E-ISSN : XXXX-XXXX
Journal of Computer Science & Intelligent Systems

Implementation of LUC Algorithm-Based Cryptography in Document File Protection

Implementasi Kriptografi Algoritma LUC Dalam Pengamanan File Dokumen

[1]Muhammad Sabiq, [2]Surya Guntur
[1, 2]Politeknik Ganesha Medan, Medan, Indonesia
*Correspondence:[1]sabiq.mhs@polgan.ac.id
https://doi.org/10.59696/nexcore.v1i1.259

Submitted: Des 25, 2025 | **Accepted**: Jan 11, 2026 | **Published**: Jan 14, 2026

ABSTRACT
In this study, the authors implement the LUC cryptographic algorithm for securing file messages. Cryptography is a scientific discipline that explores the art and methods of ensuring that a message cannot be interpreted by unauthorized parties. The purpose of this research is to apply the LUC algorithm in a user-friendly manner while utilizijng hexadecimal number system conversion so that the resulting ciphertext becomes more complex to decipher and the ciphertext file size is not excessively large, since the LUC algorithm typically produces ciphertext much larger than its plaintext. The test results show that the developed cryptographic application is easy to use (user-friendly) because the key generation process is performed automatically. The encryption process using the LUC conversion algorithm (by adding hexadecimal number conversion) produces more randomized ciphertext and keeps the encrypted file size relatively small.

Keywords*: Cryptography, LUC, LUC Conversion, Encryption, Decryption*

ABSTRAK
Dalam penelitian ini, penulis mengimplementasi kriptografi algoritma LUC dalam pengamanan pesan file. Kriptografi merupakan suatu ilmu pengetahuan yang mempelajari tentang seni ataupun cara menjamin suatu pesan agar pesan tersebut tidak dapat dimaknai oleh pihak lain yang tidak mempunyai otoritas di dalamnya. Tujuan penelitian ini adalah menerapkan algoritma LUC yang mudah digunakan serta memanfaatkan konversi sistem bilangan hexadecimal agar ciphertext yang dihasilkan lebih rumit untuk dipecahkan dan ukuran file ciphertext tidak terlalu besar karena algoritma LUC biasanya menghasilkan ciphertext yang jauh lebih besar dari plaintext-nya. Dari hasil pengujian, aplikasi kriptografi yang dibangun mudah digunakan (user friendly) karena proses pembentukan kunci dilakukan secara otomatis. Proses enkripsi dengan algoritma LUC conversion (menambahkan konversi bilangan hexadecimal) menghasilkan ciphertext lebih acak dan ukuran file enkripsi tidak terlalu besar.

Kata Kunci: Kriptografi, LUC, LUC Conversion, Enkripsi, Dekripsi

PENDAHULUAN

Sebuah pesan menjadi sangat penting jika di dalamnya terdapat banyak informasi yang bersifat private, di mana publik tidak diizinkan untuk mengetahui isi yang terdapat pada pesan tersebut. Banyak kejahatan cyber mencari celah keamanan untuk masuk dan melakukan manipulasi. Banyaknya upaya dari pihak yang berusaha ingin mengetahui informasi dari pesan tersebut, menjadi salah satu alasan dikembangkannya pengamanan data dalam bentuk digital, agar informasi yang bersifat rahasia lebih aman dan tidak dapat diketahui oleh pihak lain. Dalam menjamin keamanan dan kerahasiaan pesan tersebut perlu adanya suatu teknik untuk menyandikan pesan atau informasi yang disebut kriptografi.

Kriptografi adalah suatu ilmu pengetahuan yang mempelajari tentang seni ataupun cara menjamin suatu pesan agar pesan tersebut tidak dapat dimaknai oleh pihak lain yang tidak mempunyai otoritas di dalamnya. Terdapat dua jenis kriptografi, antara lain kriptografi klasik dan modern. Dalam mengaplikasikannya, orang lebih mengandalkan kelebihan kriptografi modern. Metode pengamanan data dalam bidang ilmu kriptografi begitu beragam dengan kunci yang berbeda pula, salah satu motode yang penulis pilih adalah Algoritma LUC.

Algoritma LUC merupakan metode kriptografi dengan menggunakan dua kunci yang berbeda untuk pengamanan data. Pada Algoritma LUC dilakukan dalam bentuk bilangan, oleh karena itu sebelum dilakukan proses enkripsi, di mana teks terlebih dahulu dikonversikan kedalam bentuk angka. Teks yang dienkripsi menggunakan Algoritma LUC menghasilkan ciphertext yang lebih besar dibanding plaintext-nya sehingga Algoritma LUC dapat meningkatkan keamanan sehingga pesan lebih sulit dipecahkan oleh kriptanalis.

Sebuah penelitian pernah dilakukan oleh Irawan (2013), bertujuan dalam membangun sebuah perangkat lunak yang dapat melakukan proses enkripsi dan dekripsi pada SMS dengan menggunakan android GingerBread (2.3.4) memakai pemrograman Java. Dan memastikan proses enkripsi dekripsi text dengan

memakai Algoritma LUC. Penelitian ini berhasil mencapai dengan mengenkripsi dan mendekripsi pesan yang berbentuk file string yang terdiri dari huruf kapital.

Algoritma LUC bekerja berdasarkan pada Lucas Function. Algoritma ini dikembangkan dari algoritma RSA setelah Smith meneliti kelemahan algoritma tersebut. Namun kemudian dalam penerapannya ditemukan bahwa teks yang dienkripsi menggunakan Algoritma LUC menghasilkan ciphertext yang lebih besar dibanding plaintext-nya.

METODE PENELITIAN

Pada penelitian ini menggunakan data text inputan langsung dengan format *.txt, *.doc, dan *.docx yang termasuk dalam tabel ASCII. Bahan penelitian didapat dari beberapa sumber, seperti jurnal, buku, prosiding, sumber bacaan elektronik.

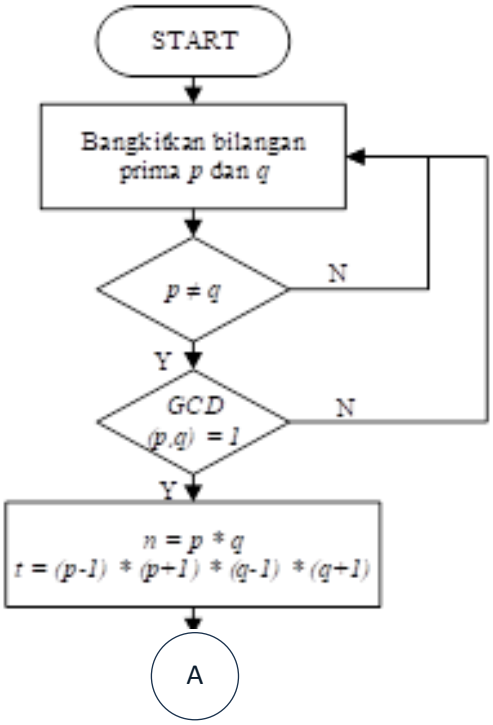
Proses Algoritma LUC

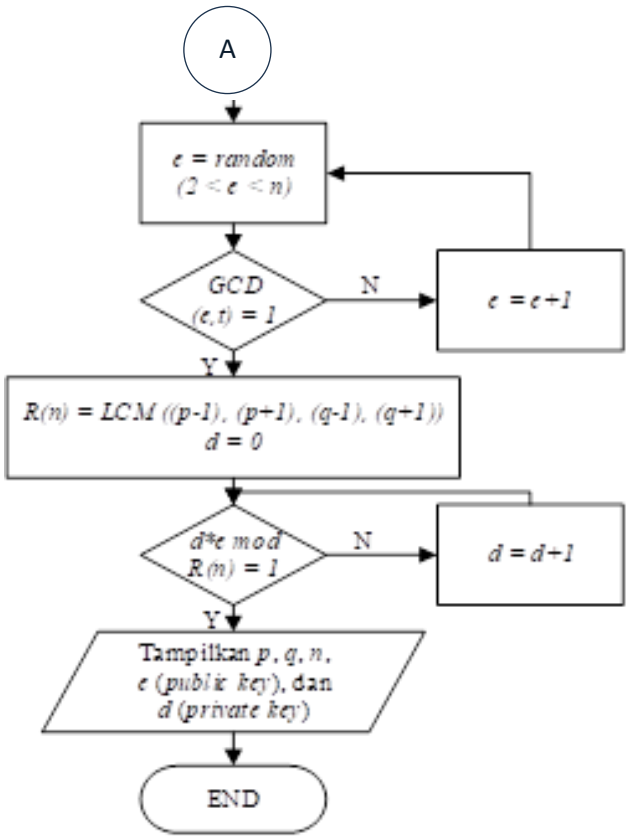
Untuk proses pengamanan pesan pada penelitian ini penulis akan menerapkan kriptografi alogritma LUC dengan menyisipkan model konversi sistem bilangan hexadecimal untuk proses enkripsi dan dekripsi pesan.

Pada teknik ini, pesan dienkripsi terlebih dahulu dengan algoritma LUC kemudian hasil enkripsi akan dikonversi menjadi sistem bilangan hexadecimal agar hasil enkripsi semakin sulit untuk dipecahkan dan menjadikan ukuran file tidak terlalu besar. Proses dekripsi dilakukan dengan cara yaitu hasil enkripsi dengan bilangan hexadecimal dikonversi kembali menjadi bilangan decimal dan kemudian dekripsi dengan algoritma LUC.

Flowchart Pembangkitan Kunci Algoritma LUC

Pada gambar 1, dapat kita lihat bahwa di dalam proses pembangkitan kunci algoritma LUC ada suatu tahap dimana perlu membangkitkan dua bilangan prima, yaitu p dan q. Setelah dibangkitkan sistem akan mencari mengecek apa GCD (p, q) = 1, jika tidak maka akan diulangi hingga GCD (p, q) =1. Lalu cari nilai n, t, e, R(n), dan d, lalu simpan e dan n sebagai public key serta d dan n sebagai private key.





Gambar 1 Flowchart Pembangkitan Kunci Algoritma LUC

Berikut ini akan dijelaskan proses-proses dalam pembangkitan kunci Algoritma LUC:

a. Tentukan p dan q, dimana p dan q merupakan bilangan prima dan p ≠ q.
   Misalkan p = 17 dan q = 19
b. Mencari nilai n.
   N = p * q
   n= 17 x 19
   = 323
c. Hitung t = (p - 1) * (q – 1) * (p + 1) * (q + 1)
   t= (17 – 1) * (19 – 1) * (17 + 1) * (19 + 1)
   = 16 * 18 * 18 * 20
   = 103,680
d. Menentukan nilai e. Nilai e merupakan bilangan relatif prima yang dipilih secara acak. Syarat nilai e di mana 2 < e < n dan hasil dari GCD(e,t) harus sama dengan 1. Nilai e akan digunakan untuk proses enkripsi.
   Misalkan kita pilih e = 319, GCD (319, 103680) = 1
e. Hitung RN = LCM (p – 1, q – 1, p + 1, q + 1)
   RN = LCM (16, 18, 18, 20)
   = 720
f. Mencari nilai d yang akan digunakan untuk proses dekripsi. Nilai d didapat dari hasil e.d mod RN = 1. Pada tabel 3.1 akan diperlihatkan perhitungan untuk mendapatkan nilai d.

Tabel 1. Perhitungan Kunci Dekripsi (d)

D	e.d mod RN = 1 319.d mod 720 = 1
1	319
2	638
3	237
4	556
...	...
...	...
79	1

Maka dari proses di atas diperoleh :
p = 17
q = 19
n = 323
t = 103.680
e = 319 → kunci public (kunci enkripsi)
RN = 720
d = 79 → kunci private (kunci dekripsi)

HASIL DAN PEMBAHASAN

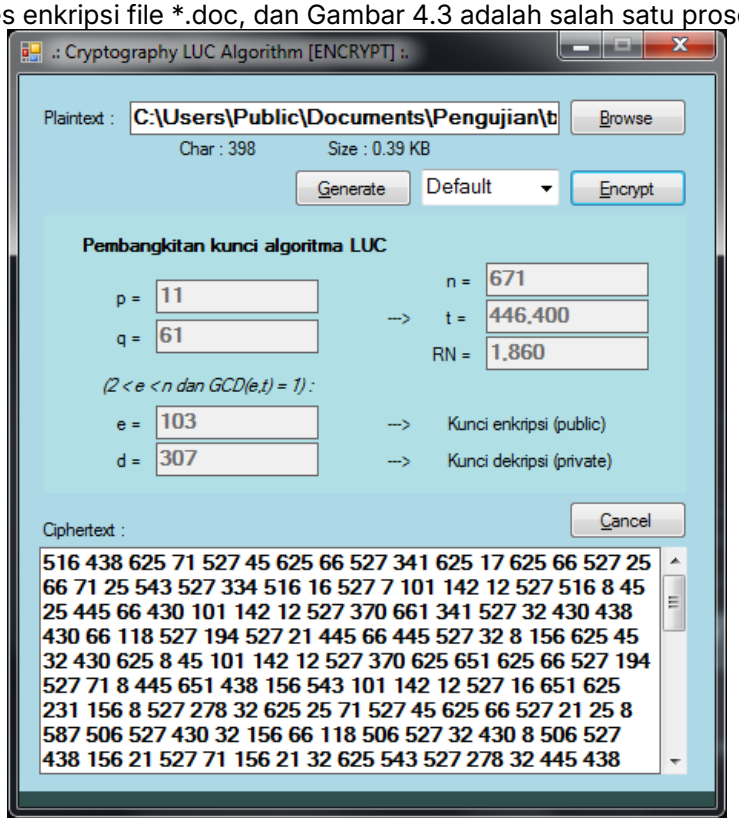
Pada penelitian ini melakukan pengujian pada proses enkripsi dan dekripsi file menggunakan algoritma LUC. Pengujian dilakukan dengan mengenkripsi file *.txt, *.doc, dan *.docx dengan Algoritma LUC default dan dengan Algoritma LUC yang dikombinasikan dengan konversi bilangan (conversion) pada hasil enkripsinya. Setelah itu penulis akan menganalisa ukuran file sebelum dienkripsi dengan ukuran file setelah dienkripsi pada masing-masing percobaan.

Proses Enkripsi dan Dekripsi dengan Algoritma LUC

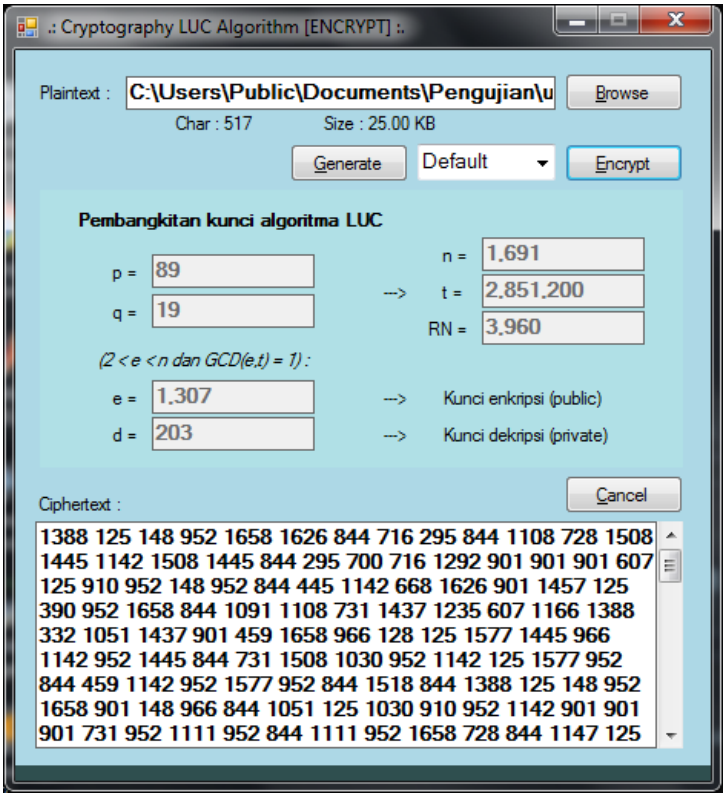
Pada tahap ini, penulis akan memperlihatkan hasil dari proses enkripsi dan dekripsi menggunakan algoritma LUC.

1. Proses Enkripsi Algoritma LUC Default

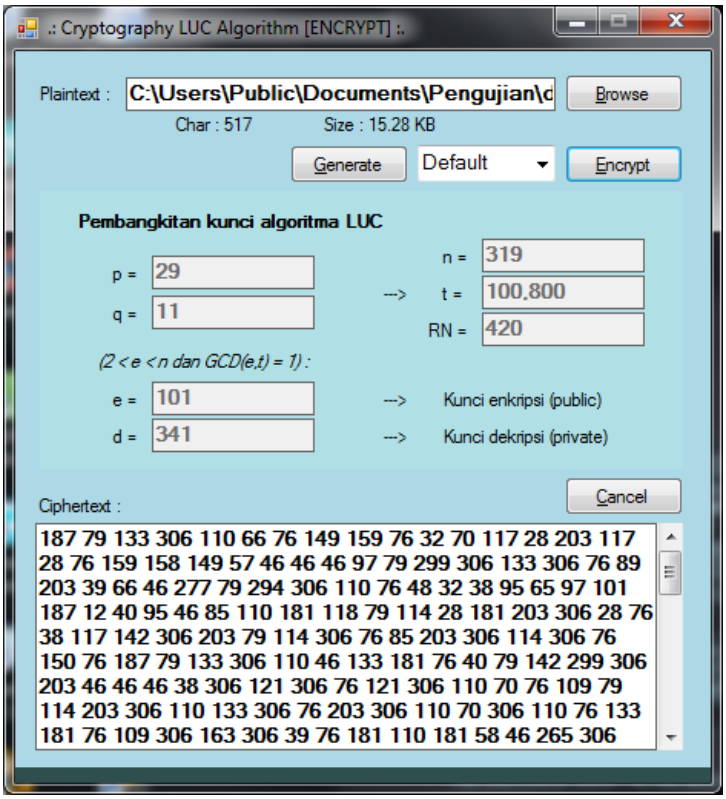
Proses enkripsi Algoritma LUC default dengan melakukan input file terlebih dahulu, kemudian dilakukan proses pembangkitan (generate) kunci kemudian pilih tipe enkripsi default, kemudian pilih encrypt maka akan dihasilkan ciphertext. Pada Gambar 2 adalah salah satu proses enkripsi file *.txt, Gambar 3 adalah salah satu proses enkripsi file *.doc, dan Gambar 4.3 adalah salah satu proses enkripsi file *.docx..



Gambar 2. Proses Enkripsi Default menggunakan Algoritma LUC (*.txt)



Gambar 3 Proses Enkripsi Default menggunakan Algoritma LUC (*.doc)



Gambar 4 Proses Enkripsi Default menggunakan Algoritma LUC (*.docx)

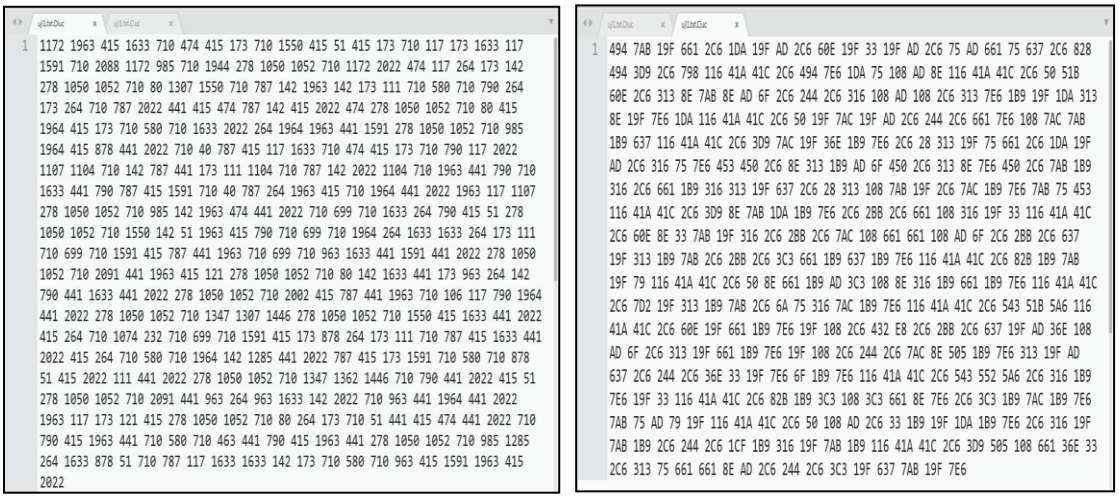
Pada Gambar 2, 3, dan 4 dapat dilihat proses enkripsi dengan Algoritma LUC default pada masing-masing format file (*.txt, *.doc, dan *.docx). Pada Tabel 2 akan dijabarkan hasil pengujian secara keseluruhan pada masing-masing file.

Tabel 2 Hasil Pengujian Enkripsi Algoritma LUC Default

No. Uji	Format file	Jumlah karakter	Ukuran (KB)	Ukuran setelah di-enkrip (KB)	Keterangan
1.	.txt	398	0.39	1.41	Berhasil
2.	.txt	2,263	2.21	9.61	Berhasil
3.	.txt	5,059	4.94	27.50	Berhasil
4.	.txt	7,769	7.59	28.30	Berhasil
5.	.txt	11,087	10.83	48.50	Berhasil
6.	.txt	12,171	11.89	58.90	Berhasil
7.	.txt	13,190	12.88	73.80	Berhasil
8.	.txt	17,170	16.77	73.90	Berhasil
9.	.doc	517	25.00	2.22	Berhasil
10.	.doc	1,379	27.00	5.25	Berhasil
11.	.doc	4,582	31.00	20.20	Berhasil
12.	.doc	8,593	41.50	33.00	Berhasil
13.	.doc	7,986	45.50	34.50	Berhasil
14.	.doc	14,283	49.00	60.80	Berhasil
15.	.doc	21,040	66.00	86.20	Berhasil
16.	.doc	29,733	71.50	126.00	Berhasil
17.	.docx	517	15.28	1.81	Berhasil
18.	.docx	4,582	17.23	18.10	Berhasil
19.	.docx	6,499	19.81	24.30	Berhasil
20.	.docx	8,593	23.62	30.80	Berhasil
21.	.docx	9,412	25.43	45.00	Berhasil
22.	.docx	21,040	25.46	89.20	Berhasil
23.	.docx	29,733	27.71	127.00	Berhasil
24.	.docx	31,179	28.23	121.00	Berhasil

Sesuai perkiraan, pada Tabel 2 dapat dilihat ukuran file setelah di-enkrip jauh lebih besar daripada ukuran file aslinya, hal ini dikarenakan Algoritma LUC mengubah setiap karakter menjadi angka-angka random yang ukurannya lebih panjang dan lebih besar. Hal ini baik untuk keamanannya, namun di satu sisi terdapat kekurangan yaitu ukuran file hasil enkripsi sangat besar sehingga banyak memori yang terpakai.

Contoh teks pesan hasil enkripsi dengan Algoritma LUC akan penulis tampilkan pada Gambar 5 Gambar 5(a) adalah teks pesan hasil enkripsi dengan Algoritma LUC default, sedangkan Gambar 5(b) adalah teks pesan hasil enkripsi dengan Algoritma LUC conversion.



Gambar 5 Contoh Teks File Hasil Enkripsi dengan Algoritma LUC

Pembahasan

Pada proses enkripsi menggunakan algoritma LUC jika dilakukan percobaan mengenkripsi secara berulang dapat menghasilkan ciphertext yang berbeda-beda. Karena nilai p dan nilai q pada proses pembangkitan kunci algoritma LUC dilakukan secara acak. Keamanan pesan file menggunakan Algoritma LUC conversion lebih baik karena ciphertext yang dihasilkan dari enkripsi berbeda-beda dan lebih panjang. Hal ini menyebabkan sangat sulit menebak isi file text maupun kunci yang digunakan. Semakin besar nilai p dan q

akan berpengaruh pada kunci enkripsi (e) maupun kunci dekripsi (d), hal ini menyebabkan semakin besar nilai e maka semakin besar pula ukuran file enkripsi yang dihasilkan.

KESIMPULAN

Dari hasil penelitian penerapan Algoritma LUC pada pengamanan pesan file, dapat disimpulkan bahwa aplikasi kriptografi yang dibangun mudah digunakan karena bersifat user friendly dan proses pembentukan kunci dilakukan secara otomatis; semakin besar nilai p dan q akan mempengaruhi besar kecilnya kunci enkripsi (e) dan dekripsi (d) yang terbentuk, serta berdampak pada ciphertext yang dihasilkan; dan proses enkripsi dengan Algoritma LUC conversion memiliki beberapa keuntungan dibandingkan enkripsi dengan Algoritma LUC default, seperti ciphertext yang lebih acak dan ukuran file enkripsi yang tidak terlalu besar. Adapun saran dari penelitian ini adalah agar penelitian selanjutnya dapat memperluas nilai p dan q sehingga kunci dan ciphertext yang dihasilkan lebih acak (random), serta mengkombinasikan Algoritma LUC dengan algoritma kriptografi lain seperti DES, RSA, BBS, RC5, dan sebagainya untuk meningkatkan keamanan.

REFERENSI

Hidayatullah, P. 2012. Visual Basic .NET Membuat Aplikasi Database dan Program Kreatif. Bandung: Informatika.

Indrajani. 2015. Database System (Case Study All in One). Jakarta: Elex Media Komputindo.

Irawan, A. F. 2013. Sistem Keamanan Pesan Pada Android GingerBread (2.3.4) Dengan Algoritma LUC. Universitas Jember.

Iswandy, E. 2014. Perancangan Sistem Informasi Tentang Pencatatan Hasil Tes Kemampuan Fisik Atlet (Studi Kasus: Fakultas Ilmu Keolahragaan (UNP) Padang. Jurnal Teknoif, Vol. 2, No. 2, ISSN: 2338-2724 : 27-36.

Kromodimoeljo, S. 2010. Teori dan Aplikasi Kriptografi. SPK IT Consulting.

Puspita, K. & Wayahdi, M. R. 2015. Analisis Kombinasi Metode Caesar Cipher, Vernam Cipher, dan Hill Cipher dalam Proses Kriptografi. Seminar Nasional Teknologi Informasi dan Multimedia. ISSN : 2302-3805.

Ramadani, D. 2018. Implementasi Algoritma LUC dalam Penyandian Teks. MEANS. Volume 3 Nomor 1. p-ISSN : 2548-6985, e-ISSN : 2599-3089.

Sadikin, R. 2012. Kriptografi untuk Keamanan Jaringan dan Implementasinya dalamBahasa Java. Andi: Yogyakarta.

Schneier, B. 2016. Applied Cryptography: Protocols, Algorithms and Source Code in C. 6td Edition. John Wiley & Sons, Inc: New Jersey.

Winarno, E. et. al. 2013. Belajar Pemrograman Populer (Java, VB, PHP). Jakarta: Elex Media Komputindo.