Indian Journal of Science and Technology, Vol 10(4), DOI: 10.17485/ijst/2017/v10i4/107893 January 2017
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Multi Aspect Sparse Time Integrated Cut-off Authentication (STI-CA) for Cloud Data Storage

R. Pragaladan* and S. Sathappan

Department of Computer Science, Erode Arts and Science College, Erode–638009, Tamil Nadu, India; pragaladanr@gmail.com, devisathappan@yahoo.co.in

Abstract

Objectives/Background: Cloud infrastructure is a pool of commuting resources such as information storage servers, application progress platforms, load balancers and virtual machines that are shared between the users for transactional processes with on demand process. However, transactional process lacks a secure authentication system, while it does not attest the trustworthiness of dynamic contents threats which outlaw the cloud system. **Methods/Statistical Analysis:** To establish the authenticity and avoiding improper data modification on cloud based data transactions, a framework called, multi aspect Sparse Time Integrated Cut-off Authentication (STI-CA) for Cloud Data Storage is designed. STI-CA framework commences with the password registry for each cloud user on the basis of two dimensional service matrices reducing the overhead incurred during user authentication by applying Sparse Vector Cloud User Registry. Next, by utilizing Time Integrated One Time Password, which is unique for each cloud user and each login reduces the execution time and space complexity as the cloud server does not maintain the password. Finally, the Cut-off Potential Cryptography prevents the unauthorized user modification on transactional data, therefore improving the security. Here the Amazon Simple Storage Service (Amazon S3) dataset is used for experiment using the JAVA coding with Cloudsim3. A series of simulation results are performed to test the data confidentiality, execution time, communication overhead and space complexity for obtaining transactional data and measure the effectiveness of STI-CA framework. **Findings:** STI-CA framework offers better performance with an improvement of the data confidentiality by 31%, reduces execution time by 20%, reduce communication overhead by 30% and also minimize space complexity by 22% compared to existing models of DRAFT and iCloud native Mac OS X respectively. **Applications/Improvements:** It can be further extended with implementation of new model with different parameters which improves more confidentiality and integrity.

Keywords: Authentication, Cloud Data Storage, Cut-off, Multi Aspect, Password registry, Potential Cryptography, Sparse, Time Integrated.

1. Introduction

In distributed computing environments, it remains crucial data acquisition on cloud server has become widely necessary to measure the authenticity of both the data and the user. Several remote attestation methods and mechanisms have been constructed for this purpose by measuring the integrity of a remote system in order to determine its authenticity and trustworthiness.

In this work, focus is made on developing a Multi aspect Sparse Time Integrated Cut-off Authentication (STI-CA) framework in cloud business data storage transaction processing. The STI-CA framework is constructed with the aim of improving the authorization of the cloud users and therefore improves the data confidentiality for cloud data request made by each cloud user.

The initial work involved in STI-CA framework is to reduce the execution time by assigning the user ID and password for the corresponding cloud user with the aid of sparse vector representation. Second part in STI-CA framework is to reduce the overhead and space complexity by applying one time password for each cloud user that changes over time and hence is said to be more secured. Finally, a Cut-off Potential Cryptography is applied in

* Author for correspondence

STI-CA framework improving the authentication and therefore improving the data confidentiality in obtaining the cloud data.

This paper provides the literature review in the Chapter 2. The Chapter 3 introduces our Sparse Time Integrated Cut-off Authentication (STI-CA) frameworkin cloud environment. Chapter 4 explains the experimental setup. Chapter 5 deals with the evaluation results of experimental studies. The paper concluded in Chapter 6.

2. Existing Work

A review has been performed for peer-reviewed articles that are relevant to this research topic. A brief summary of the research techniques, principles, findings, limitations, for each study is provided.

One of the biggest trends in cloud is the data storage. Several clients and organizations have starting storing their valuable data in remote servers in the cloud. Though integrity is verified by complete downloading, with the time consuming process, several works have been done on constructing remote data integrity. The remote data integrity grants to be verifiedpublic auditability and data dynamics without change for downloading the data.

Author in[1] explained the Dynamic Remote Attestation Framework and Tactics (DR@FT) which adopts a graph-based method to represent integrity violations and the graph-based policy analysis in cloud infrastructure. Domain-based integrity model process the system and protect the information with lesser runtime factor. DR@FT only verifies the most recent alter in a system position, instead of considering the entire system integrity level. Tolerable risk level is not explained detail in the cloud transaction system. An approach using iCloud service via the native Mac OS X system in[2] uses the file hash values to match with the original files using the MD5 algorithm. Synchronization integrity schema research was not carried out to establish the authenticity on cloud data based transactions.

Author in[3] introduced a remote data integrity checking protocol that was designed for cloud storage. The protocol also proved to be efficient in the aspects of communication, computation and storage costs. A review of models followed for addressing big data was provided in[4]. However, data level dynamics was not supported. To ensure data level dynamics, two greedy algorithms namely, Global Greedy Budget (GGB) and Gradual Refinement (GR) was provided in[5]. The design of these two algorithms in turn ensured performance optimization.

With the increasing growth of cloud computing services, there has been a massive trend toward large-scale and geographically distributed data centers. An efficient and provable online scheduling algorithm was designed in[6] model using optimal offline algorithm to ensure energy fairness cost in cloud server. A review on security and privacy issues by applying Hadoop MapReduce framework on cloud was discussed in[7] model. To ensure both authentication of user and data integrity, a Time-based One-Time Password (TOTP) model[8] was designed to fully protect the system from unauthorized third party.

More secured cloud user authentication can be ensured by designing a more decentralized storage system for accessing data with anonymous authentication. However, modification of data has to be restricted while data access.

Author in[9] designed a decentralized framework to ensure authentication and restriction of unauthorized data modification through authorization and access policy provider. A security framework using Shield was designed by[10] method for applying Merkle Hash Tree. The advantage of the method lies in extracting the data or file without modifying the underlying file system. However, the cloud data storage in Shield did not concentrated on maximizing the security on performing the transactions over cloud servers.

An investigation on the techniques for encryption and authentication followed in cloud was presented in[11] technique. Author in[12] made an attempt to make a review of encryption techniques available to ensure data confidentiality and authentication in cloud environment. Though authentication and data confidentially is the key for obtaining the cloud owner data for each cloud user, access control is another way through which authorization can be ensured.

Author in[13] designed a privacy aware access control system with the help of third party provider ensuring security and defence against collusion. An ID-based cryptography scheme by[14] provided cloud-based storage applications to address the protection of cloud storage infrastructure. Despite, authentication and security, computational aspect remained unaddressed. This aspect was covered in[15] scheme by applying light weight nature framework for Mobile Cloud Computing (MCC). On the other hand, data privacy through third party was provided in using a personal metadata management framework[16].

Author in[17] provided another model for secure data sharing based on key management. Moreover, the paper also focused on privacy and confidential data sharing aspects. However, client-side de-duplication across multiple servers helps in achieving cost savings. To address this de-duplication, a differentially client-side de-duplication protocol was designed in[18]. However, computational cost was not ensured. To avoid this, a security risk assessment framework was designed in[19] model which reduce the complexity involved during risk assessment. An efficient and secure data sharing framework with the objective of providing privatization and ensure accessibility was designed in[20] scheme.

In the analysed papers, the authors employ different mechanisms for ensuring authentication, but in all of them, there appears a single factor for authentication, which is highly susceptible to attacks or data modification. In this way, multiple aspects for authentication and system accessibility have to be explored. The work proposed in this paper takes account of multiple aspects using Sparse Time Integrated Cut-off Authentication for cloud data storage, which is discussed in the forthcoming sections.

3. Proposed Work

3.1 Sparse Time Integrated Cut-off Authentication (STI-CA) for Cloud Data Storage

In this section a MultiAspect Authentication framework was used. Here Sparse Time Integrated Cut-off Authentication (STI-CA) framework is provided. The STI-CA framework uses multiple aspects for user authorization using Password Registry, providing One Time Password (OTP) through time integrated model and Cut-Off Potential Cryptography (COPC) scheme. The STI-CA framework starts with the system model, followed by which the multi aspects are explained in the following sections.

3.2 System Model

Let us consider a Cloud Service Provider ($CSP_i = CSP_1, CSP_2, \ldots, CSP_n$) that manages Cloud Storage Server (CSS), with significant amount of storage space, ensuring preservation of the cloud owner's data. Moreover the Cloud Data Owner ($CDO_i = CDO_1, CDO_2, \ldots, CDO_n$) has large Data Files ($DF_i = DF_1, DF_2, \ldots, DF_n$) to be stored in the Cloud Storage Server (CSS).

Cloud Data Owner represents an individual or an organization that outsources the encrypt data to the Cloud Service Provider. Finally, the Cloud User ($CU_i = CU_1, CU_2, \ldots, CU_n$) registered with the Cloud Owner uses the data of Cloud Owner stored on the Cloud Storage Server (CSS). Cloud users' are the authorized persons who request data and use data stored in Cloud Storage Server based on their access rights.

The most convenient means of cloud authentication is through remote attestation. However, it does not attest the trustworthiness of dynamic contents. It is the goal of this paper is to study the measure of trustworthiness using multiple aspect authentications and improve the computation overhead and ensure authentication. In the next section we start with extending these results to the multi aspect authentication for cloud data storage.

3.3 Sparse Vector Cloud User Registry

Sparse Vector Cloud User Registry in the STI-CA framework aims to seek a compact dictionary of Cloud Data Owners to succinctly represent large-scale CDOs. Let us consider a set of services '$S = S_1, S_2, \ldots, S_n$' where each '$S_i \in R^n$' is denoted as a cloud user vector and 'n' is the size of the cloud user dictionary who can access the cloud data owner files. By mapping cloud data owners into rows and cloud users into columns, 'S' is represented by a two dimensional service matrix '$TDS \in R^{n*m}$'. Each entry corresponds to the cloud user and cloud data owner respectively. The intersection of them provides with the unique user name and password for each cloud user, ensuring data to be accessed as given below.

$$\begin{array}{c} \\ CU_i \\ CU_{i+n} \end{array} \begin{array}{c} \begin{array}{cccc} CDO_i & CDO_i & & CDO_{i+n} \end{array} \\ \begin{bmatrix} \square & \square & \ldots & \square \\ . & . & \ldots & . \\ . & . & \ldots & . \\ \square & \square & \ldots & \square \end{bmatrix} \end{array}$$

Figure 1 shows the illustration of cloud user registry using sparse vector representation. Cloud User requests for Cloud Owner data sends his/her registration request by providing a unique username, password. Accordingly, the corresponding 'U_{name}, pwd' is stored in the password registry 'PR'. Therefore, the Cloud User will be registered as a new user to have a unique username and password 'U_{name}, pwd' respectively. This test is to ensure that the user name and password is stored in the password registry and used as a measured for future authentication.

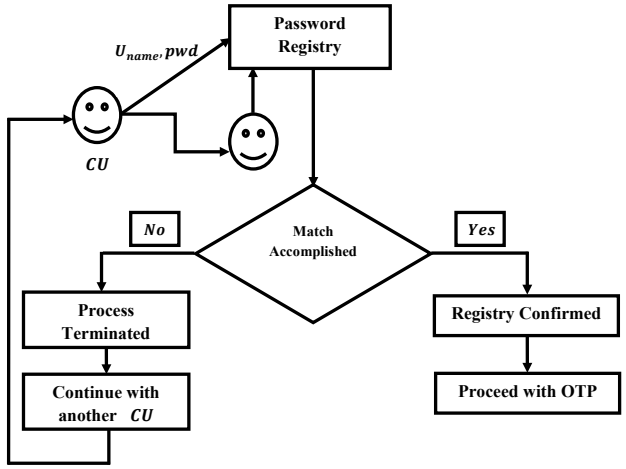

Figure 1. Illustration of cloud user registry.

As illustrated in the Figure 1, whenever a cloud user has to access the data or file of the cloud data owner, the cloud user initiates the sign up by providing a unique user name and password. This in turn checks with the 'U_{name}, pwd' stored in password registry. If a match is said to occur, the registry process is confirmed and proceeds with One Time Password discussed in the following section. Otherwise, the process is said to be terminated and continues with another cloud user. This process is repeated for all other cloud users. Algorithm 1 illustrates the registration of user name and password performed by cloud users in a cloud environment using Sparse Vector Representation (SVR).

Input: Cloud Service Provider ($CSP_i = CSP_1, CSP_2, \ldots, CSP_n$), Cloud Storage Server (CSS), Cloud Owner ($CO_i = CO_1, CO_2, \ldots, CO_n$), Data Files ($DF_i = DF_1, DF_2, .., DF_n$), Cloud User ($CU_i = CU_1, CU_2, \ldots, CU_n$)

Output: Authentication of cloud users for corresponding section

1: **Begin**
2: **For** each Cloud Owner 'CO_i' and Data Files 'DF_i'
3: **For** each Cloud User 'CU_i' who have to access Data Files 'DF_i'
4: Store U_{name} , pwd in password registry 'PR'
5: **If** match occurs
6: Registry is said to be confirmed
7: Proceed with OTP (algorithm 2)
8: **End if**
9: **If** match does not occur
10: Process terminated
11: Go to step (2)
12: **End if**
13: **End for**
14: **End for**
15: **End**

Algorithm 1. Sparse vector registry algorithm.

In the above Algorithm 1, the primary job of the Cloud Service Provider is to verify the username and password provided by the cloud user via Sparse Vector representation. Here the representation of password registry is provided through Sparse Vector where cloud user represents the rows and cloud data owner represents the columns. Upon successful match with the password registry, one time password is provided by the Cloud Service Provider that is explained in the following sections.

3.4 Time Integrated One Time Password (Reduces Space Complexity)

Once, the password registry is accomplished, the second aspect used for authentication to authorize the users is the Time Integrated OTP (TI-OTP). Accordingly, the Time Integrated OTP (TI-OTP) in the STI-CA framework has the advantage of low overhead for management as it does not involve hardware token distribution and management and in turn minimizes the space complexity. In addition, as the CSP does not keep any data regarding the OTD, the transaction involved in it is also said to be more secured.

The OTP in the STI-CA framework uses Time Integrated OTP (TI-OTP) that requires the CSS and user's password (obtained during registry) to be integrated to assemble a precise password to appropriate cloud user authentication. The advantage of using TI-OTP is that the assembled password has to be used by the cloud user within a confined time period 't', exceeding the confined time period 't' causes expiry of the OTP and authentication gets failed.

Upon every login of the cloud user, TI-OTP is a password that does not remain static, but change over time. Also, static passwords cached on the hard drives and servers are easily susceptible to attack. But, as TI-OTP is different for each user and for each login, storing the passwords are also harmless. Figure 2 illustrates the Time Integrated One Time Password.

As illustrated in the Figure 2, whenever a cloud user initiates login, the user id and password 'U_{name}, pwd' is verified by the CSS. As soon as the 'U_{name}, pwd' is verified, the CSS send a one-time password. This is called as the Time Integrated One Time Password which is valid for a fixed time 't'. This Time Integrated One Time Password has the advantage of reducing the space complexity as the password is generated for each user one time and therefore no additional space is required to

store the password. After the OTP is verified, the Cloud User successfully logs on. The Time Integrated One Time Password (TI-OTP) is illustrated in Algorithm 2.

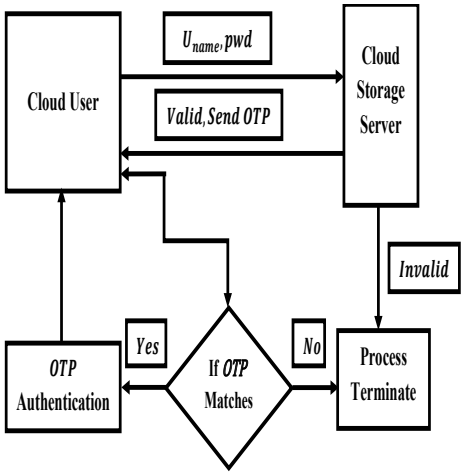

Figure 2. Illustration of time integrated one time password.

Input: Cloud Service Provider ($CSP_i = CSP_1, CSP_2, \ldots, CSP_n$), Cloud Storage Server (CSS), Cloud Owner ($CO_i = CO_1, CO_2, \ldots, CO_n$), data files ($DF_i = DF_1, DF_2, .., DF_n$), Cloud User ($CU_i = CU_1, CU_2, \ldots, CU_n$), Time '$t$'

Output: Minimized overhead

1: **Begin**
2: **For** each Cloud User CU_i
3: Send 'U_{name} , pwd' to Cloud Storage Server CSS
4: **If** 'U_{name} , pwd' matches with password registry
5: Cloud Storage Server CSS sends OTP
6: **For** each time interval 't'
7: **If** OTP Authentication verified
8: Process continues (algorithm 3)
9: **End if**
10: **End for**
11: **End if**
12: **If** 'U_{name} , pwd' does not matches with cloud storage registry
13: Process terminates
14: **End if**
15: **End for**
15: **End**

Algorithm 2. Time integrated one time password algorithm.

As illustrated in Algorithm 2, each cloud user sends the 'U_{name}, pwd ' to the CSS. Upon successful match with the password registry, the CSS sends the OTP to the cloud user. To perform an authentication process, the OTP received by the cloud user from the cloud storage server is used. Upon successful verification of OTP, the third aspect authentication called, Cut-off Potential Cryptography scheme is used as a final authorization.

3.5 Cut-off Potential Cryptography

The minimum decryption servers (potential) given to the cloud user for which the cloud user one time password is in accordance with the cloud storage servers one time password is called as the Cut-off Potential Cryptography scheme. The purpose of using Cut-off Potential Cryptography scheme as a third aspectauthentication is to provide more security to the password (i.e., 'pwd ' generated using sparse vector registry and 'OTP ' generated using Time Integrated One Time Password) by the multi aspect authentication framework. This in turn prevents the unauthorized user modification on the transactional data that is carried out using the synchronized map relationship.

In Cut-off Potential Cryptography scheme, the transactional data is partitioned into 'n ' elements and knowledge of certain elements 'c ' enables to obtain the secret transactional data. Hence, only proper authenticated and authorized cloud users are accessible to the secret transactional data. Such a scheme is called a '(c, n)' Cut-off Potential Cryptography. This in turn restricts the exploitation of the transaction data at higher level, thereby improving the security factor.

Let us consider that we use '(c, n)' Cut-off Potential to share the cloud users transactional data. Let us select a random '$k-1$ ' coefficients and '$c_0, c_1, \ldots, c_{k-1}$'. Then we can build the polynomial to divide the transactional data as[3]

$$P(TD) = p_0 + p_1 TD + p_2 TD^2 + \ldots\ldots + p_{k-1} TD^{k-1} \quad (1)$$

In Equation (1), the polynomial '$P(\square)$' model involved in the partitioning of transaction data 'TD ' is provided, where the transaction data is split into '$k-1$ ' coefficients. In order to reconstruct the transaction data 'TD ', 'c ' points are used in the STI-CA framework,

Lagrange Coefficient is used and is mathematically formulated as given below,

$$l_0 = \frac{(p - p_1)}{(p_0 - p_1)}, \frac{(p - p_2)}{(p_0 - p_2)} \quad (2)$$

$$l_1 = \frac{(p - p_0)}{(p_1 - p_0)}, \frac{(p - p_2)}{(p_1 - p_2)} \quad (3)$$

$$l_2 = \frac{(p - p_0)}{(p_2 - p_0)}, \frac{(p - p_1)}{(p_2 - p_1)} \quad (4)$$

$$f(p) = \sum_{j=1}^{n} q_j l_j s(p) \quad (5)$$

The Cut-off Potential Cryptography using Lagrange Coefficient is illustrated in Algorithm 3.

As illustrated in Algorithm 3, for each cloud user who wants to access the cloud data owners' transaction data, the transactional data is divided into polynomial model with the objective of improving the availability of transactional data and henceforth the rate of security in cloud environment. The Cut-off Potential Lagrange Coefficient Cryptography (CPLC-C) distributes the transactional data among 'n' servers so that at least 'c' servers are needed for decryption of the corresponding transaction data.

Input: Cloud User ($CU_i = CU_1, CU_2, \ldots, CU_n$), Transactional Data '$TD$', elements '$c$', total elements '$n$'

Output: Secured transaction data

1: **Begin**
2: **For** each Cloud User CU_i
3: **For** each Transactional Data 'TD'
4: Construct polynomial to divide transactional data using (1)
5: **End for**
//reconstruct
6: Obtain the values for 'c' and 'n'
7: **For** each 'c' values
8: Perform Lagrange Coefficient using (2), (3) and (4)
9: **End for**
10: **End for**
11: **End**

Algorithm 3. Cut-off potential lagrange coefficient cryptography algorithm.

4. Performance Analysis

To evaluate the efficiency of the proposed framework, comparison experiments were conducted on a PC with a 2.6GHz Pentium Dual Core Processor, Windows platform. Besides, the experiments used CloudSim3 simulator[15] to perform authentication for cloud data storage. The toolkit of CloudSim3 simulator supports modelling of virtualized environments like cloud system components namely, data centers, cloud users, VMs.

Comparison of the proposed framework is made with two already known authentication methods to ensure authentication in cloud business data storage transaction process. One of the methods is Dynamic Remote Attestation Framework and Tactics[1] (DRAFT) whose objective is to provide efficient and effective attestation that proved the authentication of measurements by verifying the has value. Another method called, iCloud service via native Mac OS X system[2] (iCloud native Mac OS X) is compared with, where acquisition of iCloud data was performed using forensically robust method. All of the results obtained by using the two methods above are compared with the multi aspect Sparse Time Integrated Cut-off Authentication (STI-CA) for Cloud Data Storage.

The multi aspect Sparse Time Integrated Cut-off Authentication (STI-CA) for Cloud Data Storage framework is developed to improve the security and authentication using the Amazon Simple Storage Service (Amazon S3) dataset. This dataset is experimented using the JAVA coding. The JAVA coded with Cloudsim3 platform easily performs authentication before implementing in the real world scenario. Amazon S3 is a warehouse of data that includes images, files and other type of useful information. Amazon S3 based storage of files are discussed and used in our experimental discussions to identify the result percentage. Also, Amazon S3 is a reliable, fast, inexpensive data storage infrastructure for efficient query processing. Amazon S3 stores data objects on multiple devices diagonally on multiple services and permit simultaneous read and write access.

The performance evaluation tests aimed at comparing the existing model with the proposed multi aspect Sparse Time Integrated Cut-off Authentication (STI-CA) framework.

The STI-CA framework using the simulator, we proposed a simulation environment that has the following parameters: the number of transactional data size varies 10 to 70 with different number of data files being required for each user. The following parameters including data confidentiality, execution time, communication overhead and space complexity in cloud environment is evaluated.

4.1 Scenario 1: Data Confidentiality

Data confidentiality is measured with the intention of guarding the cloud user's transactional data from being accessed by unauthorized persons and service providers. Data confidentiality is calculated as the difference between size of Transactional Data (TD) and the size of compromised TD by unauthorized access. Data confidentiality is mathematically formulated as follows.

$$DC = Size(TD) - Size\left(compromised\ TD\right) \quad (6)$$

In Equation (6), the data confidentiality 'DC' is measured in terms of kilobytes (KB). If data confidentiality is high, then the method is said to be efficient.

Table 1. Tabulation for data confidentiality

Transactional	Data Confidentiality (KB)		
Data size(KB)	STI-CA	DRAFT	iCloud native Mac OS X
10	14.2	11.3	7.6
20	22.3	19.5	15.5
30	33.7	27.4	23.2
40	45.4	36.3	31.4
50	56.2	46.5	42.1
60	62.8	53.7	51.1
70	72.7	67.3	59.3

Table 1 illustrates data confidentiality based on transaction data size for proposed STI-CA method and existing DRAFT and iCloud native Mac OS X methods. Transaction data size is varied from the range of 10 to 70 KB. From the Table 1, it is clear that for the increase in transaction data size, data confidentiality is also increased for all methods. However, proposed STI-CA method provides better performance in terms of improving data confidentiality when compared to existing methods.

Figure 3 shows the measure of data confidentiality using proposed STI-CA method which is compared with existing methods such as DRAFT and iCloud native Mac OS X. As shown in Figure 5, proposed STI-CA method provides higher data confidentiality when compared to existing methods. This efficient improvement of data confidentiality in proposed STI-CA method is because of using Cut-off Potential Cryptography for allocating data to each cloud user. Therefore, data confidentiality of proposed STI-CA method is improved by 19% when compared to existing DRAFT method and 43% when compared to existing iCloud native Mac OS X method respectively.

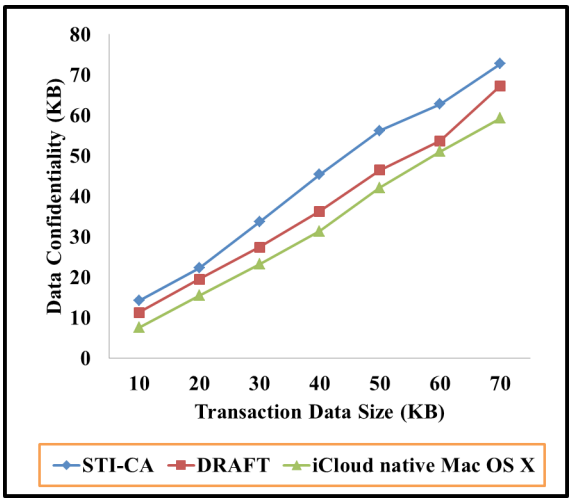

Figure 3. Measure of data confidentiality.

4.2 Scenario 2: Execution Time

Execution time is the amount of time taken by cloud user to access the cloud data owner files using Sparse Vector Representation in a more dynamic manner that stores vast unique user identification and password, making the system to perform in a more scalable manner. The transactional data size (KB) used in this experiment ranges from 10 to 70 cloud users. To evaluate the performance of execution time using the two methods, DRAFT[1]and iCloud native Mac OS X[2] are compared with STI-CA. The performance of execution time is measured in terms of milliseconds (ms). The mathematical formulation for execution time is as given below.

$$ET = \sum_{i=1}^{n} TD_{size} * T(Req_i) * T\left(confident\ data\ transaction\right) \quad (7)$$

In Equation (7), the execution time 'ET', is described using transaction data size TD_{size} and time for '$Time_{confidentdatatransaction}$' respectively. Lower execution time ensures that cloud users have access to the cloud data owner files in an easily accessible manner.

To better understand the effectiveness of the proposed STI-CA framework, the experimental result of execution time is reported in Table 2. Cloudsim3 simulator is used

to measure and experiment the factors by analyzing the percentage of result with the help of table and graph values. Results are presented for different transactional data size. The results reported here confirm that with the increase in the transactional data size, the execution time also increases.

Table 2. Tabulation for execution time

Transactional	Execution Time (ms)		
Data size(KB)	STI-CA	DRAFT	iCloud native Mac OS X
10	256	296	312
20	302	345	486
30	398	450	522
40	412	532	624
50	482	596	693
60	521	602	712
70	592	688	789

Figure 4 presents the variation of execution time with respect to transactional data size. All the results provided in figure confirm that the proposed STI-CA framework significantly outperforms the other two methods, DRAFT[1] and iCloud native Mac OS X[2]. At the same time, the curve is also linear because an increase in the transactional data size increases the execution time for allocating data files by cloud storage server. The execution time is reduced in the STI-CA framework as it uses Sparse Vector Representation that stores huge and unique user identification and password. By applying Sparse Vector Representation in STI-CA framework, a third party, cloud storage server is used where the user name and password are stored in the corresponding users' password registry. On behalf of the cloud user, the cloud storage server performs a check and ensures secure flow of cloud data as requested by the cloud user, which is less time consuming than the linear model. We obtain the individual user ID and password, by the vector representation, for each cloud user that in turn discriminates with the other cloud users' from an ensemble of cloud user. This results in the reduced execution time while assessing the cloud data owner files. This Sparse Vector representation in STI-CA framework in turn reduces the execution time by 13% compared to DRAFT[1] and 27% compared to iCloud native Mac OS X[2].

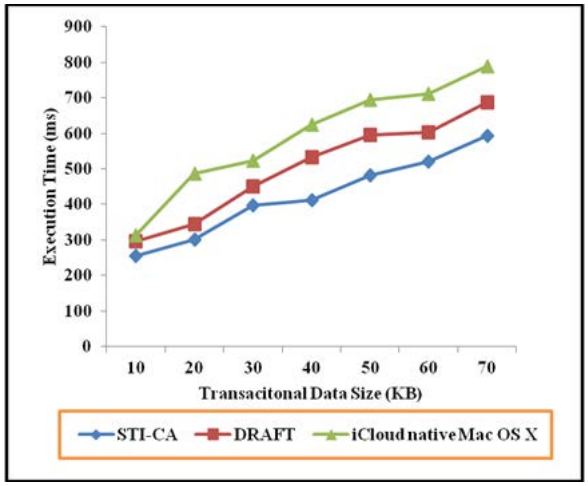

Figure 4. Measure of execution time.

4.3 Scenario 3: Communication Overhead

In order to minimize the communication overhead, One Time through Time Integrated model is presented that provides one time password to each cloud user for a particular time interval. In the experimental setup, the transactional data size considered the ranges from 10 to 70.

Communication overhead is an indirect memory required to generate one time password for each cloud user with the objective of providing authentication for cloud data storage. Once password registry is accomplished for each cloud user, a proportion of overhead is said to occur during one time password generation. This proportion of overhead incurred during one time password generation is called as the communication overhead. With the increase in the size of transactional data the Communication Overhead also increased. The mathematical formulation for communication overhead is as given below.

$$CO = \frac{\sum_{i=1}^{n} Size\,(TD_i) * \textbf{amount of data lost}}{timestamp} \quad (8)$$

In Equation (8), the communication overhead CO is measured by multiplying the transaction data size and the transaction data lost in specified timestamp. It is measured in terms of bits per second (bps). Lower the communication overhead is more efficient.

Table 3. Tabulation for communication overhead

Transactional	Communication Overhead (bps)		
Data size (KB)	STI-CA	DRAFT	iCloud native Mac OS X
10	2.89	3.67	4.72
20	3.42	5.22	6.13
30	5.13	8.24	9.72
40	7.24	11.36	13.36
50	8.45	13.42	15.72
60	10.87	14.89	16.39
70	12.24	16.72	18.35

The targeting results of communication overhead using STI-CA framework with two state-of-the-art methods[1,2] in the Table 3 presented for comparison based on the transactional data size for ensuring authentication during transactional data storage in cloud environment.

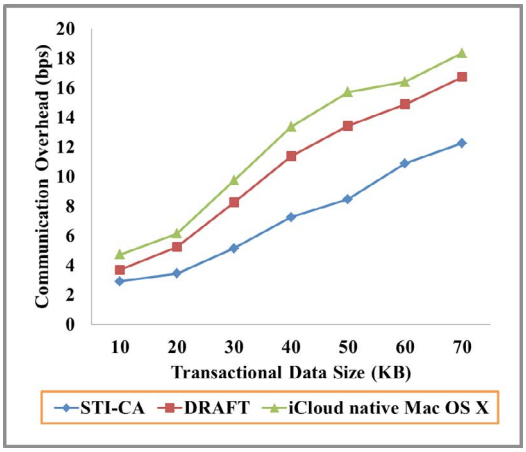

Figure 5. Measure of communication overhead.

To explore the influence of communication overhead on STI-CA framework, the experiments performed by varying the transactional data size as shown in Figure 5. It also shows that the STI-CA framework shows competitive results with the state-of-the-art methods, namely DRAFT[1] and iCloud native Mac OS X[2]. This is because of the application of Time-Integrated One Time Password. The Time-Integrated One Time Password in STI-CA framework generates different passwords for each different login and the password is not stored in the cloud storage server, as the one time password differs for the same user at a different interval time. To access the cloud data owner files, accessibility is possible only with the authenticated cloud users. Therefore, instead of using a static password, the STI-CA framework uses one time password that reduces the drudgery of hard drives of passwords being stored with additional storage. This in turn reduces the computation overhead while assessing the cloud data owner files, as the password is used with limited time period. This time integrated one time password in STI-CA framework in turn reduces the communication overhead by 26% compared to DRAFT[1] and 33% compared to iCloud native Mac OS X[2].

4.4 Scenario 4: Space Complexity

The better the time complexity of an algorithm is the faster carries the overall process of ensuring authenticity and avoiding improper data modification on cloud data based transactions. Space complexity on the hand, is the total space taken by the algorithm with respect to the input size provided, that includes both the auxiliary space and space used by input. Apart from time complexity as discussed above through two parameters namely, execution time and communication overhead, its space complexity is also important. This is essentially the number of memory cells which an algorithm requires, where a good algorithm helps in keeping this number as small as possible.

$$SC = Mem(SVR + TI - OTP + CPLC - C) * CUR_i \quad (9)$$

In Equation (9), the space complexity is attained by measuring the memory required for three algorithms used namely, Support Vector Registry (SVR), Time Integrated One Time Password (TI-OTP) and Cut-off Potential Lagrange Coefficient Cryptography (CPLC-C) with respect to the input transactional data size '⟦TDS⟧_i' respectively.

Table 4. Tabulation for space complexity

Transactional	Space Complexity (bps)		
Data size (KB)	STI-CA	DRAFT	iCloud native Mac OS X
10	9	12	17
20	17	23	29
30	24	27	35
40	32	35	41
50	41	48	52
60	48	55	65
70	59	62	70

Table 4 illustrates the measure of space complexity with respect to varied transactional data sizes. To evaluate the impact of the three algorithms, Support Vector Registry (SVR), Time integrated one time password (TI-OTP) and Cut-off Potential Lagrange Coefficient Cryptography (CPLC-C) on the space complexity with respect to varied transactional data sizes. To this end, we fix a seven stage

transactional data sizes at different time intervals is varied from 10 to 70. The results of DRAFT[1] and iCloud native Mac OSX is compared to the STI-CA are shown in the Figure 6. With the space complexity increasing with varied transactional data size, seems to be lesser when applied with STI-CA. This is because of the generation of one time password, valid for only one session for each cloud user that in reduces the memory to store the password for each user. Therefore, the space complexity is reduced using STI-CA by 15% compared to DRAFT[1] and 29% compared to iCloud native Mac OS X[2]. The Figure 6 also provides evidence that the performances of both DRAFT and iCloud native Mac OS X are very close to the optimal STI-CA, but remote attestation performed in DRAFT and forensically robust method considers latest state changes of a target system instead of considering the entire system information.

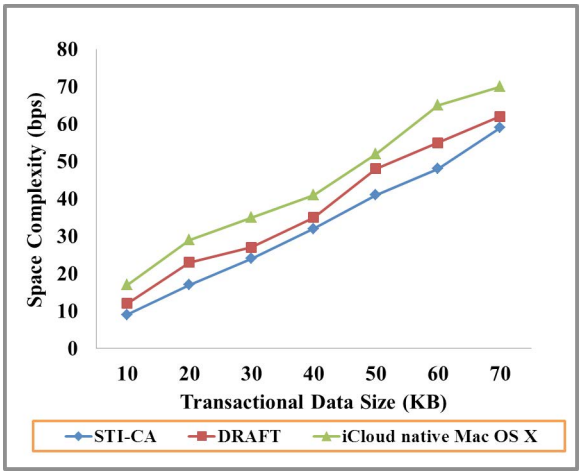

Figure 6. Measure of space complexity.

5. Conclusion

This paper proposes a multi aspect Sparse Time Integrated Cut-off Authentication (STI-CA) for Cloud Data Storage framework is provided based on the Sparse Vector Registry, Time Integrated One Time Password and Cut-off Potential Lagrange Coefficient Cryptography algorithm. This framework improves the cloud data retrieval efficiency and reduces the execution time in cloud environment. This is achieved through multi aspect authentication, where the authorization is provided from the starting of the password registry using Sparse Vector registry till the appropriate data provisioning for each cloud user using Cut-off Potential Lagrange Coefficient. The authentication property of our work is based on the sparse vector registry model based on two dimensional service matrices which is used to store corresponding user Id and password in password registry. To achieve the efficiency and effectiveness of authentication, STI-CA framework focuses on minimizing the overhead, space complexity, ensuring security by using time integrated one time password, instead of static passwords provided for each cloud user. Finally, we have extended the hash file matching for allocation data to each cloud user with a cut-off potential cryptography scheme.

6. References

1. Xu W, Zhang X, Hu H, Ahn GJ, Seifert JP. Remote Attestation with domain-based integrity model and policy analysis. IEEE Transactions on Dependable and Secure Computing. 2012 May/Jun; 9(3):429–42.
2. Oestreicher K. A forensically robust method for acquisition of iCloud data. Digital Forensics and Incident Response. 2014 Aug; 11:S106–13.
3. Hao Z, Zhong S, Yu N. A privacy-preserving remote data integrity checking protocol with data dynamics and public verifiability. IEEE Transactions on Knowledge and Data Engineering. 2011 Sep; 23(9):1432–7.
4. Hashem IAT, Yaqoob I, Anuar NB, Mokhtar S, Gani A, Khan SU. The rise of "big data" on cloud computing: Review and open research issues. Information Systems. 2015 Jan; 47:98–115.
5. Wang Y, Shi W. Budget-driven scheduling algorithms for batches of mapreduce jobs in heterogeneous clouds. IEEE Transactions on Cloud Computing. 2014 Jul-Sep; 2(3):306–19.
6. Polverini M, Cianfrani A, Ren S, Vasilakos AV. Thermal-aware scheduling of batch jobs in geographically distributed data centers. IEEE Transactions on Cloud Computing. 2014 Jan-Mar; 2(1):71–84.
7. Derbeko P, Dolevb S, Gudes E, Sharma S. Security and Privacy aspects in MapReduce on clouds: A survey. Computer Science Review. 2016 May; 20:1–28.
8. El-Booz SA, Attiya G, El-Fishawy N. A secure cloud storage system combining time-based one-time password and automatic blocker protocol. EURASIP Journal on Information Security. 2016 Jun; 1–13.
9. Mokle S, Shaikh NF. Anonymous authentication for secure data stored on cloud with decentralized access control. IEEE WiSPNET; 2016. p. 216–20.
10. Shu J, Shen Z, Xue W. Shield: A stackable secure storage system for file sharing in public storage. Journal of Parallel and Distributed Computing. Elsevier Journal. 2014.
11. Rajamani T, Sevugan P, Purushotham S. An Investigation

on the techniques used for encryption and authentication for data security in cloud computing. IIOAB Journal. 2016; 7(5):126–38.
12. Soofi AA, Khan MI, Amin FE. Encryption techniques for cloud data confidentiality. International Journal of Grid Distribution Computing. 2014; 7(4):11–20.
13. Takabi H. Privacy aware access control for data sharing in cloud computing environments. International workshop on Security in cloud computing; 2014. p. 27–34.
14. Medhioub M, Hamdi M, Kim TH. A new authentication scheme for cloud-based storage applications, International Conference on Security of Information and Networks; 2016. p. 57–60.
15. Rodrigo N, Rajiv R, Anton B, Csar A, Buyya R. CloudSim: A toolkit for modelling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software: Practice and Experience. 2011; 41(1):23–50.
16. Montjoye YA, Shmueli E, Wang SS, Pentland AS. OpenPDS: Protecting the privacy of metadata through safe answers. Plos one. 2014 Jul; 9(7):1–9.
17. Thilakanathan D, Nepal CSS, Calvo RA. Secure data sharing in the cloud. Security Privacy and Trust in Cloud Systems. Springer; 2014. p. 45–72.
18. Shin Y, Kim K. Differentially private client-side data reduplication protocol for cloud storage services. Security and Communication Networks. 2014 Oct; 8(12):2114- 23.
19. Albakri SH, Shanmugam B, Samy GN, BashahIdris N, Ahmed A. Security risk assessment framework for cloud computing environments. Security and Communication Networks. 2014 Jan; 7(11):2114–24.
20. Li J, Liu JLZ, Jia C. Enabling efficient and secure data sharing in cloud computing. Concurrency and Computation: Practice and Experience. 2013 Jun; 26(5):1052–66.