Available *online* at : http://bit.ly/InfoTekJar

InfoTekJar : Jurnal Nasional Informatika dan Teknologi Jaringan

ISSN (Print) 2540-7597 | ISSN (Online) 2540-7600

Cryptography

RC4 GGHN Cryptography Algorithm for Message Security

Shahira An-Nissa [1], *Herman Mawengkang*[2], *Syahril Efendi* [3]

[1] *Master Degree Informatics, Faculty of Computer Science and Information Technology, Universitas Sumatera Utara, Medan, Indonesia*
[2] *Faculty of Mathematics and Natural Sciences, Universitas Sumatera Utara, Medan, Indonesia*
[3] *Faculty of Computer Science and Information Technology, Universitas Sumatera Utara, Medan, Indonesia*

KEYWORDS

GGHN, Cryptography, Encryption, Decryption, Key Length, Messages

CORRESPONDENCE

Phone: +62811605879

E-mail: mawengkang@usu.ac.id

A B S T R A C T

Cryptography is a technique that is generally used in security in the process of exchanging information in the form of text messages, picture messages, or others involving two or more users. There are many types and classifications of cryptography developed to date that are able to provide security for the information sent. Modern cryptography that is popular and widely used is the Rivest Cipher (RC4) algorithm. RC4 cryptography is a type of stream cipher, which processes units or input data at one time. This research will use a combination of several message character lengths, namely 50, 100, 250, 500, 1000, 2500, 5000, 10000 characters, and tested using several key lengths, namely 5, 10, 25, 50, 100 characters. from testing with message length parameters and keys then the results will be compiled into a table and described into an image for easy understanding. Research results show it can be concluded that the length of the key used in encryption does not significantly affect the effectiveness of the system processing time. In addition, the length of the text has a big influence in determining the length of the system process in encrypting and decrypting messages. the more the number of characters that will be sent, the higher the processing time required to perform the security process once using the RC4 GGHN cryptography algorithm.

INTRODUCTION

Cryptography is a technique that is generally used in security in the process of exchanging information in the form of text messages, picture messages, or others involving two or more users. There are many types and classifications of cryptography developed to date that are able to provide security for the information sent [1]. One type of secure cryptography is asymmetric cryptography or commonly known as modern cryptography. Asymmetric cryptography uses an information security approach by using the same keyword to encrypt the plaintext and decrypt the ciphertext [2]. Symmetric encryption is the oldest and fastest encryption algorithm in terms of performance. Modern cryptography that is popular and widely used is the Rivest Cipher (RC4) algorithm [3][4]. RC4 cryptography is a type of stream cipher, which processes units or input data at one time[5]. In this way, both encryption and decryption can be performed at variable lengths [6]. This algorithm does not have to wait for a certain number of input data before processing or adding additional bytes to encrypt it. The RC4 encryption method is very fast, approximately 10 times faster than the Data Encryption Standard (DES) [7]. In several studies that review the RC4 algorithm, one of which has been developed is by finding two new ciphers, named Sheet Bend and Bowline, developed by expanding RC4 to 32 bits or from a byte-oriented model to a word-oriented model. This new algorithm is called RC4(n, m) [8]. Later, the name GGHN was adopted for this code after the initials of its designers namely Gong, Gupta, Hell, and Nawaz[9]. The GGHN algorithm has a processing time of 3-5 times faster than RC4 [10]. One source of GGHN's high security is the large size of its secret internal state. Several studies that analyzed the three RC4 floating algorithms namely RC4+, RC4 NGG and, RC4 GGHN stated that the RC4 GGHN algorithm provided a faster encryption time than the RC4+ algorithm by a difference [11]. Then the avalanche effect obtained by analyzing the five RC4 variants, namely VMPC, RC4+, RC4A, NGG, and GGHN using the SAC and BIC criteria which were extended to stream ciphers stated that the VMPC, RC4+, and GGHN variants have met both of these criteria which describe performance, security, and stable usability [12].

Based on the above background, the problem in this study is about how the performance of the application of the GGHN algorithm by using a combination of message lengths is used and then calculating the processing time of each testing stage.

https://doi.org/10.30743/infotekjar.v6i2.4531

LITERATURE REVIEW

GGHN cipher is an acronym for the name of the first character of the family name of four co-authors namely Gong, Gupta, Hell, and Nawaz [9] who then proposed a way to expand RC4 from a byte-oriented model to a word-oriented model [13]. The GGHN algorithm has two parts: Key Scheduling Algorithm (KSA) and Pseudo-Random Generation Algorithm (PRGA) as given below in Figure 1 and Figure 2. Due to the structure of RC4 being quite elegant, has attracted deep cryptanalysis in many studies, still, the cipher is quite safe for 128-bit key size if used with care [14]. One of the limitations of RC4 is that its output generation is in bytes (8-bits) and it is natural to extend the idea to word-oriented output (4 or 8 bytes in general) which will result in a faster keystream. A 32 or 64-bit processor is ready for such applications [15]. To store an 8-bit permutation, one needs only 28 bytes, but to store a 32-bit permutation, one requires a large array of 232 words (1 word = 4 bytes). Since this is impractical, there have been some attempts to work with small random arrays only, not complete random permutations. For example, one can refer to the cipher GGHN which does look as simple and elegant as follows:

Input: S-Box S: an m-bit integer array of $N = 2^n$ locations, initially loaded with certain predefined value;
Input: Key array K: an m-bit integer array of l locations;
Input: An integer r for number of rounds;
Output: Pseudorandom S-Box S and a pseudorandom m-bit integer k;
$j = 0, k = 0, t = 0, M = 2^m$;
while $t < r$ **do**
  for $i = 0, \ldots, N-1$ **do**
    $j = (j + S[i] + K[i \bmod l]) \bmod N$;
    Swap $S[i] \leftrightarrow S[j]$;
    $S[i] = (S[i] + S[j]) \bmod M$;
    $k = (k + S[i]) \bmod M$;
  end
  $t = t + 1$;
end

Fig. 1. GGHN-Key Scheduling Algorithm (KSA) [16]

Input: S-Box S and integer k from GGHN-KSA(n, m);
Output: m-bit Keystream words z
$i = 0, j = 0, M = 2^m$;
while *Keystream is generated* **do**
  $i = (i + 1) \bmod N$;
  $j = (j + S[i]) \bmod N$;
  $k = (k + S[j]) \bmod M$;
  $z = (S[(S[i] + S[j]) \bmod N] + k) \bmod M$;
  $S[(S[i] + S[j]) \bmod N] = (k + S[i]) \bmod M$;
end

Fig. 2. GGHN-Pseudo-Random Generation Algorithm (PRGA) [16]

METHOD

At this stage, the methodology for conducting tests using variations in the number of characters will be explained to determine the performance of the RC4 GGHN cryptography algorithm as follows:

A. Studying literature

At this stage regarding theoretical studies that support research

B. Data collection

Collects files containing message characters with different alphabet of 50, 100, 250, 500, 1000, 2500, 5000, 10000 characters. Collects random key for encryption and decryption with different alphabet of 5, 10, 25, 50, 100

C. Analysis and Testing Methods

At the data analysis stage, the settlement will be carried out using the RC4 GGHN cryptographic algorithm. The solution steps include:

1) Perform calculations and record comparison of key length with processing time
2) Perform calculations and record processing time with different lengths of text
3) From testing with message length parameters and keys then the results will be compiled into a table and described into an image for easy understanding

RESULT AND DISCUSSION

These experiments are conducted on the Windows 10 Ultimate which has 64-bit architecture Intel Core i5 processor and 4096 MB RAM. The results of testing with messages that have a varying number of characters is presented in table 1 and the results of testing with message length variation can be seen in table 2 as follows:

Table 1. Key Length Variables

No	Key Lentgth	Key
1	5	pQoZ9
2	10	E3oohZUGB3
3	25	swmukhXjAEuWswSpq9EACNUQc
4	50	PADCLYYFxe2M58ZAU0xDphRT RYfy5V1voGnT2M1upV1fqsiTfQ
5	100	A3pppvAIcrizwAIB5Sn0U7 fyGIdkyu9jxpMaSKOq43 EyfnfJIiVNSr8jFHMpTa RrWG4FBmMhZDK5TRm9 u5FZTZs5RnqClTugwdwy

Table 2. Message Length Variation

No	Text Length	Message
1	50	Cryptography is the study of secure communication
2	100	Cryptography is the study of secure … sender and intend
3	250	Cryptography is the study of … associated to encryption
4	500	Cryptography is the study … microdots or merging. Ancient
5	1000	Cryptography is the study … recipient for decryption
6	2500	Cryptography is the study … that way only you can
7	5000	Cryptography is the study … do not reveal the identity
8	10000	Cryptography is the study … of data through a one-way

 https://doi.org/10.30743/infotekjar.v6i2.4531

Table 3. Encryption Running Time Testing

Encryption	Text Length	Key Length				
		5	10	25	50	100
1	50	0,0449	0,0412	0,0438	0,0471	0,0401
2	100	0,0503	0,0493	0,0581	0,051	0,0527
3	250	0,0627	0,0578	0,0615	0,0597	0,0642
4	500	0,0735	0,0689	0,0797	0,0715	0,0795
5	1000	0,0819	0,0846	0,0827	0,0801	0,0869
6	2500	0,0983	0,0936	0,0954	0,0921	0,0918
7	5000	0,1041	0,1019	0,1121	0,119	0,1095
8	10000	0,1205	0,1245	0,1195	0,1301	0,1384

Table 3 describes the test using a combination of key length and text length. From the above test, it can be seen that the processing time value increases with the increasing number of characters. but the processing time with the key combination does not affect the encryption process time.

Table 4. Decryption Running Time Testing

Decryption	Text Length	Key Length				
		5	10	25	50	100
1	50	0,0367	0,0359	0,0337	0,0319	0,0268
2	100	0,0417	0,0484	0,0427	0,0426	0,0397
3	250	0,076	0,0712	0,0788	0,0729	0,0727
4	500	0,1532	0,1597	0,1499	0,1252	0,1539
5	1000	0,1986	0,1914	0,1836	0,1944	0,1982
6	2500	0,2337	0,2355	0,2334	0,2312	0,2837
7	5000	0,2712	0,2981	0,2682	0,2802	0,2638
8	10000	0,3198	0,3754	0,3137	0,3073	0,3275

Table 4 describes the test using a combination of key length and text length. From the above test, it can be seen that the processing time value increases with the increasing number of characters. but the processing time with the key combination does not affect the encryption process time.

When compared to the encryption and decryption process, it can be seen that the characters of 50 and 100 encryption have a longer processing time than decryption. however, at 250 to 1000 characters, encryption has a shorter time than decryption. so that it can be concluded that the encryption process with a large amount of time has a more concise time and the decryption process with fewer characters will result in a shorter processing time than encryption

Based on tables 3 and 4 above, the results of testing with a combination of key lengths with varying message text lengths have been obtained. To make it easier to understand, the description is explained in graphic form which can be seen in figure 3 – 7 for encryption and figure 8-13 for decryption.

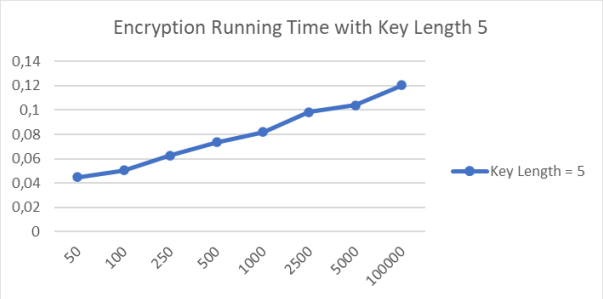

Fig. 3. Encryption Running Time Testing with Key Length 5

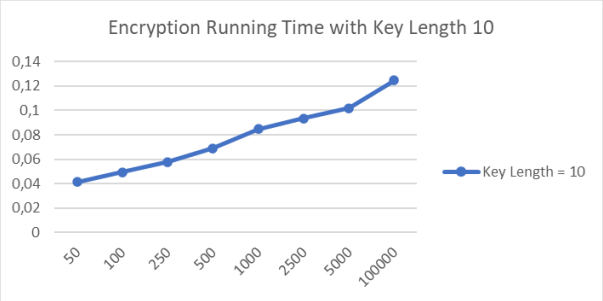

Fig. 4. Encryption Running Time Testing with Key Length 10

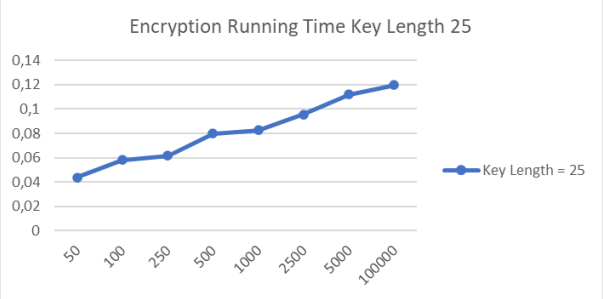

Fig. 5. Encryption Running Time Testing with Key Length 25

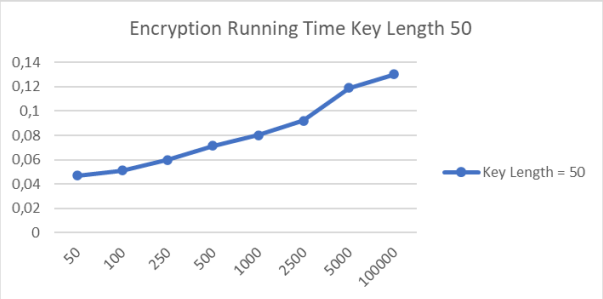

Fig. 6. Encryption Running Time Testing with Key Length 50

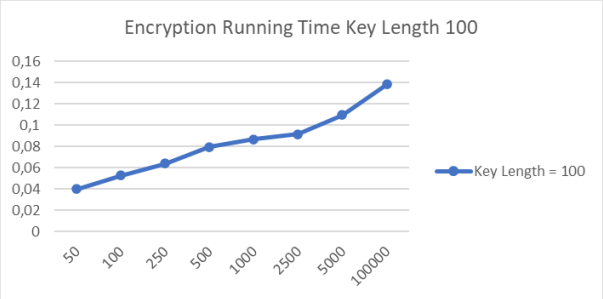

Fig. 7. Encryption Running Time Testing with Key Length 100

The combination of testing with several variations of the length of the text and the length of the key found that the length of the key used for the encryption process does not affect the running time but the length of the text does affect it. it can be seen from figure 3-7 which shows an increase in processing time as the number of text characters tested increases.

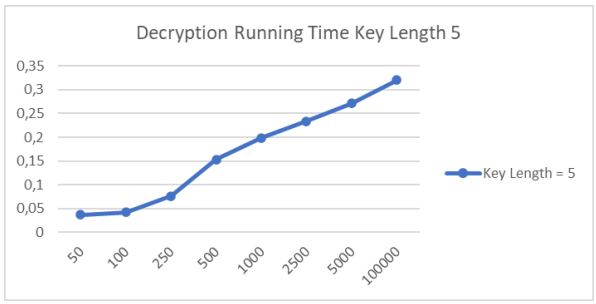

Fig. 8. Decryption Running Time Testing with Key Length 5

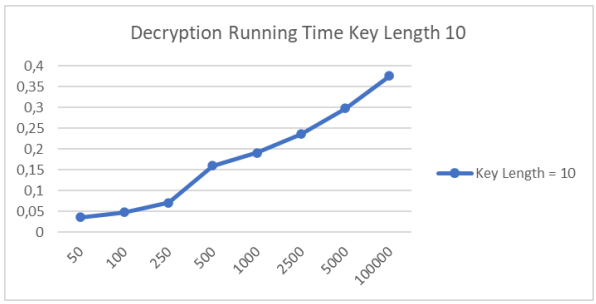

Fig. 9. Decryption Running Time Testing with Key Length 10

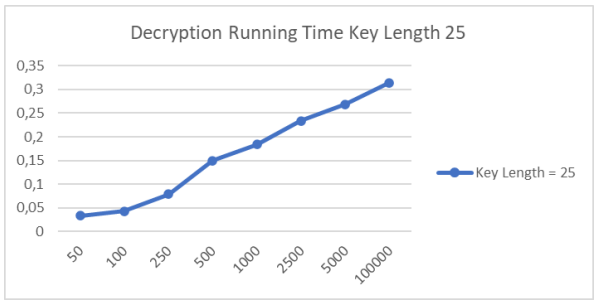

Fig. 10. Decryption Running Time Testing with Key Length 25

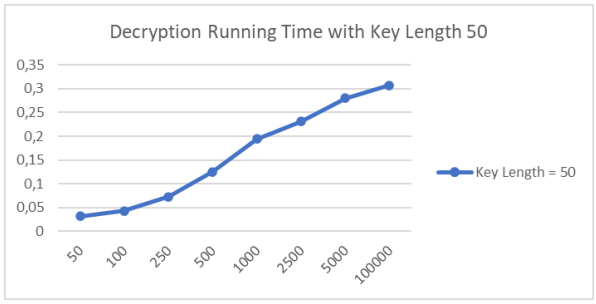

Fig. 11. Decryption Running Time Testing with Key Length 50

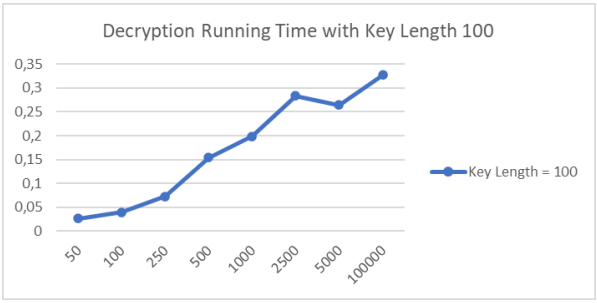

Fig. 12. Decryption Running Time Testing with Key Length 100

CONCLUSIONS

The combination of testing with several variations of the length of the text and the length of the key found that the length of the key used for the decryption process does not affect the running time but the length of the text does affect it. it can be seen from figure 8-2 which shows an increase in processing time as the number of text characters tested increases.

Based on the analysis of data testing in research, as a material for further research development, it can encrypt several types of files at once and add a picture-in-text feature to secure a more varied message

REFERENCES

[1] D. R. I. M. Setiadi, E. H. Rachmawanto, C. A. Sari, A. Susanto, and M. Doheir, "A Comparative Study of Image Cryptographic Method," *Proc. - 2018 5th Int. Conf. Inf. Technol. Comput. Electr. Eng. ICITACEE 2018*, pp. 336–341, 2018, doi: 10.1109/ICITACEE.2018.8576907.

[2] M. A. Budiman, Amalia, and N. I. Chayanie, "An Implementation of RC4+ Algorithm and Zig-zag Algorithm in a Super Encryption Scheme for Text Security," *J. Phys. Conf. Ser.*, vol. 978, no. 1, 2018, doi: 10.1088/1742-6596/978/1/012086.

[3] R. Saha, G. Geetha, G. Kumar, T. H. Kim, and W. J. Buchanan, "MRC4: A Modified RC4 Algorithm Using Symmetric Random Function Generator for Improved Cryptographic Features," *IEEE Access*, vol. 7, pp. 172045–172054, 2019, doi: 10.1109/ACCESS.2019.2956160.

[4] A. Khalid, G. Paul, and A. Chattopadhyay, "RC4-AccSuite: A Hardware Acceleration Suite for RC4-Like Stream Ciphers," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 25, no. 3, pp. 1072–1084, 2017, doi: 10.1109/TVLSI.2016.2606554.

[5] D. E. Sari, H. N. N. Muchsin, D. R. I. M. Setiadi, C. A. Sari, and E. H. Rachmawanto, "Hybrid Encryption Technique using Cyclic Bit Shift and RC4," in *2019 4th International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE)*, Nov. 2019, pp. 205–209, doi: 10.1109/ICITISEE48480.2019.9003848.

[6] N. Hayati, "Implementasi Algoritma RC4A dan MD5 untuk Menjamin Confidentiality dan Integrity pada File Teks," *J. Penelit. Tek. Inform.*, vol. 1, no. April, pp. 51–57, 2017.

[7] R. A. Mollin, *An introduction to cryptography, second edition*, 2nd Editio. New York: Chapman and Hall/CRC, 2006.

[8] M. A. Orumiehchiha, J. Pieprzyk, E. Shakour, and R. Steinfeld, "Cryptanalysis of RC4(n, m) stream cipher," *SIN 2013 - Proc. 6th Int. Conf. Secur. Inf. Networks*, vol. 4, pp. 165–172, 2013, doi: 10.1145/2523514.2523615.

[9] G. Gong, K. C. Gupta, M. Hell, and Y. Nawaz, "Towards a General RC4-Like Keystream Generator," vol. 2, 2005, pp. 162–174.

[10] A. Kircanski, R. Al-Zaidy, and A. M. Youssef, "A new distinguishing and key recovery attack on NGG stream cipher," *Cryptogr. Commun.*, vol. 1, no. 2, pp. 269–282, 2009, doi: 10.1007/s12095-009-0012-4.

[11] F. Akbar, H. Mawengkang, and S. Efendi, "Comparative analysis of RC4+ algorithm, RC4 NGG algorithm and RC4 GGHN algorithm on image file security," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 420, no. 1, 2018, doi: 10.1088/1757-899X/420/1/012131.

[12] E. J. Madarro-capó, C. M. Legón-pérez, and O. Rojas, "applied sciences Measuring Avalanche Properties on RC4 Stream Cipher Variants," 2021.

[13] B. Subhadeep, "Some Studies on Selected Stream Ciphers . Analysis , Fault Attack & Related Results by Subhadeep Banik," *(Doctoral Diss. Indian Stat. Institute, Kolkata)*, no. February, 2015.

[14] B. Mondal, N. Sinha, and T. Mandal, "A secure image encryption algorithm using LFSR and RC4 key stream generator," *Smart Innov. Syst. Technol.*, vol. 43, pp. 227–237, 2016, doi: 10.1007/978-81-322-2538-6_24.

[15] A. Khalid, G. Paul, and A. Chattopadhyay, "Study of Flexibility," 2019, pp. 127–168.

[16] D. J. Bernstein and S. Chatterjee, *Progress in Cryptology – INDOCRYPT 2011*, Vol 7107., vol. 7107. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011.