Jurnal Riset Komputer (JURIKOM), Vol. 5 No. 5, Oktober 2018
ISSN 2407-389X (Media Cetak)
Hal: 517-524

Penerapan Dan Implementasi Algoritma Kriptografi Loki97 Dalam Mengamankan File PDF

Mikain Pardede

Prodi Teknik Informatika, STMIK Budi Darma, Medan, Indonesia
Jl. Sisingamangaraja No. 338, Medan, Indonesia

Abstrak

Algoritma Loki97 ini merupakan algoritma yang sudah diakui sebagai algoritma yang dapat mengamankan data biar data yang kita simpan dalam bentuk file pdf tidak dapat dibuka sembarangan oleh pihak yang tidak bertanggungjawab. Dengan menerapkan algoritma Loki97 data yang bersifat rahasia dan data yang sangat penting dapat tersimpan dengan aman, sudah banyak yang menggunakan algoritma Loki97 untuk pengamanan data seperti pada jurnal implementasi algoritma kriptografi Loki97 pada dokumen video.

Kata Kunci : Penerapan, Implementasi, Algoritma Loki97, dan Kriptografi.

Abstract

This Loki97 algorithm is an algorithm that has been recognized as an algorithm that can secure data so that the data we store in the form of pdf files cannot be opened carelessly by irresponsible parties. By applying Loki97 algorithm data that is confidential and very important data can be stored safely, many have used the Loki97 algorithm for data security such as in the journal Loki97 cryptographic algorithm implementation on video documents.

Keywords: Implementation, Implementation, Loki97 Algorithm, and Cryptography.

1. PENDAHULUAN

Perkembangan dunia komputer dan pendukung perangkat lainnya yang serba digital telah membuat data-data digital semakin banyak digunakan. Terdapat sejumlah faktor yang membuat data digital semakin banyak digunakan yaitu mudah diduplikasi dan hasilnya sama dengan aslinya, mudah dalam penyimpanannya, serta mudah didistribusikan baik melalui media *disket* maupun melalui media *internet*. Keamanan dan kerahasiaan merupakan aspek penting yang dibutuhkan dalam proses pertukaran pesan atau informasi melalui jaringan atau Internet, dan yang paling sering di gunakan oleh orang dalam meringankan pekerjaan yaitu pertukaran data melalui jaringan karena turut berkembang pula kejahatan teknologi dengan berbagai teknik interupsi, penyadapan, modifikasi, maupun pembuatan. Tanpa adanya jaminan keamanan, orang lain dapat dengan mudah mendapatkan pesan atau informasi yang dikirim melalui jaringan.

File dengan format pdf yang berisi informasi tersebut bisa bersifat penting atau bisa juga bersifat rahasia, dan yang paling dominan digunakan untuk penyimpanan data sehingga perlu diamankan agar tidak untuk jatuh kepada pihak lain. Apabila informasi tersebut diketahui oleh umum atau pihak lain maka dapat digunakan untuk mendapatkan keuntungan dan dapat digunakan untuk merugikan orang tersebut. Untuk menghindari hal-hal tersebut dibutuhkan pengamanan data agar informasi tidak diketahui oleh pihak yang lain sehingga keamanan sebuah data rahasia bisa terjaga. Teknik pengamanan data yang bisa digunakan yaitu dengan kriptografi. Kriptografi adalah ilmu yang bersandarkan pada teknik matematika untuk berurusan dengan keamanan informasi seperti kerahasiaan, keutuhan data dan otentikasi entitas, dalam kriptografi, kunci adalah suatu informasi yang mengendalikan jalannya sebuah algoritma kriptografi. Enkripsi, kunci memberikan cara khusus bagaimana suatu algoritma mentransformasikan teks asli (*plaintext*) menjadi teks tersandi (*chipertext*), ataupun proses sebaliknya, yaitu dekripsi.

Sistem kriptografi kunci publik memiliki kunci untuk enkripsi K_e dan kunci untuk dekripsi K_d yang berbeda. Kunci untuk enkripsi K_e disebut juga sebagai kunci publik yang bersifat tidak rahasia sehingga dapat didistribusikan melalui saluran tidak aman. Sedangkan kunci dekripsi K_d disebut juga kunci private yang bersifat rahasia dan harus dijaga kerahasiaanya oleh pemegang kunci.

Algoritma loki97 adalah algortima yang sudah diakui sebagai algoritma yang dapat mengamankan data biar data yang kita simpan dalam bentuk file fdf tidak dapat dibuka sembarangan oleh pihak yang tidak bertanggung jawab, dengan menerapakan algoritma loki97 data data yang bersifat rahasia dan yang penting dapat kita enkripsi dan deskripsi kedalam bentuk kode abjad yang tidak berurutan dan dapat di ketahui oleh si penerima dengan kode kunci yang di hasilkan dari algoritma loki97. Sudah banyak orang yang menggunakan algoritma loki97 sebagai alat yang digunakan mengamankan data isi file dokumen dan file dokumen maupun video contoh penerapan algoritma loki97 tercantum dalam jurnal "Implementasi algoritma kriptografi loki97 video on demand (VOD) berbasis rights management (DRM) " oleh:Surya Michrandy S.T,M.T 2005

2. TEORITIS

2.1 Kriptografi

Kriptografi *(criptographi)* berasal dari bahasa yunani "*cryptos"* artinya "*secret*" (rahasia), sedangkan *"graphein"* artinya *"writing"* (tulisan). Sehingga kriptografi berarti "*secret writing*" (tulisan rahasia)[6]. Pengertian kriptografi secara modern adalah ilmu yang bersandarkan pada teknik matematika untuk berurusan dengan keamanan informasi seperti kerahasiaan, keutuhan data dan otentikasi entitas[2].

2.2 Metode *Loki97*

Algoritma Kriptografi LOKI97 merupakan salah satu kandidat *Advanced Encryption Standard* (AES) yang diajukan kepada NIST. Algoritma Kriptografi LOKI97 dirancang oleh L. Brown dan J. Pieprzyk. NIST mementukan beberapa criteria, diantaranya adalah kunci yang digunakan harus panjang, ukuran blok yang digunakan harus lebih besar, lebih cepat, dan fleksibel. Salah satu kandidat untuk AES adalah algoritma LOKI97.

3. ANALISA DAN PEMBAHASAN

Sistem pengamanan data menggunakan algoritma loki97 adalah dengan menggunakan algoritma *chiper* blok yang memiliki ukuran blok sebesar 128 bit dan menggunakan ukuran kunci sebesar 128, 192, atau 256 bit. Cipher ini berbentuk *Substitution-Permutation Network (SP-network)* yang merupakan rangkaian operasi-operasi matematis yang saling berhubungan. *SP-network* memiliki *S-boxes* dan *P-boxes* yang mengubah blok bit masukan menjadi suatu bit keluaran.

Loki97 mendukung masukan kunci sepanjang 128 bit, 192 bit, dan 256 bit. Kenyataannya, dalam mekanisme penjadwalan kunci dibutuhkan kunci sepanjang 256 bit. Oleh karena itu, untuk masukan kunci sepanjang 128 bit dan 192 bit memerlukan mekanisme tambahan, yaitu padding. Padding menambahkan bit "1" pada bit terpenting *(most significant bit)* dan beberapa bit "0" sampai ukuran kunci mencapai 256 bit.

Berdasarkan analisa sistem pengamanan data menggunakan algoritma di atas, maka enkripsi dan dekripsi data menjadi relatif lebih efektif dan aman, di mana dalam pengenkripsian suatu data diperlukan dua kunci yang tentunya lebih menjamin kerahasiaan data dan pengenkripsian dalam algoritma loki97 juga membutuhkan 32 round, dengan hal ini maka keamanan algoritma loki97 lebih terjamin dari pada algoritma yang lainnya.

3.1 Penerapan Metode

Proses kerja algoritma loki97 dalam melakukan enkripsi dan dekripsi data dengan masukan 128 bit dengan sebuah kunci 128, 192 atau 256 bit panjangnya. Algoritma serpent tediri dari tiga komponen utama yaitu :

1. Proses enkripsi
2. Proses deskripsi

Kriptografi metode loki97 akan membagi blok komponen penyusun *file* menjadi empat bagian, dimana akan dilakukan proses perulangan dengan proses yang sama untuk bagian blok komponen penyusun berikutnya dan kunci mempunyai ukuran yang bervariasi antara 128, 192 dan 256 *bit*. Misalkan sebuah data dengan ukuran 128 *bit*, putaran yang diperkenankan 20 dan panjang kunci 128. Karena blok komponen penyusun *file* dalam loki97 akan dibagi menjadi empat maka *file* data tersebut akan dibagi menjadi 4 buah blok 32 *bit* dan akan dilakukan pembagian blok yang sama untuk komponen penyusun berikutnya, sehingga didapatkan empat register penyusun yaitu, A, B, C, dan D. Dimana *byte* pertama dari komponen penyusun tersebut akan ditempatkan pada register A dan *byte* terakhir pada register D. Perulangan merupakan proses yang penting dilakukan agar setiap register tersebut dapat diproses secara merata tanpa ada sebuah register yang terlewatkan. Sebelum masuk ke perhitungan lebih jauh pertama dibangkitkan kunci dulu.

Langkah pertama yang dilakukan adalah mengubah *plaintext* tersebut ke dalam bentuk bilangan biner. Adapun hasil biner yang diperoleh seperti terlihat pada Tabel 1.

Tabel 1. *Plaintext* Dalam Bentuk Biner

Karakter	Nilai ASCII	Nilai Biner
C	67	01000011
O	111	01101111
N	110	01101110
T	116	01110100
O	111	01101111
H	104	01101000

(spasi)	32	00100000
P	80	01010000
E	101	01100101
S	115	01110011
A	97	01100001
N	110	01101110
N	78	01001110
I	105	01101001
H	104	01101000

Selanjutnya, dilakukan pula pengubahan kata kunci “Mikain pardede!” menjadi bilangan biner, yang hasilnya seperti terlihat pada Tabel 2.

Tabel 2. Kata Kunci Dalam Bentuk Biner

Karakter	Nilai ASCII	Nilai Biner
M	77	01001101
I	101	01100101
K	121	01111001
A	100	01100100
I	65	01000001
N	105	01101001
Spasi	32	00100000
P	105	01101001
A	108	01101100
R	70	01000110
D	105	01101001
E	116	01110100
D	114	01110010
E	97	01100001
!	33	00100001

Dari hasil nilai biner pada Tabel 1, diperolehlah deret *plaintext* dalam bentuk biner sebagai berikut :

01000011 01101111 01101110 01110100 01101111 01101000 00100000

C o n t o h (Spasi)

01010000 01100101 01110011 01100001 01101110 00100000 01001110

P e s a n (Spasi) N

01101001 01101000

i h

Plaintext dalam bentuk biner ini kemudian ditransposisikan ke dalam S-Box dengan cara membagi *plaintext* sebesar 32 bit sehingga diperoleh kolom-kolom seperti terlihat pada Tabel 3.

Tabel 3. *Plaintext* Dalam Bentuk S-Box

X_0	0	1	0	0	0	0	1	1	0	1	1	0	1	***1***	1	1	0	1	1	0	1	1	1	0	0	1	1	1	0	1	0	0
X_1	0	1	1	0	1	1	1	1	0	1	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0	1	0	1	0	0	0	
X_2	0	1	1	0	0	1	0	1	0	1	1	1	0	0	1	1	0	1	1	0	0	0	0	1	0	1	1	0	1	1	1	0
X_3	0	0	1	0	0	0	0	0	0	1	0	0	1	1	1	0	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	0

Selanjutnya, data pada kotak X_0 digeser ke kiri sebanyak 13 langkah, sehingga membentuk S-Box baru seperti terlihat pada Tabel 4.

Tabel 4. S-Box Setelah Proses Pertama

X_0	**1**	1	1	0	1	1	0	1	1	1	0	0	1	1	1	0	1	0	0	0	1	0	0	0	0	1	1	0	1	1	0	1
X_1	0	1	1	0	1	1	1	1	0	1	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0	1	0	1	0	0	0	0
X_2	0	1	1	0	0	1	0	1	0	1	1	1	0	0	1	1	0	1	1	0	0	0	0	1	0	1	1	0	1	1	1	0
X_3	0	0	1	0	0	0	0	0	0	1	0	0	1	1	1	0	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	0

Proses ke dua, dilakukan pergeseran ke kiri sebanyak 3 langkah pada data dalam kotak X_1, sehingga membentuk S-Box baru seperti terlihat pada Tabel 5.

Tabel 5. S-Box Setelah Proses Kedua

X_0	**1**	1	1	0	1	1	0	1	1	1	0	0	1	1	1	0	1	0	0	0	1	0	0	0	0	1	1	0	1	1	0	1
X_1	0	1	1	1	1	0	1	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0	1	0	1	0	0	0	0	0	1	1
X_2	0	1	1	0	0	1	0	1	0	1	1	1	0	0	1	1	0	1	1	0	0	0	0	1	0	1	1	0	1	1	1	0
X_3	0	0	1	0	0	0	0	0	0	1	0	0	1	1	1	0	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	0

Proses ke tiga, dilakukan proses X_1 XOR X_0 XOR X_2 yang hasilnya disimpan dalam kotak X_1, sehingga membentuk S-Box baru seperti terlihat pada Tabel 6.

Tabel 6. S-Box Setelah Proses Ketiga

X_0	**1**	1	1	0	1	1	0	1	1	1	0	0	1	1	1	0	1	0	0	0	1	0	0	0	0	1	1	0	1	1	0	1
X_1	1	1	1	1	0	0	1	1	1	1	1	1	1	1	0	0	1	1	1	0	1	0	1	1	1	0	0	0	0	0	0	0
X_2	0	1	1	0	0	1	0	1	0	1	1	1	0	0	1	1	0	1	1	0	0	0	0	1	0	1	1	0	1	1	1	0
X_3	0	0	1	0	0	0	0	0	0	1	0	0	1	1	1	0	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	0

Proses ke empat, dilakukan pergeseran ke kiri sebanyak 3 langkah pada kolom X_0, sehingga membentuk S-Box baru seperti terlihat pada Tabel 7.

Tabel 7. S-Box Setelah Proses Keempat

X_0	0	1	1	0	1	1	1	0	0	1	1	1	0	1	0	0	0	1	0	0	0	0	1	1	0	1	1	0	1	**1**	1	1
X_1	1	1	1	1	0	0	1	1	1	1	1	1	1	1	0	0	1	1	1	0	1	0	1	1	1	0	0	0	0	0	0	0
X_2	0	1	1	0	0	1	0	1	0	1	1	1	0	0	1	1	0	1	1	0	0	0	0	1	0	1	1	0	1	1	1	0
X_3	0	0	1	0	0	0	0	0	0	1	0	0	1	1	1	0	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	0

Proses ke lima, dilakukan proses X_3 XOR X_2 XOR X_0 yang hasilnya disimpan dalam kotak X_3, sehingga membentuk S-Box baru seperti terlihat pada Tabel 8.

Tabel 8. S-Box Setelah Proses Kelima

X_0	0	1	1	0	1	1	1	0	0	1	1	1	0	1	0	0	0	1	0	0	0	0	1	1	0	1	1	0	1	**1**	1	1
X_1	1	1	1	1	0	0	1	1	1	1	1	1	1	1	0	0	1	1	1	0	1	0	1	1	1	0	0	0	0	0	0	0
X_2	0	1	1	0	0	1	0	1	0	1	1	1	0	0	1	1	0	1	1	0	0	0	0	1	0	1	1	0	1	1	1	0

X_3	0	0	1	0	1	0	1	1	0	1	0	0	1	0	0	1	0	1	0	0	1	0	1	1	0	1	1	0	1	0	0	1

Proses ke enam, dilakukan pergeseran ke kiri sebanyak 1 langkah pada kolom X_1, sehingga membentuk S-Box baru seperti terlihat pada Tabel 9.

Tabel 9. S-Box Setelah Proses Keenam

X_0	0	1	1	0	1	1	1	0	0	1	1	1	0	1	0	0	0	1	0	0	0	0	1	1	0	1	1	0	1	**1**	1	1
X_1	1	1	1	0	0	1	1	1	1	1	1	1	1	0	0	1	1	1	0	1	0	1	1	1	0	0	0	0	0	0	0	1
X_2	0	1	1	0	0	1	0	1	0	1	1	1	0	0	1	1	0	1	1	0	0	0	0	1	0	1	1	0	1	1	1	0
X_3	0	0	1	0	1	0	1	1	0	1	0	0	1	0	0	1	0	1	0	0	1	0	1	1	0	1	1	0	1	0	0	1

Proses ke tujuh, dilakukan pergeseran ke kiri sebanyak 7 langkah pada kolom X_3, sehingga membentuk S-Box baru seperti terlihat pada Tabel 10.

Tabel 10. S-Box Setelah Proses Ketujuh

X_0	0	1	1	0	1	1	1	0	0	1	1	1	0	1	0	0	0	1	0	0	0	0	1	1	0	1	1	0	1	**1**	1	1
X_1	1	1	1	0	0	1	1	1	1	1	1	1	1	0	0	1	1	1	0	1	0	1	1	1	0	0	0	0	0	0	0	1
X_2	0	1	1	0	0	1	0	1	0	1	1	1	0	0	1	1	0	1	1	0	0	0	0	1	0	1	1	0	1	1	1	0
X_3	1	0	1	0	0	1	0	0	1	0	1	0	0	1	0	1	1	0	1	1	0	1	0	0	1	0	0	1	0	1	0	1

Proses ke delapan, dilakukan proses X_0 XOR X_1 XOR X_3 yang hasilnya disimpan dalam kotak X_0, sehingga membentuk S-Box baru seperti terlihat pada Tabel 11.

Tabel 11. S-Box Setelah Proses Kedelapan

X_0	0	0	1	0	1	1	0	1	0	0	1	0	1	0	0	0	0	0	1	0	0	0	0	0	1	1	1	1	1	0	1	1
X_1	1	1	1	0	0	1	1	1	1	1	1	1	1	0	0	1	1	1	0	1	0	1	1	1	0	0	0	0	0	0	0	1
X_2	0	1	1	0	0	1	0	1	0	1	1	1	0	0	1	1	0	1	1	0	0	0	0	1	0	1	1	0	1	1	1	0
X_3	1	0	1	0	0	1	0	0	1	0	1	0	0	1	0	1	1	0	1	1	0	1	0	0	1	0	0	1	0	1	0	1

Proses ke sembilan, dilakukan pergeseran ke kiri sebanyak 7 langkah pada kolom X_1, sehingga membentuk S-Box baru seperti terlihat pada Tabel 3.12

Tabel 12. S-Box Setelah Proses Kesembilan

X_0	0	0	1	0	1	1	0	1	0	0	1	0	1	0	0	0	0	0	1	0	0	0	0	0	1	1	1	1	1	0	1	1
X_1	1	1	1	1	1	1	0	0	1	1	1	0	1	0	1	1	1	0	0	0	0	0	0	0	1	1	1	1	0	0	1	1
X_2	0	1	1	0	0	1	0	1	0	1	1	1	0	0	1	1	0	1	1	0	0	0	0	1	0	1	1	0	1	1	1	0
X_3	1	0	1	0	0	1	0	0	1	0	1	0	0	1	0	1	1	0	1	1	0	1	0	0	1	0	0	1	0	1	0	1

Proses ke sepuluh, dilakukan proses X_2 XOR X_3 XOR X_1 yang hasilnya disimpan dalam kotak X_2, sehingga membentuk S-Box baru seperti terlihat pada Tabel 13.

Tabel 13. S-Box Setelah Proses Kesepuluh

X_0	0	0	1	0	1	1	0	1	0	0	1	0	1	0	0	0	0	0	1	0	0	0	0	0	1	1	1	1	1	0	1	1
X_1	1	1	1	1	1	1	0	0	1	1	1	0	1	0	1	1	1	0	0	0	0	0	0	0	1	1	1	1	0	0	1	1

X_2	0	0	1	1	1	1	0	1	0	0	1	1	1	1	0	1	0	1	0	1	0	1	0	1	0	0	0	0	1	0	0	0
X_3	1	0	1	0	0	1	0	0	1	0	1	0	0	1	0	1	1	0	1	1	0	1	0	0	1	0	0	1	0	1	0	1

Proses ke sebelas, dilakukan pergeseran ke kiri sebanyak 5 langkah pada kolom X_0, sehingga membentuk S-Box baru seperti terlihat pada Tabel 14.

Tabel 14. S-Box Setelah Proses Kesebelas

X_0	1	1	0	1	0	0	1	0	1	0	0	0	0	0	1	0	0	0	0	0	1	1	1	1	1	0	1	0	0	1	0	1
X_1	1	1	1	1	1	1	0	0	1	1	1	0	1	0	1	1	1	0	0	0	0	0	0	0	1	1	1	1	0	0	1	1
X_2	0	0	1	1	1	1	0	1	0	0	1	1	1	1	0	1	0	1	0	1	0	1	0	1	0	0	0	0	1	0	0	0
X_3	1	0	1	0	0	1	0	0	1	0	1	0	0	1	0	1	1	0	1	1	0	1	0	0	1	0	0	1	0	1	0	1

Proses ke duabelas, dilakukan pergeseran ke kiri sebanyak 22 langkah pada kolom X_2, sehingga membentuk S-Box baru seperti terlihat pada Tabel 15.

Tabel 15. S-Box Setelah Proses Keduabelas

X_0	1	1	0	1	0	0	1	0	1	0	0	0	0	0	1	0	0	0	0	0	1	1	1	1	1	0	1	0	0	1	0	1
X_1	1	1	1	1	1	1	0	0	1	1	1	0	1	0	1	1	1	0	0	0	0	0	0	0	1	1	1	1	0	0	1	1
X_2	0	1	0	0	0	0	1	0	0	0	0	0	1	1	1	1	0	1	0	0	1	1	1	1	0	1	0	1	0	1	0	1
X_3	1	0	1	0	0	1	0	0	1	0	1	0	0	1	0	1	1	0	1	1	0	1	0	0	1	0	0	1	0	1	0	1

Dari proses transposisi *plaintext* di atas, diperoleh *plaintext* baru sebagai berikut :

11010010 10000010 00001111 10100101 11111100 11101011 10000000

11110011 01000010 00001111 01001111 01010101 10100100 10100101

10110100 10010101

Untuk memproses *plaintext* dengan kata kunci, harus dilakukan konversi karakter kata kunci menjadi bentuk biner. Dengan menggunakan Tabel 3.2 sebagai acuan, maka diperoleh bentuk biner dari kata kunci yang digunakan sebagai berikut

01001101 01100101 01111001 01000001 01101001 01110100 00100000

M i k a i n spasi

01101100 01000001 01110010 00100000 01101001 00100000 01101001

p a r d e d e

00100001

Nilai *biner* kata kunci tersebut dipecah menjadi 32 bit yang masing-masing di masukkan ke dalam kotak Y_0, Y_1, Y_2 dan Y_3, sebagaimana terlihat pada Tabel 16.

Tabel 16. Kata Kunci Dalam Bentuk 32 Bit

Y_0	0	1	0	0	1	1	0	1	0	1	1	0	0	1	0	1	0	1	1	1	1	0	0	1	0	0	1	0	0	0	0	0
Y_1	0	1	0	0	0	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	0	1	0	0	0	1	1	0	1	0	0	1

Y_2	0	1	1	0	1	1	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	1	1	0	0	1	1	0	1	0	0	1
Y_3	0	1	1	1	0	1	0	0	0	1	1	1	0	0	1	0	0	1	1	0	0	0	0	1	0	0	1	0	0	0	0	1

Selanjutnya, dilakukan proses X_0 XOR Y_0, X_1 XOR Y_1, X_2 XOR Y_2 dan X_3 XOR Y_3, sehingga diperoleh hasil sebagaimana terlihat pada Tabel 17 sampai Tabel 20.

Tabel 17. X_0 XOR Y_0

X_0	1	1	0	1	0	0	1	0	1	0	0	0	0	0	1	0	0	0	0	0	1	1	1	1	1	0	1	0	0	1	0	1
Y_0	0	1	0	0	1	1	0	1	0	1	1	0	0	1	0	1	0	1	1	1	1	0	0	1	0	0	1	0	0	0	0	0
	1	0	0	1	1	1	1	1	1	1	1	0	0	1	1	1	0	1	1	1	0	1	1	0	1	0	0	0	0	1	0	1

Tabel 18 X_1 XOR Y_1

X_1	1	1	1	1	1	1	0	0	1	1	1	0	1	0	1	1	1	0	0	0	0	0	0	0	1	1	1	1	0	0	1	1
Y_1	0	1	0	0	0	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	0	1	0	0	0	1	1	0	1	0	0	1
	1	0	1	1	1	1	0	1	1	0	0	0	0	0	1	0	1	1	1	0	0	1	0	0	1	0	0	1	1	0	1	0

Tabel 19. X_2 XOR Y_2

X_2	0	1	0	0	0	0	1	0	0	0	0	0	1	1	1	1	0	1	0	0	1	1	1	1	0	1	0	1	0	1	0	1
Y_2	0	1	1	0	1	1	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	1	1	0	0	1	1	0	1	0	0	1
	0	0	1	0	1	1	1	0	0	0	1	0	1	1	1	1	0	0	0	0	1	0	0	1	0	0	1	1	1	1	0	0

Tabel 20. X_3 XOR Y_3

z	1	0	1	0	0	1	0	0	1	0	1	0	0	1	0	1	1	0	1	1	0	1	0	0	1	0	0	1	0	1	0	1
Y_3	0	1	1	1	0	1	0	0	0	1	1	1	0	0	1	0	0	1	1	0	0	0	0	1	0	0	1	0	0	0	0	1
	1	1	0	1	0	0	0	0	1	1	0	1	0	1	1	1	1	1	0	1	0	1	0	1	1	0	1	1	0	1	0	0

Dari proses XOR yang terlihat pada Tabel 17 sampai Tabel 20, akan dilakukan penggabungan terhadap seluruh nilainya. Hasil penggabungan inilah yang akan menjadi *chipertext* akhir dari proses enkripsi dengan metdoe *serpent* ini. Adapun *chipertext* yang diperoleh dari proses enkripsi di atas adalah sebagai berikut :

10011111 11100111 01110110 10000101 10111101 10000010 11100100 10011010 00101110 00101111 00001001 00111100 11010000 11010111 11010101 10110100

Chipertext dalam bentuk biner di atas kemudian dikonversikan kembali menjadi bentuk karakter sehingga diperoleh hasil sebagai berikut :

10011111 11100111 01110110 10000101 10111101 10000010 11100100
f τ v à ╨ é Σ
10011010 00101110 00101111 00001001 00111100 11010000 11010111
Ü . / (Tab) < ╩ ╬
11010101 10110100
F ┤

5. KESIMPULAN

Berdasarkan hasil perancangan dan implementasi sistem keamanan data menggunakan algoritma Loki97 ini, penulis menarik beberapa kesimpulan, sebagai berikut :

1. Perangkat lunak ini dapat digunakan sebagai sistem pengamanan data dalam bentuk teks sehingga bisa terjamin isi data hingga sampai ke tangan yang berhak.
2. Proses enkripsi dan dekripsi dengan menggunakan perangkat lunak ini dapat dilakukan dengan cepat.
3. *File output* dari sistem ini dihasilkan dalam bentuk *file* berektension *TXT*.

REFERENCES

[1] Rachmad C, Antonius, "Algoritma dan Pemrograman dengan Bahasa C-Konsep, Teori dan Implementasi", Penerbit Andi, Yogyakarta, Edisi 2, 2010
[2] Rifqi Sadikin, "Kriptografi untuk Keamanan Jaringan", Penerbit Andi, Yogyakarta, 2012
[3] Edy Winarno, ST, M.Eng, Ali Zaky, SmitDev Community, "VB.NET untuk Skripsi", PT. Elex Media Komputindo, Jakarta, 2015
[4] Irwan Sembiring, Theophilus Wellem, Gloria Saripah Patara, 4, 2007
[5] Rinaldi Munir, "Algoritma dan pemrograman Dalam Bahasa Pascal dan C", Informatika Bandung, Bandung, Edisi revisi 3, 2011
[6] Rinaldi Munir, "Kriptografi", Informatika Bandung, Bandung, Edisi 1, 2006
[7] Eko Budi Setiawan, Yogie Setiawan Nugraha, 2085-4552, VII, 2, 2015
[8] Mawarni Siregar, 2301-9425, V, 3, 2013
[9] Rahmat Priyanto, "Visual Basic.Net dan Database MysQL", Informatika Bandung, Bandung, 2006
[10] Wu, Hongjun, The Stream Cipher HC-128.