UTILIZING RC5 ENCRYPTION TO OPTIMIZE CONTENT SECURITY IN WEB APPLICATIONS

Wirpan Atmaja Putra, Alldino Allban Prakasa, Yohanes Dwi Cahyono
Department of Cyber Security Engineering, Army Polytechnic, Batu-Malang, East Java, Indonesia
wirfan.azillia28@gmail.com

ABSTRACT

The rapid advancement of information and communication technology has led to an increased use of web applications, but also brought challenges in data security. The widely used HTTP protocol remains vulnerable to threats such as data theft and cyber-attacks. One solution to mitigate these issues is the use of client-side encryption methods, such as the RC5 algorithm, which provides high flexibility in encryption parameter configurations. This research analyzes the implementation of the RC5 algorithm in enhancing web application data security. A web application prototype utilizing RC5 encryption was developed and tested for encryption time, decryption time, and memory consumption. Test results indicate that the RC5 algorithm enhances data security without significantly compromising application performance. However, an increase in processing load was observed on low-specification devices. This study also highlights the importance of adjusting RC5 parameters to balance security and performance.

Keywords : *encryption, RC5, data security, web application, HTTP, client-side encryption*

1. INTRODUCTION

Technology advances and the increasing use of web applications in everyday life bring new challenges in terms of data security. The HTTP protocol which is widely used for communication on the internet is still vulnerable to security threats, especially because data is often sent in plain text form. This issue may result in the leak of sensitive information, such as login credentials and personal data. Although client-side encryption methods have been widely proposed to improve security, their implementation requires further development, especially in utilizing efficient encryption algorithms to provide optimal solutions in data protection[1]**.**

The Client-side Encryption method has the advantage of Increasing the user data security by ensuring the data is encrypted before leaving the client side, making it more difficult for third parties to infiltrate or steal information. However, the downside lies in increasing the processing load on the client side, which can affect application performance, especially on devices with low specifications[2]. Methods and tools are used to protect individual privacy from surveillance, with emphasis on the importance of encryption and secure communication practices [3][6].

Key management in the HTTPS ecosystem means that sharing private keys between websites and third-party hosting providers is a common practice driven by efficiency and economic factors. However, this may increase the risk centralization of trust and security vulnerabilities, especially when many websites rely on a few hosting providers. This research also highlights that certificates managed by third parties tend to be more secure than those managed independently, but improvements in certificate management practices are needed to reduce the risk of sharing private keys[4].

The RC5 algorithm is a symmetric encryption method known for its simplicity and flexibility in parameter configuration, such as block size, key length, and number of rounds. These advantages make RC5 the ideal choice to secure data through fast and efficient encryption and decryption. Compared with other algorithms such as DES, RC5 exhibits shorter processing times to provide better security for data communications in web applications that require high performance. However, there is a limitation on the number of rounds applied, namely only 16 rounds, which can be a weak point in terms of security. Therefore, adjusting the number of rounds is important to increase the algorithm's resilience to more complex cryptographic attacks. Through its flexibility, RC5 offers a customizable solution to meet data security needs, consistently maintaining a balance between processing speed and protection against security threats [5][9].

Previous research shows that HTTP protocol weaknesses can be exploited by various types of attacks, including Man-in-the-Middle (MITM) and cookie theft, which have a significant impact on user security [1]. Additional encryption methods, such as the RC5 algorithm, have been proposed to increase data protection during transmission. RC5 is known for its flexibility in parameter configuration and efficiency of the encryption-decryption process, making it a relevant choice for integration on modern web servers and web browsers [2][10].

However, the main challenge in implementing encryption algorithms on web servers and browsers is how to maintain a balance between security and performance levels. Research by Zuhar Musliyana et al. showed that client-side encryption can reduce the risk of data theft during transmission, but can increase the processing load on the user's device [3]. Therefore, a solution is needed that is able to optimize performance without reducing the level of security.

2. LITERATURE REVIEW

2.1. Protocol HTTP

The HTTP protocol (Hypertext Transfer Protocol) is the main protocol in web communications that uses text as its data format. Because of its plaintext nature, HTTP is easy to intercept and manipulate. To increase security, the HTTPS (HTTP Secure) protocol was developed by integrating SSL/TLS to encrypt transmitted data. Research shows that although HTTPS offers higher security than HTTP, this protocol is still vulnerable to attacks such as Man-in-the-Middle (MITM) and SSL stripping[2][3].

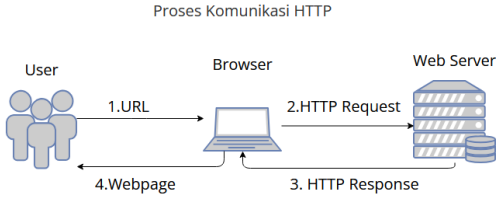

Figure1. HTTP Protocol Connetion Process

2.2. Stream Encryption

Stream encryption is an encryption method where data is encrypted per bit or per byte sequentially, making it suitable for applications that require real-time data transmission. The RC5 algorithm, which is the focus of this research, can be applied in stream encryption mode to produce fast encryption and decryption processes. Research shows that RC5 provides high performance with security levels that can be adjusted through parameters such as number of turns and key length [7][8].

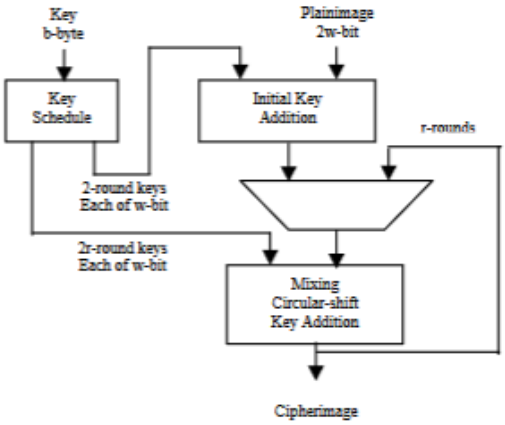

Figure 2. RC5 Encryption Algorithm

3. RESEARCH METHODS

The method involves building several layers of encryption security and also developing a web server prototype with the RC5 algorithm.. This stage involves collecting information from various journals and Scientific articles related to the RC5 algorithm, HTTP/HTTPS protocol, steganography, and stream encryption.

$$A = B = 0, \quad i = j = 0$$

Ulangi 3 x T kali:

$$A = \text{ROTL}(S[i] + A + B, 3)$$

$$B = \text{ROTL}(L[j] + A + B, (A + B)\%W)$$

$$i = (i + 1)\%T, \quad j = (j + 1)\% \left(\frac{b}{4}\right)$$

Figure 3. Random subkeys are generated using rotation and modular operations.

Web application prototype with implementation of the RC5 algorithm for data encryption. This prototype is designed to utilize a combination of steganography and stream encryption as an additional layer of security. Testing was carried out to measure the efficiency of the RC5 algorithm in web applications, including encryption time, decryption time, and resource consumption. Testing also includes attack simulations to identify the algorithm's resilience to various threats. The encryption process uses the RC5 algorithm with parameters www (word size), rrr (number of rounds), and bbb (key length):

C=ERC5(K,P)

CC = Ciphertext (encrypted data)
KK = Encryption key
PP Plaintext (original data)
ERC5ERC5 = RC5 encryption function

Figure 4. Encryption Process Using the RC5 Algorithm

P=DRC5(K,C)

DRC5DRC5 = RC5 encryption function

The decryption process to retrieve the plaintext:

Figure 5. Decryption process to obtain Plaintext.

Performance testing involves measuring the following parameters:

Resource Consumption Efficiency:
R=CM
R=MC

Tenc, TdecTenc.Tdec = Encryption and decryption time
RR = Resource consumption efficiency ratio
CC = Encrypted data capacity
MM = Resources used (e.g. CPU or memory)

Figure 6. Performance Evaluation Process.

The formulas above can be used as a basis for prototype implementation describing the web application prototype process with the implementation of the RC5 algorithm which utilizes a combination of steganography and stream encryption.

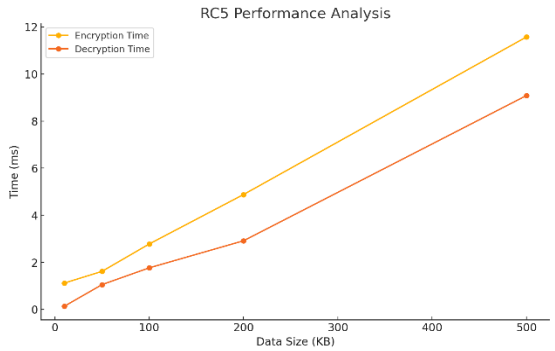

Figure 7. RC5 Performance Analysis on Various Data Sizes

This figure 7, can be based on analysis data using RC5 with various data sizes. Encryption Time and Decryption Time are measured in milliseconds for various data sizes (in kilobytes). Performa Analysis on various data size shows how processing time increases with the size of the data being encrypted or decrypted.

4. RESULTS AND DISCUSSIONI

Addition to the performance tests that have been carried out to measure encryption time, decryption time, and memory consumption on various data sizes, further tests were carried out to evaluate the security level of implementing RC5 encryption in web applications using the Wireshark network analysis tool. The purpose of this test is to ensure that data sent via web applications that have been encrypted using RC5 remains safe and protected from potential threats of data theft or unauthorized monitoring during transmission.

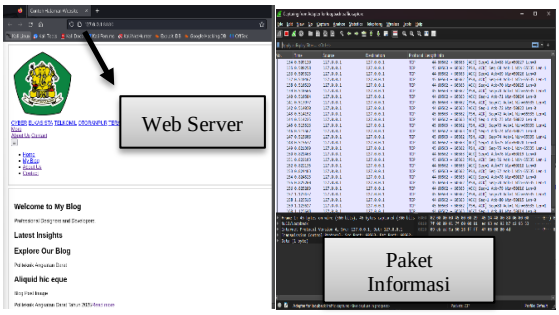

Figure 8. Web Server and Capturing Wireshark Results information related to captured packets.

The security test procedure with Wireshark begins with capturing data packets. After the web application prototype was run with RC5 encryption, Wireshark was used to capture data packets sent between the client and server. The protocol used is overlaid HTTP with RC5 encryption, and this communication is carried out via local IP 127.0.0.1:8080. Wireshark monitors packets sent during the encryption and decryption process, and analysis is performed to ensure that the data sent cannot be read without a proper decryption process.

The test results show that data packets sent after being encrypted using the RC5 algorithm cannot be read or understood by third parties, even though the data can be captured on the network. Even if the data is intercepted using Wireshark, the contents of the packet remain safe because the information contained therein has been encrypted with RC5, which has high complexity and resistance to basic cryptographic analysis attacks. Thus, implementing RC5 encryption has proven effective in maintaining the security of data sent via web applications, securing data transmission from potential threats on the network.

Table 1. Packets analyzed in Wireshark

No.	Time	Source IP	Destination IP	Protocol	Length	Info
1	0.001235	127.0.0.1	127.0.0.1	HTTP	150	HTTP Request (GET /index.html)
2	0.003245	127.0.0.1	127.0.0.1	HTTP	300	HTTP Response (200 OK)
3	0.005567	127.0.0.1	127.0.0.1	HTTP	200	HTTP Data (Encrypted using RC5)
4	0.007346	127.0.0.1	127.0.0.1	HTTP	220	HTTP Data (Encrypted using RC5)
5	0.009000	127.0.0.1	127.0.0.1	HTTP	180	HTTP Request (POST /submit)
6	0.011345	127.0.0.1	127.0.0.1	HTTP	250	HTTP Response (Encrypted data)

In the table above, the Info column shows that the data sent is the result of encryption using RC5, and although the packet can be captured using Wireshark, its contents remain unreadable without the appropriate decryption process. These packets will appear random and cannot be recognized without knowledge of the encryption key.

Even though communication uses HTTP, implementing RC5 encryption manages to secure data effectively. By using RC5, encrypted data cannot be read or manipulated without having the correct key for decryption. This encryption ensures that even if data is intercepted during transmission, the information contained within remains safe and protected. However, one of the challenges found was the increased processing load resulting from using RC5 encryption, especially on devices with low specifications. This can affect the user experience, especially in web applications that involve the transmission of large amounts of data, as encryption and decryption processes require additional time and resources.

5. CONCLUSIONS

From this research, it can be concluded that the RC5 algorithm has proven effective in improving data security in web applications by providing fast and efficient encryption and decryption. However, the processing load imposed on client devices needs to be considered, especially on devices with low specifications. Tests on various data sizes show that RC5 is able to maintain good performance without sacrificing security levels. For further research, it is recommended to optimize the RC5 implementation by adjusting parameters such as the number of

rounds and key length, as well as exploring the use of stream encryption mode for applications that require real-time data transmission. Further research could also be conducted to compare RC5 with other encryption algorithms in the context of web applications that require a balance between security and performance.

REFERENCES

[1] Zuhar Musliyana, Mahendar Dwipayana, Ayu Helinda, Zahrul Maizi, "Improvement of Data Exchange Security on HTTP using Client-side Encryption," Journal of Physics: Conference Series, vol. 1019, 2018, doi:10.1088/1742-6596/1019/1/012073.

[2] Sivakorn, S. (2016). An exploratory study on HTTP cookie hijacking attacks. 2016 IEEE Symposium on Security and Privacy. DOI: 10.1109/SP.2016.49.

[3] Freedom of the Press Foundation. (2013). Edward Snowden: NSA Files Whistleblower. Diakses dari uk/world/2013/jun/17/edward-snowden-nsa-files-whistleblower.

[4] Zuhar Musliyana, Mahendar Dwipayana, Ayu Helinda, Zahrul Maizi, "Improvement of Data Exchange Security on HTTP using Client-side Encryption," Journal of Physics: Conference Series, vol. 1019, 2018, doi:10.1088/1742-6596/1019/1/012073.

[5] Xiapu Luo. 2019. HTTPOS: HTTP or HTTPS with Obfuscation. Journal of Network Security.

[6] Pratama, A. R., Ichsan, M. H. H., & Kusyanti, A. (2019). Implementasi Algoritme AES Pada Pengiriman Data Sensor DHT11 Menggunakan Protokol Komunikasi HTTP. Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, 3(4), 3781-3789. http://j-ptiik.ub.ac.id

[7] Kurniawan, B. D., Rosid, M. A., Kautsar, I. A., & Pratama, N. E. (2023). Rancang Bangun Library Web Token untuk Enkripsi HTTP Data Menggunakan Eksklusif-OR (XOR). Physical Sciences, Life Science and Engineering, Volume: 1, Nomor 1, 2023.

[8] Nair, Dhanya B., & Maideen, Ruksana. (2013). "Chaotic Image Encryption Using RC5." International Journal of Image Processing and Vision Science, 1(4), Article 5. DOI: 10.47893/IJIPVS.2013.1047. Tersedia di: https://www.interscience.in/ijipvs/vol1/iss4/5

[9] Shinde, H. N., Raut, A. S., Vidhale, S. R., Sawant, R. V., & Kotkar, V. A. (2014). A Review of Various Encryption Techniques. International Journal Of Engineering And Computer Science, 3(9), 8092-8096. ISSN: 2319-7242.

[10] Hnin Yu Wai, Khine Myat Nwe. Encryption and Decryption by using RC5 And DES Algorithms for Data File. University of Computer Studies, Magway