying the integrity of user data reached more and more attention. As the user data has been frequently updated by the owners. So the audit schemes have the ability to maintain the static collection of data and to validate the integrity of the cloud with the help of third-party auditors (TPA). But there are still a series of problems in the existing audit schemes for dynamic data storage. Therefore, a novel technique FSCHS-TIMRSVRDA is introduced in this paper for dynamic data storage and integrity verification.

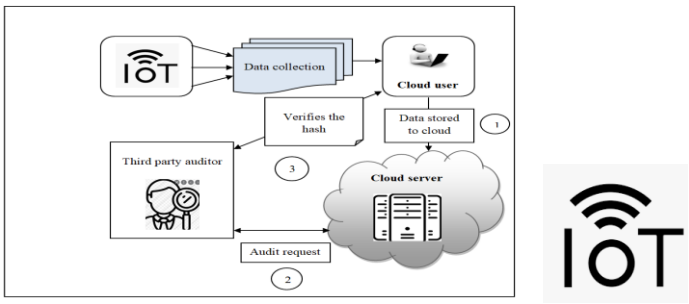

Figure 1 architecture diagram of the

FSCHS-TIMRSVRDA technique

Figure 1 demonstrates the architecture diagram of the proposed FSCHS-TIMRSVRDA technique for cloud data storage. The system model of the proposed FSCHS-TIMRSVRDA technique comprises the three types of entities such as cloud users $cu_1, cu_2, cu_3, \ldots . cu_n$ who comprises the data $d_1, d_2, \ldots d_n$ to be stored in the cloud server (CS) in the form of a hash function using Fast syndrome-based cryptographic hash to offer a data storage services, Third-party auditor (TPA) who has knowledge and abilities that the trusted users to access the cloud data. Based on the above-said system model, the proposed FSCHS-TIMRSVRDA technique is designed to provide higher integrity on cloud data storage.

3.1 Registration phase

In the FSCHS-TIMRSVRDA technique, initially, the cloud user enters and submits their details to the cloud server (CS) for data storage. The cloud server provides the storage services only to the registered users. Therefore, the cloud user needs to submit their details to the cloud server before accessing the services. The users enter their details like name, date of birth, mail ID, and so on and it is stored on the cloud server.

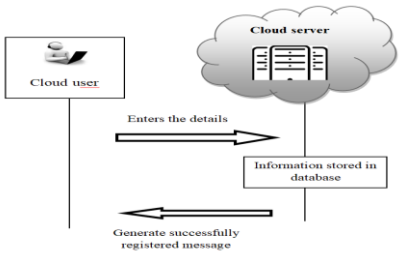

Figure 2 registration phase

Figure 2 illustrates the registration phase to store the user details into the cloud user for accessing the different services.

3.2 Fast syndrome-cryptographic hash-based data storage

The second process of the FSCHS-TIMRSVRDA technique is to perform the data storage on a cloud server. The server provides the storage services to the registered cloud users. IoT collects the information i.e. data from an entity and sends the information to the cloud server. Let us consider the number of data $d_1, d_2, \ldots d_n$ are generated from the IoT. Then the input data are converted into a sequence of bits [0,1]

$$D_1 \rightarrow b_1 b_2 b_3 .. b_k \quad (1)$$

From (1), $b_1 b_2 b_3 .. b_k$ denotes a sequence of bits. Followed by, convert the sequence of bits into an integer. Then construct the matrix based on the integer value. For example, if the integer value is 3, then we construct 3 matrices with four columns and four rows.

$$M_1 = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ m_{41} & m_{42} & m_{43} & m_{44} \end{bmatrix}$$

$$M_1 = \begin{bmatrix} c_{11} & c_{12} & c_{13} & c_{14} \\ c_{21} & c_{22} & c_{23} & c_{24} \\ c_{31} & c_{32} & c_{33} & c_{34} \\ c_{41} & c_{42} & c_{43} & c_{44} \end{bmatrix}, M_3 = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \\ p_{41} & p_{42} & p_{43} & p_{44} \end{bmatrix} \quad (2)$$

In order to obtain the final hash, from the first sub-matrix $M_1 M_1$, we pick the column based on the next number of the first integer value. We select the column from the second sub-matrix M_2 based on next number of the second integer value. From the M_3 , we select the column based on next number of the third integer value. After picking the column value, perform an XOR operation to obtain the final hash.

$$H = [m_{31} m_{32} m_{33} m_{34}] \oplus [c_{11} c_{12} c_{13} c_{14}] \oplus [p_{41} p_{42} p_{43} p_{44}] \quad (3)$$

Where H denotes a final output hash value. The XOR operation is performed based on input bits (m, c, p) and generates the output bit. If the bits are the same, the result is 0. If the bits are different, the result is 1. In this way, hash values of cloud user data are obtained. Then the hashed data are stored into the cloud server with minimum space complexity.

// Algorithm 1: Fast syndrome-cryptographic hash-based data storage
Input: Number of data $d_1, d_2, .., d_n$, cloud users $cu_1, cu_2, cu_3, \ldots . cu_n$ **Output**: Minimize space complexity
Begin **1:** Collect the number of data$d_1, d_2, .., d_n$ **2**: **For each data** d_i **3**: Convert data into a sequence of bits [0,1] **4:** Convert the sequence of bits into an integer **5:** Construct the matrix based on the integer value **6:** Pick the column value

7: Perform XOR operation
8: Obtain the final hash **'H'**
9: End for
End

Algorithm 1 given above illustrates the step-by-step process of data storage on a cloud server with lesser space complexity. First, the input data are converted into a sequence of bits string and obtain the integer value. Followed by, the matrix which is constructed based on the integer value. After that, pick the column value and perform the XOR operation. Finally, the hash value is generated and stored in the cloud server. As a result, the space complexity of the server gets minimized.

3.3 Data auditing and Margin Infused Relaxing Support Vector Regression based verification

Whenever the cloud user needs to audit the data, the request is sent to the third-party auditor (TPA). The TPA performs the data auditing only for the registered user. When the TPA receives the request from a cloud user, TPA sends the request for stored data to the Cloud Server (CS). CS sends the requested data to the TPA for data auditing. TPA generates the hash value ' H_2' using a fast syndrome-based cryptographic hash function for the received data. After that, TPA verifies the generated hash value of requested data from CS with the hash value of data stored in TPA by using Margin Infused Relaxing Support Vector Regression in the Verification phase shown in figure 3.

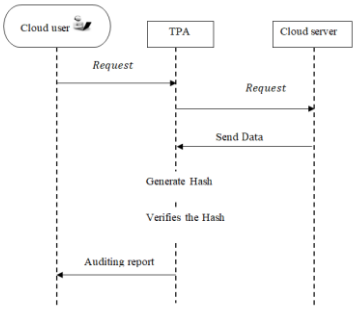

Figure 3 flow processes of Data auditing and verification

The TPA verifies the generated hash value of received data from CS with the hash value of data stored in TPA by using Margin Infused Relaxing Support Vector Regression. In this verification phase, Margin Infused Relaxing Support Vector Regression is a machine learning technique that helps to analyze the two hash functions based on the hyperplane. Followed by, the two marginal hyperplanes are formed on both sides of the hyperplane. Here, the hyperplane act as a boundary between the two classes. A separating hyperplane uses the Tanimoto index to find the similarity between two hash functions.

$$\delta = \frac{\sum H_1 H_2}{\sqrt{\sum {H_1}^2} + \sqrt{\sum {H_2}^2} - \sum H_1 H_2} \quad (4)$$

Where 'δ' represents the Tanimoto similarity coefficient, H_1 denotes a hash value of requested data from CS, H_2 denotes the hash value of data stored in TPA, $\sum {H_1}^2$ denotes a sum of the squared score of the H_1, $\sum {H_2}^2$ indicates a sum of the squared score of the H_2. $\sum H_1 H_2$ denotes a sum of the product of the paired score of H_1 and H_2. Therefore, the Tanimoto similarity coefficient provides the

output ranges from 0 to +1 where +1 indicates the higher similarity, and 0 represents the low similarity. Based on the similarity value, the verification process is performed. In the Verification phase, the output result is '1', the hash value is matched and data stored in CS is not altered by intruders. When the output result is '-1', the hash value is not matched and data stored in CS is altered by intruders. If the result falls on the marginal hyperplane instead of falling on either side of the margin, then the Margin Infused relax technique is applied to maximize the margin hyperplane between the two classes.

$$Y = \arg\max[mk_1, mk_2] \quad (5)$$

From (5), Y denotes a Margin Infused Relaxing output, $\arg max$ denotes an argument of the maximum function, mk_1 and mk_2 denotes a two marginal hyperplane.

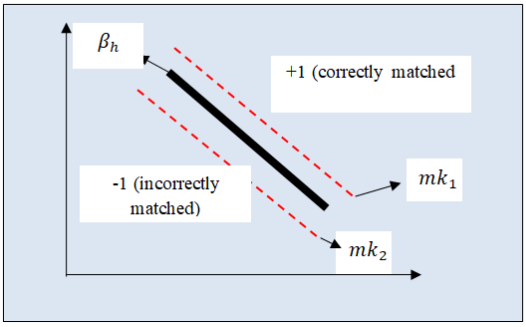

Figure 4 Tanimoto indexed Margin Infused Relaxing Support Vector Regressive Data Auditing

Figure 4 reveals a Margin Infused Relaxing support vector regression for hash verification using Tanimoto similarity coefficient based on either side of the hyperplane (β_h). As shown in figure 4, mk_1 and mk_2 denotes two marginal hyperplanes. In this way, the hash verification is performed by the TPA to increase the data integrity and minimize the auditing time. The algorithmic process of the Tanimoto index Margin Infused Relaxing Support Vector Regressive Data Auditing is described as given below,

Algorithm 2: Tanimoto index Margin Infused Relaxing Support Vector Regressive Data Auditing

Input: Generated hash values
Output: Improve data integrity rate

Begin

1. **If the User** audit the data from the server
2. Send a request to TPA
3. TPA sends a request to a cloud server
4. Cloud server sends the requested data into TPA in the form of hash ('H_1)
5. TPA generates the hash for that data ('H_2)
6. **End if**
7. TPA performs verification
8. Construct hyperplane 'β_h'
9. Find two marginal hyperplane mk_1,mk_2
10. Measure the similarity between two hash functions 'δ'

11. if ($\delta = 1$)then
12. Data was not altered by intruders
13. else
14. Data was altered by intruders
15. end if
End

Algorithm 2 describes the data auditing with higher integrity. The user sends the request to TPA for verifying the integrity of stored data in the cloud server. The third party effectively audit the multiple users' data for verifying integrity using Tanimoto index Margin Infused Relaxing Support Vector Regression. If the hash is matched correctly, then the data is not altered by any intruders increasing the data integrity rate. In this way, the proposed technique achieves higher data integrity on cloud storage with minimum auditing time.

4. Experimental Scenario

In this section, experimental evaluation of the proposed FSCHS-TIMRSVRDA technique PP-CSA [1], Dynamic data auditing scheme [2] is implemented using Java language with CloudSim simulator. The Amazon Elastic Compute Cloud (EC2) dataset http://www.ec2instances.info/ is used for data auditing in the cloud server. Amazon EC2 dataset is a web service that offers storage security on a cloud server. The user data are stored to a cloud server in the form of the hash value. Then the third-party auditor verifies the integrity of that stored data. The performance evaluation of the FSCHS-TIMRSVRDA technique is compared with existing methods with respect to a number of cloud user data.

5. Performance analysis

The result performance of the FSCHS-TIMRSVRDA technique and PP-CSA [1], Dynamic data auditing scheme [2] are described in this section with various metrics such as Space complexity, Data integrity rate, and data auditing time. The graphical results and tabulation show that the performance results of proposed and existing methods.

Space complexity: It is measured as the amount of memory space consumed for storing the hashed data into the cloud server. The space complexity is measured as given below,

$$Com_S = n * Mem\ (storing\ one\ data) \quad (6)$$

By using (6), Com_S denotes a space complexity,' n' indicates a number of cloud user data, Mem denotes a memory for storing the data. Space complexity is measured in terms of megabytes (MB).

Data integrity rate: It is defined as the number of data stored into the cloud server not altered by any intruders to the number of data. The formula for calculating the data integrity rate is given below,

$$Rate_{Int} = \left(\frac{n_{na}}{n}\right) * 100 \quad (7)$$

By using (7), '$Rate_{Int}$' denotes a data integrity rate, n_{na} denotes the number of data not altered, n' indicates a number of cloud user data. Data integrity rate is measured in the unit of percentage (%).

Data auditing time: It is defined as the amount of time consumed by the algorithm to audit the cloud user data. The overall time consumption of the algorithm is computed as given below,

$$T_{DA} = n * t\ (auditing\ one\ data) \quad (8)$$

In (8),T_{DA} represents the data auditing time and n represents the number of cloud user data, t denotes time consumption. The auditing time is measured in terms of milliseconds (ms).

Table 1 Space complexity

Number of cloud user data	Space complexity (MB)		
	FSCHS-TIMRSVRDA	PP-CSA	Dynamic data auditing scheme
40	21	24	28
80	24	28	30
120	30	32	36
160	34	38	42
200	40	44	48
240	43	48	53
280	48	50	56
320	53	56	61
360	55	58	63
400	58	60	64

Table 1 given above provides the performance results of space complexity using three methods namely the FSCHS-TIMRSVRDA technique, PP-CSA [1], Dynamic data auditing scheme [2]. In order to compute the space complexity, a number of cloud user data is taken in the ranges from 40 to 400. It is evident from the table that the proposed FSCHS-TIMRSVRDA technique provides a unique solution when compared to existing methods. However, with the experiment is conducted for 40 cloud user data, the memory consumed for storing the data was found to be $21MB$ using the FSCHS-TIMRSVRDA technique. Similarly, memory consumed for storing the data was found to be $24MB$ and $28MB$ using PP-CSA [1], Dynamic data auditing scheme [2] respectively. Likewise, various performance results are obtained with respect to a number of cloud user data. From this validation result, the space complexity using space complexity is said to be comparatively minimized by 8% and 17% than [1] and [2].

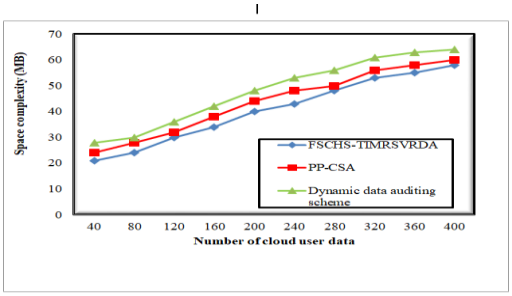

Figure 5 Graphical representation of space complexity

Figure 5 given above shows the graphical illustration of space complexity with respect to a distinct number of cloud user data. As shown in figure 5, 'x 'axis represents the number of cloud user data, and 'y' axis represents the space complexity. The graph shows that the space complexity of different methods gets increased while increasing the number of cloud user data ranging between 40 and 400. But comparatively, the space complexity of the FSCHS-TIMRSVRDA technique is found to be minimized when compared to existing methods. The reason behind the minimum space complexity was owing to the application of fast syndrome-cryptographic hash-based data storage onto the cloud server. The input cloud user data are converted into hash values and stored on the cloud server. The hash value of the data consumed lesser storage space than the original data. This helps to minimize the space complexity.

Table 2 Data integrity rate

Number of cloud user data	**Data integrity rate (%)**		
	FSCHS-TIMRSVRDA	PP-CSA	Dynamic data auditing scheme
40	95	90	88
80	94	87	84
120	95	88	85
160	94	89	86
200	95	88	85
240	96	90	87
280	95	88	84
320	96	91	87
360	97	90	88
400	95	89	86

Table 2 illustrates the comparison between data integrity rates using three different methods, FSCHS-TIMRSVRDA technique, PP-CSA [1], Dynamic data auditing scheme [2] respectively. The data integrity rate is measured with respect to a distinct number of cloud user data ranging between 40 and 400. Compared to other existing methods, the proposed FSCHS-TIMRSVRDA technique provides improved performance as compared to other related methods. This is inferred from the experiments where 38 data were not altered by any intruders using the FSCHS-TIMRSVRDA technique, 38 data

were not altered using [1] and 35 data were not altered using [2]. As a result, the overall data integrity rate using the three methods was found to be 95%, 90 [1], and 88 [2] respectively. Different performance results are observed for each method. With this, the overall data integrity rate using the FSCHS-TIMRSVRDA technique is said to be improved by 7% compared to [1] and 11% compared to [2].

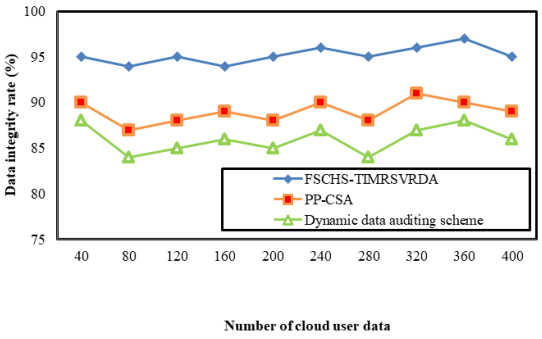

Figure 6 Graphical representation of data integrity rate

Figure 6 illustrates the data integrity rate of three auditing schemes namely the FSCHS-TIMRSVRDA technique, PP-CSA [1], Dynamic data auditing scheme [2]. As shown in the graph, the data integrity rate of the FSCHS-TIMRSVRDA technique is found to be increased when compared to existing methods. The reason behind the improvement was due to the application of Tanimoto index Margin Infused Relaxing Support Vector Regression. The Support Vector Regression uses the Tanimoto index for verifying the hash value of cloud user data by the third-party auditor. If the two hash values get matched, the input data was not altered by an intruder. This helps to obtain a high integrity rate.

Table 3 Data auditing time

Number of cloud user data	**Data auditing time (ms)**		
	FSCHS-TIMRSVRDA	PP-CSA	Dynamic data auditing scheme
40	16	20	24
80	20	22	26
120	24	26	28
160	27	32	34
200	32	34	36
240	34	36	38
280	36	39	42
320	38	42	44
360	40	43	45
400	42	44	48

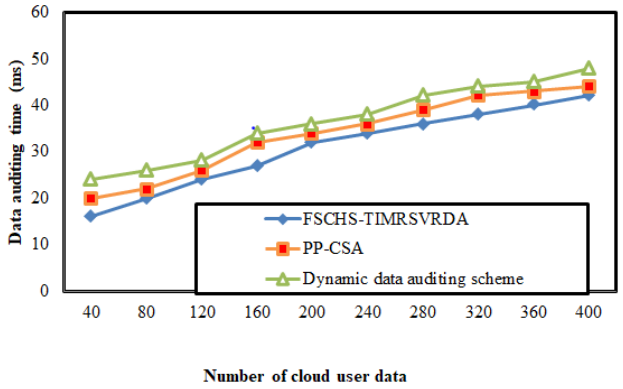

Figure 7 Graphical representations of data auditing time

Finally, Table 3 and Figure 7 the graphical illustration of data auditing time with respect to a distinct number of data. In the graph, the horizontal axis represents the number of data, and the vertical axis represents the data auditing time. As shown in figure 7, an increasing trend is found while increasing the cloud user data ranging between 40 and 400. Among three different auditing schemes, the FSCHS-TIMRSVRDA technique minimizes the data auditing time. This is proved through statistical estimation. The experimentation is conducted for 40 data, the time consumption of auditing the data using the FSCHS-TIMRSVRDA technique was found to be $16ms$. therefore, the time consumption of data auditing using PP-CSA [1], Dynamic data auditing scheme [2] are $20ms$ and $24ms$ respectively. As a result, the overall data auditing time was said to be minimized by 9% using the FSCHS-TIMRSVRDA technique when compared to [1] and 16% when compared to [2] respectively. The reason behind the application of Tanimoto index Margin Infused Relaxing Support Vector regression. The third party verifies the multiple users' data by using the Regression function. The regression function uses the Tanimoto index for measuring the similarity between the two hash values such as the hash value of requested data from the cloud server with the hash value of data stored in TPA. Based on the similarity value, the Regression function provides the verification results. This process helps to minimize the data auditing time.

6. Conclusion

In this paper, a proposed FSCHS-TIMRSVRDA technique is developed for data storage and auditing in the cloud environment. The proposed FSCHS-TIMRSVRDA technique achieved higher data integrity, by hashing the data before storing it on the cloud server. The FSCHS-TIMRSVRDA technique first performs the data storage with the help of fast syndrome-cryptographic hash-based data storage on to the cloud server. The input cloud user data are converted into hash values and it stored on the cloud server. The cloud server provides storage services to cloud users. Secondly, the Tanimoto index Margin Infused Relaxing Support Vector regression is applied to perform the integrity verification. In this way, efficient data auditing is performed with minimum time. Experimental assessment is carried out for analyzing the performance of the FSCHS-TIMRSVRDA technique and the conventional auditing methods with different metrics such as space complexity, data integrity rate, and data auditing time The implementation and discussed results confirm that the proposed FSCHS-TIMRSVRDA technique is performed well and is more efficient than the earlier ones, having a higher data integrity rate, and lesser time as well as space complexity.

References

[1] Yan Xu, Long Ding, Jie Cui, Hong Zhong, Jia Yu, "PP-CSA: A Privacy-Preserving Cloud Storage Auditing Scheme for Data Sharing", IEEE Systems Journal , Volume 15, Issue 3, 2021, Pages 3730 – 3739

[2] Xingyue Chen, Tao Shang, Feng Zhang, Jianwei Liu & Zhenyu Guan, "Dynamic data auditing scheme for big data storage", Frontiers of Computer Science, Springer, Volume 14, 2020, Pages 219-229

[3] Jiguo Li, Hao Yan, Yichen Zhang, "Identity-Based Privacy Preserving Remote Data Integrity Checking for Cloud Storage", IEEE Systems Journal, Volume 15, Issue 1, 2021, Pages 577 - 585

[4] Xuelian Li, Lisha Chen, and Juntao Gao, "An Efficient Data Auditing Protocol With a Novel Sampling Verification Algorithm", IEEE Access, Volume 9, 2021, Pages 95194 – 95207

[5] Li Li and Jiayong Liu, "SecACS: Enabling lightweight secure auditable cloud storage with data dynamics", Journal of Information Security and Applications, Volume 54 , 2020, Pages 1-10

[6] Baidaa Abdulrahman Jalil, Taha Mohammed Hasan, Ghassan Sabeeh Mahmood, Hazim Noman Abed, "A secure and efficient public auditing system of cloud storage based on BLS signature and automatic blocker protocol", Journal of King Saud University - Computer and Information Sciences, Elsevier, 2021, Pages 1-14

[7] Neenu Garg , Seema Bawa, Neeraj Kumar, "An efficient data integrity auditing protocol for cloud computing", Future Generation Computer Systems, Elsevier, Volume 109, August 2020, Pages 306-316

[8] Qingqing Gan, Xiaoming Wang, Jianwei Li, Jiajun Yan, Suyu Li, "Enabling online/offline remote data auditing for secure cloud storage", Cluster Computing, Springer, Volume 24, 2021, Pages 3027-3041

[9] Junfeng Tiana,Haoning Wang , Meng Wang, "Data integrity auditing for secure cloud storage using user behavior prediction", Computers & Security, Elsevier, Volume 105, 2021, Pages 1-13

[10] Reem Almarwani, Ning Zhang, James Garside, "A novel approach to data integrity auditing in PCS: Minimising any Trust on Third Parties (DIA-MTTP)", PLoS ONE, Volume 16, Issue 1, 2021, Pages 1-54

[11] Sameen Fatima, Dr. Shafiq Hussain and Rana Abu Bakar, "An Efficient Secure Auditing Framework for Big Data Storage in Cloud Computing Environment", International Journal of Computing and Digital Systems, Volume 10, Issue 01, 2021, Pages 817-827

[12] M. Thangavel and P. Varalakshmi, "Enabling Ternary Hash Tree Based Integrity Verification for Secure Cloud Data Storage", IEEE Transactions on Knowledge and Data Engineering , Volume 32, Issue 12, 2020, Pages 2351 – 2362

[13] Xiaodong Yang, Xizhen Pei, Meiding Wang, Ting Li, Caifen Wang, "Multi-Replica and Multi-Cloud Data Public Audit Scheme Based on Blockchain", IEEE Access, Volume 8, 2020, Pages 144809 – 144822

[14] Lunzhi Deng, Benjuan Yang, Xiangbin Wang, "A Lightweight Identity-Based Remote Data Auditing Scheme for Cloud Storage", IEEE Access, Volume 8, Pages 206396 – 206405

[15] Yang Xu, Ju Ren, Yan Zhang, Cheng Zhang, Bo Shen, Yaoxue Zhang, "Blockchain Empowered Arbitrable Data Auditing Scheme for Network Storage as a Service", IEEE Transactions on Services Computing , Volume 13, Issue 2, 2020, Pages 289 – 300

[16] Yang Yang, Yanjiao Chen, Fei Chen, "A Compressive Integrity Auditing Protocol for Secure Cloud Storage", IEEE/ACM Transactions on Networking , Volume 29, Issue 3, 2021, Pages 1197 – 1209

[17] Zhenpeng Liu, Yi Liu, Xianwei Yang,Xiaofei Li, “Integrity Auditing for Multi-Copy in Cloud Storage Based on Red-Black Tree”, IEEE Access , Volume 9, 2021, Pages 75117 – 75131
[18] Rama Krishna Kalluri and Guru C.V, “An effective analytics of third party auditing and Trust architectures for integrity in cloud environment”, Materials Today: Proceedings, Elsevier, 2021, Pages 1-5
[19] Wenting Shen, Jing Qin, Jia Yu, Rong Hao, Jiankun Hu, Jixin Ma, “Data Integrity Auditing without Private Key Storage for Secure Cloud Storage”, IEEE Transactions on Cloud Computing Volume 9, Issue 4, 2021, Pages 1408 – 1421
[20] Lu Rao, Hua Zhang, Tengfei Tu, “Dynamic Outsourced Auditing Services for Cloud Storage Based on Batch-Leaves-Authenticated Merkle Hash Tree”, IEEE Transactions on Services Computing , Volume 13, Issue 3, 2020, Pages 451 - 463