ISSN No. 0976-5697

Volume 6, No. 5, May - June 2015

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

BCES Encryption Algorithm

Khdega A.Yosef Galala
Computer Science Department,
Faculty of Education, Waddan, Sirte University,
Waddan, Libya

Abstract: A fundamental problem in cryptography is how to communicate securely over an insecure channel. This problem has been there for a long time and has become even more important especially with the proliferation of computers and communication systems. Achieving information and communications security in an electronic society requires a vast array of technical and algorithms, without which the information security objectives deemed necessary cannot be adequately met. For this reason, this paper aims to propose a new algorithm to improve the security level. That is called as BCES algorithm. Thus, the rest of this paper is organized to offer an overview of basic issues of cryptography and describes in detail a new algorithm.

Keywords: BCES, encryption, public key, decryption, algorithm, ASCII

I. INTRODUCTION

In an encryption algorithm the sender and the receiver are the two main parties which must involve. Encryption algorithm can be done in two main stages; they are encryption and decryption. First stage is called encryption and it starts whenever a sender wants to send a message, called the plaintext to the receiver, and converts it to an encrypted form, called the ciphertext, finally sends it to the receiver. However, the method used to convert the plaintext into a cipher text (by performing various substitutions and permutations on it) which is called the encryption algorithm[1]. This is shown in Figure 1.

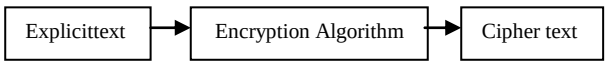

Figure1. An encryption algorithm

In the second stage, upon receiving the cipher text, the receiver uses a decryption algorithm to recover the original plaintext from the ciphertext it received. The method used to convert the ciphertext into a plaintext is called the decryption algorithm. In sample means decryption algorithm takes the ciphertext and the secret key and produces the original plaintext. This is shown in Figure 2.

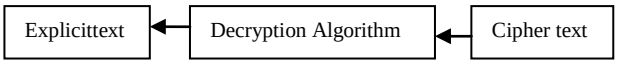

Figure2. A decryption algorithm

In addition, to finished encryption performance, it also required to generate encryption key. A key in encryption is a long sequence of bits used by encryption and decryption algorithms[1]. The keys used by the encryption and decryption algorithms are called the encryption key and the decryption key respectively. The decryption key should always be kept secret and, for this reason, it is also called the secret key[2].

Although the major issue to design any encryption and decryption algorithm is to improve the security level there are several objectives to design it; some of them are; it aims to data integrity: to ensure data wasn't altered between sender and recipient. Another aim is confidentiality: to ensure data is remained private (read only by authorized parties). Confidentiality is usually achieved using encryption that is because cryptography is used to convert plain text into cipher text and the equivalent decryption algorithm is used to convert the cipher text back to plain text. Lastly, Authentication: to ensure data originated from a particular party[2].

II. BRIEF HISTORY OF ENCRYPTION

Encryption has a long history and it goes back to at least Egyptian times about 4000 years ago. Moreover, Caesar's shift cipher was introduced more than 2000 years ago. In the 1970s, Martin Hellman, Whitfield Diffie, invented a powerful cryptographic idea. Their idea was to solve the key exchange and trust problems of symmetric encryption by replacing the single shared secret key with a pair of mathematically related keys, one of which can be made publicly available and another one must be kept secret by the individual who generated the key pair. Furthermore, Ron Rivest, Adi Shamir, and Leonard Adleman invented the RSA cipher in 1978; response to the ideas proposed by Hellman, Diffie, and Merkel, and it is widely used Public-Key algorithm[2]. The more common secret-key Encryption algorithm used today is the Data Encryption Standard (DES), designed by IBM in the 1970s and adopted by the National Bureau of Standards (NBS) [now the National Institute for Standards and Technology (NIST)] in 1977 for commercial and unclassified government applications in the early 1970 s[3]. Another class of powerful public-key schemes was found by ElGamal in 1985. In addition, in 1991 the first international standard for digital signatures (ISO/IEC9796) was adopted. It is one of the most significant contributions provided by public-key cryptography. In 1994 the U.S. Government adopted the Digital Signature Standard, a mechanism based on the ElGamal public key scheme[2].

III. TYPES OF ENCRYPTION

There are several ways of classifying encryption algorithms. For the purposes of this paper, they will be categorized into two basic types of encryption: symmetric algorithms ("private key") and asymmetric algorithms ("public key").

a. *Symmetric-key algorithm (Secret Key Encryption)*

Symmetric key algorithms are the quickest and most commonly used type of encryption. Symmetric key algorithms require both the sender and the recipient to have the same key [4]. This key is used by the sender to encrypt the data, and again by the recipient to decrypt the data. Most well-known symmetric encryptions are DES (data encryption standard) and AES (advanced encryption standard).

b. *Asymmetric-key algorithm (Public Key Encryption)*

Another type is called asymmetric key encryption or public key encryption. It is used to solve the problem of key distribution. In this type each user has both a public key and a private key [5]. The public key is used for encryption and private key is used for decryption. The most widely used symmetric key cryptographic method is the RSA. However, asymmetric algorithms have several advantages the main advantage of an asymmetric algorithm with respect to a symmetric algorithm is that the parties do not need to agree on a common key before communicating, but it has the disadvantage of being hundreds of times slower than symmetric algorithms [4]. This study adapted symmetric key algorithm that a single key is used for both encryption and decryption. This paper describes a new symmetric key algorithm in detail. The advantages of this new algorithm are also explained in the next section.

IV. NEW SYMMETRIC KEY ALGORITHM

In this section the implementation of Binary Count Encryption System (BCES) algorithm is introduced. In this way, Section A: includes explanation of encryption algorithm, Section B includes explanation of decryption algorithm and Section C includes explanation of decryption key. The structure of this algorithm can be explained by the following steps:

A. *Encryption algorithm*

The Encryption process involves converting the original data to its encrypted form. The proposed encryption algorithm consists of the following processes as explained below:
Step 1: Read a character from the file of explicit text.
Step 2: Generate the ASCII value of that character.
Step 3: Generate the corresponding binary value of it.
Step 4: Divide the binary data into blocks in which each block equals 8 bits.
Step 5: Last step in this algorithm is process of a counting and a testing (at the same time, count the total of similar bits, and followed by the type of bits, which is zero or one).

a. *For a counting process*

A count process in proposed encryption algorithm means calculating the total of similar and contiguous bits, stop counting and print the total when the first different bit appearance. In this step, two counters of Boolean type are used as such:

- Use flag1 for excellence the bit equal zero.
- Use flag2 for excellence the bit equal one.

b. *For a testing process*

The testing process is to determine the type of bits (the bits countered in the previous step), for this purpose the ASCII table was divided into two parts as follows:

1) To determine the type of bits equal zero, generate a random number from the ASCII table between (160-254).
2) To determine the type of bits equal one, generate a random number from the ASCII table between (59-157).
3) Numbers from (1-59) are exceptional.

So, this would be the cipher text or encrypted text. Figure3. shows steps followed in the proposed encryption algorithm.

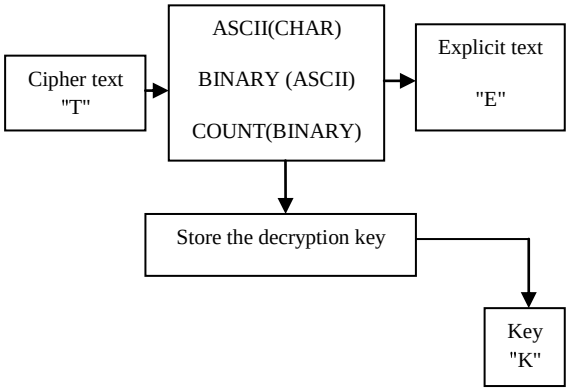

Figure3. Encryption in BCES algorithm

However, the encryption process can be expressed as:

$$F(T) = (E, K)$$

Where T: explicit text, E: cipher text, F: encryption algorithm, K:decryption key.

B. *Decryption Algorithm*

The decryption process involves converting the encrypted data back to its original form. The proposed decryption algorithm consists of the following processes as explained below:
Step 1: Read pairs of characters from the file of ciphertext.
Step 2: Generate the corresponding binary value of them. As follows: converting the second variable into binary data using ASCII code, if the number between (160 -254) return zero else return one. Then increase the number of the bit (0 or 1) produced in the previous step, according to the value of the first variable.
Step 3: Divide the binary data into blocks in which each block equals 8 bits.
Step 4: Generate the ASCII value of the binary data.
Step 5: Generate the corresponding ASCII code of that value.

So, this would be the original text or the explicit text. Figure 4 Shows steps followed in the proposed decryption algorithm.

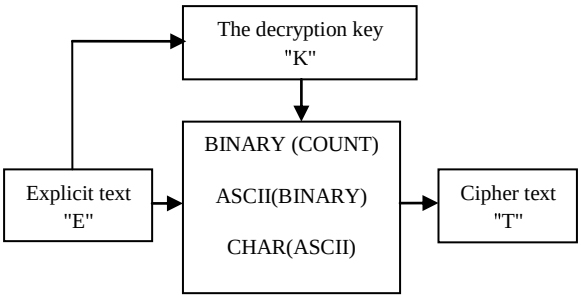

Figure4. Decryption in BCES algorithm

The decryption process can be expressed as:

$$F(T) = (E, K)$$

Where T: explicit text, E: Cipher text, F: encryption algorithm, K: encryption key.

C. *BCES Key Generation*

Generating keys is extremely important. If the security of an encryption system is reliant on the security of keys then clearly care has to be taken when generating keys. In addition, designing key length also is essential for any encryption algorithm that is because a large key will indeed be more difficult to break under most circumstances, on the other hand, encrypting and decrypting data with large keys may have bad effects on the system. However, in the BCES algorithm the length of the decryption key could generate efficiently from both long and short messages. It's also not fixed because it depends on the length of the message. The decryption key algorithm proposed in this paper is introduced in four different steps. They are:

Step 1: Count the number of lines in the ciphertext.

Step 2: Count the number of characters in the cipher text.

Step 3: Divide the number of characters on the number of lines multiplied by2.

D = character/(2*rows)

Step 4: Compare the result produced in the previous step. If it is less than 35, we need to make it 35. After that, repeat from the first digit produced in the previous step and quotient in next 8 digits. Then that get and store the squareness of the result produced in the previous step.

```
If D < 35 then
Key=D+8
for ( i=D; i<D+8; i++)
Key=i*i
```

Otherwise, if the result produced in the previous step is greater than 35, repeat from the first digit produced in the previous step and quotient in next 8 digits. After that get and store the squareness of the result produced in the previous step.

```
If D >= 35 then
For( i=D; i<D+8; i++)
Key= i*i
```

a) Example

This example explains how we could generate the key from short message.

Assuming that the message length is one line and number of characters in the message are 3. Now, according to the steps above we will get the following:

Step 1: Dividing the number of characters on the message by the number of lines multiplied by 2.

D = int (3/(2*1))=1.5

Step 2: the result produced in the previous step is compared with 35. Since it is less than 35, we need to make it 35 according to this algorithm, so it would be 36 (added 1).

D < 35 THEN 35+1=36

Step 3: repeat from the first digit produced in the previous step and quotient in next 8 digits.

for (i=D; i < D+8; i++)

(36,37,38,…43)

Step 4: get and store the final values of the key by getting the squareness of the result produced in the previous step.

Key= i * i

(1296- 1369-1444-1521-1600-1681-1764-1849)

b) Another case study

This example explains how we could generate the key from a long message.

It is assumed that the message contains 400 lines, each line contains 80 characters. Now, according to the steps above, we will get the following:

Step 1: calculate the number of characters in the message.

400x80=32000 Characters.

Step 2: Dividing the number of characters on the number of lines multiplied by 2.

D= int (32000/(2*400))

int (32000)/(800)=40

Step 3: the result produced in the previous step is compared with 35, and it is greater than 35.

40>35

Step 4: repeat from the first digit produced in the previous step and quotient in next 8 digits.

For (i=D; i<D+8; i++)

(40,41,42,43,…47)

Step 5: get and store the final value of the key by getting the squareness of the result produced in the previous step.

Key = i * i

(1600,1681,1764,1849,1936,2025,2116,2209)

V. ADVANTAGES OF THE NEW ALGORITHM

The main advantages of the new algorithm are as follows:

1. The BCES algorithm is fast and unbreakable for both encrypting and decrypting processes.
2. The several complex operations which are present in this algorithm would make it meet necessary security adequately.
3. Key-length is easily changed and makes it hard to break and hard to guess. This offers better security.

VI. CONCLUSION

With the increasing number of users connected to large communication networks, such as the Internet, many more of security communication problems exist. In order to avoid them various techniques and algorithms are developed. Among these techniques, the encryption algorithms appear as one of the most likely to be used and they are growing significantly. This paper introduced a new algorithm for complex encrypting and decrypting data. It falls under secret key encryption algorithm. It offers a very good and efficient security to the data which is needed to be protected.

VII. REFERENCES

[1] V. Magesh Babu, T. Shankar Ganesh, K. Ramraj. A Comparative Analysis on Encryption and Decryption Algorithms. International. Journal of Scientific and Research Publications, Volume 4, Issue 12, December 2014 1 ISSN 2250-3153.

[2] Alfred J. Menezes, Scott A. Vanstone, Paul C. van Oorschot . Handbook of Applied Cryptography. CRC Press, 1996.

[3] Idrizi, Florim, Dalipi, Fisnik & Rustemi, Ejup. Analyzing the speed of combined cryptographic algorithms with secret and public key. International Journal of Engineering Research and Development, e-ISSN: 2278-067X, p-ISSN: 2278-800X, Volume 8, Issue 2, pp. 45

[4] Abdul.Mina, D.S, Kader, H.M. Abdual & Hadhoud, M.M. Performance Analysis of Symmetric Cryptography.pp.1.

[5] Sunitha K, Prashanth K.S. Enhancing Privacy in Cloud Service Provider Using Cryptographic Algorithm. IOSR Journal of Computer Engineering (IOSR-JCE) e-ISSN: 2278-0661, p- ISSN: 2278- 8727Volume 12, Issue 5. pp. 64.

