Fast and Secure Data Accessing by using DNA Computing for the Cloud Environment

Suyel Namasudra, *Member, IEEE*

Abstract—In a cloud environment, traditional approaches are used to encrypt any data by using 0 and 1 that increase data security issues because of the presence of numerous malicious users and hackers over the internet. Deoxyribonucleic Acid (DNA) computing can be one of the best solutions to improve data security in which data are encrypted using the DNA bases: Thymine (T), Guanine (G), Cytosine (C) and Adenine (A). Along with data security, access control is another major issue of a cloud environment as the searching time of the owner of any data, the system overheard and the accessing time of a file or data are high during data access. A novel DNA computing based secure and fast Access Control Model (ACM) is proposed in this paper to solve all these major problems. In the proposed scheme, the Cloud Service Provider (CSP) keeps a table or list for fast data accessing. Here, a 1024-bit DNA computing based random key is generated by using the user's secret information, and the same key is utilized for data encryption. Theoretical analysis along with many experimental results prove the efficiency and effectiveness of the proposed access control model over some well-known existing models.

Index Terms—Cloud Service Provider, Complementary Pair Rule, DNA Encoding Rule, DNA-EXOR Rule, Decimal Encoding Rule, CloudSim.

1 Introduction

THE advances of cloud computing technologies attract Information Technology (IT) sector for shifting the IT resources to the cloud environment. Cloud computing is basically the on-demand availability of the computer resources or services, mainly computing power and data storage without using any external hardware or software. Here, numerous distributed computers are combined to provide various kinds of services, such as a server, storage and many more [1]. Cloud computing has very attractive benefits, such as pay-per-use, on-demand service, unlimited storage capacity, flexibility and many more. However, it also has many disadvantages, namely security, access control, limited control, downtime, etc. A cloud computing environment consists of three entities: (i) Data Owner (DO) (ii) CSP, and (iii) user. The users and DOs use numerous cloud services delivered by the CSP [2-4]. Here, many DOs use the cloud server to store their file or data, and users access data from the cloud server based on the requirements. When a user accesses any data from a cloud server, there are some major issues: (i) high time for finding a DO (ii) high time to data access (iii) high system overhead and (iv) data security. After getting a data accessing demand from an authorized user, the CSP may initiate a query in the entire cloud database to get the details of a DO, so that it can deliver the Public Key of the DO (PBKDO) to the requested user. Hence, the searching time can be high in many instances. The user or customer is allowed to request to the DO only with the corresponding PBKDO to get the secret key or data decryption key and Access Right (ASR). In case the CSP consumes time for finding the PBKDO, the user has to wait for a long time, which results in increasing the data accessing time. Therefore, the intended user must pay extra for the usage of cloud services. In [2], high system overhead is an additional issue because the DO has to be continuously online throughout the data communication process.

Recently, many researchers have proposed many schemes to improve the performance of a cloud environment [3-5]. Role based ACM is one of the popular ACMs, which is suggested by Ferraiolo and Kuhn [6]. Here, users access data based on the assigned job role, and all the tasks are controlled by using the job role. Role based ACM is vulnerable to many security attacks. The traditional Mandatory ACM is not flexible because the CSP manages all the aspects as per its wishes [7]. Therefore, this scheme is not suitable. Wan *et al*. [8] have proposed a novel encryption scheme in 2012 to improve the data security of a cloud environment, where users' ASRs are expired after a predefined time. Mian *et al*. [9] have proposed a Provisioning Data Analytic Workloads scheme for the cloud environment. However, this scheme does not consider the data security issue. Activity Based Access Control [10] model suggests giving user's ASR in an organization for any file or data based on the designation of the customer or user. But, the presence of

S. Namasudra is with the Department of Computer Science and Engineering, National Institute of Technology Patna, Bihar, India, PIN: 800005. E-mail: suyelnamasudra@gmail.com.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2020.3046471, IEEE Transactions on Services Computing

several components of this scheme escalates the overhead of system. In [11], an Index Generation based ACM is proposed by the substring index generation technique. The main aim of this scheme is to secretly share the data. Here, the time to generate the secret key is high. In 2017, a Cross Tenant Access Control model (CTAC) is proposed by Alam *et al*. [12]. In CTAC, the CSP acts as a trusted third party to generate the keys as well as manage many tasks. However, CTAC consumes time for data accessing. For a multitenant cloud environment, a novel technique is proposed in [13], where notion of compassion at the cloud data centre's end is introduced. This scheme is also slow for data accessing. Xie *et al*. [14] have developed a Modified Hierarchical Attribute-Based Encryption (MHABE) system for the cloud environment for data processing, data accessing and data storing. The data encryption time is high in MHABE. Zhao and Jiang [15] have proposed an ACM based on distributed machine learning for maintaining data integrity. Here, the two-step key generation technique and identity based cryptography are used for generating the DO's Public-Private Key Pair (PPKP). Some common issues of all these existing schemes are high time to search a DO, high system overhead, high time to data access and data security issue.

Data security is another critical issue of any cloud computing environment because of the existence of many attackers. The CSP must support a robust security model for securing user's confidential files or data as the attackers always wish to get unauthorized access of data, and they try to replace user's personal data. Nowadays, DNA based cryptography is one of the advanced fields to enhance information or data security. DNA based cryptography is mainly based on DNA computing in which DNA sequence, hardware and biochemistry are utilized to encode the genetic details in a personal computer. The concept of DNA was first used by Adleman [16] in 1994 for computation. The nitrogen bases (T, G, A and C) of human DNA are blended for forming a long DNA sequence. DNA based cryptography has received much attention to improve the security of a cloud environment because of DNA's complex structure. Clelland *et al*. [17] have proposed Hiding Messages in Microdots in which DNA computing was first used for hiding any data. Here, microdot is a technique to hide the data or message. But, this method is very slow. In 2005, a novel Public Key System is proposed using DNA computing and a one-way function [18]. This cryptosystem increases the data accessing time, and it is insecure against the password guessing attack. Chaos Based Image Encryption [19] scheme has been proposed by using the genetic algorithm, DNA masking and logistic map. This scheme does not support strong security. The Probabilistic Secret Sharing [20] model is proposed in 2016 by utilizing DNA-EXOR operation. Here, a cover image is used as the secret key and data are separated into three different secret shares. This scheme is insecure against the side channel attack. Wang *et al*. [21] have developed a reversible data hiding scheme based on DNA computing. This scheme fails to support a strong security technique to protect DO's file. In [22], a novel ACM is proposed in which complementary pair rule, decimal encoding rule, DNA encoding rule and American Standard Code for Information Interchange (ASCII) values are used for secure and efficient data accessing. However, DNA-EXOR operation is not considered in this scheme. So, it does not support much randomness. In addition, the data encryption time and data decryption time are not considered in this scheme. In 2018, a Coupled Map Lattice (CML) based ACM [23] was suggested by Wang *et al*. based on DNA sequence and coupled map lattice. Here, the SHA-256 algorithm and keys are amalgamate for generating the initial values of CML as well as other parameters. However, CML consumes time to encrypt any data. Double Layer Data Hiding (DLDH) method [24] is proposed based on the recombinant DNA technique, where any data is hidden in a random length fake DNA sequence. DLDH is secured against the password guessing attack. In [5], a novel DNA based data encryption scheme is proposed. The secret key generation process and data encryption process of this scheme are very simple. It does not support much randomness for not using DNA-EXOR operation. The data accessing issues are also not considered in this scheme. Reddy *et al.* [25] have proposed a bio-inspired cryptosystem based on DNA computing. There are three phases in this scheme: (i) key generation (ii) encryption, and (iii) decryption. Here, Central Dogma Molecular Biology is used for encryption and decryption. This scheme does not support fast data accessing.

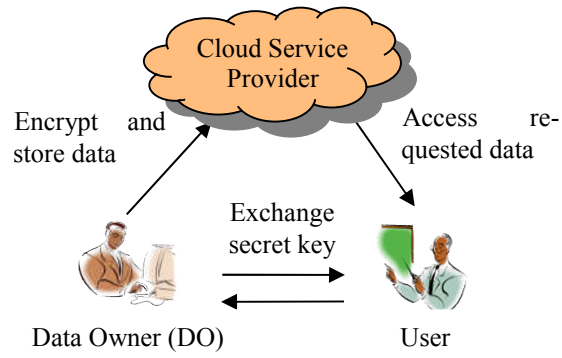

Fig. 1. System model of DNASF

To solve the above mentioned problems, in this paper, a novel ACM is proposed for the cloud environment, namely DNA based Secure and Fast (DNASF) ACM. A table is maintained in DNASF by the CSP on the basis of the data or file size of the data owner. Here, the CSP initiates a query in its table based on the desired size for finding the DO, when an end user sends a data access request. Thus, the service provider needs not to examine the complete database to get the details of the DO and easily can provide the PBKDO in a less time. Therefore, the time to search the PBKDO can be reduced, and in result, the data accessing time can also be repeatedly reduced and the customers can be charged less. A data encryption scheme is proposed in this paper to improve data security. Here, a random secret key or data decryption key (1024-bit) is generated by the data owner based on DNA computing and user's characteristics. Then, this generated key is utilized to encrypt the data. The DO provides the secret key and other credentials to the authorized users in an encrypted form. Hence, in DNASF, data or file security can be enhanced. The foremost contributions of this proposed work are:

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2020.3046471, IEEE Transactions on Services Computing

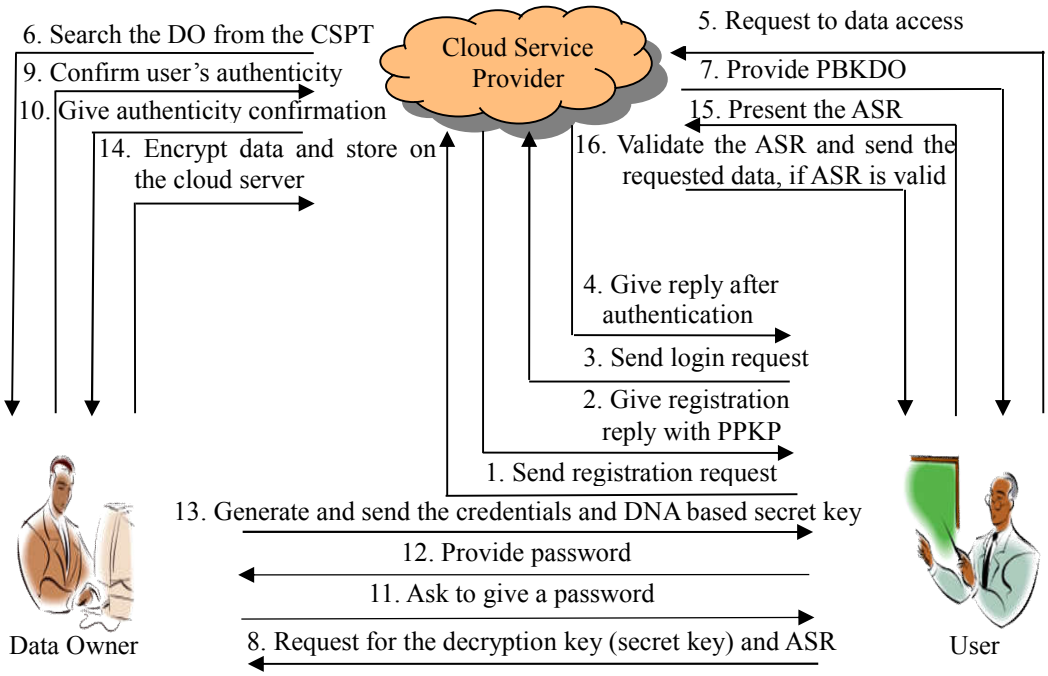

Fig. 2. Workflow of DNASF

1) A novel secure and fast ACM (DNASF) is proposed in this paper by using DNA computing and a table. The table can support to reduce the data accessing time and the searching time of data owner.
2) A data encryption scheme is proposed by using DNA computing in which a random 1024-bit DNA computing based Data Decryption Key (DNADK) is generated.
3) Security analysis and experimental results of DNASF are presented in this paper.

The rest of the paper is structured into several parts. Section 2 deals with the problem statements. The proposed scheme is discussed in detail in section 3. Section 4 represents the security analysis of the proposed scheme and the performance analysis is given in section 5. At last, conclusions and future works are presented in section 6.

2 Problem Statements

2.1 System Model

There are 3 entities in the proposed model to provide efficient and secure data accessing:

1) *CSP:* In a cloud environment, CSP is the overall administrator to support infrastructure and several cloud services by using abundant servers having much power as well as sufficient memory space.
2) *DO:* The DO is one special kind of user in a cloud environment. They store normal (ordinary) data as well as confidential data on the cloud database.
3) *User:* The authorized party or entity, who desires to get a service or data content from a cloud server is considered as a user. The system model of DNASF is depicted in Figure 1.

2.2 System Requirements

Few system requirements of the proposed scheme are as follows:

1) *Fine-grained data access control:* The service provider must support assurance that the authorized users are only permitted to get data from the cloud environment. However, the DO must give access rights to the users for data accessing, and the end users or customers are not permitted to access any data, when the respective ASR is not valid.
2) *Minimization of the data or file accessing time:* The users continually pay based on the usage of services in a cloud environment. Therefore, to minimize the data accessing time is always a top most requirement of all users.
3) *Security:* As any cloud environment works over the internet, attackers always wish to get unauthorized accesses of data. Therefore, the CSP must support a strong security technique for securing users' confidential files.
4) *Reduce the system overhead*: The DOs must be online through the complete data accessing process in the existing schemes that increases the system overhead. So, reducing the system overhead can be helpful to improve the performance of a cloud environment.

3 Proposed Method

In the proposed method i.e. DNASF, the service provider maintains a CSP Table (CSPT) with 6 attributes: Data Size (Sz), Group Identity (GP_ID), DO's ID (DO_ID), Popularity Value (PV), GP_ID's Time & Date (GP_T&D)) and DO's Time & Date (DO_T&D). Sz field keeps the sort of data size of a particular GP_ID, which is determined by the CSP. GP_ID maintains a unique number of a group. The details of

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2020.3046471, IEEE Transactions on Services Computing

Algorithm: Generate a secret key by using DNA computing

Input: User's attributes and a password
Output: 1024-bit data decryption key or secret key

```
if UsID == Authorized user's ID
    COLLECT all the user's information
    ASK to give a password
    APPLY decimal encoding rule
    CHANGE into 8-bit binary
    ADD MAC address and password in the middle
    SEPARATE into four equivalent parts
    EXECUTE EXOR between the first two parts
    EXECUTE EXOR between the last two parts
    EXECUTE EXOR between the resultant of EXORs
    MAKE the binary number of 1024-bit
    SEPARATE into four equivalent parts
    ASSIGN a random DNA base to each part
    APPLY complementary pair rule
    SEPARATE into two equivalent parts
    EXECUTE DNA-EXOR between two parts
else
    STOP
```

Algorithm: Insert a DO (ID) in the CSPT

Input: DO's ID
Output: Updated CSPT

```
if DOID == Authorized DO's ID
  a0 ← X, a1 ← Y, n[ ]
    if a0 ≠ FULL
        for all GID_x ∈ T
            if DOID_Sz == GID_xSz
                for all DOID_x ∈ U in GID_x
                    if a1 ≠ FULL
                        GID_xDO_ID ← GID_xDO_ID ∪ DOID
                        DOID_PV ← 1
                    else
                        DEL_DO_ID (GP_ID)
                        GID_xDO_ID ← GID_xDO_ID ∪ DOID
                        DOID_PV ← 1
                end for
        end for
    else
        m ← DEL_GP_ID()
        GID_mDO_ID ← DOID
        GID_mSz ← DOID_Sz
        GID_mPV ← 1
        GID_mDO_T&D ← DOID_T&D
        GID_mGP_T&D ← DOID_T&D
        UPDATE CSPT
else
    STOP
```

the DOs with the same data size are saved in a DO_ID. PV holds the popularity value of each DO up to a threshold value. In DNASF, threshold value of PV is 10. GP_T&D keeps the formation information of a GP_ID. The data provided details (time and data) of a DO are maintained in DO_T&D.

A random 1024-bit DNADK is used in DNASF for encrypting the user's sensitive file or data. In the proposed DNASF, the DO uses attributes of the end users, such as Date of Birth (DOB), MAC address, etc. and a password to make the DNADK. DNASF has 6 phases: (i) registration (ii) system setup (iii) login (iv) data encryption based on DNA computing (v) processes on the CSPT and (vi) data access. The entire proposed scheme has been discussed in this section.

3.1 System Setup

In this phase, at first, the service provider computes a multiplicative group (Z_p^*) by selecting a big prime number (p). The CSP chooses PPKP of all the entities from Z_p^*. The PPKP of the users and DOs are provided at the time of the registration. In DNASF, the User's Public Key (PBKUSR) and CSP's Public Key (PBKCSP) can be accessed by the authorized parties, and the PBKDO is provided only to the respective users. Figure 2 shows the workflow of DNASF.

3.2 User Registration

All the users must be registered at the CSP by sending registration invitations to the CSP. After getting the registration invitation, the CSP uses some secret details of the users like MAC address, DOB, etc. to make a user's profile. Then, the PPKP and a reply is given to the particular user by establishing a Secure Socket Layer.

3.3 User Login

The registered users are only allowed for logging in the server (CSP) after a successful registration phase. After log-in in the cloud server, the user can request to data access.

TABLE 1
DECIMAL ENCODING RULE

Alphabet / Special symbol	Decimal number	Alphabet / Special symbol	Decimal number
Blankspace	01	A	25
`	02	B	26
.	.	.	.
.	.	.	.
~	24	Z	50

3.4 Data Encryption based on DNA Computing

In this phase, the DO creates a random 1024-bit DNADK based on DNA computing. Then, this DNADK is used by the DO to encrypt file or data.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2020.3046471, IEEE Transactions on Services Computing

TABLE 2
DNA-EXOR RULE

DNA-EXOR	A	C	T	G
A	A	C	T	G
C	C	A	G	T
T	T	G	A	C
G	G	T	C	A

TABLE 3
ASCII VALUE CONVERSION RULE

Binary number	ASCII
00000000	13
00000001	110
.	.
.	.
11111111	104

TABLE 4
DNA ENCODING RULE

2-bit binary number			
00	**01**	**10**	**11**
G	C	T	A
G	A	T	C
.	.	.	.
.	.	.	.
A	G	T	C

3.4.1 Key generation based on DNA computing

In the key generation phase, the DO first verifies users' authenticities, when s/he receives requests from the end users to give the DNADK and ASR. The DO gathers corresponding secret details of the users like MAC address, DOB and first teacher name, and asks the end users to give a password. Then, the DO organizes all the collected information in a sequential manner to apply decimal encoding rule as per Table 1. The DO converts them into respective 8-bit binary numbers. Then, binary values of MAC address and password are added in the middle and separates the entire binary numbers into four equal parts. The DO then executes 3 EXOR operations. Here, the first EXOR operation is executed between the first and second parts of the binary numbers, and then, the second EXOR operation is performed between the third and fourth parts of the binary numbers. Finally, the last EXOR operation is performed between the results of the first and second EXOR operations. The DO deletes some binary digits as per his/her wishes to make it 1024-bit. In the next phase or step, the resultant binary number is divided into 4 equal size parts to randomly assign the DNA bases to each part, which means there are 256-bit in each part. The DO uses the complementary pair rule on the resultant DNA sequence, which can be signified as $W(V) = C$, where $V \neq C$. The following complementary pair rule is used in DNASF:

(AC) (TG) (CA) (GT) i.e. $W(A) = C$ and so on.

The DO then separates the resultant DNA sequence or bases into two equivalent parts and performs DNA-EXOR process amid both the equivalent parts using Table 2, which is randomly generated by DO's wish. DO can assign any rule for DNA-EXOR operation. A more complex and random DNA sequence is generated by using the DNA-EXOR rule. The final resultant complex DNA bases are sent to the users as DNADK, which is encrypted using the DO's Private Key (PTKDO) and PBKUSR. The DO also provides other Credentials (CDL), namely Table 1, Table 2, Table 3, Table 4, ASR and complementary pair rule with the same message.

3.4.2 Data encryption

To encrypt any data, the DO executes some steps. At first, the plaintext of the data is separated into several equivalent blocks with 1024-bit each. Zero is added to the last block (right side), if it lacks 1024-bit. The DO then performs EXOR operation between 1024-bit DNADK and 1024-bit block (plaintext). Table 3 is used to convert 8-bit binary number into corresponding ASCII values to make the ciphertext complex as Table 3 is randomly generated in which 13 or 31 or any other value can be the ASCII value of 00000000. These ASCII values are then changed into their actual binary numbers, which are (each 2-bit) then again changed into the DNA bases using a random Table 4. DNA bases are rotated in the next step, which is followed by applying the same complementary pair rule used in subsection 3.4.1. Then, the owner of the data divides the DNA sequence or DNA base into 2 equivalent parts and executes DNA-EXOR operation by Table 2 amid both the equal parts.

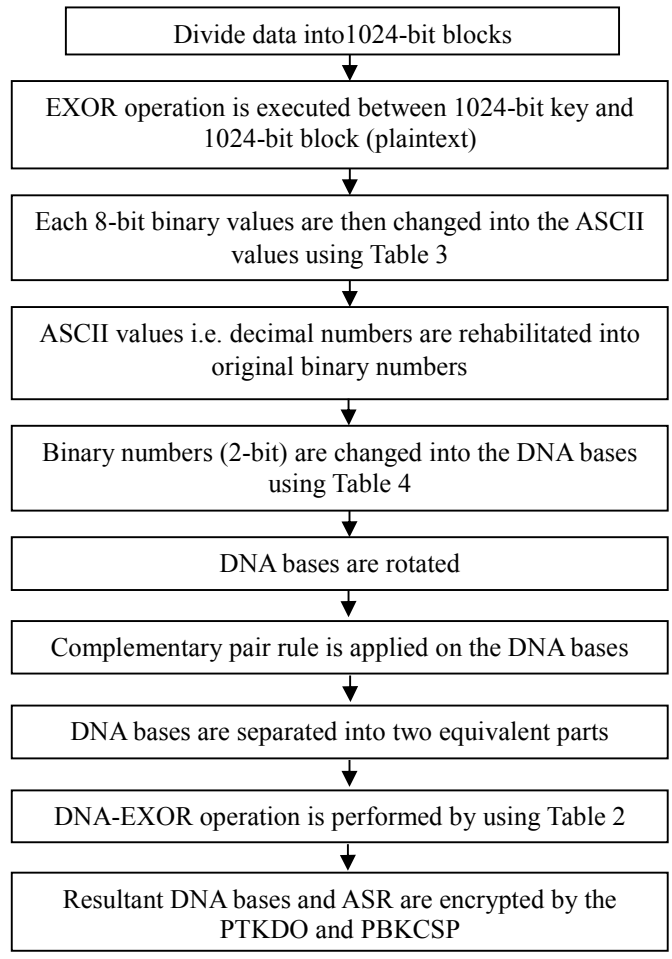

Fig. 3. Data encryption based on DNA computing

Finally, the DO utilizes the PTKDO and PBKCSP for encrypting the resultant DNA bases and respective ASR, and sends the entire encrypted message or file to the CSP. Upon receiving the message, the CSP uses its (CSP) Private Key (PTKCSP) and PBKDO to incompletely decrypt the encrypted message and saves it on cloud database. Figure 3 shows steps to data encrypt based on DNA computing.

3.5 Processes on the CSPT

The concept of Least Recently Used (LRU) is used in CSPT

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2020.3046471, IEEE Transactions on Services Computing

TABLE 5
CSP TABLE (CSPT)

GP_ID	DO_ID	PV	Sz	GP_T&D	DO_T&D
1	113	9	$2^{-33} < Sz \leq 2^{-30}$	14:11 20-11-2012	15:00 01-06-2019
	016	9			09:14 29-12-2018
	102	7			07:10 15-09-2016
	166	3			12:54 30-07-2014
	592	1			14:11 20-11-2012
2	541	10	$2^{-30} < Sz \leq 2^{-10}$	12:18 09-08-2013	11:59 24-12-2017
	342	9			10:31 13-05-2015
	864	7			17:11 22-01-2015
	019	6			21:52 11-10-2014
	450	2			12:18 09-08-2013
.	.		.	.	.
	.	.			.
	.	.			.
	.	.			.
	.	.			.
50	760	8	$1 < Sz \leq 3$	11:32 11-10-2017	19:01 22-04-2020
	399	7			17:41 30-11-2018
	343	7			10:42 09-11-2018
	234	5			21:57 26-05-2018
	456	4			11:32 11-10-2017

TABLE 6
NOTATIONS AND DESCRIPTIONS

Notation	Description	Notation	Description	Notation	Description
$a0$, $a1$, m, i, j	Variable	$DOID_{Sz}$	Requested data size of DOID	$GID_{x_{Sz}}$	Data size of x^{th} GP_ID
$GID_{x_{DO\_ID}}$	DO_ID of x^{th} GP_ID	$DOID_{PV}$	PV of authorized DO	$GID_{m_{DO\_ID}}$	DO_ID of m^{th} GP_ID
$GID_{m_{Sz}}$	Data size of m^{th} GP_ID	$GID_{m_{PV}}$	PV of m^{th} GP_ID	$GID_{m_{DO\_T\&D}}$	DO_T&D of m^{th} GP_ID
$DOID_{T\&D}$	T&D of authorized DO	$GID_{m_{GP\_T\&D}}$	GP_T&D of m^{th} GP_ID	$cou\ n(n)$	A function for computing DOs' IDs in $n[\,]$
$SEARCH_{LRU}\ (DOID_x)$	Search $DOID_x$ based on LRU	DO_ID_{NEW}	DO_ID after processing	DO_ID_{PREV}	DO_ID before processing
$SEARCH_{LRU}\ (GID_x)$	Search GID_x based on LRU	$UsID_{Sz}$	User's requested data size	$DOID_{x_{PV}}$	PV of x^{th} DO

Algorithm: Delete a DO's ID (*DEL_DO_ID* (*GP_ID*))

Input: DO's ID
Output: Updated CSPT

1 $n[\,] \leftarrow SEARCH_{minimum}(DOID_{x_{PV}})$
2 if $cou\ n(n) > 1$
3 $SEARCH_{LRU}\ (DOID_x)$
4 $DO\_ID_{NEW} \leftarrow DO\_ID_{PREV} - DOID_x$
5 else
6 $DO\_ID_{NEW} \leftarrow DO\_ID_{PREV} - DOID_x$
7 UPDATE CSPT
8 STOP

Algorithm: Delete an entire group (*DEL_GP_ID*())

Input: DO's ID
Output: Updated CSPT

1 $SEARCH_{LRU}\ (GID_x)$
2 DELETE entries of GID_x
3 UPDATE CSPT
4 STOP

Algorithm: Search a PBKDO

Input: User's ID
Output: Provide the PBKDO to the user

```
if UsID == ID of an authorized user
    for i = 1 to X
        if UsID_Sz == GID_{x_Sz}
            for j = 1 to Y
                SEARCH data owner from GID_{x_DO_ID}
                PROVIDE PBKDO to UsID
                if DOID_PV ≠ Threshold
                    DOID_PV = DOID_PV + 1
            end for
        else
            Show INVALID data type
    end for
else
    STOP
```

to place the DOs' IDs. Here, when a DO delivers the requested data, its ID is maintained at the top of a corresponding group. The CSPT consists of 3 operations: (i) insertion of a DO's identity (ii) deletion, and (iii) searching a PBKDO. In the proposed scheme, DO_ID and GP_ID are defined by 2 sets, $U = \{DOID_1, DOID_2, DOID_3, \dots, DOID_Y\}$ and $T = \{GID_1, GID_2, GID_3, \dots, GID_X\}$, respectively. Here, $DOID_x$ and GID_x signify x^{th} DO's ID and x^{th} GP_ID, respectively. In the CSPT, Y and X signify the highest number of DOs' IDs in each DO_ID and GP_ID, respectively. An illustration of CSPT is given in Table 5.

3.5.1 Insertion

To insert a DO's ID in the CSPT, the authenticity of the DO is tested by the CSP after receiving a data sharing request. Then, when the data size exists in the CSPT and the particular DO_ID is not completed i.e. not full, the service provider adds the DO at the corresponding place of the CSPT. Otherwise, if the CSPT is not completed and the data size is not matched, a different GP_ID is added in the CSPT with all the information of DO. Notations along with the descriptions used in this paper are presented in Table 6

3.5.2 Deletion

The CSP can remove an entire GP_ID or identity of a DO from the CSPT based on the requirements. As both the deletion algorithms are parts of the insertion algorithm, so ID of an authorized DO is the input of both algorithms.

1) DO's identity deletion (***DEL_DO_ID*** (***GP_ID***)) algorithm: The identity of a DO is removed from DO_ID after searching GP_ID as well its respective DO_ID. Here, LRU DO's ID is searched and assigned to n, and at last, it is removed from DO_ID. $SEARCH_{minimum}(DOID_{x_{PV}})$ means searching a $DOID_x$ with minimum PV.
2) Entire GP_ID deletion (***DEL_GP_ID***()) algorithm: The LRU concept is used to remove a long time unused GP_ID from the CSPT. Here, the CSP searches the LRU GP_ID from the CSPT, and then, all the entries are removed.

3.5.3 Searching a PBKDO

In this operation, the CSP first checks user's authenticity prior to provide the PBKDO. Here, the CSP matches the $UsID_{Sz}$ with the GP_IDx_{Sz}. Then, the CSP searches the PBKDO from the CSPT, and delivers the PBKDO to the end user in the encrypted form using the PTKCSP and PBKUSR. Otherwise, the CSP indicates an unsound or invalid data size and executes the normal process to search the PBKDO. After providing the PBKDO, if the PV of the DO has not reached the threshold value, it is increased by 1.

3.6 Data Access

The following steps are executed among the CSP, DO and user for accessing any requested data from the CSP:

Step 1: At first, the end users must be registered at the cloud server by registration requests.
Step 2: The information or details of the users are brought together by the CSP. Then, the CSP sends a registration acknowledgement to the user along with the PPKP, if the registration phase is successful.
Step 3: In the third step, the users try to login in the cloud server.
Step 4: The CSP accepts the login requests and sends acknowledgements, if the users are valid ones.
Step 5: In the fifth step, the users send data or file access appeals to the CSP encrypted by the User's Private Key (PTKUSR) and PBKCSP.
Step 6: Here, the CSP queries the particular PBKDO from the CSPT.
Step 7: In the seventh step, the CSP provides the PBKDO to the end user encrypted by using the PTKCSP and PBKUSR. The user retrieves the PBKDO, when decrypts it by the PTKUSR and PBKCSP.
Step 8: The users use the PBKDO to request the corresponding DOs for DNADK and ASR. The requests are encrypted by the PTKUSR and PBKDO.
Step 9: In the ninth step, the DOs verify users' authenticities

at the CSP's end.

Step 10: Here, the CSP provides the authenticity confirmations to the DOs.

Step 11: In this step, the data owner appeals to the users to deliver a password, which is used to generate the DNADK.

Step 12: In the twelfth step, the users provide passwords to the DOs, which is encrypted by the PTKUSR and PBKDO.

Step 13: The random 1024-bit DNADKs are generated by the DOs by some secret details and passwords of the users. Then these are sent to the users as encrypted messages by the PTKDO and PBKUSR. The DOs also send CDL in the same encrypted messages. The users decrypt the messages using the corresponding PTKUSR and PBKDO.

Step 14: In the fourteenth step, the DOs use the DNADKs to encrypt the requested data before storing the corresponding ASR and ciphertext on the cloud database encrypted using the PTKDO and PBKCSP. The CSP partly decrypts the message using the PTKCSP and PBKDO.

Step 15: Here, the users send the ASRs to the CSP encrypted using the PTKUSR and PBKCSP. The CSP uses the PTKCSP and PBKUSR for decryption and retrieves the ASRs.

Step 16: This is the final step in which the CSP finds the demanded data or file from the cloud server. The ASRs of the users are then verified with the corresponding ASRs provided by the DOs. If these are validated, the CSP sends the ciphertext of the requested data encrypting by the PTKCSP and PBKUSR. If not, the data or file accessing requests are rejected.

The users decrypt the ciphertext using the PTKUSR and PBKCSP. At last, the users utilize the DNADKs and CDL to retrieve the original data.

4 Security Analysis

DNASF can resist many security attacks. In this section, security analysis of DNASF is presented.

4.1 Password Guessing Attack

In this attack, malicious users attempt all probable combinations of a password to achieve a truthful password. In DNASF, the DO creates the data decryption key based on a random password given by the user, decimal encoding rule, secret details of the user, complementary pair rule and DNA-EXOR operation. In the data encryption phase, DNA-EXOR rule, ASCII value, DNA encoding rule and complementary pair rule are used by the DO. Here, all the rules i.e. tables are randomly generated by the DO's own wish to enhance data or file security. The DO is able to use different tables for the same file. It is very difficult for the attackers to guess all the secret details of a user. In addition, CDL are sent in the encrypted method. The authorized corresponding customer or user is only allowed to get the CDL after decryption. Hence, DNASF can resist this attack.

4.2 Distributed Denial of Service (DDoS) Attack

In a DDoS attack, attackers or malicious users sent a number of requests to the CSP from different geographical regions. The main reason to send these requests is to make some cloud services unreachable for the valid end users. In DNASF, the owner of the data randomly creates an elongated DNADK and utilizes the same key i.e. DNADK for encryption purpose. In both the data encryption and key generation phases, the DO is allowed to arbitrarily assign the DNA bases to enhance data security. In addition, the DO also makes use of the PTKDO and PBKCSP to convert the data into its ciphertext. All the entities except the intended user do not aware of the DNADK and CDL. If a DDoS attack is occurred by numerous attackers, they also must have CDL and DNADK to get the original data. The CDL are only sent to the authorized end users or customers after making it ciphertext, so there is no data security concern in DNASF. Therefore, DNASF is secured against the DDoS attack.

4.3 Masquerade Attack

In this attack, unauthorized users use some fake information or identity for getting unauthorized access. In DNASF, all end users must be registered at the cloud server's end, and they have to login into the server to access any cloud service, when a registration phase is completed. Thus, unauthorized or malicious users are not allowed to login into the CSP using a forged identity, and they cannot get any cloud services without passing the login phase. If a hacker logs into the server in the worst case and demands to access any data, the CSP sends the PBKDO to obtain the DNADK and ASR from the corresponding DO. Prior to generate and provide the DNADK and ASR, the DO verifies users' authenticities from the CSP. In this case, if the malicious user uses a forged identity for data accessing, then, the authenticity of the malicious user is not proved at the CSP's end. So, the DO declines the request for providing the DNADK and ASR. Thus, a malicious user or hacker is failed for accessing any unauthorized data using the forged identity, and DNASF can resist the masquerade attack.

4.4 Stolen Verifier Attack

In this attack, malicious users attempt to acquire a hashed password. They then recuperate the truthful key from the hashed password. After achieving the password, the attackers access data from the CSP. Here, in the proposed DNASF, the DO first verifies users' authenticities from the CSP to create the DNADK based on the random passwords provided by the users and other secret details. The DO encrypts the DNADK using the PTKDO and PBKUSR before sending it to the user. The particular end user uses the PTKUSR and PBKDO for decrypting the encrypted message and gets the DNADK. The DNADK is not saved on the cloud database or anywhere, and the other entities do not receive the DNADK. Therefore, DNASF is not vulnerable to this stolen verifier attack.

4.5 Phishing Attack

In a phishing attack, malicious users obtain confidential details of the authorized user, such as voter ID number, user ID, first school name, etc. Then, the attackers make use of these confidential information for accessing an unauthorized cloud service. In DNASF, the CSP assembles some necessary secret details of the user, when a request for registration is initiated by the user. The service provider then delivers

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2020.3046471, IEEE Transactions on Services Computing

the PPKP. The same processes are followed to register a DO. The service provider (CSP) solitary sends the PBKDO of the particular DO to the end user in the ciphertext form, when the end user initiates a data access request, and the DO always confirms the user's authenticity prior to deliver the DNADK. The DO and CSP do not disclose any secret details about themselves to even valid or authorized users. So, malicious users are unable to obtain details about the authorized users, and thus, DNASF resists the phishing attack.

5 Performance Analysis

This section deals with the performance analysis of DNASF in detail.

5.1 Experimental Setup

A simulation environment is constructed for evaluating the efficiency of the proposed model. To prove the efficiency, CloudSim 4.0 framework [26] is used for developing a cloud environment (simulation) on a Dell OptiPlex 7070 Tower desktop with the following configurations:

1) RAM: 16GB
2) Storage capacity: 2TB
3) Processor: 3.0GHz Intel Core i7 9700 eight core
4) Operating system: Windows 10

On the above mentioned desktop, Java version 8 [27] is installed and Apache Commons Math 3.6.1 [28] is fixed up with the CloudSim framework to implement mathematical and statistical components. Hundred datacenters are used to set up a heterogeneous cloud environment with a memory capacity of 4GB and 3000 physical nodes. In the simulation, the following 4 different VMs (VM1, VM2, VM3 and VM4) are used, and each VM has 1GB/s bandwidth:

1) 2500MIPS, 2.5GB (VM1)
2) 3000MIPS, 3GB (VM2)
3) 3500MIPS, 3.5GB (VM3)
4) 4000MIPS, 4GB (VM4)

1000 cloudlets with ID i.e. cloudletId from 1 to 1000 are considered in the simulations. CloudletId is used to distinguish a cloudlet from a submitted list of cloudlets. Each cloudlet has a length i.e. cloudletLength varying from 1000MI to 10000MI. The total size of input data i.e. cloudletFileSize and the total size of output data i.e. cloudletOutputSize is 265631 (byte). All cloudlets use the CPU cores of all VMs. Time-shared policy (CloudetSchedulerTimeShared) is used for sharing processing power among the cloudlets in VMs. Cloudlet is submitted to the datacenter broker module and a VM with optimal configuration is assigned for each cloudlet.

5.2 Results and Discussions

The proposed scheme i.e. DNASF is compared with 3 existing schemes: (i) CML [23] (ii) CTAC [12] and (iii) MHABE [14]. 200 experiments are performed in different situations for getting the searching time of PBKDO, key generation time, data decryption time, data encryption time, key retrieval time, data accessing time and CPU utilization. Then, average values are considered out of all 200 results. In the experiments, the weather dataset of CityPulse Dataset Collection [29] is used.

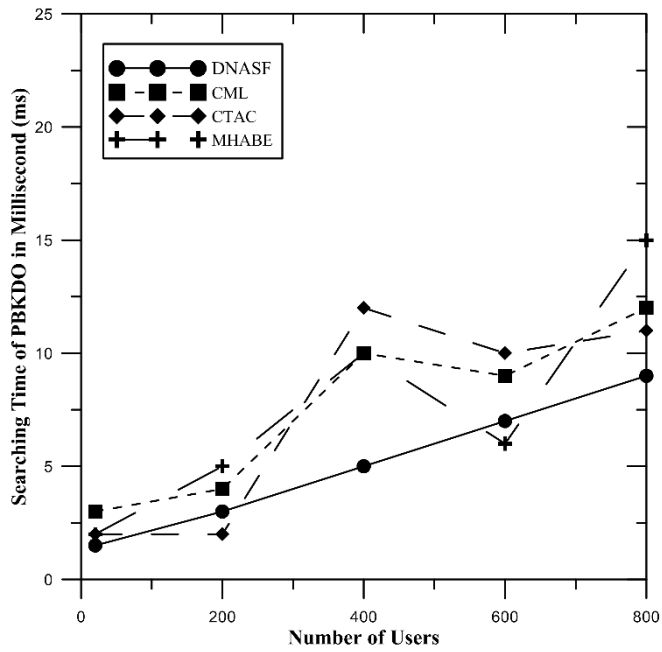

Fig. 4. Searching time of PBKDO

Figure 4 shows the results of the first simulation to find a PBKDO from the server. DNASF offers improved result than CML, CTAC and MHABE schemes to provide the PBKDO to the end users. Upon receiving data sharing demands from the users, in DNASF, the CSP obtains the DO's details from the CSPT by using the demanded file or data size. So, the exact position of the DO in the CSPT is searched by the CSP without searching the entire database. When the number of end users increases exponentially, the CSP takes time to obtain the details of DO i.e. PBKDO due to the increases of load on the server's side, which resulted a linearly growing curve for the proposed DNASF. In CML, CTAC and MHABE, the CSP does not use any unique manner or a table like CSPT to maintain the information of all DOs, which can reduce the time to search the DOs. The CSP sometimes obtains the details in a reduced amount of time. However, sometimes, it takes a large amount of time due to searching the entire database. Thus, zigzag curves are experienced.

The result to compute the data decryption key generation time is displayed in Figure 5. In DNASF, the DO makes use of CDL i.e. Table 1, complementary pair rule and Table 2 to create the DNADK, which takes time in DNASF. In any cloud server, two types of users are presented: (i) new users and (ii) old users. The new users are those types of users, who just login into the cloud server for data accessing for the first time, and the old are those types of users, who already have retrieved file or data from the server. For all the new users of DNASF, the DO creates the DNADK. So, the time to create the data decryption key is high. In the case of old users, the DO needs not to create the DNADK as the old users already have the data decryption key. This key can be easily used by the users for the subsequent data accessing from the same cloud server. In a cloud environment, both

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2020.3046471, IEEE Transactions on Services Computing

types of users periodically access data. Therefore, it is hard to guess prior the type of an end user, which results to generate a zigzag curve for DNASF. In the existing considered schemes, both types of users need to get the DNADK from the DO. Furthermore, these schemes are more complex compare to DNASF. Therefore, the DO has to create the data decryption key for all users and linearly growing curves are experienced.

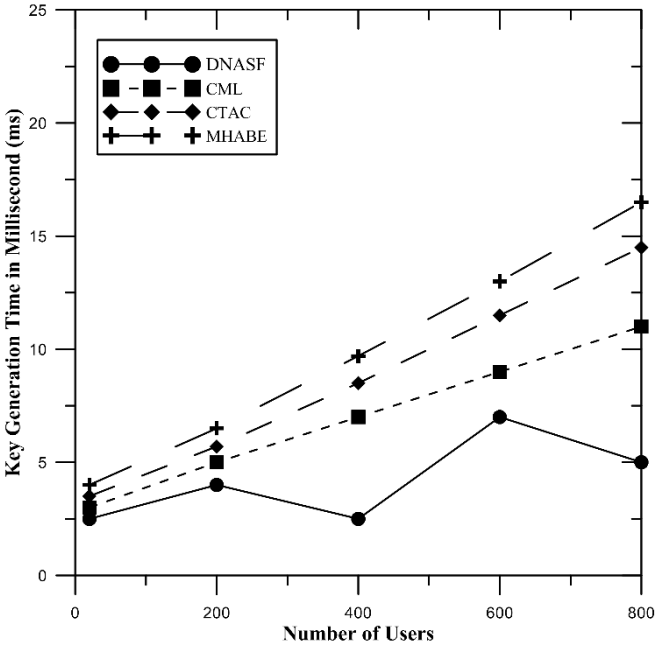

Fig. 5. Key generation time

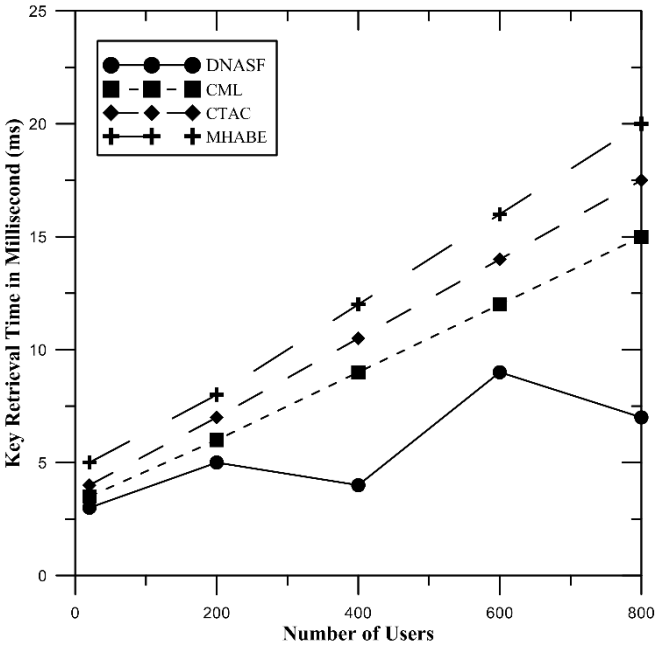

Fig. 6. Key retrieval time

The next experiment is executed for verifying the data decryption key retrieval time. It can be simply understandable from Figure 6 that the retrieval time of DNADK in DNASF is better than CML, CTAC and MHABE. In DNASF, upon receiving a file or data providing request from an end user, if the CSP finds that the user already has accessed the data, the CSP does not provide the PBKDO to the user. As the user does not request for the DNADK, the DO needs not to generate it, and in return, the user does not regain the data decryption key. Therefore, in DNASF, the time to key retrieve is reduced. However, when a user demands the DNADK in the existing schemes, the DO has to generate it. When the DO provides the DNADK to the user, for each data communication process, all the end users or customers must retrieve the DNADK. Hence, the data decryption key retrieval time is more for CML, CTAC and MHABE.

Figure 7 illustrates the next result to compute the data encrypting time. The DO uses the necessary CDL in DNASF for encrypting the requested data or file. There are 3 EXOR and 1 DNA-EXOR processes in the suggested secret key generation phase, which consume time to encrypt the data. In CML, the coupled map lattice and DNA sequence or strand are utilized to encrypt any data. Here, a user must provide a key, which is integrated with the SHA-256 algorithm, and in result, the time to data encrypt is enlarged. In CTAC, the service provider controls the entire data accessing process, and CTAC performs several operations for encrypting any requested data of the user. In MHABE, the hierarchical identity-based encryption and attribute-based encryption are combined to achieve a fast and efficient ACM for the cloud environment. MHABE is ample complex with the presence of many EXOR operations due to combining both schemes. So, in the existing schemes, the time to encrypt any data is more than DNASF.

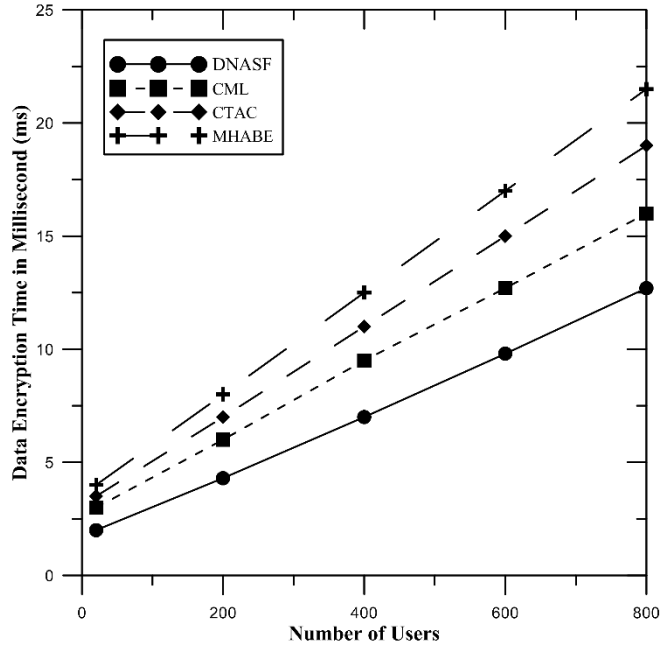

Fig. 7. Data encryption time

Figure 8 depicts the experimental results to compute the data decryption time. When an end user sends a request to the DO for the DNADK and ASR, the DO first verifies user's authenticity. The DNADK and CDL are only provided to the user in the encrypted form, if user's authenticity is proved. The intended user decrypts the encrypted message by using the PTKUSR and PBKDO to obtain the CDL and DNADK. The end user then can effortlessly retrieve the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2020.3046471, IEEE Transactions on Services Computing

original requested data or file in a less time because there is no difficult or complex process in DNASF. One EXOR and one DNA_EXOR operations are executed during data decryption phase, which reduce the time of data decryption in DNASF. CML, CTAC and MHABE contain many EXOR processes to provide a data to the users, and theses existing methods are more complex than DNASF due to the occurrence of multifaceted operations. Therefore, the time to decrypt any data is high in CML, CTAC and MHABE.

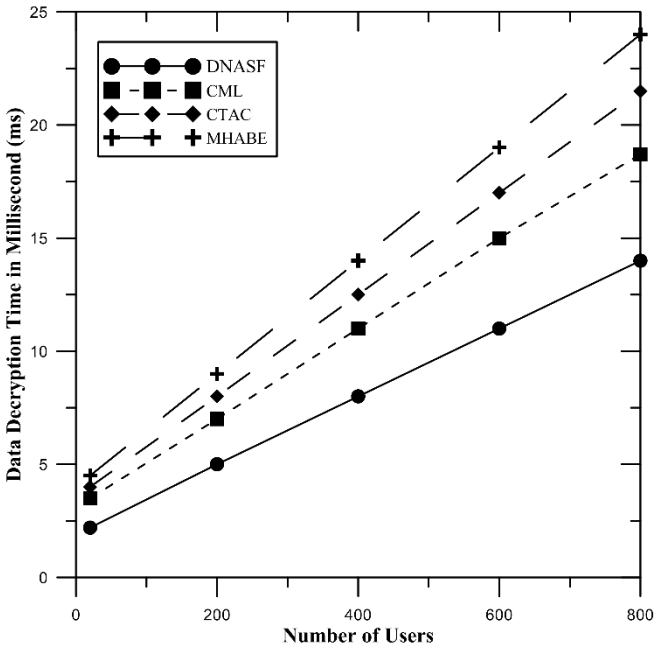

Fig. 8. Data decryption time

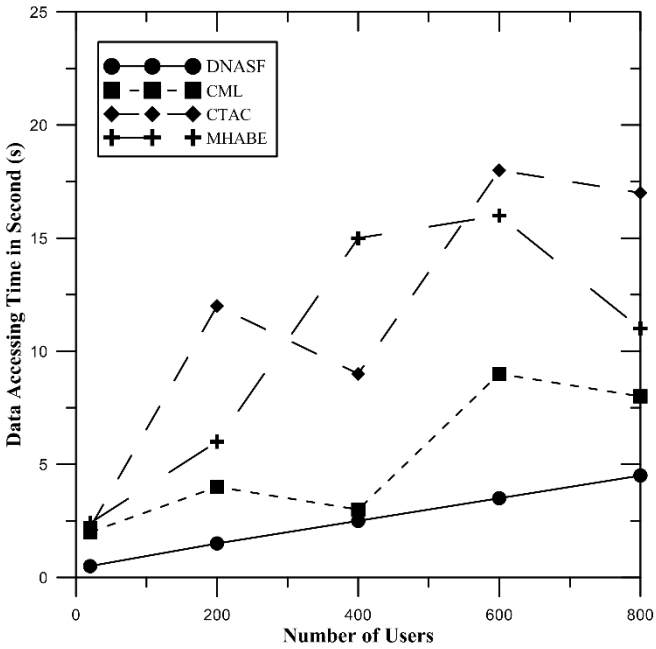

Fig. 9. Data accessing time

The next experiment is executed to confirm the time for data access. The CSP can simply find the PBKDO from the CSPT in DNASF in a less time, which results in the minimization of the waiting time of a user to get the required PBKDO. The users can use the corresponding PBKDO to get the data decryption key and ASR. The accessing time of data is repeatedly less in DNASF as the key retrieval time, key generation time, data decryption time and data encryption time are less. Hence, a linearly growing curve in Figure 9 is generated. There are four main algorithms in CTAC: (i) activation (ii) delegation (iii) backward revocation and (iv) forward revocation, which escalate the time to data access. In CML, CTAC and MHABE, the CSP consumes time to search the PBKDO because of searching the entire database. In addition, the key generation time, data encryption time, key retrieval time and data decryption time are also high. Therefore, the time for data access is ample high. Here, data accessing time is calculated as the time interval between the data access request time and data decryption time.

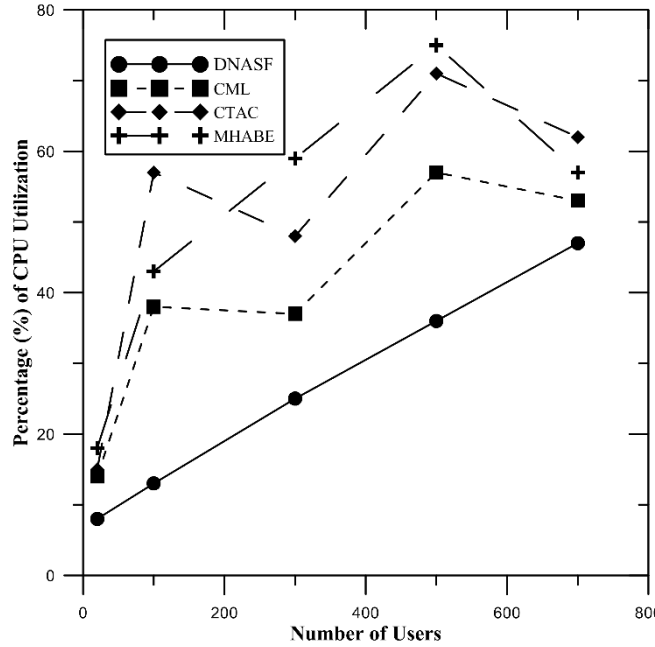

Fig. 10. CPU utilization

The last simulation process is for calculating CPU utilization. In Figure 10, it can be easily seen that the utilization of CPU of the proposed DNASF is linearly increased with the number of users. The number of data accessing requests is also increased, when the number of user is increased. As the data accessing time is less in the proposed DNASF, the cloud service provider does not take a large amount of time to provide the service. Therefore, the utilization of CPU is less. However, the CSP takes an abundant time for responding to the users in the existing schemes because the accessing time of data is high. The CPU utilization has been calculated per 5 second that implies 12 values per minutes. Finally, the average value has been taken out of all these values. The following Eq. (1) has been utilized to compute CPU utilization:

$$U_n = \frac{\sum_{i=1}^{n} U_i(n)}{n} \quad (1)$$

where, n is the complete utilization of CPU for each experiment. So, $n \in 1,2,3,\ldots$ and $\sum_{i=1}^{n} U_i(n)$ is the total CPU usage.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TSC.2020.3046471, IEEE Transactions on Services Computing

6 Conclusions and Future Works

In this paper, a novel access control model is proposed for the cloud computing environment by using a table and DNA computing for secure and efficient data accessing. In the proposed technique, a 1024-bit secret key is created based on the user's attributes to improve data security, and the same 1024-bit key is used to encrypt the data in the proposed DNA based data encryption phase. The proposed scheme is secured against some security attacks, namely password guessing attack, DDoS attack, masquerade attack, stolen verifier attack and phishing attack by supporting randomness in the data encryption and secret key generation phase. Here, the system overhead is reduced as the data owners can go offline, when s/he provides the data decryption key to the user. Many experiments are executed to evaluate the performance of the proposed scheme, which show its efficiency. The identity management technique of a cloud environment can be improved in the future.

References

[1] A. Huthand J. Chebula, "The basics of cloud computing," Carnegie Mellon University, 2011.

[2] X. Gao, Z. Jiang, and R. Jiang, "A novel data access scheme in cloud computing," in *Proceedings of the 2nd International Conference on Computer and Information Applications*, 2012, pp. 124-127.

[3] S. Namasudra, "An improved attribute based encryption technique towards the data security in cloud computing," *Concurrency and Computation: Practice and Exercise*, vol. 33, no. 3, 2019.

[4] S. Namasudra, "Taxonomy of DNA-based security models," in *Advances of DNA Computing in Cryptography*, S. Namasudra and G.C. Deka, Eds., Taylor & Francis, pp. 53-68, 2018.

[5] S. Namasudra *et al.*, "Towards DNA based data security in the cloud computing environment," *Computer Communications*, vol. 151, pp. 539-547, 2020.

[6] D. F. Ferraiolo and D. R. Kuhn, "Role-based access controls," in *Proceedings of the 15th National Computer Security Conference*, Baltimore, USA, 1992, pp. 554-563.

[7] R. S. Sandhu and P. Samarati, "Access control: Principles and practice," *IEEE Communications Magazine*, 1994.

[8] Z. Wan, J. Liu, and R. H. Deng, "HASBE: a hierarchical attribute-based solution for flexible and scalable access control in cloud computing," *IEEE Transactions on Information Forensics and Security,* vol. 7, no. 2, pp. 743-754, 2012.

[9] R. Mian, P. Martin, and J. L. V. Poletti, "Provisioning data analytic workloads in a cloud," *Future Generation Computer Systems*, vol. 29, no. 6, pp. 1452-1458, 2013.

[10] S. Ajgaonkar, H. Indalkar, and J. Jeswani, "Activity based access control model for cloud computing," *International Journal of Current Engineering and Technology*, vol. 5, no.2, pp. 708-713, 2015.

[11] S. Raghavendra *et al*., "Index generation and secure multi-user access control over an encrypted cloud data," *Procedia Computer Science*, vol. 89, pp. 293-300, 2016.

[12] Q. Alam *et al*., "A cross tenant access control (CTAC) model for cloud computing: formal specification and verification," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 6, pp. 1259-1268, 2017.

[13] A. Almutairi, M. I. Sarfraz, and A. Ghafoo, "Risk-Aware Management of Virtual Resources in Access Controlled Service-Oriented Cloud Datacenters," *IEEE Transactions on Cloud Computing*, vol. 6, no. 1, pp. 168-181, 2018

[14] Y. Xie *et al.*, "A Modified Hierarchical Attribute-Based Encryption Access Control Method for Mobile Cloud Computing," *IEEE Trans. Cloud Computing*, vol. 7, no. 2, 2019.

[15] X. P. Zhao and R. Jiang, "Distributed machine learning oriented data integrity verification scheme in cloud computing environment," *IEEE Access*, vol. 8, pp. 26372-26384, 2020.

[16] L. M. Adleman, "Molecular computation of solutions to combinatorial problems," *Science,* vol. 266, no. 5187, pp. 1021-1024, 1994.

[17] C. T. Clelland, V. Risca, and C. Bancroft, "Hiding messages in DNA microdots," *Nature*, vol. 399, no. 6736, pp. 533-534, 1999.

[18] K. Tanaka, A. Okamoto, and I. Saito, "Public-key system using DNA as a one-way function for key distribution," *Biosystems*, vol. 81, no.1, pp. 25-29, 2005.

[19] R. Enayatifar, A. H. Abdullah, and I. F. Isnin, "Chaos-based image encryption using a hybrid genetic algorithm and a DNA sequence," *Optics and Lasers in Engineering*, vol. 56, pp. 83-93, 2014.

[20] U. Tuncer and E. Avci, "A reversible data hiding algorithm based on probabilistic DNA-XOR secret sharing scheme for color images," *Displays*, vol. 41, pp. 1-8, 2016.

[21] B. Wang *et al.*, "Reversible data hiding based on DNA computing," *Computational Intelligence and Neuroscience*, 2017. DOI: https://doi.org/10.1155/2017/7276084.

[22] S. Namasudra *et al.*, "Time efficient secure DNA based access control model for cloud computing environment," *Future Generation Computer Systems*, vol. 73, pp. 90-105, 2017.

[23] X. Wang, Y. Hou, S. Wang, and R. Li, "A New Image Encryption Algorithm Based on CML and DNA Sequence," *IEEE Access*, vol. 6, pp. 62272-62285, 2018.

[24] Y. Wang, Q. Han, G. Cui, J. Sun, "Hiding Messages Based on DNA Sequence and Recombinant DNA Technique," *IEEE Transactions on Nanotechnology*, vol. 18, pp. 299-307, 2019.

[25] M. I. Reddy, A. P. S. Kumar, K. S. Reddy, "A secured cryptographic system based on DNA and a hybrid key generation approach," *Biosystems*, vol. 197, 2020. DOI: https://doi.org/10.1016/j.biosystems.2020.104207

[26] R. N. Calheiros, R. Ranjan, A. Beloglazov, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Software-Practice and Experience (SPE)*, vol. 41, no. 1, pp. 23-50, 2011.

[27] Java. Available: http://java.com/en/download/index.jsp., 2020 [Accessed on 12 January 2019]

[28] Apache Commons Math. Available: http://commons.apache.org/proper/commons-math/download_math.cgi., 2020 [Accessed on 1 January 2019]

[29] CityPulse Dataset Collection. Available: http://iot.ee.surrey.ac.uk:8080/datasets.html., 2020 [Accessed on 12 January 2020]

Dr. Suyel Namasudra (M'19) is an Assistant Professor in the Department of CSE at the National Institute of Technology Patna, Bihar, India. Prior to joining the National Institute of Technology Patna, Dr. Namasudra was an Assistant Professor in the Department of CSE at the Bennett University, India. He has received PhD in Computer Science and Engineering from the National Institute of Technology Silchar, Assam, India. His research interests include Cloud Computing, Information Security and DNA Computing. Dr. Namasudra has edited 2 books and 30 publications in refereed journals, book chapters and conference proceedings. He has participated in many international conferences as an Organizer and Session Chair. Dr. Namasudra is an Editorial Board member and a Reviewer of many journals.