Seminar Nasional Teknologi Komputer & Sains (SAINTEKS)
SAINTEKS 2020
ISBN: 978-602-52720-7-3
Februari 2020
Hal 438-446

Pengamanan File DOCX Menerapkan Algoritma International Data Encryption Standart

Oktavian Laksamana Suryo, Fuady Mahbub, Mirawati

Program Studi Teknik Informatika, STMIK Budi Darma, Medan Indonesia
Email: [1]oktavianlaksamanasuryo@gmail.com, [1]fuadymahbub15@gmail.com, [1]mirawati2512@gmail.com

Abstrak−Seiring dengan Berkembangnya Teknologi, hampir semua orang menggunakan Microsoft Office Word dalam menyelesaikan tugas sehari-hari. Format penyimpanan file dokumen yang ada pada Microsoft office word adalah DOCX. Bisa jadi file yang dikerjakan merupakan file penting yang tidak boleh dilihat oleh sembarangan orang. Namun format file ini sangat mudah sekali untuk dicuri atau dipalsukan oleh pihak ketiga yang sangat tidak bertanggung jawab. Untuk menghindari hal tersebut, maka penulis bermaksud mengamankan file DOCX dengan mengenkripsi plainteks menjadi cipherteks agar pihak ketiga yang seharusnya tidak diperkenankan mengetahui, tidak bisa dengan mudah mencuri file tersebut. Namun hanya orang yang tahu kuncinya yang dapat mendekripsikan kembali cipherteks tersebut menjadi plainteks. Adapun algoritma yang digunakan yaitu Algoritma *International Data Encryption Standard* atau sering disebut dengan IDEA. Algoritma IDEA ini digunakan karena memberikan suatu keamanan tingkat tinggi dengan berdasarkan kunci rahasia yang kompleks.

Kata Kunci: DOCX, Pengamanan, Enkripsi, Dekripsi, IDEA

1. PENDAHULUAN

Era Teknologi penyebaran informasi sangat cepat, banyak informasi yang dibutuhkan guna membantu menyelesaikan perkerjaan. Kemudahan seseorang untuk mengakses informasi ini tentulah merupakan hal baik, namun ada juga hal buruk yang dapat terjadi. Ini dikarenakan tidak semua informasi yang ada itu boleh diakses sembarangan oleh semua pihak. Terdapat dua jenis informasi yaitu informasi yang rahasia dimana hanya orang-orang tertentu yang dapat mengaksesnya dan juga ada informasi tidak rahasia dimana semua orang boleh mengakses informasi tersebut. Informasi rahasia dapat berupa file-file penting perusahaan, data pribadi, skripsi dan lain sebagainya. Tentunya tidak sembarangan orang yang boleh melihat informasi ini. Oleh karena itu diperlukan suatu cara dalam mengamankan data tersebut sehingga hanya osrang-orang yang diizinkan yang dapat mengaksesnya. Untuk itu kita harus bisa mengamankan suatu data ataupun informasi rahasia tersebut menggunakan kriptografi. Kriptografi adalah suatu ilmu ataupun seni yang mempelajari bagaimana cara membuat atau menciptakan suatu pesan yang telah dikirim oleh pengirim dapat tersampaikan kepada penerima dengan aman (Scheneier 1996), Kriptografi bertujuan untuk menjaga suatu kerahasiaan dari informasi yang terkandung didalam sebuah data yang dimiliki sehingga informasi didalam data tersebut tidak dapat dengan mudah diketahui ataupun dibuka oleh pihak yang tidak sah[1].

Dengan kriptografi data atau informasi dapat di *enkripsi* ataupun mengamankan informasi *(plaintext)* dengan mengubah nya menjadi *Chipertext*, dan dibutuhkan *key* (kunci) untuk *mendekripsi* data tersebut dari *ciphertext* menajdi *plaintext* kembali. sehingga oknum ketiga yang tidak diperkenankan mengetahui informasi tersebut, tidak bisa membuka informasi yang ada pada data tersebut dan hanya orang yang memiliki *key* sajalah yang dapat membuka informasi yang ada. Untuk itu penulis berencana mengamankan *File* bertipe dokumen menggunakan kriptografi guna menjaga keamaan dari data yang ada. Kriptografi memliki berbagai macam algoritma dalam penggunaannya untuk itu penulis memilih menggunakan Algoritma IDEA ataupun (*International Data Encryption Algorithm)*. IDEA adalah sebuah algoritma dalam bentuk ciper blok yang telah dibuat oleh Xuejia Lai dan James Massey dari ETH Zurich dan kemudian pada tahun 1991 pertama kalinya diperkenalkan. Dirancang dengan tujuan menggantikan DES[2]. Algoritma IDEA memberikan keamanan informasi yang cukup tinggi dimana tidak tingkat keamanan bukan berdasarkan dari kerahasiaan algoritmanya melainkan lebih ditekankan terhadap kerahasiaan ataupun keamanan kunci yang digunakan. Untuk itu penulis berharap dengan menggunakan algoritma IDEA ini merupakan solusi terbaik dalam mengamankan data tertentu.

2. METODOLOGI PENELITIAN

2.1 International Data Encryption Algorithm (IDEA)

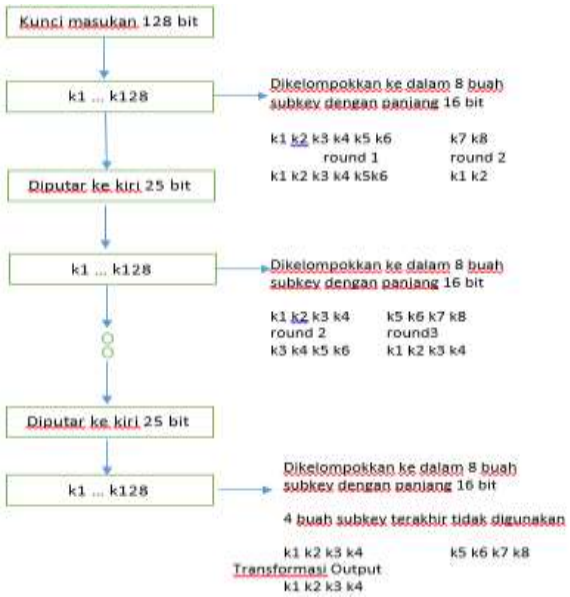
Algoritma IDEA merupakan suatu algoritma simetris yang dapat bekerja untuk sebuah blok pesan dengan lebar pesan 64 bit dan panjang kuncinya berukuran 128 bit. Algoritma IDEA dapat mengenkripsi dan mendekripsi kan sebuah DOCX menjadi lebih aman dan lebih bersifat rahasia dikarenakan ukuran dari algoritma IDEA sepanjang 64 bit dan dengan menggunakan tiga bentuk operasi aljabar yang berberbeda, yaitu Operasi Perkalian Modulo ($2^{16}+1$) , Operasi Penjumlahan Modulo (2^{16}) dan Operasi XOR($\oplus$). Operasi yang terdapat pada Algoritma IDEA dilakukan pada sebuah subblok 16 bit dan algoritma IDEA bisa melakukan sebanyak 8 iterasi[3]. Algoritma IDEA dapat mengenkripsi pesan terbuka yang dapat berupa citra, audio, ataupun dokumen dengan menggunakan penyandian simetris dimana *key* lebih panjang dibanding pesan[4]. Pada IDEA terdapat arsitektur dasar yang terdiri dari putaran (*round)* IDEA, Penjadwalan Kunci (*key)* dan unit Kontrol[5].

a. Pembangkit Kunci

Proses pembangkit kunci pada algoritma IDEA dimulai dari membagi ke-128 bit *key* yang ada menjadi 8 buah (16 bit *subkey*). *Subkey* ini adalah delapan *subkey* yang pertama untuk algoritma IDEA dengan 6 *subkey* pertama untuk *round* (putaran) 1 dan 2 *subkey* yang terakhir untuk *round* 2. Rotasikan *key* ke kiri 25 bit dan bagi lagi menjadi 8 *subkey*. Hasilnya

ISBN: 978-602-52720-7-3
Februari 2020
Hal 438-446

adalah 8 *subkey* yang kedua untuk algoritma berikutnya dengan 4 *subkey* pertama untuk *round 2* dan 4 *subkey* terakhir untuk *round 3.* Algoritma ini hanya menggunakan 52 *subkey* dengan menggunakan perincian 6 buah hasil *subkey* untuk 8 *round* ditambah lagi 4 buah *subkey* untuk perhitungan transformasi *output.* Proses pembangkit kunci (*key)* dapat dilihat pada gambar 1:



Gambar 1. proses pembangkit kunci IDEA [2]

Algoritma IDEA memiliki beberapa kelebihan diantaranya:

1. Standar keamanan penyandian data lebih sederhana namun bisa dibilang cukup ampuh dalam mencegah serangan *cryptanalysis* terhadap *key* enkripsi dan dekripsi.
2. Segi efisiensi waktu cukup baik untuk men-enkripsi dan men-dekripsi data. Dikarenakan struktur algoritma yang singkat namun tidak berpengaruh terhadap kemampuan dalam mengamankan data.[6]

b. Proses Enkripsi Algoritma IDEA

Enkripsi dipakai untuk menyandikan suatu data atau informasi. Namun Berdasarkan cara dalam memproses *plaintext* cipher dapat digolongkan menjadi dua jenis cipher yaitu Stream chiper untuk memproses masukan dan menghasilkan suatu data pada saat bersamaan dan blok chiper yang berkerja dengan memproses suatu data secara blok, dengan menggabungkan beberapa karakter menjadi satu blok[7]. Pada proses enkripsi ini kita menggunakan cipher blok. Proses Meng-enkripsi algotima IDEA dapat dilakukan sebagai berikut, Langkah awal ,plaintext 64 bit akan dibagi atau dipecah menjadi 4 buah subblok, masing-masing subblok panjangnya 16 bit, yaitu X_1, X_2, X_3, X_4. Keempat subblok tersebut menjadi masukan bagi setiap iterasi tahap pertama pada algoritma IDEA dan dengan total 8 iterasi. Lalu subblok yang 16 bit ditransformasikan menjadi sebuah subblok 16 bit dimana subblok yang ditransformasikan tersebut menjadi pesan rahasia 64 bit. Adapun subblok yang ditransformasikan kedalam 16 bit yaitu Y_1, Y_2, Y_3, Y_4. Setiap iterasi, keempat subblok tersebut di XOR kan, dengan menggunakan 6 buah subkey yang panjangnya 16 bit. Pertukarkan sub blok yang kedua dan ketiga. kemudian 4 buah dari sub blok dikombinasi dengan 4 *subkey* didalam transformasi *output* nya. Tahapannya adalah sebagai berikut:

1. Kalikan blok X1 dengan K1 mod $(2^{16} + 1)$.
2. Tambahkan blok X2 dengan K2 mod 2^{16}.
3. Tambahkan blok X3 dengan K3 mod 2^{16}.
4. Kalikan blok X4 dengan K4 mod $(2^{16} + 1)$.
5. XORkan hasil dari step 1 dan 3.
6. XORkan hasil dari step 2 dan 4.
7. Hasil dari step 5 dikalikan dengan K5 mod $(2^{16} + 1)$.
8. Hasil dari step 6 ditambahkan dengan step 7 mod 2^{16}.
9. Kemudian step 8 dikalikan dengan K6 mod $(2^{16} + 1)$.
10. Kemudian hasil dari step 7 Tambahkan dengan Step 9.
11. XORkan hasil dari step 1 dan 9.
12. XORkan hasil dari step 3 dan 9.
13. XORkan hasil dari step 2 dan 10.
14. XORkan hasil dari step 4 dan 10.

Outputnya per*round* adalah 4 sub blok yang telah dihasilkan dari langkah 11-14. *Swap* sub blok dari langkah 12 dan 13 (kecuali pada putaran yang terakhir) sehingga *input* dari putaran yang berikutnya adalah hasil dari kombinasi langkah 11-14. Lalu setelah 8 *round* lakukan transformasi *output* kalikan X1 dengan *Subkey* pada K1 mod $(2^{16} + 1)$, lalu tambahkan

ISBN: 978-602-52720-7-3
Februari 2020
Hal 438-446

X2 dengan *subkey* pada K3 mod 2^{16}, kemudian tambahkan lagi X3 dengan *subkey* K3 mod 2^{16}, terakhir kalikan X4 yang telah didapat dengan *subkey* K3 mod (2^{16} + 1)[8].

c. Proses *Dekripsi* International Data Encryption Algorithm (IDEA)

Proses dekripsi IDEA menggunakan suatu algoritma yang sama dengan Proses *enkripsi*. Perbedaannya yaitu pada 52 *subkey* yang digunakan masing-masing adalah hasil turunan 52 *subkey enkripsi*. Urutan *subkey* dibalik dengan proses *enkripsi* kemudian *subkey*-nya di *inverse*-kan. *Subkey* pada *round* 8 di *inverse*-kan dan kemudian digunakan sebagai *subkey round* 1 dan 2 pada sebuah proses dekripsi.

3. ANALISA DAN PEMBAHASAN

3.1 Proses Pembangkit Kunci

Algoritma IDEA memiliki *input*-an 128 *bit key* yang identik dengan 32 digit heksadesimal ataupun 16 karakter untuk menghasilkan 52 *subkey*. Adapun proses pembangkitan kunci pada algoritma IDEA adalah sebagai berikut:
Input key : OKTAVIANFUADMIRA
Ubah kedalam bentuk Biner, Desimal, dan Hexadesimal seperti tabel berikut:

Tabel 1. proses pembangkitan kunci IDEA

Char	Desimal	Hexadesimal	Biner
O	79	4F	01001111
K	75	4B	01001011
T	84	54	01010100
A	65	41	01000001
V	86	56	01010110
I	73	49	01001001
A	65	41	01000001
N	78	4E	01001110
F	70	46	01000110
U	85	55	01010101
A	65	41	01000001
D	68	44	01000100
M	77	4D	01001101
I	73	49	01001001
R	82	52	01010010
A	65	41	01000001

Putaran ke 1:
Input kunci:
01001111010010111010101000100000101010110010010010100000101001110010001100101010101000001010001000100111010100100101010010010000001
Pecah menjadi 8 kelompok:
KE 1 (Putaran 1) 0100111101001011
KE 2 (Putaran 1) 0101010001000001
KE 3 (Putaran 1) 0101011001001001
KE 4 (Putaran 1) 0100000101001110
KE 5 (Putaran 1) 0100011001010101
KE 6 (Putaran 1) 0100000101000100
KE 1 (Putaran 2) 0100110101001001
KE 2 (Putaran 2) 0101001001000001

Putaran ke 2:
Left shift
(01001111010010111010101000100000101010110010010010100000101001110010001100101010101000001010001000100111010100100101010010010000001, 25)
=
10000010101011001001001010000010100111001000110010101010100000101000100010011010100100101010010010000010100111101001011010101000
Pecah menjadi 8 kelompok:
KE3 (Putaran 2) =1000001010101100
KE4 (Putaran 2) =1001001010000010
KE5 (Putaran 2) =1001110010001100
KE6 (Putaran 2) =1010101010000010

ISBN: 978-602-52720-7-3
Februari 2020
Hal 438-446

KE1 (Putaran 3) =1000100010011010
KE2 (Putaran 3) =1001001010100100
KE3 (Putaran 3) =1000001010011110
KE4 (Putaran 3) =1001011010101000

Putaran ke 3:
Left shift
(10000010101011001001001010000010100111001000110010101010100000101000100010011010100100101001000001010011110100101101010100000, 25)
=
00000101001110010001100101010101000001010001000100110101001001010100100100000101001111010010110101010001000010101011001001001010

Pecah menjadi 8 kelompok:
KE 5 (Putaran 3) = 0000010100111001
KE 6 (Putaran 3) = 0001100101010101
KE 1 (Putaran 4) = 0000010100010001
KE 2 (Putaran 4) = 0011010100100101
KE 3 (Putaran 4) = 0100100100000101
KE 4 (Putaran 4) = 0011110100101101
KE 5 (Putaran 4) = 0101000100000101
KE 6 (Putaran 4) = 0101100100100101

Putaran ke 4:
Left shift
(00000101001110010001100101010101000001010001000100110101001001010100100100000101001111010010110101010000100010101011001001001010, 25)
=
10101010000010100010001001101010010010101001001000001010011110100101110101010001000001010101100100100101010000101001110010001100101010

Pecah menjadi 8 kelompok:
Ke 1 (Putaran 5) = 1010101000001010
Ke 2 (Putaran 5) = 0010001001101010
Ke 3 (Putaran 5) = 0100101010010010
Ke 4 (Putaran 5) = 0000101001111010
Ke 5 (Putaran 5) = 0101101010100010
Ke 6 (Putaran 5) = 0000101010110010
Ke 1 (Putaran 6) = 0100101000001010
Ke 2 (Putaran 6) = 0111001000110010

Putaran ke 5:
Left shift
(10101010000010100010001001101010010010101001001000001010011110100101110101010001000001010101100100100101010000101001110010001100101010, 25)
=
11010100100101010010010000010100111101001011010101000100000101010110010010010100000101001110010001100101010101000001010001000100

Pecah menjadi 8 kelompok :
KE 3 (Putaran 6) = 1101010010010101
KE 4 (Putaran 6) = 0010010000010100
KE 5 (Putaran 6) = 1111010010110101
KE 6 (Putaran 6) = 0100010000010101
KE 1 (Putaran 7) = 0110010010010100
KE 2 (Putaran 7) = 0001010011100100
KE 3 (Putaran 7) = 0110010101010100
KE 4 (Putaran 7) = 0001010001000100

Putaran ke 6:
Left shift
(11010100100101010010010000010100111101001011010101000100000101010110010010010100000101001110010001100101010101000001010001000100, 25)
=
00101001111010010110101010001000001010101100100100101000001010011100100011001010101000001010001000100

Pecah menjadi 8 kelompok :
KE 5 (Putaran 7) = 0010100111101001
KE 6 (Putaran 7) = 0110101010001000

ISBN: 978-602-52720-7-3
Februari 2020
Hal 438-446

KE 1 (Putaran 8) = 0010101011001001
KE 2 (Putaran 8) = 0010100000101001
KE 3 (Putaran 8) = 1100100011001010
KE 4 (Putaran 8) = 1010100000101000
KE 5 (Putaran 8) = 1000100110101001
KE 6 (Putaran 8) = 0010101001001000

Putaran ke 7:
Left shift
(0010100111101001011010101000100000101010110010010010100000101001110010001100101010101000000101000100100110101001001010100100100 0, 25)
=
0001000001010101100100100101000001010011100100011001010101010000010100010001001101010010010101001001000001010011110100101101010101

Pecah menjadi 8 kelompok (4 kelompok terakhir tidak digunakan):
KE 1 (Transformasi *Output)* = 0001000001010101
KE 2 (Transformasi *Output)* = 1001001001010000
KE 3 (Transformasi *Output)* = 0101001110010001
KE 4 (Transformasi *Output)* = 1001010101010000

KUNCI DEKRIPSI:
Kunci = OKTAVIANFUADMIRA
KD1(Putaran 1) = *Inverse*(KE1-Putaran9) = 1110111110101010
KD2(Putaran 1) = *Minus*(KE2-Putaran9) = 0110110110110000
KD3(Putaran 1) = *Minus*(KE3-Putaran9) = 1010110001101111
KD4(Putaran 1) = *Inverse*(KE4-Putaran9) = 0110101010101111
KD5(Putaran 1) = (KE5-Putaran8) = 1000100110101001
KD6(Putaran 1) = (KE6-Putaran8) = 0010101001001000

KD1(Putaran 2) = *Inverse*(KE1-Putaran8) = 1101010100110110
KD2(Putaran 2) = *Minus* (KE2-Putaran8) = 1101011111010111
KD3(Putaran 2) = *Minus* (KE3-Putaran8) = 0011011100110110
KD4(Putaran 2) = *Inverse*(KE4-Putaran8) = 0101011111010111
KD5(Putaran 2) = (KE5-Putaran7) = 0010100111101001
KD6(Putaran 2) = (KE6-Putaran7) = 0110101010001000

KD1(Putaran 3) = *Inverse*(KE1-Putaran7) = 1001101101101011
KD2(Putaran 3) = *Minus* (KE2-Putaran7) = 1110101100011100
KD3(Putaran 3) = *Minus* (KE3-Putaran7) = 1001101010101100
KD4(Putaran 3) = *Inverse*(KE4-Putaran7) = 1110101110111011
KD5(Putaran 3) = (KE5-Putaran6) = 1111010010110101
KD6(Putaran 3) = (KE6-Putaran6) = 0100010000010101

KD1(Putaran 4) = *Inverse*(KE1-Putaran6) = 1011010111110101
KD2(Putaran 4) = *Minus* (KE2-Putaran6) = 1000110111001110
KD3(Putaran 4) = *Minus* (KE3-Putaran6) = 0010101101101011
KD4(Putaran 4) = *Inverse*(KE4-Putaran6) = 1101101111101011
KD5(Putaran 4) = (KE5-Putaran5) = 0101101010100010
KD6(Putaran 4) = (KE6-Putaran5) = 0000101010110010

KD1(Putaran 5) = *Inverse*(KE1-Putaran5) = 0101010111110101
KD2(Putaran 5) = *Minus* (KE2-Putaran5) = 1101110110010110
KD3(Putaran 5) = *Minus* (KE3-Putaran5) = 1011010101101110
KD4(Putaran 5) = *Inverse*(KE4-Putaran5) = 1111010110000101
KD5(Putaran 5) = *(*KE5-Putaran4) = 0101000100000101
KD6(Putaran 5) = (KE6-Putaran4) = 0101100100100101

KD1(Putaran 6) = *Inverse*(KE1-Putaran4) = 1111101011101110
KD2(Putaran 6) = *Minus* (KE2-Putaran4) = 1100101011011011
KD3(Putaran 6) = *Minus* (KE3-Putaran4) = 1011011011111011
KD4(Putaran 6) = *Inverse*(KE4-Putaran4) = 1010111011111010
KD5(Putaran 6) = (KE5-Putaran3) = 0000010100111001
KD6(Putaran 6) = (KE6-Putaran3) = 0001100101010101

KD1(Putaran 7) = *Inverse*(KE1-Putaran3) = 0111011101100101

ISBN: 978-602-52720-7-3
Februari 2020
Hal 438-446

KD2(Putaran 7) = *Minus* (KE2-Putaran3) = 0110110101011100
KD3(Putaran 7) = *Minus* (KE3-Putaran3) = 0111110101100010
KD4(Putaran 7) = *Inverse*(KE4-Putaran3) = 1001011010101000
KD5(Putaran 7) = (KE5-Putaran2) = 1001110010001100
KD6(Putaran 7) = (KE6-Putaran2) = 1010101010000010

KD1(Putaran 8) = *Inverse*(KE1-Putaran2) = 1011001010110110
KD2(Putaran 8) = *Minus* (KE2-Putaran2) = 1010110110111111
KD3(Putaran 8) = *Minus* (KE3-Putaran2) = 0111110101010100
KD4(Putaran 8) = *Inverse*(KE4-Putaran2) = 0110110101111101
KD5(Putaran 8) = (KE5-Putaran1) = 0100011001010101
KD6(Putaran 8) = (KE6-Putaran1) = 0100000101000100

KD1(Putaran 9) = *Inverse*(KE1-Putaran1) = 1011000010110100
KD2(Putaran 9) = *Minus* (KE2-Putaran1) = 1010101110111111
KD3(Putaran 9) = *Minus* (KE3-Putaran1) = 1010100110110111
KD4(Putaran 9) = *Inverse*(KE4-Putaran1) = 1011111010110001

3.2 Proses Enkripsi

sebelumnya *Plainteks* dalam bentuk Doc yang ingin di *enkripsi* terlebih dahulu di *convert* menggunakan *binary viewer* kemudian dikonversikan lagi menjadi bilangan biner. Adapun *Plainteks* yang ingin di-*enkripsi* adalah:
50 4B 03 04
5C 8F CA 01

Tabel 2. Konversi *Plainteks* ke bilangan Binner

Hexadesimal	Binner
50	01010000
5C	01011100
4B	01001011
8F	10001111
03	00000011
CA	11001010
04	00000100
01	00000001

X1 = 0101000001011100
X2 = 0100101110001111
X3 = 0000001111001010
X4 = 0000010000000001

Putaran 1:
1) L#1 = (X1 * K1) mod (2^16 + 1) = 0101000001011100 * 0100111101001011 mod (2^16 + 1) = 1101011000010001
2) L#2 = (X2+K2) Mod 2^16 = 0100101110001111 + 0101010001000001 mod (2^16) = 1001111111010000
3) L#3 = (X3 + K3) mod 2^16 = 0000001111001010 + 0101011001001001 mod (2^16) = 0101101000010011
4) L#4 = (X4 * K4) mod (2^16 + 1) = 0000010000000001 * 0100000101001110 mod (2^16 + 1) = 0111100001001001
5) L#5 = L#1 XOR L#3 = 1101011000010001 XOR 0101101000010011 = 1000110000000010
6) L#6 = L#2 XOR L#4 = 1001111111010000 XOR 0111100001001001 = 1110011110011001
7) L#7 = (L#5 * K5) mod (2^16 + 1) = 1000110000000010 * 0100011001010101 mod (2^16 + 1) = 1110001000110100
8) L#8 = (L#6 + L#7)) mod 2^16 = 1110011110011001 + 1110001000110100 mod 2^16 = 1100100111001101
9) L#9 = (L#8 * K6) mod (2^16 + 1) = 1100100111001101 * 0100000101000100 mod (2^16+1) = 0111010000000010
10) L#10 = (L#7 + L#9) mod 2^16 = 1110001000110100 + 0111010000000010 mod 2^16 = 0101011000110110
11) L#11 = L#1 XOR L#9 = 1101011000010001 XOR 0111010000000010 = 1010001000010011
12) L#12 = L#3 XOR L#9 = 0101101000010011 XOR 0111010000000010 = 0010111000010001
13) L#13 = L#2 XOR L#10 = 1001111111010000 XOR 0101011000110110 = 1100100111100110
14) L#14 = L#4 XOR L#10 = 0111100001001001 XOR 0101011000110110 = 0010111001111111

Untuk Putaran Berikutnya :
X1 = L#11 = 1010001000010011
X2 = L#12 = 0010111000010001
X3 = L#13 = 1100100111100110
X4 = L#14 = 0010111001111111

Putaran 2:
1) L#1 = (X1 * K1) mod (2^16 + 1) = 1010001000010011 * 0100110101001001 mod (2^16 + 1) = 1011110101111110

ISBN: 978-602-52720-7-3
Februari 2020
Hal 438-446

2) L#2 = (X2+K2) Mod 2^16 = 0010111000010001 + 0101001001000001 Mod 2^16 = 1000000001010010
3) L#3 = (X3 + K3) mod 2^16 = 1100100111100110 + 1000001010101100 Mod 2^16 = 0100110010010010
4) L#4 = (X4 * K4) mod (2^16 + 1) = 0010111001111111 * 1001001010000010 mod (2^16 + 1) = 10001011011110011
5) L#5 = L#1 XOR L#3 = 1011110101111110 XOR 0100110010010010 = 1111000111101100
6) L#6 = L#2 XOR L#4 = 1000000001010010 XOR 10001011011110011 = 1001011010100001
7) L#7 = (L#5 * K5) mod (2^16 + 1) = 1111000111101100 * 1001110010001100 mod (2^16 + 1) = 1000100100100001
8) L#8 = (L#6 + L#7) mod 2^16 = 1001011010100001 + 1000100100100001 mod 2^16 = 0001111111000010
9) L#9 = (L#8 * K6) mod (2^16 + 1) = 0001111111000010 * 1010101010000010 mod (2^16 + 1) = 1101111101011110
10) L#10 = (L#7 + L#9) mod 2^16 = 1000100100100001 + 1101111101011110 mod 2^16 = 0110100001111111
11) L#11 = L#1 XOR L#9 = 1011110101111110 XOR 1101111101011110 = 0110001000100000
12) L#12 = L#3 XOR L#9 = 0100110010010010 XOR 1101111101011110 = 1001001111001100
13) L#13 = L#2 XOR L#10 = 1000000001010010 XOR 0110100001111111 = 1110100000101101
14) L#14 = L#4 XOR L#10 = 10001011011110011 XOR 0110100001111111 =10111111010001100

Untuk Putaran Berikutnya :
X1 = L#11 = 0110001000100000
X2 = L#12 = 1001001111001100
X3 = L#13 = 1110100000101101
X4 = L#14 =10111111010001100

Terus lanjutkan perhitungan sama seperti putaran 1 dan putaran 2 hingga sampai 8 putaran dan berhasil menemukan transformasi *output*-nya

Putaran 8:
1) L#1 = (X1 * K1) mod (2^16 + 1) = 1101011101110010 * 0010101011001001 mod (2^16 + 1) = 1011100010000001
2) L#2 = (X2+K2) Mod 2^16 = 0111011111010011 + 0010100000101001 Mod 2^16 = 1001111111111100
3) L#3 = (X3 + K3) mod 2^16 = 0111110100010010 + 1100100011001010 mod 2^16 = 10100010111011100
4) L#4 = (X4 * K4) mod (2^16 + 1) = 1011010100110000 * 1010100000101000 mod (2^16 + 1) = 0101100001111101
5) L#5 = L#1 XOR L#3 = 1011100010000001 XOR 10100010111011100 = 11111110101011101
6) L#6 = L#2 XOR L#4 = 1001111111111100 XOR 0101100001111101 = 1100011110000001
7) L#7 = (L#5 * K5) mod (2^16 + 1) = 11111110101011101 * 1000100110101001 mod (2^16 + 1) = 1111010110000000
8) L#8 = (L#6 + L#7) mod 2^16 = 1100011110000001 + 1111010110000000 mod 2^16 = 1011110100000001
9) L#9 = (L#8 * K6) mod (2^16 + 1) = 1011110100000001 * 0010101001001000 mod (2^16 + 1) = 0011001100010001
10) L#10 = (L#7 + L#9) mod 2^16 = 1111010110000000 + 0011001100010001 mod 2^16 = 0010100010010001
11) L#11 = L#1 XOR L#9 = 1011100010000001 XOR 0011001100010001 = 1000101110010000
12) L#12 = L#3 XOR L#9 = 10100010111011100 XOR 0011001100010001 = 10111011011001101
13) L#13 = L#2 XOR L#10 = 1001111111111100 XOR 0010100010010001 = 1011011101101101
14) L#14 = L#4 XOR L#10 = 0101100001111101 XOR 0010100010010001 = 0111000011101100

Untuk Putaran Berikutnya:
X1 = L#11 = 1000101110010000
X2 = L#12 = 10111011011001101
X3 = L#13 = 1011011101101101
X4 = L#14 = 0111000011101100

Transformasi *output* :
Y1 = (X1*K1) Mod (2^16+1) = 1000101110010000 * 0001000001010101 Mod (2^16+1) = 0100110111101001
Y2 = (X2+K2) Mod (2^16) = 10111011011001101 + 1001001001010000 Mod (2^16) = 0000100100011101
Y3 = (X3+K3) Mod (2^16) = 1011011101101101 + 0101001110010001 Mod (2^16) = 0000101011111110
Y4 = (X4*K4) Mod (2^16+1) = 0111000011101100 * 1001010101010000 Mod (2^16+1) = 0110001111100100

Hasil *Enkripsi:*
Y1 = 01001101 11101001 = 4D E9
Y2 = 00001001 00011101 = 09 1D
Y3 = 00001010 11111110 = A FE
Y4 = 01100011 11100100 = 63 E4

3.3 Proses Dekripsi.

Proses dekripsi adalah kebalikan dari proses men-enkripsi. Proses dekripsi menggunakan perhitungan algoritma yang serupa dengan proses enkripsi. Sebagai contoh, misalkan ingin didekripsikan kembali hasil enkripsi yang telah didapatkan diatas, maka proses dekripsinya adalah sebagai berikut:
Chiper Text: 4D 09 A 63
E9 1D FE E4

ISBN: 978-602-52720-7-3
Februari 2020
Hal 438-446

Putaran 1:
1) L#1 = (X1 * K1) mod (2^16 + 1) = 0100110111101001 * 1110111110101010 mod (2^16 + 1) = 1111101011001011
2) L#2 = (X2+K2) Mod 2^16 = 0000100100011101 + 0110110110110000 Mod 2^16 = 0111011011001101
3) L#3 = (X3 + K3) mod 2^16 = 0000101011111110 + 1010110001101111 mod 2^16 = 1011011101101101
4) L#4 = (X4 * K4) mod (2^16 + 1) = 0110001111100100 * 0110101010101111 mod (2^16 + 1) = 1000011100111100
5) L#5 = L#1 XOR L#3 = 1111101011001011 XOR 1011011101101101 = 0100110110100110
6) L#6 = L#2 XOR L#4 = 0111011011001101 XOR 1000011100111100 = 1111000111110001
7) L#7 = (L#5 * K5) mod (2^16 + 1) = 0100110110100110 * 1000100110101001 mod (2^16 + 1) = 1110111011010110
8) L#8 = (L#6 + L#7)) mod 2^16 = 1111000111110001 + 1110111011010110 mod 2^16 = 1110000011000111
9) L#9 = (L#8 * K6) mod (2^16 + 1) = 1110000011000111 * 0010101001001000 mod (2^16 + 1) = 1011100011011001
10) L#10 = (L#7 + L#9) mod 2^16 = 1110111011010110 + 1011100011011001 mod 2^16 = 1010011110101111
11) L#11 = L#1 XOR L#9 = 1111101011001011 XOR 1011100011011001 = 0100001000010010
12) L#12 = L#3 XOR L#9 = 0111011011001101 XOR 1011100011011001 = 1100111000010100
13) L#13 = L#2 XOR L#10 = 0111011011001101 XOR 1010011110101111 = 1101000101100010
14) L#14 = L#4 XOR L#10 = 1000011100111100 XOR 1010011110101111 = 0010000010010011

Untuk Putaran Berikutnya:
X1 = L#11 = 0100001000010010
X2 = L#12 = 1100111000010100
X3 = L#13 = 1101000101100010
X4 = L#14 = 0010000010010011

Terus lanjutkan proses putaran hingga putaran ke-8 sampai ditemukan transformasi *output*-nya seperti dibawah ini :

Putaran 8:
1) L#1 = (X1 * K1) mod (2^16 + 1) = 0100101010110110 *1011001010110110 mod (2^16 + 1) = 0111010100111101
2) L#2 = (X2+K2) Mod 2^16 = 1101110001110111 + 1010110110111111 Mod 2^16 = 1000101000110110
3) L#3 = (X3 + K3) mod 2^16 = 0000011010111110 + 0111110101010100 Mod 2^16 = 1000010000010010
4) L#4 = (X4 * K4) mod (2^16 + 1) = 1110001011100100 * 0110110101111101 mod (2^16 + 1) = 0111110001001011
5) L#5 = L#1 XOR L#3 = 0111010100111101 XOR 1000010000010010 = 1111000100101111
6) L#6 = L#2 XOR L#4 = 1000101000110110 XOR 0111110001001011 = 1111011001111101
7) L#7 = (L#5 * K5) mod (2^16 + 1) = 1111000100101111 * 0100011001010101 mod (2^16 + 1) = 1010110001011001
8) L#8 = (L#6 + L#7) mod 2^16 = 1111011001111101 + 1010110001011001 Mod 2^16 = 1010001011010110
9) L#9 = (L#8 * K6) mod (2^16 + 1) = 1010001011010110 * 0100000101000100 mod (2^16 + 1) = 0110110101010101
10) L#10 = (L#7 + L#9) mod 2^16 = 1010110001011001 + 0110110101010101 Mod 2^16 = 0001100110101110
11) L#11 = L#1 XOR L#9 = 0111010100111101 XOR 0110110101010101 = 0001100001101000
12) L#12 = L#3 XOR L#9 = 1000010000010010 XOR 0110110101010101 = 1110100101000111
13) L#13 = L#2 XOR L#10 = 1000101000110110 XOR 0001100110101110 = 1001001110011000
14) L#14 = L#4 XOR L#10 = 0111110001001011 XOR 0001100110101110 = 0110010111100101

Untuk Putaran Berikutnya:
X1 = L#11 = 0001100001101000
X2 = L#12 = 1110100101000111
X3 = L#13 = 1001001110011000
X4 = L#14 = 0110010111100101

Transformasi *Output:*
Y1 = (X1*K1) Mod (2^16+1) = 0001100001101000 * 1011000010110100 Mod (2^16+1) = 0101000001011100
Y2 = (X2+K2) Mod (2^16) = 1110100101000111 + 1010101110111111 Mod (2^16) = 0100101110001111
Y3 = (X3+K3) Mod (2^16) = 1001001110011000 + 1010100110110111 Mod (2^16) = 0000001111001010
Y4 = (X4*K4) Mod (2^16+1) = 0110010111100101 * 1011111010110001 Mod (2^16+1) = 0000010000000001

Hasil Dekripsi:
Y1 = 01010000 01011100 = 50 5C
Y2 = 01001011 10001111 = 4B 8F
Y3 = 00000011 11001010 = 03 CA
Y4 = 00000100 00000001 = 04 01

4. KESIMPULAN

IDEA adalah sebuah algoritma berbentuk ciper blok yang diciptakan oleh Xuejia Lai dan James Massey dari ETH Zurich dan pada tahun1991 diperkenalkan. Algoritma IDEA menggunakan 128 bit kunci ataupun 32 digit hexadesimal dan 16 karakter yang menghasilkan 52 *subkey* dengan hasil 6 buah *subkey* digunakan untuk 8 *round* dan 4 *subkey* untuk transformasi output dan digunakan untuk mengenkripsi *plaintext* 64 bit kemudian mendekripsikannya kembali dengan *key* yang serupa.

ISBN: 978-602-52720-7-3
Februari 2020
Hal 438-446

Algoritma IDEA ini dapat digunakan untuk mengenkripsi ataupun mengamankan *file* bertipe DOCX, agar tidak sembarang dibuka oleh pihak ketiga yang tidak berkepentingan.

REFERENCES

[1] M. K. Emy Setyaningsih, S.Si., *Kriptografi dan Implementasinya menggunakan MATLAB*. Yogyakarta: CV.ANDI OFFSET, 2015.
[2] M. Kholidya Yuli Wardani, M.Zen S.Hadi, ST, MSc, Mike Yuliana, ST, "IMPLEMENTASI METODE KRIPTOGRAFI IDEA pada PRIORITY DEALER untuk LAYANAN PEMESANAN dan LAPORAN PENJUALAN HANDPHONE BERBASIS WEB," *IMPLEMENTASI Metod. KRIPTOGRAFI IDEA pada Prior. Deal. untuk LAYANAN PEMESANAN dan Lap. PENJUALAN HANDPHONE Berbas. WEB*, p. 1, 2015.
[3] D. L. C. P. Tri Andriyanto, "STUDI PERBANDINGAN ALGORITMA IDEA DAN ALGORITMA BLOWFISH," *KOMMIT 2008*, no. 1411–6286, 2008.
[4] A. T. B. Sarwono Sutikno, "PENERAPAN ALUR DESAIN ALLIANCE DALAM PERANCANGAN CORE PROSESOR KRIPTO IDEA."
[5] P. F. S. Raka Yusuf, "PENERAPAN METODE ENKRIPSI IDEA, FUNGSI HASH MD5, DAN METODE KOMPRESI HUFFMAN UNTUK KEAMANAN DAN EFISIENSI RUANG DOKUMEN," *SINAPTIKA*, no. 2086–8251, 2010.
[6] H. F. Abdul Hanan, "METODE ENKRIPSI DAN DEKRIPSI DATA MENGGUNAKAN KRIPTOGRAFI IDEA."
[7] J. Sasongko, "PENGAMANAN DATA INFORMASI MENNGUNAKAN KRIPTOGRAFI KLASIK," *Teknol. Inf. Din.*, vol. X, no. 0854–9524, p. 3, 2005.
[8] A. W. Raka Yusuf, "Pengamanan Data dengan Kompresi LZSS dan Enkripsi IDEA Berbasis Web," *Semin. Nas. Pengaplikasian Telemat. SINAPTIKA*, no. 2086–8251, p. 5, 2010.