Journal of Artificial Intelligence and Engineering Applications

Website: https://ioinformatic.org/

15th October 2025. Vol. 5. No. 1; e-ISSN: 2808-4519

Development of Dynamic Key Based on Pseudo-Random Algorithm in Vigenere Cipher for Hybrid Vigenere-ElGamal Encryption to Secure Documen Data

Rianda Putra[1*], Achmad Fauzi[2], Rusmin Saragih[3]

[1,2,3]*Informatics Engineering, STMIK Kaputama*
Jl. Veteran No. 4A-9A, Binjai, North Sumatra, Indonesia
riandaputra43@gmail.com[1*], fauzyrivai88@gmail.com[2], evitha12014@gmail.com[3]

Abstract

The security of digital documents has become increasingly critical in line with the growing threats to the confidentiality and integrity of information. This study aims to develop a hybrid cryptographic system that combines the Vigenère Cipher with dynamic keys generated through the Linear Congruential Generator (LCG) and the ElGamal algorithm. The application of dynamic keys in the Vigenère Cipher enhances randomness and reduces predictability, while the ElGamal algorithm strengthens protection through public-key cryptography. The system was designed and implemented as a desktop application using Visual Basic .NET Framework 4.0, supporting the encryption and decryption of PDF, DOCX, and XLSX files. Each data byte is processed according to the algorithm sequence stored in a hybrid file format, ensuring accurate decryption results. Experimental testing demonstrated that the system can successfully restore encrypted documents to their original form without data loss, thereby maintaining document integrity. However, the ciphertext size increased to nearly three times that of the original file due to the ElGamal mechanism, which produces two values for each byte. This study concludes that the hybrid Vigenère–ElGamal method with LCG-based dynamic keys can effectively enhance the security of digital document encryption, although it results in a larger ciphertext size.

Keywords: *ElGamal, Hybrid encryption, Dynamic key, Cryptography, Vigenère Cipher*

1. Introduction

The security and confidentiality of digital documents have become crucial factors in modern information systems, given the increasing risks of data theft and unauthorized access in line with technological advancements. Sensitive data such as personal identities, financial records, and classified documents can cause severe harm if misused by irresponsible parties. One of the most effective methods to address this issue is the application of cryptography, which conceals information into an unreadable form that can only be restored to its original state by authorized users.

Cryptography consists of two main processes: encryption and decryption. Encryption is the process of converting plaintext into ciphertext using a specific algorithm, while decryption is the process of transforming ciphertext back into its original form. Classical algorithms such as the Vigenère Cipher are often utilized due to their simplicity, which works by shifting each character based on a key value. However, the Vigenère Cipher with a static key has significant weaknesses, particularly due to its repetitive key patterns that make it vulnerable to cryptanalysis. To strengthen its security, a dynamic key generator such as the Linear Congruential Generator (LCG) can be employed to enhance randomness and reduce predictability in the encryption process.

On the other hand, modern algorithms such as ElGamal provide stronger security by employing public-key cryptography. ElGamal operates on modular arithmetic and the discrete logarithm problem, making it extremely difficult to break without the private key. Nevertheless, this algorithm also has limitations, particularly in producing ciphertext that is larger than the original plaintext, which may lead to inefficient storage usage. This indicates that relying on a single algorithm is often insufficient to balance efficiency and security, making hybrid approaches a more promising alternative.

Several previous studies have also highlighted the importance of combining cryptographic algorithms to enhance data protection. One such study was conducted by Ulfa Br. Mtd et al. (2017), which integrated the Vigenère Cipher with the One-Time Pad (OTP) for securing digital images. The results demonstrated that this combination successfully preserved data confidentiality and made the system more resilient to attacks compared to using a single algorithm. However, the method still had drawbacks, particularly in reduced performance when applied to large images, which limited its efficiency[1].

2. Theoretical Fondation

2.1. Cryptography

The term *cryptography* originates from the Greek words *kryptos*, meaning "secret," and *graphein*, meaning "to write." Thus, cryptography can be defined as the art of writing in a concealed manner. In general, cryptography is employed to ensure the confidentiality and integrity of messages within modern information systems[2].

Cryptography is the science of transforming data or messages into an unintelligible form for unauthorized parties. Beyond preserving confidentiality, cryptography also ensures integrity, authentication, and non-repudiation, thereby allowing only those with the correct key to restore the message to its original form[3].

2.2. Hybrid Cryptosystem

Hybrid A hybrid cryptosystem is a cryptographic method that combines two or more algorithms in order to leverage the strengths of each cryptosystem[4]. In general, this system combines classical or symmetric algorithms, which are efficient in the data encryption process, with asymmetric algorithms, which provide stronger key management[5].

This approach aims to enhance security while maintaining efficiency, ensuring that data can be securely encrypted yet remain practical for real-world applications.

2.3. Vigenere Cipher Algorithm

The Vigenère Cipher is one of the classical cipher algorithms that operates by substituting each character in the message with a different shift according to the sequence of the key [6]. This technique belongs to the category of polyalphabetic substitution, as the letter substitution process is not uniform but instead depends on the length of the key used. Its advantage lies in its ability to obscure text patterns more effectively compared to monoalphabetic ciphers. However, it still presents vulnerabilities when the key is repetitive, as this allows for potential cryptographic analysis[7]. The equation used to perform the computation in the Vigenère Cipher algorithm is as follows:

$$\mathbf{Ci = (Pi + Ki)\ Mod\ M} \quad \textbf{(1)}$$

Ci refers to the ciphertext, which is the transformed result of a plaintext character. Pi represents the plaintext, namely the original character before the encryption process. Ki denotes the key, obtained from the conversion table in the form of a decimal number that represents the shift value of the key character. M is the modulus used in both the encryption and decryption operations to ensure that the computation results remain within the specified range.

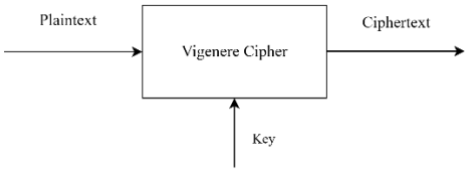

Figure 1: Vigenere Cipher Encryption

Figure 1. illustrates that the Vigenère Cipher requires an input in the form of plaintext and a key to transform the plaintext into ciphertext. In the decryption process, the Vigenère Cipher also requires the same key used during encryption in order to reconstruct the original plaintext. The key is then converted into a decimal number using a conversion table. The ciphertext (Ci) is also converted through the same table to produce a decimal value. Subsequently, the plaintext (Pi) can be retrieved using the corresponding equation:

$$\mathbf{Pi = (Ci - Ki)\ Mod\ M} \quad \textbf{(2)}$$

Pi represents the original plaintext of the encrypted message. Ci is the ciphertext produced through the encryption process. Ki denotes the key used in both the encryption and decryption processes. M is the modulus applied to ensure that the computation results remain within the defined range.

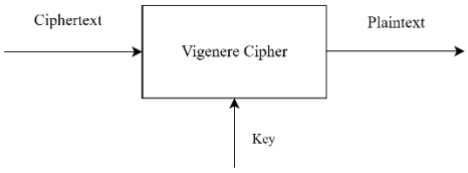

Figure 2: Vigenere Cipher Decryption

Figure 2. illustrates the decryption process of the Vigenère Cipher, which requires the ciphertext to be processed using the Vigenère decryption equation with the same key employed during encryption in order to restore the ciphertext to its original plaintext form.

2.4. ElGamal Algorithm

The ElGamal algorithm was introduced by Taher ElGamal in 1985 as a form of public-key cryptography. Initially, this algorithm was designed for digital signatures; however, it was later extended to support the encryption and decryption of messages. ElGamal operates on blocks of plaintext, which are encrypted into ciphertext blocks, then decrypted and recombined to reconstruct the original message. The security level of this algorithm relies on the computational difficulty of solving the discrete logarithm problem in very large prime numbers.[8].

The strength of the ElGamal algorithm lies in its key generation mechanism based on the discrete logarithm problem and its high level of security. However, its weaknesses include the need for significant computational resources and the fact that the ciphertext size can reach up to twice the length of the plaintext. The ElGamal algorithm employs several important parameters in its computation, particularly when generating public and private keys. These parameters are p, g, x, and y[9]. Y can be calculated using the equation y = g^x mod p. The encryption process in the ElGamal algorithm is divided into two components, namely *a* and *b*, where *a* can be calculated using the equation:

$$\mathbf{a = g^k \bmod p} \quad \textbf{(3)}$$

a represents the first value of the ciphertext. *g* is the generator chosen to be smaller than the prime number *p*. *k* is a secret random number that differs in each encryption process. *p* is a large prime number used as the modulus in the computation. Meanwhile, *b* can be calculated using the equation:

$$\mathbf{b = y^k . Mi \ Mod \ p} \quad \textbf{(4)}$$

b represents the second value of the ciphertext. *y* is the public key derived from the generator *g* and the private key *x*. *k* is the secret random number used during the encryption process. *M* denotes the message or plaintext to be encrypted. *p* is the large prime number used as the modulus in the computation.

Meanwhile, the decryption process can be calculated using the equation:

$$\mathbf{Pi = bi . ai^{(p-1-x)} \ Mod \ p} \quad \textbf{(5)}$$

2.5. Linear Congruential Generator

The Linear Congruential Generator (LCG) is one of the simplest methods for generating pseudo-random numbers. It is referred to as pseudo-random because the numbers produced appear random, but in fact can be predicted if the initial parameters are known[10]. Due to its lightweight nature and ease of implementation, LCG is widely used in educational applications, games, and quiz systems to randomize the order of questions or displays.

The working process of LCG begins by defining an initial value or seed along with three main parameters: the multiplier, the increment, and the modulus. The initial value is then processed using the LCG formula to generate the first pseudo-random number, which is subsequently used as the input for the next computation. This process continues iteratively, producing a sequence of numbers that appear random, with values always remaining within the modulus range[11]. LCG can generate pseudo-random numbers using the following equation:

$$\mathbf{X_{n+1} = (a \times X_n + c) \bmod M} \quad \textbf{(6)}$$

$X_{(n+1)}$ It represents the newly generated pseudo-random number. *a* is the multiplier constant. X_n refers to the previous pseudo-random number or the initial seed value. *c* is the increment constant. *m* is the modulus that defines the upper limit of the generated random numbers.

3. Research Method

3.1. Dinamic Key Generator LCG

In this study, the LCG algorithm is employed to generate pseudo-random numbers that are subsequently used as keys in the encryption and decryption processes of the Vigenère Cipher. The stages involved in generating random numbers using LCG are illustrated in Figure 3 as follows:

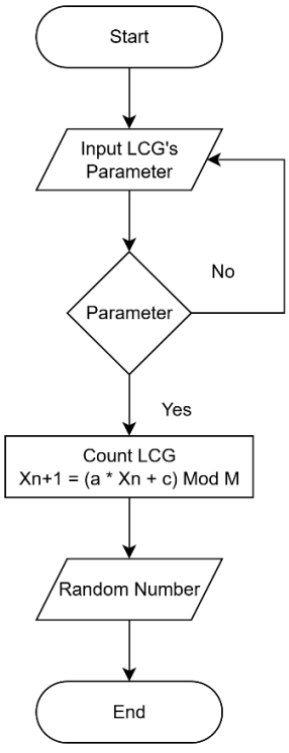

Figure 3: Dinamic Key Generator LCG Process

Figure 3. illustrates that LCG requires several parameters: the seed or initial value (X_0), *a* as the multiplier factor, *c* as the increment, and *m* as the modulus that regulates the computation results to generate pseudo-random numbers.

3.1.1. Dinamic Key Generator Process

The computation to generate pseudo-random numbers using LCG, which are then utilized as keys in the Vigenère Cipher algorithm, is as follows:

Seed (X_0) = 7
A = 5
C = 3
M = 257

The computation process is carried out as follows:

$X_1 = (5 \times 7 + 3) \bmod 257 = (35 + 3) \bmod 257 = 38$
$X_2 = (5 \times 38 + 3) \bmod 257 = (190 + 3) \bmod 257 = 193$
$X_3 = (5 \times 193 + 3) \bmod 257 = (965 + 3) \bmod 257 = 968 \bmod 257 = 197$
$X_4 = (5 \times 197 + 3) \bmod 257 = (985 + 3) \bmod 257 = 988 \bmod 257 = 217$

3.2. Hybrid Encryption

Hybrid cryptography is a technique that combines two cryptographic algorithms to work together. In this study, the algorithms used are the Vigenère Cipher and ElGamal. The hybrid process applied is a cross-hybrid cryptosystem variant. The stages involved in the Hybrid Vigenère–ElGamal encryption process are as follows:

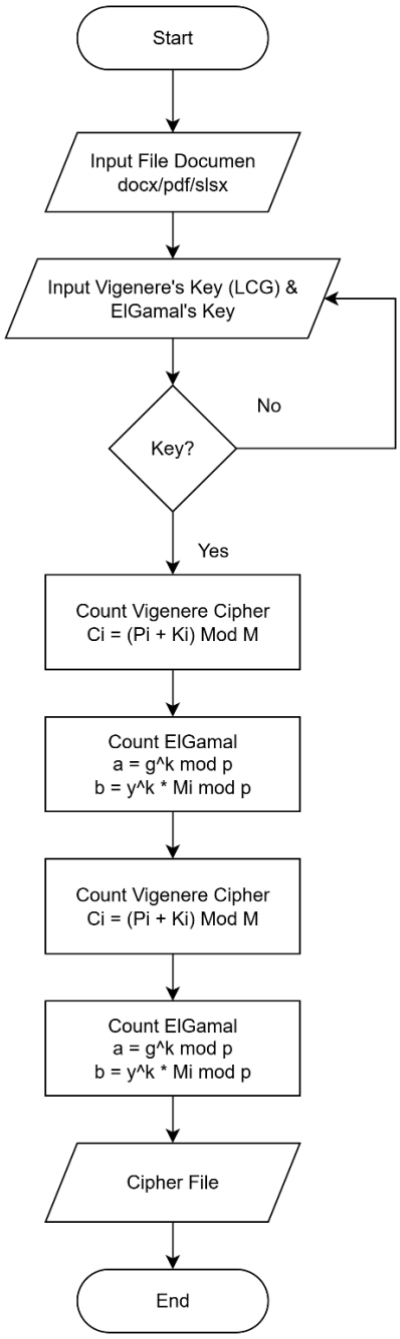

Figure 4: Vigenere-ElGamal Hybrid Encryption Process

3.2.1. Hybrid Encryption Process

The analysis of the Hybrid Encryption process is as follows:

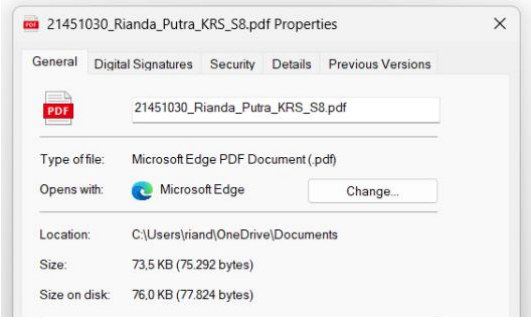

Figure 5: File To Be Secured

Before proceeding to the encryption and decryption steps, the file is first extracted into its hexadecimal values, with 8 bytes taken as a sample using the Binary Viewer software, as shown in the following figure:

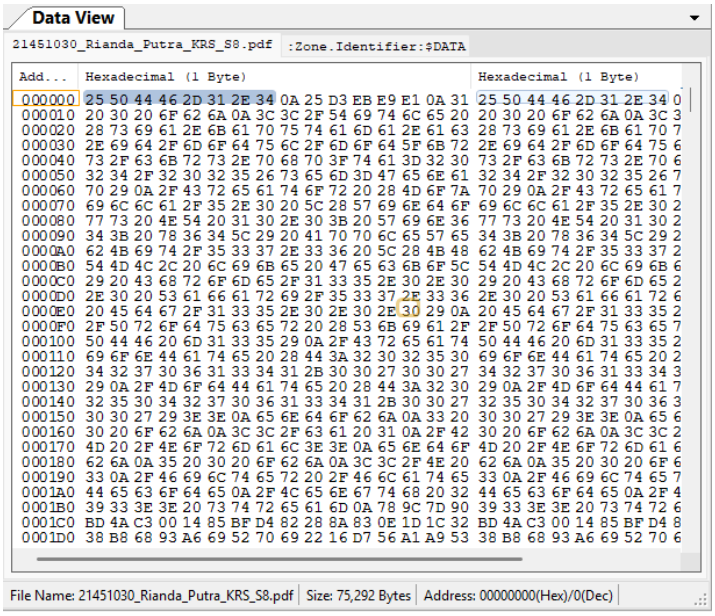

Figure 6: Hexadecimal From File

After obtaining the bytes to be processed, namely [25, 50, 44, 46, 2D, 31, 2E, 34], these hexadecimal values are first converted into decimal numbers to be processed in the encryption and decryption stages, resulting in [37, 80, 68, 70, 45, 49, 46, 52].

The 8 bytes are then divided into two parts: the first 2 bytes are encrypted using the Vigenère Cipher, the next 2 bytes are encrypted using ElGamal, and the process alternates between Vigenère Cipher and ElGamal for every 2 bytes until the entire plaintext file is fully encrypted. The Vigenère Cipher key is generated using the LCG algorithm, namely [38, 193, 197, 217]. Meanwhile, the ElGamal key is defined with the following parameters:

p = 263
g = 5
x = 13
y = g^x mod p
  = 5^{13} mod 263
  = 197
k = [7, 11, 15, 17]

The computation process of the Hybrid Vigenère–ElGamal Encryption is as follows:

Vigenere Cipher Encryption

[P1, P2] = [37, 80]
C1 = 37 → (37 + 38) MOD 256 = 75
C2 = 80 → (80 + 193) MOD 256 = 273 MOD 256 = 17

ElGamal Encryption

[P3, P4] = [68, 70]
C3 = 68
a3 = 5^7 MOD 263 = 14
b3 = (197^7 × 68) MOD 263 = 134
C3 = (14, 134)
C4 = 70
a4 = 5^11 MOD 263 = 71
b4 = (197^11 × 70) MOD 263 = 135
C4 = (71, 135)

Vigenere Cipher Encryption

[P5, P6] = [45, 49]
C5 = 45 → (45 + 197) MOD 256 = 242
C6 = 49 → (49 + 217) MOD 256 = 266 MOD 256 = 10

ElGamal Encryption

[P7, P8] = [46, 52]
C7 = 46
a7 = 5^15 MOD 263 = 191

b7 = (197^15 × 46) MOD 263 = 41
C7 = (191, 41)
C8 = 52
a8 = 5^17 MOD 263 = 41
b8 = (197^17 × 52) MOD 263 = 193
C8 = (41, 193)

The cipher file obtained from the Hybrid Vigenère–ElGamal encryption process is as follows: [75, 17, (14,134), (71,135), 242, 10, (191,41), (41,193)].

3.3. Hybrid Decryption

The stages involved in the Hybrid Vigenère–ElGamal decryption process are as follows, as illustrated in Figure 7:

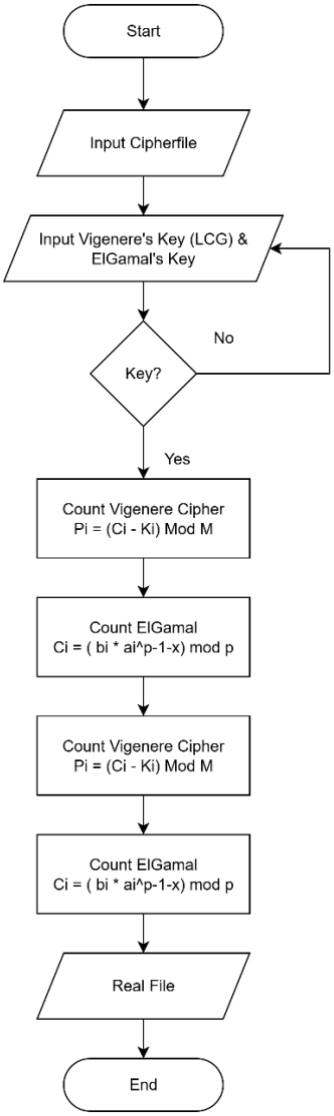

Figure 7: Vigenere-ElGamal Decryption Process

3.3.1. Hybrid Decryption Process

After obtaining the cipher file generated from the Hybrid Vigenère–ElGamal encryption process, namely [75, 17, (14,134), (71,135), 242, 10, (191,41), (41,193)], the next step is to restore this cipher file to its original form through the decryption process.

The computation process of the Hybrid Vigenère–ElGamal Decryption is as follows:

Vigenère Cipher Decryption

[C1, C2] = [75, 17]
P1 = (75 – 38 + 256) MOD 256 = 37
P2 = (17 – 193 + 256) MOD 256 = 80

ElGamal Decryption

[C3, C4] = (14,134) (71,135)
For C3 = (14,134) with k = 7
M3 = 134 × 14^(263–1–13) MOD 263 = 68
For C4 = (71,135) with k = 11
M4 = 135 × 71^(263–1–13) MOD 263 = 70
Result = 68 70

Vigenère Cipher Decryption

[C5, C6] = [242, 10]
P5 = (242 – 197 + 256) MOD 256 = 45
P6 = (10 – 217 + 256) MOD 256 = 49

ElGamal Decryption

[C7, C8] = (191,41) (41,193)
For C7 = (191,41) with k = 15
M7 = 41 × 191^(263–1–13) MOD 263 = 46
For C8 = (41,193) with k = 17
M8 = 193 × 41^(263–1–13) MOD 263 = 52
Result = 46 52

After being processed using the hybrid decryption method, the original plaintext obtained is [37, 80, 68, 70, 45, 49, 46, 52]. The final step is to convert all these decimal values back into hexadecimal form so that all bytes return to their original values, resulting in [25, 50, 44, 46, 2D, 31, 2E, 34].

4. Implementation and Discussion

4.1. Implementation

The implementation stage is carried out by realizing the system design into an application for document encryption and decryption based on the hybrid Vigenère Cipher and ElGamal algorithms. The application is developed using Visual Basic and consists of several forms: the main form, the dynamic key generator form based on LCG, the encryption form, and the decryption form. Each form has its own function according to its purpose, such as the encryption form for selecting files and generating ciphertext, and the decryption form for restoring the encrypted file to its original form.

4.2. Testing

Testing was conducted to ensure that the hybrid encryption and decryption system operates according to the design. The objective was to evaluate the implementation of the Vigenère Cipher algorithm with a dynamic key based on LCG and the ElGamal algorithm, as well as to verify that the decryption results can restore documents to their original form without data corruption. Test files in PDF, DOCX, and XLSX formats were used to assess performance consistency, while file size and processing time were recorded as metrics to evaluate the system’s effectiveness and efficiency.

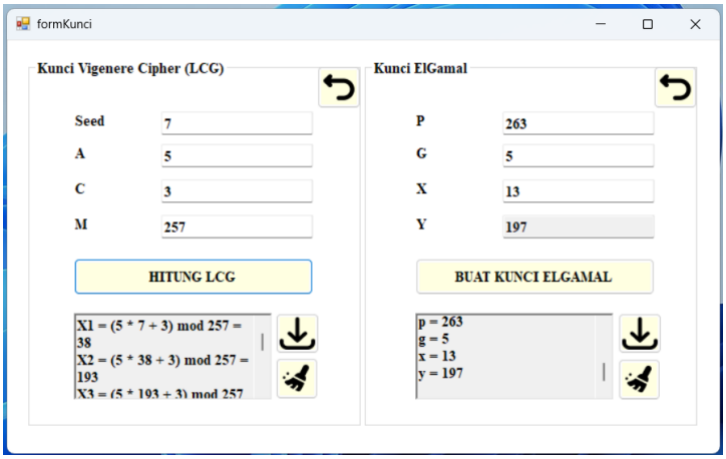

Figure 8: Dinamic Key Generator Testing

In Figure 8. users are required to input LCG parameters such as the seed or initial value (X_0), *a*, *c*, and *m* to generate pseudo-random numbers when the "Calculate LCG" button is pressed. The results and calculations are then displayed in the RichTextBox, and users can download the key file by clicking the button with the download icon. A similar process applies when users want to generate the ElGamal key.

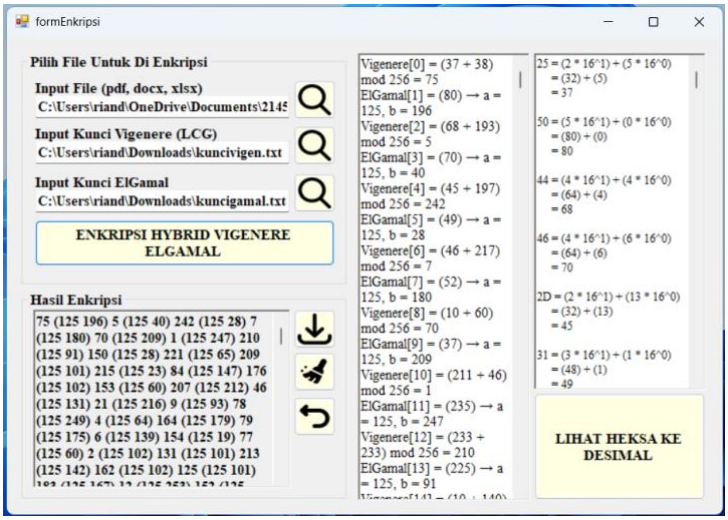

Figure 9: Encryption Form Testing

Figure 9 shows the encryption process of a file, where a button with a magnifying glass icon is used to browse and upload the file for encryption. The user can then press the hybrid encryption button to perform the computation, and the results will be displayed in the RichTextBox below. Additionally, the user can view the conversion of hexadecimal values to decimal by pressing the "View Hex to Decimal" button, and the output will be displayed in the RichTextBox located above the button.

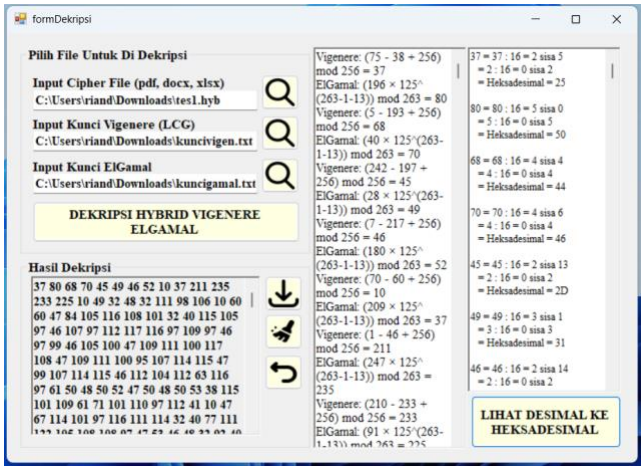

Figure 10: Decryption Form Testing

Figure 10 displays the interface of the decryption form used to restore encrypted files to their original format. The decryption process begins by selecting an encrypted file with the extension .hyb, which contains a combination of ciphertext data and the algorithm sequence. The user then inputs the Vigenère key generated by the LCG, along with the public and private keys for the ElGamal algorithm. After pressing the "Hybrid Vigenère–ElGamal Decrypt" button, the system processes the data alternately according to the encryption sequence: bytes encrypted with Vigenère are decrypted using Vigenère, while byte pairs encrypted with ElGamal are decrypted using the ElGamal private key.

The decryption results are displayed in the RichTextBox both as sequences of decimal numbers and detailed calculation steps. On the right side of the form, the computation stages are shown—from the subtraction with the Vigenère key to the modular calculations in ElGamal—allowing users to transparently observe the data recovery process. Additionally, a "View Decimal to Hexadecimal" button is available to facilitate the conversion of decrypted output into hexadecimal form for easy verification against the original data.

Based on this testing, the decrypted documents were successfully restored to their original formats (PDF, DOCX, or XLSX) and could be opened without errors or data loss. This demonstrates that the hybrid Vigenère–ElGamal algorithm implemented in the system effectively maintains data integrity.

4.3. Test Result

The test results indicate that the developed hybrid system is capable of performing encryption and decryption processes as designed. Test files in PDF, DOCX, and XLSX formats were successfully encrypted, rendering them inaccessible through standard applications. After decryption, the files were restored to their original state without any changes to their content. Additionally, the tests recorded the file sizes before and after encryption, as well as the processing time required, to evaluate the system's performance in terms of effectiveness and efficiency. The trial results are presented in the table below:

Table 1: Test Result

NO	Real File	Vigenere-ElGamal Hybrid Encryption	Vigenere ElGamal Hybrid Decryption
1	21451030_Rianda_Putra_KRS_S8.pdf Type of file: Microsoft Edge PDF Document (.pdf) Opens with: Microsoft Edge Location: C:\Users\riand\OneDrive\Documents Size: 73,5 KB (75.292 bytes) Size on disk: 76,0 KB (77.824 bytes)	tes1.hyb Type of file: HYB File (.hyb) Opens with: Pick an app Location: C:\Users\riand\Downloads Size: 257 KB (263.534 bytes) Size on disk: 260 KB (266.240 bytes)	tes1.pdf Type of file: Microsoft Edge PDF Document (.pdf) Opens with: Microsoft Edge Location: C:\Users\riand\Downloads Size: 73,5 KB (75.292 bytes) Size on disk: 76,0 KB (77.824 bytes)
2	tes2.xlsx Type of file: Microsoft Excel Worksheet (.xlsx) Opens with: Excel Location: C:\Users\riand\Downloads Size: 8,42 KB (8.626 bytes) Size on disk: 12,0 KB (12.288 bytes)	tes2.hyb Type of file: HYB File (.hyb) Opens with: Pick an app Location: C:\Users\riand\Downloads Size: 29,4 KB (30.204 bytes) Size on disk: 32,0 KB (32.768 bytes)	tes22.xlsx Type of file: Microsoft Excel Worksheet (.xlsx) Opens with: Excel Location: C:\Users\riand\Downloads Size: 8,42 KB (8.626 bytes) Size on disk: 12,0 KB (12.288 bytes)
3	tes3.docx Type of file: Microsoft Word Document (.docx) Opens with: Word Location: C:\Users\riand\Downloads Size: 14,5 KB (14.854 bytes) Size on disk: 16,0 KB (16.384 bytes)	tes3.hyb Type of file: HYB File (.hyb) Opens with: Pick an app Location: C:\Users\riand\Downloads Size: 50,7 KB (52.002 bytes) Size on disk: 52,0 KB (53.248 bytes)	tes33.docx Type of file: Microsoft Word Document (.docx) Opens with: Word Location: C:\Users\riand\Downloads Size: 14,5 KB (14.854 bytes) Size on disk: 16,0 KB (16.384 bytes)

5. Conclusion

Based on the results of the research conducted, it can be concluded that the hybrid cryptographic system combining the Vigenère Cipher and ElGamal algorithms with a dynamic key based on the Linear Congruential Generator (LCG) was successfully designed and implemented. The application of dynamic keys has proven effective in mitigating the weaknesses of repetitive key patterns in the classical Vigenère Cipher, making the encryption process more random and harder to predict. On the other hand, the use of the ElGamal algorithm provides an additional layer of security as a public-key cryptography scheme with a higher level of complexity. The combination of these two algorithms results in a hybrid mechanism capable of maintaining data confidentiality and integrity.

The system testing showed that documents in PDF, DOCX, and XLSX formats could be effectively encrypted, rendering them inaccessible through standard document readers. After the decryption process, the files were successfully restored to their original form and could be opened and read without data corruption. This demonstrates that the designed system meets the research objectives, which are to secure digital documents without altering their original content.

Testing also revealed that the size of the encrypted files was larger than the original files, with the encrypted files being nearly three times bigger. This increase is due to the nature of the ElGamal algorithm, which produces two ciphertext components for each byte of data, thereby increasing the file size. Nevertheless, the encryption and decryption processes were completed in a relatively fast and stable time frame. This indicates that the developed hybrid system functions correctly while maintaining reasonable efficiency for medium-sized documents.

Overall, this research confirms that a hybrid approach can overcome the limitations of individual algorithms when used separately. The Vigenère Cipher with a dynamic key based on LCG offers speed and stronger key variability, while ElGamal enhances the system with a higher level of security. Therefore, the Vigenère–ElGamal hybrid system developed in this study can serve as an alternative solution for securing data and digital documents in modern information systems.

References

[1] R. M. Ulfa Br Mtd, A. Fauzi, and H. Sembiring, "Kombinasi Algoritma Vigenere Cipher Dan One Time Pad Pada Keamanan Citra Digital," *J. Inform. Kaputama*, vol. 5, no. 1, pp. 137–146, 2021, doi: 10.59697/jik.v5i1.312.

[2] A. Fauzi, "Analisa Perancangan Aplikasi Penyandian Pesan Pada Email Menggunakan Algoritma Kriptografi Blowfish," *MEANS (Media Inf. Anal. dan Sist.*, vol. 1, no. 2, pp. 72–77, 2016, doi: 10.54367/means.v1i2.13.

[3] M. S. Dairi, M. Setiani Asih, and corespondent author, "Implementasi Algoritma Kriptografi RSA Dalam Aplikasi Sistem Informasi Perpustakaaan," *J. Ilmu Komput. dan Sist. Inf.*, vol. 2, no. 1, pp. 98–107, Jan. 2023, doi: 10.70340/JIRSI.V2I1.44.

[4] Jamaludin and Romindo, "Implementation of Combination Vigenere Cipher and RSA in Hybrid Cryptosystem for Text Security," *Int. J. Inf. Syst. Technol. Akreditasi*, vol. 4, no. 1, pp. 471–481, 2020.

[5] A. Y. Suseno, N. Sulistiyowati, and P. -, "Analisis Peningkatan hybrid Cryptosystem Untuk Enkripsi dan Dekripsi Menggunakan Vigenere Cipher dan RSA Pada Text," *STRING (Satuan Tulisan Ris. dan Inov. Teknol.*, vol. 6, no. 2, p. 142, 2021, doi: 10.30998/string.v6i2.10309.

[6] Q. Kester, "A hybrid cryptosystem based on Vigenère cipher and columnar transposition cipher," *Int. J. Adv. Technol. Eng. Res.*, vol. 3, no. 1, pp. 141–147, 2013.

[7] N. A. Nanda, S. M. S. Silalahi, D. Fatricia Nasution, M. Sari, and I. Gunawan, "Kriptografi dan Penerapannya Dalam Sistem Keamanan Data," *J. Media Inform.*, vol. 4, no. 2, pp. 90–93, 2023, doi: 10.55338/jumin.v4i2.428.

[8] A. Fauzi, Y. Maulita, and N. Novriyenni, "Perancangan Aplikasi Keamanan Pesan Menggunakan Algoritma Elgamal Dengan Memanfaatkan Algoritma One Time Pad Sebagai Pembangkit Kunci," *JTIK (Jurnal Tek. Inform. Kaputama)*, vol. 1, no. 1, pp. 1–9, 2017, doi: 10.59697/jtik.v1i1.680.

[9] A. M. H. Pardede *et al.*, "Aplikasi Pengamanan File Gambar Menggunakan Algoritma Elgamal," *J. Inf. Syst. Dev.*, vol. 3, no. 2, 2018.

[10] D. Deslianti and A. Fahry, "Aplikasi Pembelajaran Bahasa Inggris Dengan Menggunakan Algoritma Linear Congruent Generator Berbasis Android," *JUSIBI (Jurnal Sist. Inf. dan Bisnis)*, vol. 5, no. 1, pp. 9–15, 2023, doi: 10.54650/jusibi.v5i1.496.

[11] Ahmad Thariq and Mita Paramitha, "Aplikasi Game Edukasi Pembelajaran Seni Budaya Menggunakan Linear Congruential Generator (Lcg)," *Tek. Teknol. Inf. dan Multimed.*, vol. 5, no. 1, pp. 11–16, 2024, doi: 10.46764/teknimedia.v5i1.168.