Al-Mustansiriyah Journal of Science
ISSN: 1814-635X (print), ISSN:2521-3520 (online) Volume 31, Issue 3, 2020 DOI: http://doi.org/10.23851/mjs.v31i3.728

Research Article Open Access

Propose a new approach for key generation based on Chicken Swarm Optimization and HTML Parser

Suhad Malalla*, Ayat Hadi Ali

Department of Computer Science, University of Technology, Baghdad, IRAQ.

*Correspondent contact: suhad_malalla@yahoo.com

Article Info

Received
20/08/2019

Accepted
03/04/2020

Published
20/08/2020

ABSTRACT

The random key plays a basic role in the design of any cryptographic algorithms. In this paper, a proposed model for generating a random key is presented, which will be used for security purposes. Chicken swarm optimization (CSO) and HTML parser are used for scattering the bits of the key. The statistical tests of randomness have given good and acceptable security results. The proposed key generation method was programmed by JavaScript language.

KEYWORDS: Cryptography; Key generation; Parser; Chicken swarm optimization; HTML code; Symmetric Key.

INTRODUCTION

More technology is developed, so more necessary security is required. Everyone knows the dangers of technology that are used by some people in wrong way [1].

To know what is the key? The key is a set of random bits; the algorithm uses the key to add randomization to the encryption process [2]. It's important to think about how to generate a key and make it safe from the attacker. Keys must be created randomly [3]. There are three steps for creating encryption keys: "Obtain a key generator object for the algorithm you want to use, Initialize the key generator, and Ask the key generator to generate a key or key pair" [4].

Encryption is a process of transforming the data to another shape, that achieving security by encoding messages. There are two types of encryption: Symmetric key cryptography uses the same key for encryption and decryption. The second type is use two different keys, one key used for encryption and another different key used for decryption known as asymmetric key cryptography [5].

The parser or syntax analyzer takes the input from the lexical analyzer and check the syntax by depending on the grammar [6].

HTML code contains tags and attributes that are used to generate the proposed key in this paper.

Swarm intelligence is a branch of artificial intelligence that studies the collective behavior of self-organized systems. The main idea came for swarm intelligence from the study of the behavior of animals in nature [7].

Chicken Swarm Optimization (CSO) is one of the main swarm intelligence. In CSO there are sundry groups, each group include a dominant rooster, a couple of hens, and chicks [8]. In this paper, we will depend on the roosters because the best chickens would be assumed to be roosters and it gives the best fitness rather than the hen or chick.

In this paper a new approach for key generation that used for security purpose (symmetric key cryptography) is proposed. The proposed approach is based on CSO and HTML parser.

The rest of the paper is organized as follows: The second section will present the related works. The third section concerned with discussing the proposed method.

Experimentation and results are demonstrated in section four and the last section provides the main conclusions.

RELATED WORK

There have been many researches in the field of key generation, some of them are:

- In 2013, H. B. Abdul Wahab et at. [9], proposed an approach to generate key (mask key) for symmetric cryptography that relied on combine between the meta-heuristic algorithms (GRASP and VNS) and the features of elliptic curve arithmetic, Discrete Logarithm. This method has yielded good and effective results.
- In 2015 H. A. Zaki [10], proposed a method for key generating, It is depends on fingerprint features to protect the system from hackers. The proposed technology consists of two parts, The first part is "the EPROM memory filled with information of fingerprint after processed by the enhancement, binarization and thinning operations and then 512 numeric values has been extracted." and the second part is a set of linear shift registers.
- In 2015 A. K. Abdul Hassan [3], proposed an approach for key generation based on the bases of RNA translation to protein chain. This method has the key size (9, 18, 27, ...) Where a key chain is generated with an appropriate length along with the message length.
- In 2016 J. Zhang et at. [11], proposed an approach for wireless key generation, where a comprehensive survey will be conducted for the principles, performance metrics and key generation procedure. This method is a promising technology for sharing cryptographic keys securely between users with authority. Emphasis was placed on techniques of key generation systems.
- In 2019 Wirawan and Suwadi [12] proposed a method to generating a secret key (SKG) scheme, where they behave an inquiry on PHYSEC which utilizes channel exchange in generating a secret key. Through this proposed idea, a secret key will be generated between two users who have authority and do not need to pass through the information reconciliation step.

THE PROPOSED KEY GENERATION APPROACH

Figure (1) illustrates the block diagram for the proposed key generation approach.

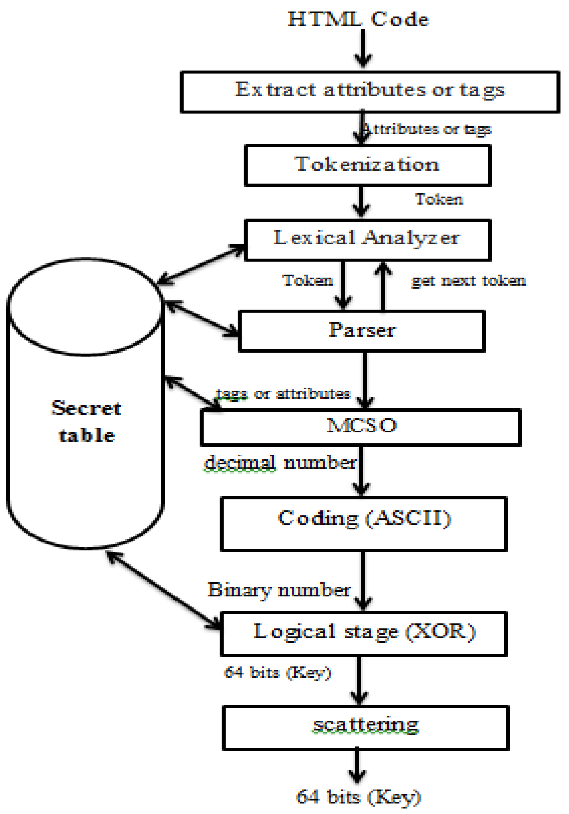

Figure 1. Block diagram for proposed key generation.

The input of this stage is a code of HTML file (such that the sender and receiver know the URL of this HTML file) that used to extract the privet information between the sender and the receiver, and the output of this stage will be 64 bits that represents the generated key.

The idea of generating the key is based on a special parser that depends on special rules about HTML tags arrangements (S is a secret table that represents the HTML grammar with its corresponding secret values with range between 100 and 254, see Table 1). Algorithm (`1`) represents the main steps of key generation.

The first step in this algorithm is uploading the HTML file that can be loaded from any website, then after extracting the main HTML code there is a need to normalize this code's letters into lower case. Tokenization will produce tokens of HTML code, lexical will make a matching between these HTML tokens with tags or attributes that found in the secret table (S), and the parser will check the pattern of these tags or attributes with the grammar found in table S and store their indexes in an array (A).

Now there is a need to compute the length of each specified tags or attributes (Their indexes found in A), with their indexes and their secret values (That found in S) in order to use the modified rooster equation of the CSO.

Table 1. secret table (S)

Tags and Attributes	Value
Width	133
Link	220
Align	189
Src	194
Shap	216
body	111
: : :	: : :

Rooster equation is one of CSO equations. The reason for choosing this equation because the roosters have better fitness values rather than hens or chicks. There is a need to make a modification on rooster equation because the rooster equation uses a random function and this function is not useful in generating the same value between the sender and the receiver. After the modifying, the equation becomes as below:

$$x_i = x * (1 + bj) \qquad (1)$$

$$bj = e\ ((FK1 - Fi1)/Fi1) \qquad (2)$$

- bj the random replaced by a value bj. Which is calculated from the equation (2)
- i represents a rooster's index and it represent the index for the tags or attributes.
- F represents the length of the tag or attributes.
- X has the value 8

The output from this equation will be converted to binary using its ASCII (after using round function). And in order to increase the complexity, the output is processed by XOR with the value for tags or attributes. These steps will be repeated eight times, each step produce 8 bits and as a result the 64 bits generated that represent the random key.

Lexical analysis is to read input line in the HTML code and produce tokens. Lexical analyzer scans the entire source code of the program. It identifies each token one by one. We obtain the next token is a command which is sent from the parser to the lexical analyzer. On receiving this command, the lexical analyzer scans the input until it finds the next token and it returns the token to Parser. The parser checks the token.

Then, applies the scattered for the 64 bits. Each 9 bits was scattered by rule (A) is the rule (2, 3, 1). Example to illustrate Scattering bits:

```
000101010   000010100   101010100   --------- the key
    1           2           3       ------------index

000101010      000010100      101010100     ------- the key
 1  2  3
 2  3  1         2  3  1        2  3  1     ------------rule

Apply Scattering

101010000      000010100      101010100     ------- the key
 2  3  1         2  3  1        2  3  1      ----------rule
```

Algorithm (1): The proposed key generation

```
Input: web URL (U)
Output: random key (k)
Process:
Step 1: Upload the HTML code.
Step 2: Extract the source code of the HTML page.
Step3: normalize HTML code into lower case, i=0.
Step4: while I < 8 do
4.1 Search for attributes or tags using table 1 (S).
4.2 If found attribute or tags then
4.2.1 i = i + 1
4.2.2 Store attributes or tags in array (A).
Step5: for j = 0 to 7 do
5.1: let fi be the length of A[i]
5.2: let FK = j
5.3: Apply the equation for Rooster equation(1) and equation (2).
5.4: Round the results (X).
5.5: convert x to binary code (B).
5.6: Apply XOR operation between the binary code (B) and with values that found in table1(S).
End for
Step 6: convert B to stream of 64 bit to get KK.
```

6.1: Take the first nine bits and each three bits from it their locations are switched by rule (A). It is one of the crossover ways.
6.2: Take the following nine bits and apply the same rule (A) to them and so on.
Step7: apply Scattering bits for 64 bit to get K.
Step8: return (K).
End.

The following steps represent processing that the proposed approach.

After uploading the HTML page and matching between it is with the secret table (grammatical rules) in order to get the tags or attributes, it was obtained:

1→img = 3
2→body = 4
3→background = 10
4→a = 1
5→font = 4
6→width = 5
7→link = 4
8→shap = 4

The length was calculated for each the tags and attributes. Then, apply the modified rooster equation (1 and 2).

Img→ X1=8 *(1 + bj)
bj= exp ((3-17)/17) = 2.71
X1=8 * (3.71) = 58.66
Round the number to 59
Convert the result to binary number
59=00111011

Body→ X1=8 *(1 + bj)
bj= exp((4-6)/6) = 2.71
X1=8 * (3.71) = 58.66
Round the number to 59
Convert the result to binary number
59=00111011

background→ X1=8 *(1 + bj)
bj= exp((10-15)/15) = 2.71
X1=8 * (3.71) = 58.66
Round the number to 59
Convert the result to binary number
59=00111011

a→ X1=8 *(1 + bj)
bj= exp((1-14)/14) = 2.71
X1=8 * (3.71) = 58.66
Round the number to 59
Convert the result to binary number
59=00111011

font→ X1=8 *(1 + bj)
bj= exp((4-27)/27) = 2.71
X1=8 * (3.71) = 58.66
Round the number to 59
Convert the result to binary number
59=00111011

width→ X1=8 *(1 + bj)
bj= exp((5-1)/1) = 10.87
X1=8 * (11.87) = 94.98
Round the number to 95
Convert the result to binary number
95=01011111

link→ X1=8 *(1 + bj)
bj= exp((4-2)/2) = 2.71
X1=8 * (3.71) = 58.66
Round the number to 59
Convert the result to binary number
59=00111011

shap→ X1=8 *(1 + bj)
bj= exp((1-5)/5) = 2.71
X1=8 * (3.71) = 58.66
Round the number to 59
Convert the result to binary number
59=00111011

After this process we apply the XOR between the results from the rooster equation with the value for each tags and attributes that used in the example (after convert to binary).

img	= 173	= **10101101**
body	= 111	= **01100100**
background	= 168	= **10101000**
a	= 128	= **10000000**
font	= 200	= **11001000**
width	= 133	= **10000101**
link	= 220	= **11011100**
shap	= 216	= **11011000**

Now, apply the XOR
1 = 00111011 XOR **10101101 = 01011111**
2 = 00111011 XOR **01100100 = 01011111**
3 = 00111011 XOR **10101000 = 10010011**

4 = 00111011 XOR **10000000 = 10111011**
5 = 00111011 XOR **11001000 = 11110011**
6 = 01011111 XOR **10000101 = 11011010**
7 = 00111011 XOR **11011100 = 11100111**
8 = 00111011 XOR **11011000 = 11100011**
Obtained the key 64 bits

Key=
0101111101011111100100111011101111111001111011010
1110011111100011
Key=0101111101011111100100111011101111111001111011
10101110011111100011

After the obtained the key we apply the scattering to the key

Key=
1111100101111101010111010101111111101110
1100111100101011000111111

DISCUSSION AND EXPERIMENTAL RESULTS

To verify the generated string. Random statistical tests used for this purpose. We will use the four basic statistical tests that benefit us for this purpose. They are: [13,14] Frequency test, Poker test, Serial test, Runs test. Statistical tests will be applied to the keys generated in the proposed method to determine how effective the method it is.

1) Frequency test: Is the first necessary test, as the rest of the tests depend on the success of this test. The purpose of this test is to determine the number of 0's as 1's in sequence are the same as expected in random sequence.
2) Poker test: In this test the m-bit groups are considered. sequence of m-bit groups of length n/m. the poker test defined the sequences of length m each appear approximately the same number of times in s, as would be expected for a random sequence.
3) Runs test: The aim of this test is to defined the number either 0's or 1's of various lengths string in the sequence S is as predictable for a random sequence.
4) Serial test: The aim of this test determines number of times of the 2mm-bit (00, 01, 10, and 11) Nested patterns in sequence are the same as expected in random sequence.
5) In each block the m is represent the length in bits.

Table 2. Experimental Results for Key Generation.

key	Key1	Key2	Key3	Key4	Key5	Pass value
Frequency test	Pass=1.000	Pass=2.250	Pass=0.563	Pass=0.063	Pass=0.063	must be <= 3.84
Runs test	T0=pass=2.500 T1=pass=2.500	T0=pass=4.750 T1=pass=2.625	T0=pass=1.125 T1=pass=8.250	T0=pass=7.375 T1=pass=4.750	T0=pass=4.000 T1=pass=12.250	must be <=12.309
Poker test	Pass=3.550	Pass=4.300	Pass=1.550	Pass=1.550	Pass=4.300	must be <=11.1
Serial test	Pass=5.000	Pass=5.250	Pass=0.750	Pass=6.250	Pass=3.250	must be <=7.81

The tests were performed for five keys, to determine the randomness ratio, which gave good results, as shown in Table 2. The way is good but not 100%. We also had other keys up to 5 keys, 3 succeeded and 2 keys failed. Of the total of 10 keys, 8 keys succeeded and two keys failed 80%.

This test was in the first iteration and when we changed the value of the one tag in the second iteration for both keys who failed, it is succeeded the two keys who failed first and gave very good results 100%.

CONCLUSIONS

The proposed key generation method that based on HTML parser and MCSO produces a good randomization (According to the four statistical tests), it is useful in generating the key. Depending on rooster's equation rather than hen's or chick's equation of CSO is suitable for key generation purpose because it has the best fitness. In addition to the Scattering of the bits that was born by the Parser. It is giving a good random scale according to the statistics table.

REFERENCES

[1] Hala B. , Suhad M. , and Ekhlas K., "Generate Cryptographic key using generated 3D- Digital Image", Eng. & Tech. Journal ,Vol.27, No.2, 2009.

[2] Israa T. and Shatha H., "Proposal New Approach for Blowfish Algorithm by Using Random Key Generator", Journal Of Madent Alelem College, Vol 4 No 1 Year 2012.

[3] Alia K., " Proposed Approach for Key Generation Based on the RNA ", Journal of the Faculty of Basic Education, 2015.

[4] Jonathan B., "Java Cryptography", Book, First Edition May 1998.

[5] Ayushi, "A Symmetric Key Cryptographic Algorithm", International Journal of Computer Applications (0975 - 8887) Volume 1 – No. 15, 2010.

[6] "Tutorial point", 2014 by Tutorials Point (I) Pvt. Ltd.

[7] Konstantinos E. and Michael N., "Particle Swarm Optimization and Intelligence Advances and Applications", 2010.

[8] Xianbing M., Yu L., Xiaozhi G., and Hengzhen Z., " A New Bio-inspired Algorithm: Chicken Swarm Optimization " International Conference, ICSI 2014, Hefei, China, October 17-20, pp. 86–94.

[9] Hala B. , Suhad M. , and Estabraq A., "Proposed Approach for Key Generation Based on Elliptic Curve (EC) Algebra and Metaheuristic Algorithms Eng. & Tech. Journal, Vol.32, Part (B), No.2, 2014.

[10] Huda A. Zaki," Cryptographic key Generation Using Fingerprint Biometrics", J.Thi-Qar Sci. Vol.5 (2) May/2015.

[11] Zhang, J., Duong, T. Q., Marshall, A., & Woods, R., " Key Generation from Wireless Channels: A Review", IEEE Access, 4, 614-626.,2016.

[12] Wirawan and Suwadi , "A Simple Secret Key Generation by Using a Combination of Pre-Processing Method with a Multilevel Quantization", Journal Entropy , Published: 18 February 2019.

[13] Andrew Rukhin, Juan Soto, James Nechvatal, Miles Smid, Elaine Barker, Stefan Leigh, Mark Levenson, Mark Vangel, David Banks, Alan Heckert, James Dray, SanVo, " A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications", National Institute of Standards and Technology Special Publication 800-22 Revision 1a, Revised:April2010

[14] Ioana R., “Statistical Assessment Of Binary Sequences Generated By Cryptographic Algorithms”, Social Economic Debates Volume 5, Issue 2, ISSN 2360-1973; ISSN-L 2248-3837, PP.23-31, 2016.