Computer Communications 151 (2020) 539–547

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

# Towards DNA based data security in the cloud computing environment

Suyel Namasudra [a,*], Debashree Devi [b], Seifedine Kadry [c], Revathi Sundarasekar [d], A. Shanthini [e]

[a] Department of Computer Science and Engineering, Bennett University, Greater Noida, UP, India
[b] Department of Computer Science and Engineering, NIT Silchar, Silchar, Assam, India
[c] Department of Mathematics and Computer Science, Faculty of Science, Beirut Arab University, Lebanon
[d] Anna University, India
[e] Department of Information Technology, SRM University, Kattankulathur, India

ARTICLE INFO

*Keywords:*
Cloud computing
DNA computing
Big data security
MAC address
Complementary rule
CloudSim

ABSTRACT

Nowadays, data size is increasing day by day from gigabytes to terabytes or even petabytes, mainly because of the evolution of a large amount of real-time data. Most of the big data is transmitted through the internet and they are stored on the cloud computing environment. As cloud computing provides internet-based services, there are many attackers and malicious users. They always try to access user's confidential big data without having the access right. Sometimes, they replace the original data by any fake data. Therefore, big data security has become a significant concern recently. Deoxyribonucleic Acid (DNA) computing is an advanced emerged field for improving data security, which is based on the biological concept of DNA. A novel DNA based data encryption scheme has been proposed in this paper for the cloud computing environment. Here, a 1024-bit secret key is generated based on DNA computing, user's attributes and Media Access Control (MAC) address of the user, and decimal encoding rule, American Standard Code for Information Interchange (ASCII) value, DNA bases and complementary rule are used to generate the secret key that enables the system to protect against many security attacks. Experimental results, as well as theoretical analyses, show the efficiency and effectivity of the proposed scheme over some well-known existing schemes.

## 1. Introduction

Computers are advancing day by day from the era of a vacuum tube to artificial intelligence, which involves many advanced technologies for data communication over the internet. Cloud computing is one of the benchmarks in the era of large-scale computation, where a huge number of distributed and parallel computers are interconnected. Here, many technologies like utility computing, virtualization, network system and distributed processing are combined to allow many services, such as a server, space, network hardware, pay-per-use and many more [1]. There are three main entities in a cloud environment: (i) Data Owner (DO) (iii) Cloud Service Provider (CSP), and (iii) user. Here, many users store their confidential data on the cloud server, and the DOs and users use the cloud services provided by the CSP [2–4]. Managing big data is an important expected capability of today's cloud environment. Nowadays, big data is considered as one of the sophisticated areas for research. Big data is a large size of data including both unstructured and structured data. It needs to be collected, processed, analyzed and visualized. Big data involves advanced techniques and methods for information extraction and data transformation. However, access control and data security are two major concerns of big data in a cloud environment due to the presence of many attackers and malicious users. Access control can be defined by a technique by which a user or customer can access any data or any kind of service from the cloud server. By an efficient Access Control Model (ACM), the CSP recognizes all the attempts to access any service from the cloud server and allows only the authorized users to access big data or cloud service. Since CSP is the central authority or administrator that manages all the tasks of a cloud server, it should provide a strong security of the user's confidential or sensitive big data.

Cryptography is one of the most prominent solutions for big data security. The primary goal of cryptography is to transmit the data or message between the sender and receiver over an untrusted medium in such a manner that an attacker or malicious user is unable to read the original data content. Cryptosystem can be divided into two ways based on the use of encryption and decryption keys: symmetric key encryption and asymmetric key encryption. In the symmetric key encryption, the encryption and decryption processes are executed by using the same key [5,6]. In asymmetric key encryption, different keys or values are used to encrypt and decrypt a data or message. Here, each party is provided with a pair of keys, namely private key and public key. The private key is always kept secret, while the public key can be shared with all. Recently, many researchers have proposed many schemes to improve data or information security and to improve the performance of the cloud environment [7–23].

Shamir [24] has proposed a novel Identity-Based Encryption (IBE) technique, where the sender of any data or message specifies a unique

* Corresponding author.
*E-mail address:* suyelnamasudra@gmail.com (S. Namasudra).

https://doi.org/10.1016/j.comcom.2019.12.041
Received 6 October 2019; Received in revised form 6 December 2019; Accepted 21 December 2019
Available online 2 January 2020

identity that should match at the receiver's end for decrypting the data. In 1992, Role Based Access Control (RBAC) model has been proposed based on the job role. Here, based on the job rule, user's accesses are controlled [25]. However, RBAC is not secured. The ciphertext is assigned with several attributes in Key Policy based Attribute-Based Encryption (KPABE) [26]. Here, an access structure is associated with the private key of the user. In KPABE, the DOs cannot control access policies, and they must trust on the key generator. In Cipher Policy Attribute-Based Encryption (CPABE) [27], an advanced model of KPABE is used in which each ciphertext is associated with an access policy. In CPABE, user's attributes are used to create the private key of the user, and the system must be rebooted to modify the private key of the user. In 2013, Mian et al. [28] have proposed a new Provisioning Data Analytic Workloads (PDAW) model for the cloud environment. However, PDAW does not consider the security issue of confidential data. In Activity Based Access Control (AtBAC) model [29], user's Access Right (AR) for any data is provided based on the designation of the user in an organization. Since there are several components in AtBAC, the system overhead can be increased. Mutual Trust Based ACM (MTBACM) [30] was proposed to achieve mutual trust among different entities of cloud environment. In 2016, Auxilia and Raja [31] have suggested a novel Ontology Centric Based ACM (OCBACM) model in which ontology is maintained for assigning role. This scheme does not address the data security issue. Index Generation Based Access Control model (IGBAC) [32] has been introduced by using substring index generation process. IGBAC faces high key generation time. Alam et al. [33] have proposed a Cross Tenant Access Control (CTAC) model in which the CSP plays the role of a trusted third party. This scheme supports resource sharing technique between two different tenants. In 2018, Almutairi et al. [34] have proposed a novel mechanism for multitenant cloud environment by introducing notion of sensitivity in the cloud data center. This model is based on the concept of RBAC. Xie et al. [35] have proposed a Modified Hierarchical Attribute-Based Encryption (MHABE) scheme to focus data processing, data storing and data accessing in the multi-user data-shared environment. However, in all these existing schemes, the data accessing time is high and data security is a major issue.

DNA computing is one of the advanced fields in which DNA, hardware, molecular biology and biochemistry are used to encode the genetic details in computers. DNA was first used in the field of computation by Adleman [36]. In the beginning, this advanced approach has been utilized to solve NP-hard problems. However, very soon it has been realized that DNA computing may not be the best solution to solve this type of problems. After that many techniques have been developed by using DNA computing to solve many problems, such as SAT problem, 0–1 planning problem, integer planning problem, graph theory, optimal problem, and many Turing machines are proved.

Nowadays, DNA computing is widely using in the field of cryptography. In DNA cryptography, DNA is used as a computational and informational carrier along with the molecular technologies. The concept of DNA has got much attention in the field of information security due to its complex structure. In DNA encryption, unlike the traditional approaches i.e. 0 and 1, data are encrypted and stored by using the DNA bases, namely A (Adenine), C (Cytosine), G (Guanine) and T (Thymine). So, the sender or who encrypts the data can chose any combination of the DNA bases during the encryption process, which improves data security. Many researchers have proposed many schemes to improve data security by using DNA computing. In 1999, Clelland et al. [37] have proposed the first method for data hiding by using DNA computing, namely Hiding Messages in Microdots (HMM). Here, microdot can be defined by the process for concealing the messages known as steganography. However, this scheme is very slow. Leier et al. [38] have proposed a scheme, namely Cryptography with DNA Binary Strands (CDNABS) to improve data security. Tanaka et al. [39] have proposed a Public Key System by using DNA (PKSDNA) computing and one-way function. This scheme is mainly developed for solving the key distribution issue between the sender and receiver. PKSDNA increases the data accessing time and it is not secured against the malware injection attack. In 2007, MingXin et al. [40] have proposed a symmetric cryptosystem, namely DNA Symmetric-key Cryptosystem (DNASC) in which decryption and encryption keys are generated by using DNA computing. Enayatifar et al. [41] have proposed a Chaos Based Image Encryption (CBIE) scheme based on the genetic algorithm, logistic map and DNA masking. The main goal of CBIE is to find the best DNA mask for image encryption. This scheme does not provide strong security. In 2017, Wang et al. [42] have proposed a Reversible Data Hiding Scheme (RDHS) by using DNA-Exclusive OR (DNA-EXOR) operation rule. In RDHS, information is embedded, and after the extraction process, information is exactly restored as the original host signal. However, disclosing of the DNA-EXOR operation rule can create issue in RDHS. Murugan and Thilagavathy [43] have proposed a novel model, namely Zigzag Morse Code based ACM (ZMCACM) for the cloud computing environment. Here, DNA computing and Morse code is used to store the encrypted data in a zigzag pattern that improve data security. In 2019, a novel Client Side Data Encryption Scheme (CSDES) [44] has been proposed based on DNA computing. CSDES can be considered as a multifold symmetric-key cryptography scheme in which data are encrypted before uploading on the cloud server. All these existing schemes face security issues. In addition, these schemes also take much time for secret key generation, key retrieval, data encryption and data decryption.

To solve all the aforementioned problems, a data encryption technique has been introduced in this paper for the cloud computing environment based on DNA computing, namely DNA Based Data Security (DNABDS). In the proposed scheme, the DOs encrypt big data by utilizing DNA sequence unlike the traditional approaches (0 and 1). The DOs use a long 1024-bit DNA based Secret Key (DNASK) for data encryption. This key is randomly generated based on the attributes of the authorized users. After encrypting by the DNA based secret key, the DO encrypts big data by using the DO's Private Key (DOPrK). Then, the DO again encrypts the encrypted big data by the Public Key of the CSP (CSPPuK). The DO stores the encrypted big data on the cloud database, and only the authorized users are able to access the encrypted big data from the server. This proposed scheme can take less time for secret key generation, key retrieval, data encryption and data decryption. The main contributions of this paper can be summarized as below:

(1) A novel 1024-bit DNA based secret key generation technique is introduced in this paper by using DNA computing.
(2) To improve big data security, a novel DNA based encryption method is proposed in the proposed scheme by using the 1024-bit DNA based secret key, data owner's private key and user's public key. The proposed encryption technique can resist many security attacks.
(3) Experimental results and security analysis of DNABDS have been presented in this paper to prove its efficiency over the existing techniques.

The rest of the paper is structured into several parts. Background of the proposed scheme has been presented in Section 2. The proposed scheme has been discussed in Section 3. Section 4 deals with the security analysis of the proposed scheme. Results and discussions have been presented in Section 5. At last, conclusions and future works are given in Section 6.

## 2. Background of the proposed scheme

This section deals with two aspects of the proposed scheme: (i) system model, and (ii) design goals.

### 2.1. System model

There are three entities in the proposed DNABDS:

(1) *Cloud service provider:* This entity is the overall administrator or central authority of any cloud environment that supports infrastructure and provides cloud services by utilizing numer-

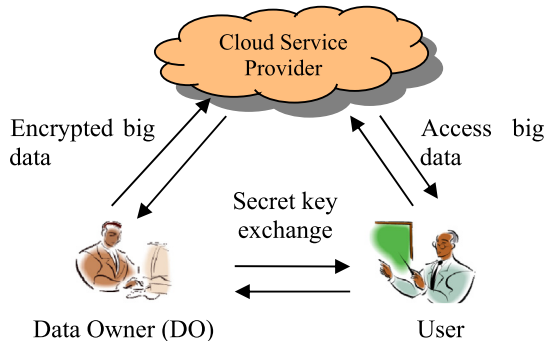


**Fig. 1.** System model of DNABDS.

ous servers having much power as well as adequate memory space [45].

(2) *Data owner:* DOs are the entities, who store their confidential or normal data as well as big data on the database of the cloud environment and depend on the CSP to manage the data.

(3) *User:* Users can be considered as the authorized entities or parties, who desire to get data or any service from the cloud server. Fig. 1 depicts the system model of the proposed scheme.

### 2.2. Design goals

There are three main design goals of DNABDS, which are discussed in this sub-section.

(1) To achieve an efficient and scalable data storage scheme for the cloud environment that must provide a strong security to the users' confidential big data.

(2) To provide the data access right by generating a long 1024-bit password or secret key based on DNA computing along with a corresponding certificate or right.

(3) The DOs must be online during the entire big data communication phase in the existing schemes, which increases system overhead. An effort is made here in DNABDS for minimizing the overhead.

## 3. Proposes scheme

The main purpose of the proposed scheme is to provide a strong security of big data by using a long 1024-bit DNA based password or secret key, which is generated through several processes. In DNABDS, when an authorized user sends a big data access request, the CSP sends the corresponding Public Key of the DO (DOPuK) to the user to get the DNASK and Certificate (C) or access right from the DO. The user can only request to the corresponding DO after getting the respective DOPuK. The DO verifies the authenticity of the user before providing the DNASK of the requested big data and C, and sends all the credentials to the user in the encrypted form. The user must present a valid C to the CSP to access any big data. Otherwise, user's data access request is declined. There are four phases in the proposed approach: (i) system setup (ii) registration and login (iii) DNA based big data storage, and (iv) big data access. Fig. 2 shows the entire workflow of DNABDS.

### 3.1. System setup phase

At first, a big prime number ($p$) is chosen in this system setup phase by the service provider to identify a multiplicative group $Z_p^*$. The CSP selects his/her own public and private key pair from $Z_p^*$. Then, public and private key pairs are chosen by the CSP for both the DO and user from $Z_p^*$. These key pairs are given to the entities at the time of their registration. The users' public keys are publicly available, and the authorized users know the DOPuK. Only the authorized entities know the CSPPuK and individual entity's private key is kept secret from other entity.

**Table 1**
Decimal encoding rule.

| Digits/symbols/ alphabets | Decimal value | Digits/symbols/ alphabets | Decimal value | Digits/symbols/ alphabets | Decimal value |
|---|---|---|---|---|---|
| 1 | 01 | A | 11 | ‘ | 37 |
| 2 | 02 | B | 12 | ~ | 38 |
| 3 | 03 | C | 13 | ! | 39 |
| 4 | 04 | D | 14 | @ | 40 |
| . | . | . | . | . | . |
| . | . | . | . | . | . |
| . | . | . | . | . | . |
| 8 | 08 | X | 34 | ( | 48 |
| 9 | 09 | Y | 35 | ) | 49 |
| 0 | 10 | Z | 36 | Blank space | 50 |

### 3.2. User registration and login phase

The user must send a request to the service provider for registering in the cloud server. When the CSP receives the request, it collects the personal or secret information of the user, such as first school name, date of birth, first teacher name, address, etc., and generates a user's profile. Then, a secure communication medium i.e. Secure Socket Layer (SSL) is established to deliver the key pairs and registration reply. The authorized user or customer is only permitted to login into the system or server after the registration phase. The CSP replies an acknowledgment to the user after a login process. A big data accessing request can only be sent to the service provider after a successful login process.

### 3.3. DNA based big data storage phase

This big data storage phase is based on DNA computing, and it is divided into two phases, namely DNA based big data encryption and DNA based secret key generation.

#### 3.3.1. DNA based secret key generation

The data owner carries the secret key generation process. The DO generates the DNASK only for the authorized or valid users. The DO fetches essential user's attributes after checking the user's authenticity from the CSP.

Then, the decimal encoding rule is applied to the attributes' values to make it in the decimal format as per Table 1. The DO can apply any decimal encoding rule that improves data security. The resultant decimal numbers are transformed to their corresponding 8-bit binary numbers. The binary number is divided into two parts, and Exclusive OR (EXOR) operation is performed between two parts. Then, each 8-bit binary number is transformed into its corresponding ASCII value as per Table 2. ASCII values are normally used to securely transfer any information between the sender and receiver. In DNABDS, ASCII values are randomly generated. As shown in Table 2, ASCII value of 00000001 is 34. However, ASCII value of 00000001 can be 36 or 100 or anything that supports randomness during key generation. These ASCII values are then again converted to their real 8-bit binary numbers, and then, the entire binary number is divided into two parts. Both the parts are added to the both side of the MAC address of the user. The DO then again converts the resultant to the binary number. The length of the binary number may be less or more than 1024-bit. If it is less than 1024-bit, then, extra zeros are added to its right side to make it 1024-bit. Otherwise, extra bits are deleted from the left side. The generated 1024-bit binary number is grouped into four parts i.e. 256-bit in each group. Each group is termed randomly by using the DNA bases, namely A, C, G and T. The complementary rule is then applied to the DNA sequence to make it more complex. Here, the following complementary rule is used in DNABDS:

(AC) (TA) (GT) (CG) that means $V(C) = A$ and so on.

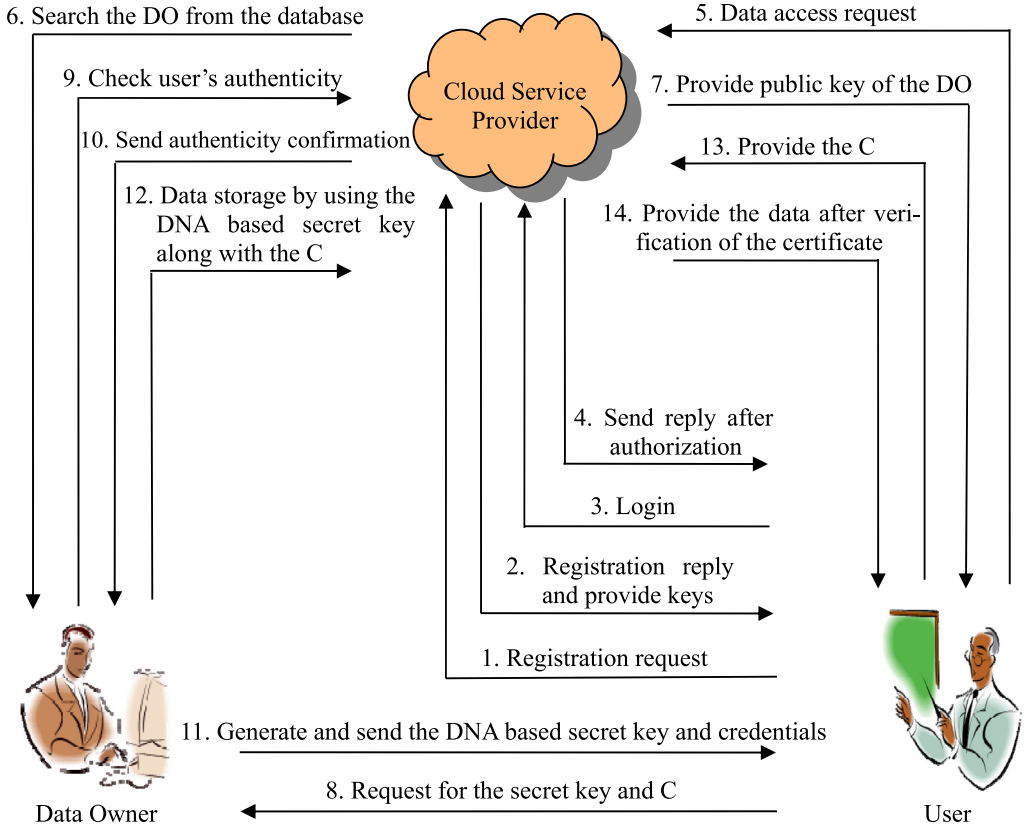


**Fig. 2.** Workflow of the proposed DNABDS.

The resultant DNA sequence is provided to the user as the DNA based secret key. There are the following steps in the key generation process:

***Step 1:*** In the first step, the user communicates with the DO, and a session is generated

***Step 2:*** The DO verifies the user's authorization. The DO continues the further processes, if the user is a valid one.

***Step 3:*** The DO fetches all the information of the user, such as date of birth, user id, address of the user and MAC address of the user. For an example:

20-02-1990 001 ABCD 00-14-22-01-23-45

***Step 4:*** In the fourth step, the DO converts these values in the 8-bit binary form by using decimal encoding rule as mentioned in Table 1.

$$\underbrace{00000010}_{2}\ \underbrace{00000000}_{0}\ \ldots\ldots\ \underbrace{00000100}_{4}\ \underbrace{00000101}_{5}$$

***Step 5:*** The DO then divides the binary value of the resultant of step 4 into two parts and EXOR operation is performed between these two parts. For an example, if two parts of the binary value are 0000001000000000 and 0000010000000101, EXOR operation is performed between 0000001000000000 and 0000010000000101. The result of EXOR operation is:

0000011000000101

***Step 6:*** In the sixth step, each 8-bit binary numbers are transformed into their corresponding ASCII values by using Table 2. For an example, in the above step, there are two 8-bit binary values: 00000110 and 00000101. ASCII values of both these binary numbers can be 53 and 165 or it can be anything. The DO randomly generates Table 2 as per his/her wish.

$$\underbrace{00000110}_{53\ (ASCII value)}\quad\underbrace{00000101}_{165\ (ASCII value)}$$

***Step 7:*** ASCII values are again transformed into their actual 8-bit binary values.

0011010110100101

***Step 8:*** In the eight step, the DO then divides the resultant of step 7 into two parts and adds the first part to the left side of the MAC address, while the second part is added to the right side of the MAC address. So, it becomes:

$$\underbrace{00110101}\ \underbrace{00-14-22-01-23-45}_{MAC\ Address}\ \underbrace{10100101}$$

***Step 9:*** Then, the DO converts the resultant of step 8 into binary number.

***Step 10:*** In the tenth step, extra bits are deleted from the left side or 0 is added to the right side to make it 1024-bit. For an example, the result (1024-bit) of this step is:

00110101 ……..10100101

***Step 11:*** Group the binary number into four parts (256-bit block) and assign the DNA bases to each part.

$$\underbrace{00110101\ \ldots\ldots}_{A}\ \underbrace{\ldots\ldots\ldots}_{G}\ \underbrace{\ldots\ldots\ldots}_{T}\ \underbrace{\ldots\ldots 10100101}_{C}$$

***Step 12:*** In the twelfth step, complementary rule is applied on the resultant of step 11 to make it the final DNA based secret key. So, the final DNA key sequence is:

CTAG

**DNA Based Secret Key Generation Algorithm**

**Input:** Attributes and MAC address of an authorized user.
**Output:** 1024-bit DNA based secret key.

```
If Y is an authorized user
    COLLECT attributes and MAC address of the user
    APPLY decimal encoding rule
    CONVERT into the 8-bit binary number
    DIVIDE into two parts
    EXECUTE EXOR operation between two parts
    CONVERT into the ASCII values
    CONVERT into its actual 8-bit binary number
    DIVIDE into two parts
    ADD both parts in the both side of the MAC address
    CONVERT into the binary number
        If length of binary number > 1024-bit
            DELETE extra bit from the left end
        Else
            APPEND zero at the right end to make it 1024-bit
    DIVIDE the binary number into four blocks
    ASSIGN DNA base for each block
    APPLY complementary rule on the DNA sequence
Else
    STOP
```

**Table 2**
ASCII values.

| 8-bit binary number | ASCII |
|---|---|
| 00000000 | 46 |
| 00000001 | 34 |
| 00000010 | 123 |
| 00000100 | 94 |
| . | . |
| . | . |
| . | . |
| 11111111 | 111 |

When the DO generates the secret key, the session is over. The resultant of step 10 is used by the DO as secret key for data encryption and the resultant of step 12 is sent to the user as DNASK after encrypting by the DOPrK and User's Public Key (USRPuK). In addition, the DO also sends Tables 1, 2, 3 and complementary rule in the same encrypted message. As the user has all the credentials, s/he can quickly recover the original key after decrypting by utilizing the Private Key of the User (USRPrK) and DOPuK.

#### 3.3.2. Big data encryption by using DNA sequence

In the big data encryption phase, the DO splits the plaintext into 1024-bit blocks. Here, each 1024-bit block is then grouped into four equal blocks i.e. 256-bit in each block. Then, EXOR operation is executed between the 256-bit block of the plaintext and the 256-bit block of the key. So, there are four EXOR operations for each 1024-bit block of plaintext. Each 2-bit binary number is then transformed into the corresponding DNA bases as per Table 3. Since there are 4 DNA bases, there may be maximum 24 combinations of the DNA bases, and the DO can choose any combination out of 24 combinations to transform the encrypted binary form of plaintext into the DNA bases. The data owner then uses the same complementary rule used in the previous Section 3.3.1 on the resultant DNA bases.

At last, the DO again encrypts the achieved DNA sequence by using the DOPrK and CSPPuK. The DO also encrypts the corresponding data access right or certificate along with the big data and saves the entire encrypted file or data on the cloud database. As the CSP does not have the secret key and other credentials, the CSP cannot decrypt the encrypted big data. Fig. 3 shows data encryption process based on DNA computing.

### 3.4. Big data access phase

To access big data from the cloud server, the user must send a registration request to the CSP. The CSP sends the registration reply, and then, the user can login into the cloud server to access big data. There are several steps for big data accessing from the cloud server, which are mentioned in this sub-section:

***Step 1:*** The user sends a request to the CSP encrypted by using USRPrK and CSPPuK to access big data.
***Step 2:*** In the second step, the CSP searches the DOPuK of the requested big data.
***Step 3:*** Then, the CSP provides the corresponding DOPuK to the user after making it ciphertext by using the Private key of the CSP (CSPPrK) and USRPuK. The user decrypts the message by using the USRPrK and CSPPuK, and gets the DOPuK.
***Step 4:*** Here, the user uses the DOPuK and initiates a demand to the respective DO encrypted by using USRPrK and DOPuK to get the secret key and C.
***Step 5:*** The DO verifies the user's authenticity from the cloud service provider.
***Step 6:*** In the sixth step, an authentication reply is sent to the DO by the CSP.
***Step 7:*** Here, the DO generates the DNASK and sends all the credentials to the user encrypted by DOPrK and USRPuK.
***Step 8:*** In the eight step, at first, the DO encrypts the big data by using the DNASK. Then, the DO stores the encrypted big data as well as the corresponding C on the cloud server encrypted by using DOPrK and CSPPuK.
***Step 9:*** The user presents the C to the CSP encrypted by using USRPrK and CSPPuK. The encrypted message is decrypted by the CSP for getting the C.
***Step 10:*** In the tenth step, the CSP verifies the C with the C of the requested big data. When the C is correct or authorized, the rest of the processes are performed.
***Step 11:*** The CSP encrypts the user's requested big data by utilizing CSPPrK and USRPuK and sends it to the user. The user gets the encrypted message and decrypts it by the USRPrK and CSPPuK. At last, the decryption process is again executed by using the DNASK and other credentials to get the original data content.

## 4. Security analysis

The proposed DNA based data encryption scheme is secured against many attacks. Security analysis of DNABDS has been presented in this section.

**Table 3**
DNA bases of 2-bit binary number.

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

### 4.1. Malware injection

In the malware injection [46], attackers and malicious users inject scripts of malicious code into the cloud server and when this code is running in the cloud server, they access confidential data from the cloud server. Sometimes, this type of injection is unnoticed for a long time, which creates a serious issue in the cloud environment. In DNABDS, the DO randomly generates a DNASK and uses this secret key for encrypting big data. The DO uses complementary rule, Tables 1, 2, 3 for key generation and big data encryption. All these credentials are shared only with the authorized user in the encrypted form after checking the user's authenticity and the respective user can only decrypt the message to get these credentials. No one can decrypt the data without all the credentials. So, if an attacker tries to access any big data by malware injection, s/he still cannot decrypt the data content. Therefore, DNABDS can be treated secure and protected against this attack.

### 4.2. Side channel attack

In the side channel attack [47], hackers and malicious users are used to place a malicious or unauthorized Virtual Machine (VM) on the same host to get any confidential data or information. Here, hackers mainly attack the system implementation of the encryption algorithm. In DNABDS, the DO generates the DNASK through several processes to make it secure and this long 1024-bit key is used to encrypt big data. The encrypted big data is again encrypted by using DOPrK and CSPPuK before storing it on the cloud database. The DNASK is only provided to the authorized users encrypted by using DOPrK and USRPuK, and the DO generates the secret key after verifying the authenticity of the user from the CSP. The respective user is only able to get the secret key, when s/he decrypts by using USRPrK and DOPuK. The DO does not store the DNASK anywhere, not even on the cloud server. If any hacker places or replaces any VM, data security still cannot be compromised. So, the proposed scheme can resist this attack.

### 4.3. Phishing attack

In the phishing attack [48], malicious user, unauthorized user or hacker gets the authorized user's sensitive information, such as user ID, user voter ID number and many more. Then, the attacker utilizes these sensitive or confidential information for getting any unauthorized service from the cloud server. In DNABDS, the CSP assembles all the details or information of the user after getting a registration request from the user. Then, the CSP sends the public and private key pair by using SSL. The CSP also follows the same approach for registering the DOs. When the user sends an access request for big data, the service provider only gives the respective DOPuK to the user in the ciphertext form. The DO only provides the DNASK after confirming the user's authenticity. Both the DO and CSP do not expose any sensitive or confidential data of themselves to others. So, the hackers or malicious users cannot get the information of the authorized customers or users, and DNABDS can resist phishing attack.

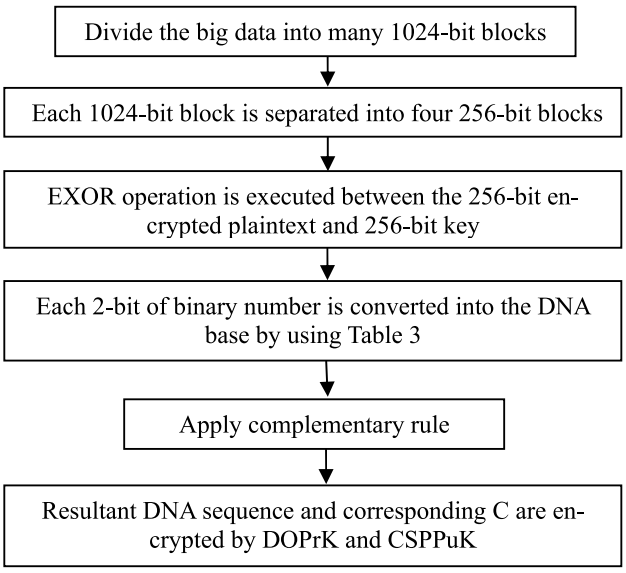


**Fig. 3.** DNA computing-based big data encryption.

### 4.4. Insider attack

In the insider attack [49], a malicious user or attacker used to violate any security policy to access any data or service. In a cloud environment, any entity can be an attacker to attempt an insider attack. Even the CSP also can execute insider attack. In the proposed scheme, the DO creates the DNASK based on user's attributes and MAC address of the user by using Tables 1, 2 and complementary rule. The DO uses Table 3 and complementary rule during the big data encryption process. Since the DNASK is generated by using user's secret information, it is very challenging for the hackers or unauthorized users to get all the secret information. If a hacker gets all the secret information or confidential data of an authorized user in a worst case, s/he also must have Tables 1, 2, 3 and complementary rule to retrieve the original data. The DO shares these credentials only with the authorized or valid user in the encrypted form by using DOPrk and USRPuK. The CSP or any other internal entity does not have any idea about all these credentials. Therefore, DNABDS can be treated secure against insider attack.

### 4.5. Denial of Service (DoS) attack

In the DoS attack [50], attackers or malicious users send numerous requests to the cloud server to make the services unavailable for the intended users. In the proposed scheme, the DO randomly generates a long 1024-bit DNASK and encrypts big data by using this long 1024-bit key. The DO also uses DOPrK and CSPPuK to encrypt the big data before storing it on the cloud server. If the attackers attack the cloud server by DoS, they cannot get original data content because the DO shares all the credentials, such as Tables 1, 2, 3 and complementary rule in the encrypted form only with the authorized users after checking the user's authenticity. Thus, user's confidential or sensitive data does not face any security issue and the proposed scheme can be considered secure against DoS attack.

## 5. Performance analysis

Performance analysis of the proposed DNABDS has been presented in this section in detail.

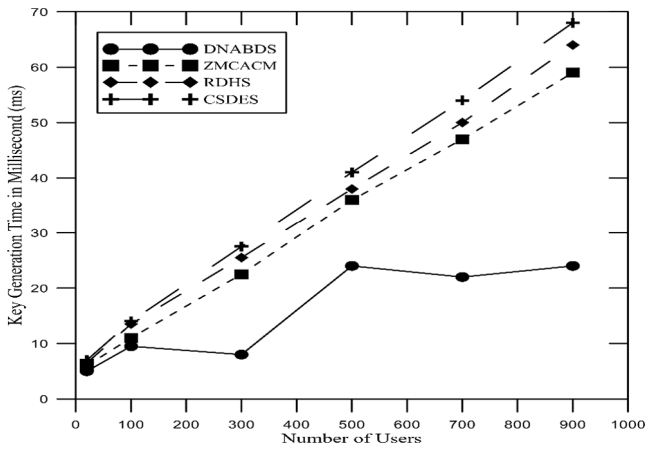


**Fig. 4.** Number of users vs. key generation time.

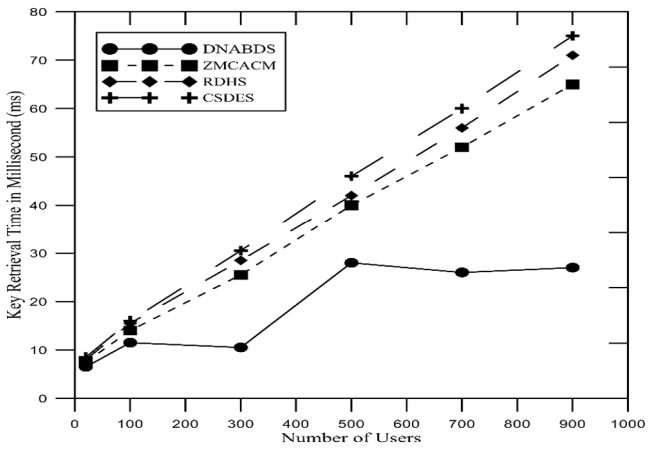


**Fig. 5.** Number of users vs. key retrieval time.

### 5.1. Simulation environment

For evaluating the performance of DNABDS, a cloud simulation environment has been built by using CloudSim 3.0.3 [51]. CloudSim has been installed on a DELL OPTIPLEX 9020 SFF desktop, which consists of the following configuration:

(1) RAM: 8 GB
(2) Storage capacity: 1 TB
(3) Processor: 3.40 GHz Intel corei5
(4) Operating system: Windows 7

Apache Common-math [52] is mounted with the CloudSim. Ten datacenters have been considered during simulation to create a heterogeneous cloud environment. In the heterogeneous cloud environment, 2000 physical nodes are considered, and 1 GB/s of network bandwidth and storage capacity is 4 GB. Four types of VM with 1 GB/s bandwidth have been considered as mentioned below:

(1) 1000 MIPS, 1 GB
(2) 1500 MIPS, 1.5 GB
(3) 2000 MIPS, 2 GB
(4) 2500 MIPS, 2.5 GB

In CloudSim, few classes, namely *VMScheduler, powerhost, VM* and *CloudletScheduler* are modified for the simulation purpose to achieve dynamic resource distribution and a class, *dynamicmemory* is newly added to guess the required resource. This class also updates the resources of VMs after the execution of a cloudlet.

### 5.2. Results and discussions

Many experiments are executed for evaluating the performance of the proposed scheme with respect to the existing schemes, namely reversible data hiding scheme [42], zigzag Morse code based ACM [43] and client side data encryption scheme [44]. These schemes are proposed recently and almost analogous to the proposed scheme, so these are treated as the baseline schemes for comparisons. To calculate key generation time, key retrieval time, encryption time of big data and decryption time of big data, 50 experiments are executed in different situations to get each result. Finally, average values are considered from those 50 results. For the simulation, datasets are collected from publicly available CityPulse Dataset Collection [53], which provides different types of datasets, namely pollution, weather, road traffic, etc. Here, vehicle traffic dataset (3 GB size) has been used for the experiments. In this sub-section, results and discussions of the proposed DNABDS have been presented.

Fig. 4 depicts the results of the first experiment to calculate the key generation time. In DNABDS, all the users have to send the requests to the respective DOs for receiving the secret key and user's access right, when they wish to get any big data. The DO collects a set of secret user's attributes and MAC address of the user, and uses Tables 1, 2 and complementary rule for generating the DNASK. So, the proposed scheme takes a little bit of time to generate the DNASK. In DNABDS, once the secret key is provided to the users, they can use it in future. Thus, for the existing or old users, who already have the DNA based secret key, the key generation time is reduced. Thus, DNABDS produces a zigzag curve. In ZMCACM, secret key is generated through several processes, which consumes much time. There are many EXOR operations in both RDHS and CSDES to generate the DNA sequence based secret key. Moreover, all these existing schemes are very complex compared to the proposed scheme, and all the users or customers must get the secret key in each accessing time of big data for further data accessing processes. Therefore, linear increasing curves are experienced in the case of all these existing schemes.

The second experiment validates the key retrieval time. It can be easily recognized from Fig. 5 that DNABDS gives much better results for retrieving the secret key by the user. In the proposed scheme, the old or existing users have the secret key of the corresponding data and they do not need to get the DNASK from the DO. So, they do not retrieve the DNASK for the subsequent time of big data accessing. However, in DNABDS, all the new users or customers, who want to access the big data for the first time must get the DNASK from the DO. So, they have to retrieve the key during the first accessing, and thus, the key retrieving time is increased in DNABDS. In the existing schemes, namely ZMCACM, RDHS and CSDES, users must execute many operations to retrieve the secret key. Here, whenever the users want to get a big data from the cloud server, every time they have to get the secret key from the DO and they have to retrieve it. Thus, the key retrieving time is high in ZMCACM, RDHS and CSDES, and it produces linearly increasing curves.

In Fig. 6, it can be easily seen that the proposed scheme takes less time to encrypt big data than the existing schemes in the cloud computing environment. In DNABDS, after generating the DNASK, the DO uses Table 3 and complementary rule to encrypt big data, which is time consuming. In ZMCACM, Morse code is used to encrypt big data and encrypted data is stored in zigzag lines after many operations by using 'dot' and 'dash'. Thus, this scheme takes much time to encrypt big data. The classical algorithm of reversible data protecting or data hiding approach is used in RDHS, where using of histogram modification technique increases the time of big data encryption compare to the proposed scheme. In addition, watermarking process of RDHS consumes time to encrypt a big data. Client side data encryption scheme

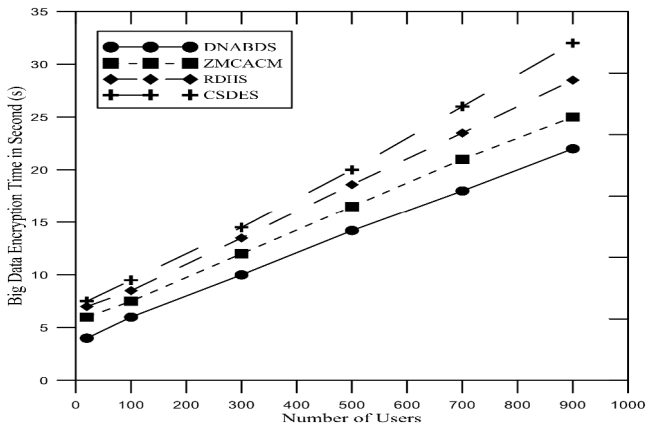


**Fig. 6.** Number of users vs. data encryption time.

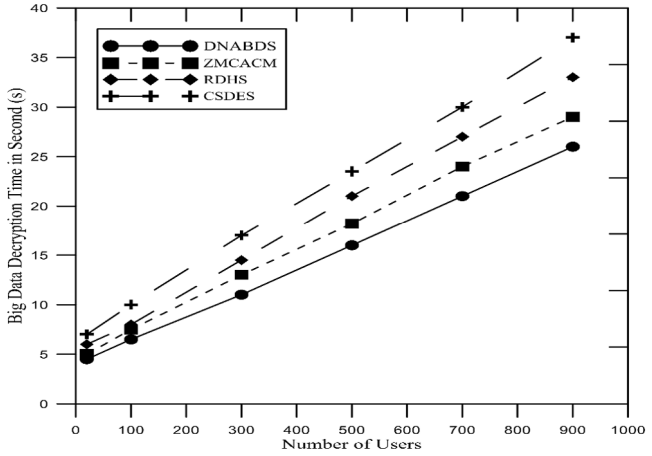


**Fig. 7.** Number of users vs. data decryption time.

is complex and there are many EXOR operations to encrypt any big data. Thus, the encryption time of big data is more in all these existing schemes.

The last experiment shows the results for big data decryption time. In DNABDS, the DO provides all the credentials to the users at the time of providing the secret key. The users can easily decrypt the big data by using these credentials in a less time. In DNABDS, there are no complex operations and four EXOR operations are required for big data decryption, which decreases the time of decryption. However, in the existing schemes, namely ZMCACM, RDHS and CSDES, many complex operations and many EXOR operations are required to decrypt any big data, which increases the time as shown in Fig. 7. Therefore, the decryption time of big data is high in these existing schemes compare to the proposed DNABDS.

## 6. Conclusions and future works

The security issues of big data in the cloud computing environment are increasing day by day. A long 1024-bit DNA based secret key or password generation technique is proposed in this paper based on the secret attributes of the user. The same key is used for encrypting big data based on DNA computing. The secret key is secured as it labels the randomness during assign of the DNA bases and the confidential credentials are only shared with the authorized users. The proposed scheme can resist many attacks in the cloud environment. Here, the data owners can be online to provide the DNA based secret key and access certificate, and they can go offline after delivering the credentials. Thus, the system overhead is decreased. Experimental results prove that the proposed technique is more efficient and effective than the other well-known existing schemes. Mathematical proof for the security analysis of the proposed scheme will be considered as a future work. Moreover, there is a considerable scope to improve the authentication process of the cloud computing environment.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## References

[1] A. Huthand J. Chebula, The Basics of Cloud Computing, Carnegie Mellon University, 2011.
[2] S. Namasudra, S. Nath, A. Majumder, Profile based access control model in cloud computing environment, in: Proceedings of the International Conference on Green Computing, Communication and Electrical Engineering, IEEE, Coimbatore, India, 2014, pp. 1–5.
[3] Q. Zhang, L. Cheng, R. Boutaba, Cloud computing: state-of-the-art and research challenges, J. Internet Serv. Appl. 1 (1) (2010) 7–18.
[4] V. Chang, Y.H. Kuo, M. Ramachandran, Cloud computing adoption framework: a security framework for business clouds, Future Gener. Comput. Syst. 57 (2016) 24–41.
[5] S. Kumar, T. Wollinger, Fundamentals of symmetric cryptography, in: K. Lemke, C. Paar, M. Wolf (Eds.), Embedded Security in Cars, Springer, Berlin, Heidelberg, 2006, pp. 125–143.
[6] B. Schneier, Applied Cryptography, second ed., Wiley & sons, 1996.
[7] S. Namasudra, An improved attribute-based encryption technique towards the data security in cloud computing, in: Concurrency and Computation: Practice and Exercise, 2017, http://dx.doi.org/10.1002/cpe.4364.
[8] M. Sarkar, K. Saha, S. Namasudra, P. Roy, An efficient and time saving web service based android application, SSRG Int. J. Comput. Sci. Eng. 2 (8) (2015) 18–21.
[9] S. Namasudra, P. Roy, B. Balamurugan, Cloud computing: fundamentals and research issues, in: Proceedings of the 2nd International Conference on Recent Trends and Challenges in Computational Models, IEEE, Tindivanam, India, 2017.
[10] G.C. Deka, P.K. Das, An overview on the virtualization technology, in: Handbook of Research on Cloud Infrastructures for Big Data Analytics, 2014, http://dx.doi.org/10.4018/978-1-4666-5864-6.ch012.
[11] S. Namasudra, Taxonomy of DNA-based security models, in: S. Namasudra, G.C. Deka (Eds.), Advances of DNA Computing in Cryptography, Taylor & Francis, 2018, pp. 53–68.
[12] S. Namasudra, P. Roy, B. Balamurugan, P. Vijayakumar, Data accessing based on the popularity value for cloud computing, in: Proceedings of the International Conference on Innovations in Information, Embedded and Communications Systems (ICIIECS), IEEE, Coimbatore, India, 2017.
[13] D. Devi, S.K. Biswas, B. Purakayastha, Redundancy-driven modified Tomek-link based undersampling: a solution to class imbalance, Pattern Recognit. Lett., 93, 3-122017.
[14] S. Namasudra, P. Roy, Secure and efficient data access control in cloud computing environment: a survey, Multiagent Grid Syst.- Int. J. 12 (2) (2016) 69–90.
[15] G.C. Deka, M. Kathing, D.P. Kumar, Library automation in cloud, in: Proceedings of the International Conference on Computational Intelligence and Communication Networks, Mathura, 2013.
[16] S. Namasudra, G.C. Deka, Advances of DNA Computing in Cryptography, Taylor & Francis, 2018.
[17] S. Namasudra, D. Devi, S. Choudhary, R. Patan, S. Kallam, Security, privacy, trust, and anonymity, in: S. Namasudra, G.C. Deka (Eds.), Advances of DNA Computing in Cryptography, Taylor & Francis, 2018, pp. 153–166.
[18] S. Namasudra, P. Roy, Time saving protocol for data accessing in cloud computing, IET Commun. 11 (10) (2017).
[19] S. Namasudra, G.C. Deka, Introduction of DNA computing in cryptography, in: S. Namasudra, G.C. Deka (Eds.), Advances of DNA Computing in Cryptography, Taylor & Francis, 2018, pp. 27–34.
[20] S. Namasudra, P. Roy, PpBAC: popularity based access control model for cloud computing, J. Organ. End User Comput. 30 (4) (2018) 14–31.
[21] S. Namasudra, G.C. Deka, Rohan Bali, Applications and future trends of DNA computing, in: S. Namasudra, G.C. Deka (Eds.), Advances of DNA Computing in Cryptography, Taylor & Francis, 2018.
[22] G.C. Deka, M.D. Borah, Cost benefit analysis of cloud computing in education, in: Proceedings of the International Conference on Computing, Communication and Applications, 2012, pp. 1–6.
[23] S. Namasudra, P. Roy, A new table based protocol for data accessing in cloud computing, J. Inf. Sci. Eng. 33 (3) (2016) 585–609.

[24] A. Shamir, Identity-based cryptosystems and signature schemes, in: G.R. Blakley, D. Chaum (Eds.), Advances in Cryptology, Springer, 1985, pp. 47–53.
[25] D.F. Ferraiolo, D.R. Kuhn, Role-based access controls, in: Proceedings of the 15th National Computer Security Conference, Baltimore, USA, 1992, pp. 554–563.
[26] V. Goyal, O. Pandey, A. Sahai, B. Waters, Attribute-based encryption for fine-grained access control of encrypted data, in: Proceedings of the 13th ACM Conference on Computer and Communications Security, New York, USA, 2006, pp. 89–98.
[27] J. Bethencourt, A. Sahai, B. Waters, Ciphertext-policy attribute based encryption, in: Proceedings of the IEEE Symposium on Security and Privacy, Berkeley, CA, 2007, pp. 321–334.
[28] R. Mian, P. Martin, J.L.V. Poletti, Provisioning data analytic workloads in a cloud, Future Gener. Comput. Syst. 29 (6) (2013) 1452–1458.
[29] S. Ajgaonkar, H. Indalkar, J. Jeswani, Activity based access control model for cloud computing, Int. J. Curr. Eng. Technol. 5 (2) (2015) 708–713.
[30] L. Guoyuan, W. Danru, B. Yuyu, L. Min, MTBAC: a mutual trust based access control model in cloud computing, China Commun. 11 (4) (2014) 154–162.
[31] M. Auxilia, K. Raja, Ontology centric access control mechanism for enabling data protection in cloud, Indian J. Sci. Technol. 9 (23) (2016) 1–7.
[32] S. Raghavendra, et al., Index generation and secure multi-user access control over an encrypted cloud data, Procedia Comput. Sci. 89 (2016) 293–300.
[33] Q. Alam, et al., A cross tenant access control (CTAC) model for cloud computing: formal specification and verification, IEEE Trans. Inf. Forensics Secur. 12 (6) (2017) 1259–1268.
[34] A. Almutairi, M.I. Sarfraz, A. Ghafoo, Risk-aware management of virtual resources in access controlled service-oriented cloud datacenters, IEEE Trans. Cloud Comput. 6 (1) (2018) 168–181.
[35] Y. Xie, H. Wen, B. Wu, Y. Jiang, J. Meng, A modified hierarchical attribute-based encryption access control method for mobile cloud computing, IEEE Trans. Cloud Comput. 7 (2) (2019) 383–391.
[36] L.M. Adleman, Molecular computation of solutions to combinatorial problems, Science 266 (5187) (1994) 1021–1024.
[37] C.T. Clelland, V. Risca, C. Bancroft, Hiding messages in DNA microdots, Nature 399 (6736) (1999) 533–534.
[38] A. Leier, C. Richter, W. Banzhaf, H. Rauhe, Cryptography with DNA binary strands, Biosystems 57 (1) (2000) 13–22.
[39] K. Tanaka, A. Okamoto, I. Saito, Public-key system using DNA as a one-way function for key distribution, Biosystems 81 (1) (2005) 25–29.
[40] L. MingXin, L. XueJia, X. GuoZhen, Q. Lei, Symmetric-key cryptosystem with DNA technology, Sci. China F 50 (3) (2007) 324–333.
[41] R. Enayatifar, A.H. Abdullah, I.F. Isnin, Chaos-based image encryption using a hybrid genetic algorithm and a DNA sequence, Opt. Lasers Eng. 56 (2014) 83–93.
[42] B. Wang, Y. Xie, S. Zhou, C. Zhou, X. Zheng, Reversible data hiding based on DNA computing, Comput. Intell. Neurosci. (2017) http://dx.doi.org/10.1155/2017/7276084.
[43] A. Murugan, R. Thilagavathy, Cloud storage security scheme using DNA computing with morse code and zigzag pattern, in: Proceedings of the IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI-2017), IEEE, 2018, pp. 2263–2268.
[44] M. Sohal, S. Sharma, BDNA-A DNA inspired symmetric key cryptographic technique to secure cloud computing, J. King Saud Univ.-Comput. Inf. Sci. (2019) http://dx.doi.org/10.1016/j.jksuci.2018.09.024.
[45] S. Namasudra, Cloud computing: a new era, J. Fundam. Appl. Sci. 10 (2) (2018) 113–135.
[46] A.O. Eze, C.C. E, Malware analysis and mitigation in information preservation, IOSR J. Comput. Eng. 20 (4) (2018) 53–62.
[47] K. Tiri, Side-channel attack pitfalls, in: Proceedings of the 44th ACM/IEEE Design Automation Conference, IEEE, San Diego, USA, 2007.
[48] T.N. Jagatic, N.A. Johnson, M. Jakobsson, F. Menczer, Social phishing, Commun. ACM 50 (10) (2007) 94–100.
[49] M.B. Salem, S. Hershkop, S.J. Stolfo, A survey of insider attack detection research, in: S.J. Stolfo, S.M. Bellovin, A.D. Keromytis, S. Hershkop, S.W. Smith, S. Sinclair (Eds.), Insider Attack and Cyber Security, in: Advances in Information Security, vol. 39, Springer, Boston, MA, 2008.
[50] A. Prakash, M. Satish, T.S.S. Bhargav, N. Bhalaji, Detection and mitigation of denial of service attacks using stratified architecture, Procedia Comput. Sci. 87 (2016) 275–280.
[51] R.N. Calheiros, R. Ranjan, A. Beloglazov, CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms, Softw.-Pract. Exp. (SPE) 41 (1) (2011) 23–50.
[52] Apache commons math, 2018, Available: http://commons.apache.org/proper/commons-math/download_math.cgi [Accessed on 14 2018].
[53] CityPulse dataset collection, 2018, Available: http://iot.ee.surrey.ac.uk:8080/datasets.html [Accessed on 15 2018].