A Novel Genetic Algorithm-Based DES Key Generation Scheme

Min-Yan Tsai[1], Hsin-Hung Cho[1(✉)], Chi-Yuan Chen[1], and Wei-Min Chen[2]

[1] Department of Computer Science and Information Engineering, National Ilan University, Yilan, Taiwan
wer9u623@gmail.com,awp.boom@gmail.com,chiyuan.chen@gmail.com
[2] Department of Information Management, School of Management, National Dong Hwa University, Hualien, Taiwan
wmchen88@gms.ndhu.edu.tw

Abstract. The data encryption is closely related to our daily lives, especially after the concept of e-commerce has become popular. The main reason is that although various network service platforms facilitate our lives, they also bring many potential problems such that there are many ways that people with bad intentions can tamper or steal data via Internet and further decrypt it. In order to maintain user privacy, encryption methods are adopted to prevent stealers from easily reading data. Data encryption standard (DES) is currently the most widely used encryption method in commercial financial units. However, since DES has a short password length and the generated key is quite linear, weak keys may be generated so that the security is still insufficient. In this paper, a genetic algorithm (GA) method is used to generate high-variability keys to solve the problem of a high probability that DES generates weak keys.

Keywords: Information security · Encryption · Symmetric keys · Genetic algorithms

1 Introduction

The habits of human beings have undergone a revolutionary change due to the development of network and multimedia technologies so that the Internet has become inseparable from everyone [4,17]. Nowadays, human beings can obtain desired information and services, through various network media such that E-commerce and the digital economy are newly developed transaction methods in recent years [1]. However, transferring the field of traditional transactions to Internet has led to some potential risks. While the data is being transmitted, some people with bad intentions can steal some related private information through various network tools. If the data is not protected by any encryption mechanism, all kinds of transaction-related information, including the personal information of the trader, the amount of the transaction, and various private information such as identity authentication will be exposed that it seriously

Y. Chen et al. (Eds.): BICT 2020, LNICST 329, pp. 199–211, 2020.
https://doi.org/10.1007/978-3-030-57115-3_17

Table 1. Comparison of symmetric/asymmetric encryption

Types	Symmetric	Asymmetric
Encryption and decryption ways	With the same key	With the different key
Common key	With the same key	With the different key
Advantage	Shorter operation time	Higher privacy; undeniable
Disadvantage	Difficult to manage keys	computing resources and time are massive
Representative method	AES,DES	RSA

affects the rights and interests of traders [2,3]. It illustrates the importance and necessity of encryption [11].

Many methods of cryptography have been proposed so far. The field of cryptography can be divided into classical cryptography, symmetric encryption and asymmetric encryption [14] that a brief comparison is shown in Table 1. The common symmetric encryption includes DES and Advanced Encryption Standard (AES) [13]. The common asymmetric encryption is like RSA [12]. Among these three types of cryptosystems, the symmetric encryption system is the most widely used. Because the characteristics of this type of cryptosystem have faster encryption speed and lower computing costs. DES is a classic method of the symmetric encryption system proposed by the National Security Agency (NSA) [16]. Up to the present, NSA has also been revised many times, which also shows that the symmetric encryption system still has a high degree of occupancy and practicality. such that banking [9], the medical industry [8], etc. are still using DES as the main encryption core, hence how to strengthen the symmetric DES encryption system is still a very important issue.

Since DES has a long history [7], many scholars have questioned the security of DES. The major questions include the insufficient length of DES and the insufficient resistance to brute-force attacks especially in the era of rapid growth in hardware computing performance, security threats have also increased rapidly. In addition, DES also has the problem of weak keys. If a weak key is used as an encryption key, an attacker will greatly increase the chance to crack the password. Although the DES method is old and cannot provide high security, it is fast and convenient. Therefore, many services still use symmetric DES as the main encryption mechanism. Therefore, this study will also adopt the symmetric encryption concept and then design a new encryption method to replace the shortcomings of DES.

The rest of the paper is organized as follows. Section 2 introduces the background and related works. Section 3 will give the problem definition via linear programming. Then we will present our proposed GA-based DES (GADES) method in Sect. 4. Finally, we will show the experimental results and summarize research contributions and discussing the future works in last two sections.

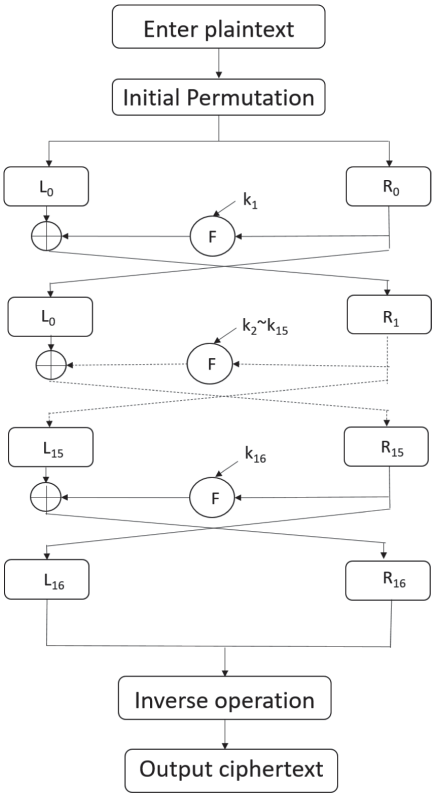

Fig. 1. DES encryption process.

2 Background and Related Works

2.1 Data Encryption Standard

DES main architecture is created from Feistel ciphers and rotated and placed through XOR. The method of DES encryption is to cut the plaintext into multiple blocks with 64-bits, and alternate each block to perform the encryption and decryption actions. If the plaintext is less than 64-bits, 0 is added until the block reaches 64-bits. The password is also 64-bit that the valid key is 56-bit and the remaining 8-bit is the check code. DES is 16-round encryption. The main process is to initialize the input plaintext block by an initial permutation to disrupt the original order of the data. Then cut into left part $L0$ and right part $R0$ which are 32-bits respectively. After $R0$ is extended and the sub-key $k1$ is calculated by $f(x)$, it is thrown into the Exclusive-OR (XOR) logic gate with $L0$. Then sequentially perform $L1 = R0$ and $R1 = L0 \oplus f(R0, k1)$ operations. The f function is to expand the block data from 32-bit to 48-bit and perform XOR operation with the sub-key and then corresponding output through Sbox. The sub-key generation process is that the user enters the 64-bit master key, and

selects the same bit check code through permutation selection, and then leaves the 56-bit master key and divides it into two blocks $C0$ and $D0$. Then after the left-hand bits get the merged block of $C1$ and $D1$, and then make a permutation selection, a 48-bit subkey $\{k1, k2, ..., k16\}$ can be obtained. The detailed process is shown in Fig. 1. However, because the DES key generation method is too standardized, the password we choose has a higher probability of weak keys during the replacement process, resulting in the use of the encryption key to lose the effect of encryption. This situation is not what we like.

2.2 Metaheuristics

Metaheuristic algorithm is a solution search method in the field of artificial intelligence (AI). It solves the combinatorial optimization problem in acceptable time and space. Although it is not guaranteed that the best solution can be obtained, it can be approximated in relative time. The genetic algorithm (GA) is a classical metaheuristic algorithm which used in this research is an algorithm that simulates natural evolution. The concept of GA is derived from Darwin's theory of evolution: "The survival of the fittest, the elimination of the unfit" [10]. In GA, each solution is called a chromosome, and each chromosome is composed of several genes. Through GA's native mechanisms which are selection, crossover, and mutation, GA can balance global search and local search. In other words, GA has the ability to converge in a more diverse environment. Based on this perspective, this study attempts to map key generation problem to a pure solution search problem and then using GA's diverse search feature to avoid falling into the local optimum so that generating weak keys to find the keys with more high security.

2.3 GA-Based Encryption Mechanism

Many scholars have proposed the use of GA for key generation, such as [5]. The author uses GA to generate the key. The first step must be the design of a fitness function so that the key can be distinguished from the chromosome. Because GA is a random process, the generated key can largely avoid the possibility of weak keys generation. However, the author uses Shannon entropy as the fitness function so that even if the process of key generation is based on a random process, the continuous characters in the key are continuously increasing so that Shannon entropy will still determine that it is a legal and secure key combinations. Although the crossover and mutation of GA can avoid this situation for legal keys, it is impossible to avoid this situation for keys that are originally chaotic. In short, GA cannot completely change the key combination, and still has a certain chance to encounter the deliberately designed attack. Therefore, this study will not only make full use of GA characteristics to find the best key combination, but also design the fitness function to exclude any potential factors that the key is maliciously cracked as far as possible.

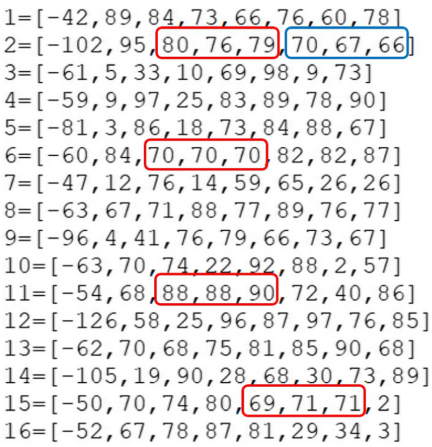

Fig. 2. 16 keys generated by GA-based encryption [5].

3 Problem Definition

The fitness function based on Shannon entropy cannot effectively rule out the generation of continuous keys. The so-called continuous key means that there is only a slight translation relationship in the key combination, so the gap between adjacent bits is very small. TIn other words, the key combination is often a gap of several bits so that the appearance of the keys is relatively similar. We can find from Fig. 2 that 16 pieces of data were randomly obtained from the 192 pieces of data generated using GA-based encryption [5] that the cases of continuous key combinations will exist in these samples. We can find that there are serious problems in the four records that the gap of less than 4 occurred between two adjacent positions that are marked with the red boxes and the blue box. These results are easy to be solved regularly and brute force attacks, which are also called continuous keys.

In order to visualize the similarity relationship of the keys, we converted the plaintext and the key into the form of bytes and mapped them to the Euclidean space as shown in Fig. 3. The red line is the plaintext and the black line is the generated key. The area enclosed in the middle is smaller which means that both of them have a higher similarity. Therefore, to determine whether the key is good or bad as long as just calculate the size of the area formed between the two lines. The normalized Linear Programming model can be shown as

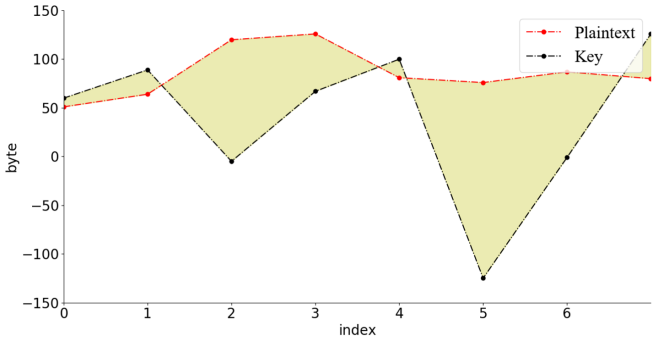

Fig. 3. Key similarity model based on area. (Color figure online)

Maxmize S

s.t.

$$0 \leq \alpha_{i,j} \leq 255$$
$$\beta \geq 0$$
$$\gamma \geq 0$$

The area S can be defined as:

$$S = (1+\gamma)\sum_{i=0}^{n} a_{i,j} \times (1+\beta), \tag{1}$$

Where S is the total area. γ is the number of times the plaintext line and the key line. $\alpha_{i,j}$ is the area of line segments i to j. n is the length of the key. In order to highlight the characteristics of the area and the case where there is no intersection, we define the β variable:

$$\beta = \begin{cases} \frac{|\alpha_{i,k} - \alpha_{k,j}|}{127}, & \text{Segments} i,j \text{ has intersection } k \\ 0, & \text{otherwise} \end{cases}, \tag{2}$$

Line segment $\bar{ij}$ will be divided into the two areas by the reference point k. Because the two curves may have an oscillating relationship, this feature will affect the similarity judgment, so the two areas are subtracted to take the absolute value and divided by 127. If the line segment $\bar{ij}$ does not have an intersection point, it indicates that the two curves may not have a oscillating relationship, which indicates that the similarity is large, and the value is 0. At this moment, the generated solutions are at their extreme values. Although they will oscillate, they are still too easy to be deciphered due to regularity. Therefore, in order to avoid that the generated solutions are always at their extreme values, we first determine the value in the key to the opposite value, and decide whether

to accept the penalty mechanism according to the threshold τ:

$$\tau = 127 - r, \tag{3}$$

The random value r ranges from 0 to 10, and 127 is the maximum value of each unit. Assuming a value greater than τ indicates that it may be closer to the extreme. If more bits of the key are close to the extreme value, the resulting area may seem large, but it is unbelievable. Therefore, a punishment mechanism must be used. When a number approaches the extreme value, the lower the key score:

$$S = S \times \omega^x, \tag{4}$$

x is the number of occurrences where the value is greater than τ, and ω represents a penalty value between 0 and 1. This study is set as 0.88. The intention is to make the negative response of this value greater than τ decrease exponentially. It means that the number of occurrences is more so that the area is smaller and the influence is also lower.

4 Proposed Mechanism

The main process of the proposed GADES mechanism in this study is to convert the plaintext into bytes then map to 2-D Euclidean space. GA is a main component for key generation. Next, the generated key and the plaintext will be converted into bits to operate XOR to obtain the ciphertext. This ciphertext will be treated as the plaintext and then perform XOR operation with the next key. This process will be repeated 16 times sequentially. The complete process is shown in Fig. 4.

The GADES will first randomly generate 8 chromosomes as the parent population. Each gene is a random number between $-128 \sim 127$. Every chromosome must calculate its fitness value. Then all of the chromosomes will enter the three GA key step:

Algorithm 1. GADES Algorithm

```
Input : bog
Output : key_i(byte)
Randomly generate the initial 8 keys;
repeat
    Fitness function calculation;
    Selection;
    Crossover;
    Mutation;
    Reproduction;
until Termination condition is met;
```

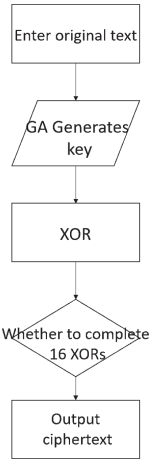

Fig. 4. GADES encryption process.

1. **Selection:** Two chromosomes will be selected from the parent population as the father and the mother for operating crossover by tournament selection.

2. **Crossover:** The chromosome is first converted into binary, and a random number is selected from 0–62 as the intersection point.

3. **Mutation:** Randomly select 10 positions for mutation. The mutation method changes 0 to 1 and 1 to 0. Finally, replace the two chromosomes which have the worst fitness value in the parent population.

Finally, the best key can be reproduced that is shown in Algorithm 1.

5 Experimental Results

5.1 Experimental Setup

This experiment uses java 8 for encryption algorithm implementation and uses Intel i7 8700 h processor as the main computing platform. In order to maintain high fairness, we use the original DES algorithm published on GitHub [6]. The existing work [5] used GA-based encryption which adopted the same hardware setting with this study as well as the GA parameter uniformly uses 12 size parent population and executes 16 rounds. Each round is executed 200 times. The selection operation of GA adopts tournament selection. The crossover and mutation triggering ratios are 95% and 50% respectively.

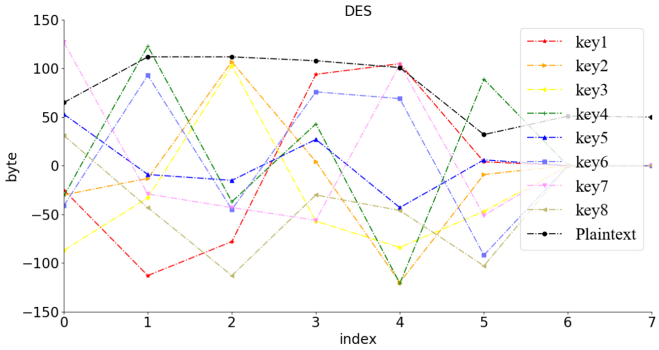

Fig. 5. Similarity of DES keys (1–8)

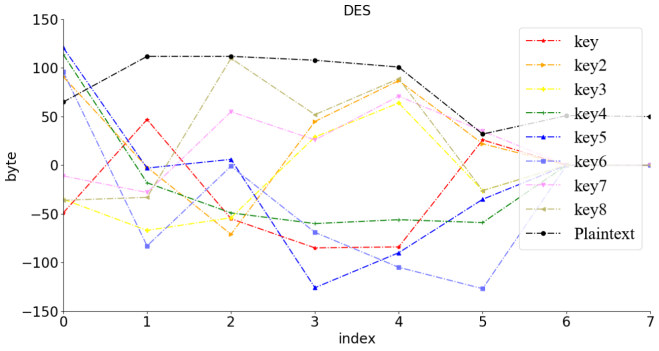

Fig. 6. Similarity of DES keys (9–16)

5.2 Experimental Results

Figures 5, 6, 7, 8, 9 and Fig. 10 show the similarity of the 16 keys generated by the original DES, GA-based encryption [5] and our proposed GADES. From Fig. 5 and Fig. 6, we can find that although there are some larger areas occurred between DES curve and the plaintext curve, occasionally, there are a few sets of keys that still have the smaller size of areas. It means that the security distribution of the original DES is uneven, causing some keys to be safe, but others are easy to be guessed.

GA-based encryption can improve security through a random process, but there is still a chance to generate a continuous combination. If someone with a bad intention thorough understanding of the GA process, it is still possible to expose the data to the danger that is shown in Fig. 7 and Fig. 8.

As shown in Fig. 9 and Fig. 10, the area of GADES has a large difference. It indicates that the similarity between the keys and the keys are small, so the probability of each key being guessed is low. Additionally, in the case of the plaintext is **"Apple 32"**, the interleaving situation of 16 sets of solutions generated by our GADES algorithm is more significant than DES. It also avoids

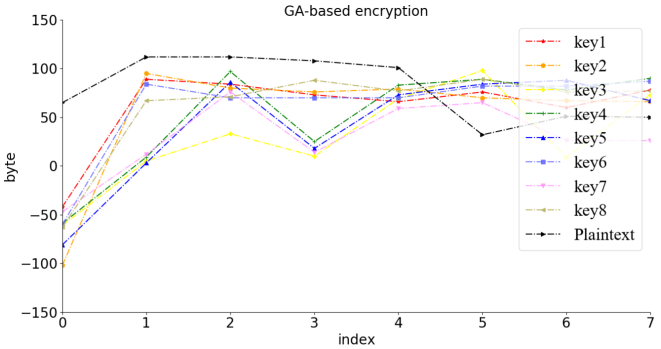

Fig. 7. Similarity of GA-based encryption keys (1–8)

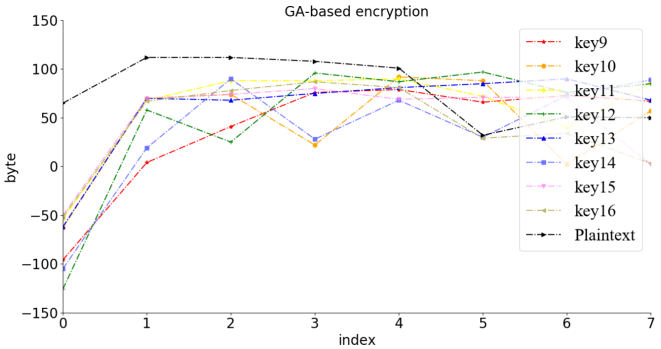

Fig. 8. Similarity of GA-based encryption keys (9–16)

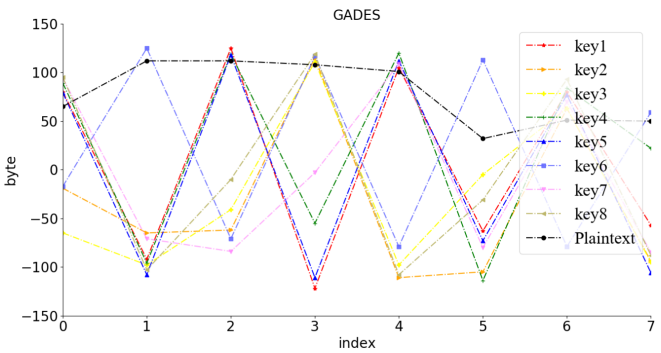

Fig. 9. Similarity of GADES keys (1–8)

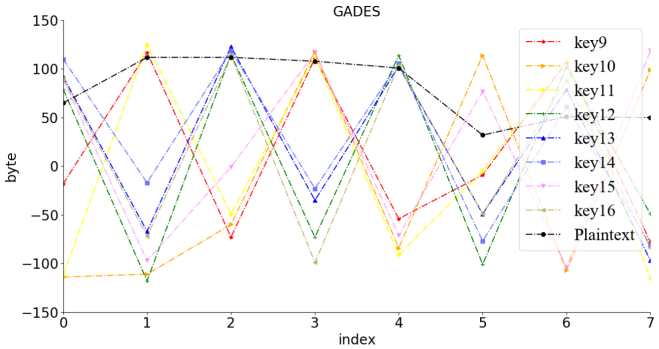

Fig. 10. Similarity of GADES keys (9–16)

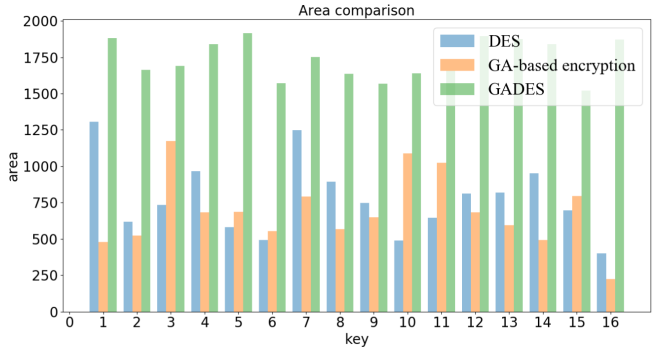

Fig. 11. Comparison of original area

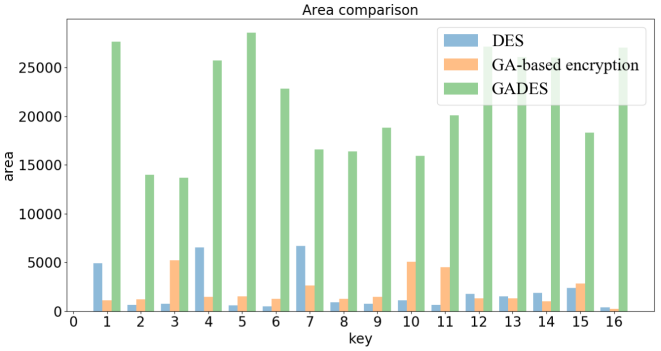

Fig. 12. Area of GADES with new fitness function

the appearance of weak keys. Because the designed fitness function will penalize key combinations which over the threshold, each key is intersected with the plaintext curve, and there is more than one intersection, and the highest points of bits of the key do not frequently approach 127 or −128. In this way, the complexity of keys is increased and be reduced the risk of cracking. It is in line with the idea we designed at the time.

Figure 11 shows the original area. The area of GADES is also larger than the other two methods. Because GADES has a higher degree of variability and diversity so that the probability of being cracked is greatly reduced. Figure 12 summarizes the area of the key combination. In order to increase the characteristics of GADES, we use the Eq. (4) to exclude similar features to highlight the similarity of each key. It can be seen that GADES has the largest dissimilarity.

6 Conclusion

In this paper, a new idea is proposed based on the genetic algorithm to generate the key. The key and the plaintext are limited to 64 bits and the XOR operation is repeated 16 times, which makes the key combination more diversified. And improved the defects of GA-based encryption which using Shannon entropy to design the fitness function that the linear continuous keys have a larger opportunity to occur. The experimental results show that our proposed GADES can significantly improve the security of data.

Although this method can increase the complexity of each key, when the 16 keys are finally combined, it can be found that some of the same values sometimes appear so that although it does not cause too many problems, it also greatly reduces the diversity of the overall portfolio. In the future, we will design corresponding mathematical functions to solve this problem. In addition, we will try to expand the length of the key and the plaintext to increase the difficulty of being cracked to increase security.

Acknowledgment. This research was partly funded by the National Science Council of the R.O.C. under grants 108–2221-E-197 -012 -MY3 and MOST 107–2221-E-197-005-MY3.

References

1. Suliman, A., Husain, Z., Abououf, M., Alblooshi, M., Salah, K.: Monetization of IoT data using smart contracts. IET Netw. **8**(1), 32–37 (2019)
2. Min, Z., Yang, G., Wang, J., Kim, G.: A privacy-preserving BGN-type parallel homomorphic encryption algorithm based on LWE. J. Internet Technol. **20**(7), 2189–2200 (2019)
3. Boussif, M., Aloui, N., Cherif, A.: Secured cloud computing for medical data based on watermarking and encryption. IET Netw. **7**(5), 294–298 (2018)
4. Zhang, C., Cho, H.-H., Chen, C.-Y., Shih, T.K., Chao, H.-C.: Fuzzy-based 3D stream traffic lightweighting over mobile P2P network. IEEE Syst. J. **14**(2), 1840–1851 (2019)

5. Nazeer, M.I., Mallah, G.A., Shaikh, N.A., Bhatra, R., Memon, R.A., Mangrio, M.I.: Implication of genetic algorithm in cryptography to enhance security. (IJACSA) Int. J. Adv. Comput. Sci. App. **9**(6), 375–379 (2018)
6. DES. https://github.com/nkengasongatem/java-des-mplementation. Accessed Dec 2019
7. Needham, R.M., Schroeder, M.D.: Using encryption for authentication in large networks of computers. Commun. ACM **2**, 993–999 (1978)
8. Raja, S.P.: Secured medical image compression using DES encryption technique in bandelet multiscale transform. Int. J. Wavelets Multiresolut. Inf. Process. **16**(04), 1850028 (2018)
9. Estrellado, M.E.L., Sison, A.M., Tanguilig III, B.T.: Test bank management system applying rasch model and data encryption standard (DES) algorithm. Int. J. Mod. Educ. Comput. Sci. **8**(10), 1–8 (2016)
10. Tsai, C.W., Rodrigues, J.J.: Metaheuristic scheduling for cloud: a survey. IEEE Syst. J. **8**(1), 279–291 (2014)
11. Chen, C., Chao, H.: A survey of key distribution in wireless sensor networks. Secur. Commun. Netw. **7**(12), 2495–2508 (2014)
12. Deng, L., Yang, Y., Chen, Y., Wang, X.: Aggregate signature without pairing from certificateless cryptography. J. Internet Technol. **19**(5), 1479–1486 (2018)
13. Niu, Y., Zhang, J., Wang, A., Chen, C.: An efficient collision power attack on AES encryption in edge computing. IEEE Access **7**, 18734–18748 (2019)
14. Bellare, M., Rogaway, P.: Optimal asymmetric encryption - how to encrypt with RSA. In: Proceeding of Eurocrypt, pp. 92–111 (1994)
15. Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 537–554. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48405-1_34
16. Marin, L.: Fast generation of DES-like S-boxes. J. Internet Technol. **17**(2), 301–308 (2016)
17. Chao, H.-C., Cho, H.-H., Shih, T.K., Chen, C.-Y.: Bacteria-inspired network for 5G mobile communication. IEEE Netw. **33**(4), 138–145 (2019)