Terbit *online* pada laman : https://journal.fkpt.org/index.php/BIT

BULLETIN OF INFORMATION TECHNOLOGY (BIT)

ISSN (Media Online) 2722-0524

Implementasi Algoritma Xtea (*Extended Tyni Encryption Algoritma*) Dalam Pengamanan Data File Dokumen Teks

Handayani Pandiangan

Program Studi Teknik Informatika, STMIK Budi Darma, Medan, Indonesia
Email: handayanipandiangan96@gmail.com

INFORMASI ARTIKEL

Sejarah Artikel:
Diterima Redaksi : 07 November 2020
Revisi Akhir : 15 November 2020
Diterima : 20 November 2020
Diterbitkan *Online* : 28 November 2020

KATA KUNCI

Cryptography, XTEA Algorithm, Securing Document Files

KORESPONDENSI

E-mail:
handayanipandiangan96@gmail.com

A B S T R A C T

At this time the security of the data stored on the computer is necessary, one of which is to secure the document file can be done is cryptography. Cryptography techniques can be used one that can be done is encryption, namely the encryption process before the file is sent and the description is the encryption process after the file is received, so that the process can be kept confidential, and only the person concerned can find out. Confidentiality is a service that is used to maintain information from any party that is not authorized to access it. Thus information can only be accessed by those who have the right to guarantee the security of false information.

1. PENDAHULUAN

Pemakaian teknologi informasi sudah menjadi suatu kebutuhan. Dengan komputer banyak pekerjaan dalam berbagai organisasi dapat diselesaikan dengan lebih cepat, akurat dan efisien. Namun selain berbagai macam keuntungan yang ditawarkan dengan pemakain teknologi informasi tersebut, terdapat bahaya yang ditimbulkan dengan adanya kemungkinan kebocoran data atau penyalahgunaan data yang dapat berakibat fatal bagi suatu organisasi.

File dokumen adalah objek digital yang berfungsi untuk menampung data baik teks, gambar, audio, program, dan lain-lain, yang diberi nama tertentu secara digital. Keinginan pihak lain untuk mengetahui apa saja isi yang ada di dalam dokumen komputer pemilik merupakan suatu ancaman, pemilik akan merasa terganggu dan terancam privasinya apabila pihak lain mengakses komputernya tanpa sepengetahuan atau pemberitahuan terlebih dahulu. Dimana di dalam komputer tersebut tersimpan beberapa arsip rahasia data-data penting atau data pribadi sehingga apabila pihak lain membuka atau bahkan mencuri data maka akan merugikan pemilik komputer. Dokumen-dokumen penting dan rahasia agar tidak mudah diakses oleh orang lain sebaiknya diberikan perhatian khusus berupa pemberian proteksi demi keamanan data yang ada didalam dokumen tersebut.

Ada pun Teknik lain yang dapat digunakan dalam pengamanan data file dokumen teks adalah dengan menggunakan kriptografi. Kriptografi ialah ilmu yang mempelajari tentang pengamanan suatu pesan atau dokumen, sehingga data tersebut tidak dapat dibaca oleh pihak yang tidak berhak (*anauthorized persons*). Kriptografi terdiri dari dua proses yaitu enkripsi dan dekripsi. Proses enkripsi dilakukan agar data tidak dapat diketahui oleh pihak yang berwenang, dengan cara mengubah data asli menjadi suatu kode dengan suatu kunci tertentu yang tidak dapat dipecahkan oleh orang lain.

XTEA (*Extended Tiny Encryption Algoritma*) merupakan kriptografi *symmetric blok chiper* karena menggunakan kunci rahasia dan dioperasikan pada 64 *bit* pesan sekaligus dimana kunci rahasia dengan panjang 128 *bit* dibagi antara pengguna.[1]

2. METODOLOGI PENELITIAN

2.1 Kriptografi

Kriptografi (*cryptography)* berasal dari bahasa Yunani, yaitu dari kata *crypto* dan *graphia* yang berarti "penulisan rahasia". Kriptografi adalah ilmu atau pun seni yang mempelajari bagaimana membuat suatu pesan yang dikirim oleh pengirim dapat disampaikan kepada penerima dengan aman. Kripografi merupakan bagian dari suatu cabang ilmu matematika yang disebut kriptologi (cryptology) Kriptografi bertujuan menjaga kerahasiaan informasi yang terkandung dalam data sehingga informasi tersebut tidak dapat diketahui oleh pihak yang tidak sah.[2]

BULLETIN OF INFORMATION TECHNOLOGY (BIT)
Volume 1, No. 3, November 2020, pp 122- 133
ISSN 2722-0524 (media online)

2.2. Enkripsi dan Dekripsi

Proses penyandian plainteks menjadi chiperteks disebut enkripsi *(encryption)* atau *enchiphering* (standart nama menurut ISO 7498-2). Sedangkan proses pengembalian chiperteks menjadi plainteks semula dinamakan dekripsi *(decryption)* atau *deciphering* (standart nama menurut ISO 7498-2). Enkripsi dan dekripsi dapat diterapkan baik pada pesan yang dikirim maupun pada pesan tersimpan. Istilah *encryption of data in motion* mengacu pada enkripsi pada pesan yang ditranmisikan melalui saluran komunikasi, sedangkan istilah *encryption of data at-rest* mengacu pada enkripsi dokumen yang disimpan di dalam storage. Contoh encrytion of data in motion adalah pengiriman nomor PIN dar mesin ATM ke komputer *server* di kantor pusat bank. Contoh *encryption of data at-rest* adalah enkripsi *file* basis data di dalam *hard disk.*[3]

2.3. Extended Tiny Encryption Algoritma (XTEA)

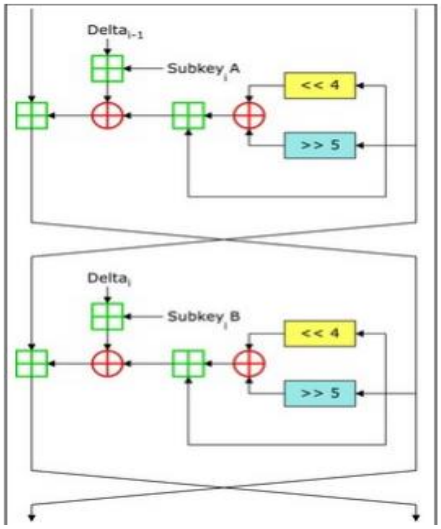
Algoritma XTEA (*Extended Tiny Encryption Algoritma)* adalah salah satu algoritma yang dapat digunakan untuk melakukan enkripsi data sehingga data asli hanya dapat dibaca oleh seseorang yang memiliki kunci enkripsi tersebut. Algoritma ini merupakan mengembangan dari TEA (*Tiny Encryption Algoruthm*). Dan dikembangkan untuk memperbaiki kelemahan dari algoritma tersebut. Perbedaan dengan algoritma sebelumnya adalah menggunkan key yang lebih kompleks dan pengaturan urutan dari operasi shift, XOR dan penambahan.

Wheeler dan Needham menciptakan XTEA *(Extended Tiny Encryption Algoritma)* pada tahun 1997 untuk menutupi kelemahan pada TEA. Sama seperti TEA, XTEA juga beroperasi dalam ukuran blok 64 bit dan panjang kunci 128 bit. Bentuk jaringan Feistel nya pun masih sama, yang membedakan adalah fungsi Feistel dan penjadwalan kunci yang digunakan. Pada XTEA, p ada ronde ganjil digunakan K[sum & 3], sedangkan pada ronde genap digunakan K[sum >> 11 & 3]. Algortima ini merupakan algortima penyandian block chiper yang dirancang untuk penggunaan memory yang seminimal mungkin dengan kecepatan proses yang maksimal.

Sebagai studi sistematis pertama, telah dicatat bahwa ukuran kode yang kecil dan rendah persyaratan penyimpanan memenuhi syarat untuk operasi enkripsi perangkat lunak yang biasanya di-host oleh sistem *embedded* kecil. Selanjutnya, *enkripsi* XTEA algoritma dikembangkan dari TEA asli oleh para ahli yang sama sebagi mantan ketegangan, dimana dilaporkan sebagai alternatif yang berharga dan inovatif untuk di keamanan berkerut ketika dilengkapi dengan operasi penguncian utama. Meskipun XTEA dianggap sebagai salah satu algoritma ringan yang paling penting, ia menderita dari kelemahan keamanan putaran rendah dan harus dapat menampung 32 putaran untuk mengakomodasi aplikasi keamanan tinggi. Secara rinci, XTEA mengimplementasikan *enkripsi* menggunakan blok 64-bit dibagi menjadi dua bagian 32-bit v0 dan v1 yang merupakan masukan untuk rutin algroritmik itu membentuk 32 putaran (Nr =32).

Langkah-langkah penggunaan algoritma ini adalah :

1. Tentukan kalimat yag akan dienkrip
2. Tentukan kata kunci enkripsi yang digunkan
3. Inisialisasi variabel yang digunakan oleh motede ini.
4. Enkripsi kalimat awal menggunakan algoritma ini.
5. Lakukan dekripsi dari kalimat yang telah terenkripsi



Gambar 1 Satu putaran enkripsi dalam *Jaringan Feistel*. [6]
Sumber : International Journal of Scientific and Research Publication.

Gambar 2.2 merupakan satu putaran enkripsi dalam jaringan *feistel* pada algoritma XTEA. XTEA memamfaatkan operasi aritmatika dan logika. Peningkatan pertama adalah untuk menyesuaikan jadwal kunci dan kedua adalah untuk memperkenalkan materi kunci secara perlahan.

BULLETIN OF INFORMATION TECHNOLOGY (BIT)
Volume 1, No. 3, November 2020, pp 122- 133
ISSN 2722-0524 (media online)

Perancangan algoritma kriptografi yang berbasis blok mempertimbangkan salah satu prinsip jaringan *feistel*. Hampir semua algoritma *chiper* blok bekerja dalam model jaringan *Feistel.* jaringan *feistel* ditemukan oleh Horst Feistel tahun 1970. Model jaringan *feistel* adalah sebagai berikut:

1. Bagi blok yang panjangnya *bit* menjadi dua bagian, kiri (*L*) dan kanan (*R*), yang masing-masing panjangnya n/2 (hal ini mensyaratkan *n* harus genap).
2. Definisikan *chiper* blok berulang dimana hasil dari putaran ke-*i* ditentukan dari hasil putaran sebelumnya.

3. ANALISA DAN PEMBAHASAN

3.1 Analisa

Analisa adalah sebuah tugas perancangan perangkat lunak yang menjambatani jurang diantara pengalokasian perangkat lunak tingkat sistem dan perancangan perangkat lunak tigkat program, dalam hal ini perancagan perangkat lunak tingkat program, dalam hal ini perancangan *interface* program aplikasi Proses teknik kriptografi modren meggunakan algortima XTEA untuk menunjang mata kuliah keamanan komputer, memungkinkan perancangan sistem dalam menentukan fungsi dan kinerja perangkat lunak, menunjukkan *interface* perangkat lunak dengan elemen-elemen sistem yang lain dan membangun batasan yang harus dipenuhi oleh perangkat lunak.

3.2 Proses Pembentukan Kunci Algoritma XTEA

Proses pembangkit kunci merupakan proses dimana *cipherkey* dibentuk untuk menghasilkan nilai penjadwalan kunci yang digunakan untuk proses enkripsi dan dekripsi pada algoritma XTEA. Berikut proses pembentukan kunci pada algoritma XTEA, jika diketahui kunci yang digunakan untuk enkripsi dengan panjang 16 byte yaitu : *cipherkey* : 1234567890123456

Tahap awal ubah *cipherkey* kedalam bentuk *decimal* menjadi sebagai berikut :

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**0**	**1**	**2**	**3**	**4**	**5**	**6**
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
49	**50**	**51**	**52**	**53**	**54**	**55**	**56**	**57**	**48**	**49**	**50**	**51**	**52**	**53**	**54**

Tahap selanjutnya masukkan nilai *decimal cipherkey* kedalam fungsi pengacakan kunci. Dengan melakukan pergeseran dan operasi OR untuk menghasilkan *subkey.* Masukkan *cipherkey* kedalam blok 16 byte, dimana 1 proses pengacakan membutuhkan 4 byte *cipherkey.*

s [0] = ((49 AND 255) shl 24) OR ((50 AND 25 5) shl 16) OR ((51AND 255) shl 8) OR ((51 AND 255)
s [0] = 822083584 OR 3276800 OR 13056 OR 52
s [0] = 825373492

S[0] ini merupakan pembangkit kunci yang digunakan untuk proses enkripsi dan dekripsi pada algoritma XTEA, selanjutnya dilakukan perulangan hingga didapat nilai S[0] hingga S[3].

s[1] = ((53 AND 255) Shl 24) OR ((54 AND 255) Shl 16) OR ((55 AND 255) Shl 8) OR ((56 AND 255))
s[1] = 889192448 OR 3538944 OR 14080 OR 56
s[1] = 892745528
s[2] = ((57 AND 255) Shl 24) OR ((48 AND 255) Shl 16) OR ((49 AND 255) Shl 8) OR ((50 AND 255))
s[2] = 956301312 OR 3145728 OR 12544 OR 50
s[2] = 959459634
s[3] = ((51 AND 255) Shl 24) OR ((52 AND 255) Shl 16) OR ((53 AND 255) Shl 8) OR ((54 AND 255))
s[3] = 855638016 OR 3407872 OR 13568 OR 54 s[3] = 859059510

Sehingga didapat nilai penjadwalan kunci, sebagai berikut :
s[0] = 825373492
s[1] = 892745528
s[2] = 959459634
s[3] = 859059510

Subkey ini yang digunakan untuk proses enkripsi atau dekripsi pada algoritma XTEA.

3.3 Proses Enkripsi Algoritma XTEA

Proses enkripsi pada algoritma XTEA dilakukan dengan cara mengambil setiap *plaintext* per blok 8 byte dan memecahnya menjadi round ganjil dan genap. Berikut enkripsi algortima XTEA, jika diketahui kunci dengan panjang 16 byte dan plaintext yang akan digunakan untuk enkripsi dengan panjang 16 byte.

Cipherkey : 1234567890123456
Plaintext : " STMIK BUDI DARMA"
Tahap awal ubah plaintext kedalam bentuk decimal menjadi sebagai berikut:

S	**T**	**M**	**I**	**K**	**spasi**	**B**	**U**	**D**	**I**	**spasi**	**D**	**A**	**R**	**M**	**A**
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
83	**84**	**77**	**73**	**75**	**32**	**66**	**85**	**68**	**73**	**32**	**68**	**65**	**82**	**77**	**65**

Volume 1, No. 3, November 2020, pp 122- 133
ISSN 2722-0524 (media online)

Pecah plaintext per blok 8 byte. Untuk setiap blok dipecah menjadi round ganjil dan round genap yang masing-masing memiliki 4 byte karakter

v0 = ((83 AND 255) Shl 24) OR ((84 AND 255) Shl 16) OR ((77 AND 255) OR (73 And 255)
v0 = 1392508928 OR 5505024 OR 19712 OR 73
v0 =1398033373

v1 = ((75 AND 255) shl 24) OR ((32 AND 255) shl 16) OR ((66 AND 255) shl 8) OR ((85 AND 255)
v1 = 1258291200 OR 2097152 OR 16896 OR 85
v1 = 1260405334
v2 = ((68 AND 255) shl 24) OR ((73 AND 255) shl 16) OR ((32 AND 255) shl 8) OR ((68 AND 255)
v2 = 1140850688 OR 4784128 OR 8192 OR 68
v2 = 1145643076

v3 = ((65 AND 255) shl 24) OR ((82 AND 255) shl 16) OR ((77 AND 255) shl 8 ((65 AND 255)
v3 = 1090519040 OR 5373952 OR 19456 OR 65
v3 = 1095892265

Inisialisasikan DELTA dengan nilai 0x9E3779B9 (-1640531527 dalam integer). Hitung nilai v0 dan v1 dengan nilai awal sum = 0. Perhitungan dilakukan dengan menggunakan nilai integer yang memiliki batas 2147483648 s/d 2147483648. Nilai S[0..3] diambil dari hasil pembangkit kunci yang telah didapatkan dari proses pembangkitan kunci.

v0 += (((v1 Shl 4) XOR (v1 Shr 5) + v1) XOR (sum + S[sum AND 3])

v0 + = (((1260405334 shl 4) XOR (1260405334 shr 5) XOR + 1260405334 XOR(0 + 5[0 AND 3])
v0+= (((-1308351136 XOR 39389072) +1260405334) XOR S[0]
v0 + = ((-1336073488) + 1260405334) XOR 852373492
v0 + = 75668154 XOR 825373492
v0 + = 900770190
v0 = 1398033737 + 900770190 =
-1996163369
sum + = Delta
sum = 0 + (-1640531527) =
-1640531527

v1 + = (((v0 Shl 4) XOR (v0 Shr 5) + v0) XOR (sum + S[sum >> 11 AND 3]

v1+=(((-1996163369shl4)XOR (1996163369 shr 5)+
(-1996163369)XOR (-1640531527 + s[(-1640531527) shr 11 AND 3])
v1+=((-1937222937) + (-1996163369) XOR(-1640531527+ 859059510)
v1 + = -3933389306 XOR -781472017
v1 + = -3303466835
v1 + = 1260405334 + (-3303466835) = -2043061501

Maka dapat nilai akhir v0 dan v1 yaitu
V0 = 1996163369 ; v1 = -2043061501
Ubah nilai akhir v0 dan v1 menjadi karakter ASCII dengan melakukan pergeseran. Nilai desimal yang diambil berupa nilai byte (8 bit).
V0 shr 24
-1996163369 shr 24 = -118
V0 shr 16
-1996163369 shr 16 = 31
V0 shr 8
-1996163369 shr 8 = 78
V0 =34

V1 shr 24
2043061501 shr 24 = 60
V0 shr 16
2043061501 shr 16 = 32
V0 shr 8
2043061501 shr 8 = 80
v1 = 102

BULLETIN OF INFORMATION TECHNOLOGY (BIT)
Volume 1, No. 3, November 2020, pp 122- 133
ISSN 2722-0524 (media online)

Desimal (byte)	**-118**	**31**	**78**	**34**
Biner	**0111 0110**	**0001 1111**	**0111 1111**	**0010 0010**
Karakter	**V**		**N**	**“**

Desimal (byte)	**60**	**32**	**80**	**102**
Biner	**0010 1100**	**0010 0000**	**0101 0000**	**0110 0110**
Karakter	**<**		**P**	**f**

Sehingga di dapat 8 karakter *chipertext*, selanjutnya lakukan setiap blok dipecah round ganjil dan round genap yang masing-masing memiliki 4 *byte* karakter.
v0 += (((v1 Shl 4) XOR (v1 Shr 5) + v1) XOR (sum + S[sum AND 3])

v0+ = (((1095892265 shl 4) XOR (1095892265 shr 5) + 1095892265) XOR (0 + s[0 AND 3])
v0+ = (((17534276240 XOR 34246633) + 1095892265) XOR 825373492
v0+ = 18663037602 XOR 825373492
v0+ = 18947074454
v0 = 1145643076 + 18947074454 = 2009271753
Sum+ = Delta
Sum=0+(-1640531527)= -1640531527

v1 += (((v0 Shl 4) XOR (v0 Shr 5) + v0) XOR (sum + S[sum >> 11 AND 3])
v1+ = (((2009271753 shl 4) XOR (2009271753 shr 5) XOR 2009271753)+s[(- 1640531527 + s[3])
v1+ = ((3214834804 XOR 62789742) + 2009271753) XOR (-1640531527) + 859059510
v1+ = 5165511139XOR (-781472017)
v1+ = -5400849140
v1 = 1095892265 + (-5400849140) =
-4304 956879
Maka dapat nilai akhir v0 dan v1 yaitu

v0 = 2009271753; v1 = -4304956875

Ubah nilai akhir v0 dan v1 menjadi karakter ASCII dengan melakukan pergeseran. Nilai desimal yang diambil berupa nilai byte (8 bit).
Vo shr 24v1 shr 24
2009271753 shr 24 = 119
V0 shr 16
2009271753 shr 16 = 31
V0 shr 8
2009271753 shr 8 = 79
V=63
24v1 shr 24
-4304956875 shr 24 = 56
V0 shr 16
-4304956875 shr 16 =65

V0 shr 8
-4304956875 shr 8 = 17
v1 = 152

Desimal (byte)	**119**	**31**	**79**	**63**
Biner	**0111 0111**	**0001 1111**	**0100 1111**	**0011 1111**
Karakter	**w**		**O**	**?**

BULLETIN OF INFORMATION TECHNOLOGY (BIT)
Volume 1, No. 3, November 2020, pp 122- 133
ISSN 2722-0524 (media online)

Desimal (byte)	**-56**	**65**	**17**	**152**
Biner	**0011 1000**	**0100 0001**	**0001 0001**	**1001 1000**
Karakter	**8**	**A**		**ÿ**

Sehingga di dapat *chipertext*, dari *Plaintext :*
"**STMIK BUDI DARMA**" adalah " **vN"<PfWO?8Aÿ** "

3.4 Analisis Proses Dekripsi Algoritma XTEA

Proses enkripsi pada algoritma XTEA dilakukan dengan cara mengambil setiap plaintext per blok 8 byte dan memecahnya menjadi round ganjil dan genap. Contoh dekripsi pada algoritma XTEA.

1. *Chipertext* : **vN"<PfWO?8Aÿ**
2. Konversi karakter *plaintext* ke dalam bentuk desimal

V		**N**	**"**	**<**		**P**	**f**
118	**31**	**78**	**34**	**60**	**32**	**80**	**102**
W		**O**	**?**	**8**	**A**		**ÿ**
119	**31**	**79**	**63**	**56**	**65**	**17**	**152**

3. Masukkan kunci dekripsi yang sama dengn kunci enkripsi yaitu 1234567890123456 sehingga di dapat nilai penjadwalan kunci S[0..3]
   yaitu:
   S[0] = 825373492
   S[1] = 892745528
   S[2] = 959459634
   S[3] = 859059510
4. Pecah plaintext per blok 8 byte. Untuk setiap blok dipecah menjadi round ganjil dan round genap yang masing-masing memiliki 4 byte karakter.

v0 = ((118 AND 255) shl 24) OR ((31 AND 255) shl 16) OR ((78 AND 255) shl8) OR((34 AND 255)
v0 = ((1979711488 OR 2031616 OR 19968 OR 34
v0 = 1981763106

v1 = ((60 AND 255) shl 25) OR ((32 AND 255) shl 16) OR ((80 AND 255) shl 8) OR (102 AND 255)
v1 = (1006632960 OR 2097152 OR 20480 OR 102
v1 = 1008750694

Inisialisasikan DELTA dengan nilai 0x9E3779B9 (-1640531527 dalam integer) dan putaran (n) sebanyak 32 round. Hitung nilai v0 dan v1 dengan nilai awal n = 32 dan nilai awal sum = DELTA * round = -957401312. Perhitungan dilakukan dengan menggunakan nilai integer yang memiliki batas -2147483648 s/d 2147483648.

v1 - = (((v0 Shl 4) XOR (v0 Shr 5) + v0) XOR (sum + S[sum Shr 11 AND 3])
v1- = (((1981763106 shl 4) XOR (1981763106 shl 5) + 1981763106) XOR (-957401312 + s[2])
v1- =7600285114XOR (-1326417662)
v1- = -6611051336
v1 = 1008750694 – (-6611051336) = 7619802030

sum- = Delta
sum = (-957401312) – (-1640531527) = 683130215

v0 - = (((v1 Shl 4) XOR (v1 Shr 5) + v1) XOR (sum + S[sum AND 3])

v0- = (((1008750694 shl 4) XOR (1008750694 shr 5) + 1008750694 XOR (683130215 + s[683130215 AND 3])
v0- = (((1614001110 XOR 31523459) + (1008750694 + s[683130215 AND 3])
v0- = 2649948603 XOR 1542189725
v0- = 3323523878
v0 = 1008750694 – 3323523878 = -2314773184

maka dapat nilai akhir v1 dan v0 yaitu;

BULLETIN OF INFORMATION TECHNOLOGY (BIT)
Volume 1, No. 3, November 2020, pp 122- 133
ISSN 2722-0524 (media online)

v0 = 7619802030 ; v1 = 2314773184
v0 shr 24
7619802030 shr 25 = 83
v1 shr 16
7619802030 shr 16 = 84
v1 shr 8
7619802030 shr 8 = 77
7619802030 = 73

v1 shr 24
-2314773184 shr 24 = 75
v0 shr 16
-2314773184 shr 16 = 32
v1 shr 8
-2314773184 shr 8 = 66
-2314773184 shr = 56

Desimal (byte)	**83**	**84**	**77**	**73**
Biner	**0101 0011**	**0101 0100**	**0100 1101**	**0100 1001**
Karakter	**S**	**T**	**M**	**I**

Desimal (byte)	**75**	**32**	**66**	**85**
Biner	**0100 1011**	**0010 0000**	**0100 0010**	**0101 0101**
Karakter	**K**	**spasi**	**B**	**U**

Sehingga di dapat 8 karakter *chipertext*, selanjutnya lakukan setiap blok dipecah round ganjil dan round genap yang masing-masing memiliki 4 *byte* karakter.
V0- = (((119 AND 255) shl 24) OR (31 AND 255) shl 16) OR (79 AND 255) shl 8) OR (63 AND 255)
V0- = ((1996488704 OR 2031616 OR 20224 OR 63)
V0- = 1998540607
V1- = (((56 AND 255) shl 24) OR (65 AND 255) shl 16) OR (17 AND 255) shl 8) OR(152 AND 255)
V1- = ((939524096 OR 4259840 OR 4352 OR 152)
V1- = 939528695

v1 -= (((v0 Shl 4) XOR (v0 Shr 5) + v0) XOR (sum + S[sum Shr 11 AND 3])
v1- =(((1998540607 shl 4) XOR (1998540607shl5)+1998540607XOR(-957401312)+ 56957401312) shr 11 AND 3)
v1- = (3197664971 XOR 6395329942) XOR 1998540607) XOR (-957401312 + s[2])
v1- = (7577568093 +1998540607) XOR(-957401312+959459634)
v1 = 9576108700–2058322 = 9574050378
sum = Delta
sum = (-957401312) – (-1640531527) = 683130215

v0 -= (((v1 Shl 4) XOR (v1 Shr 5) + v1) XOR (sum + S[sum AND 3])
v0- = (((939528695 shl 4) XOR (939528965 shr 5) + 939528695 XOR(683130215 + s[683130215 and 3])
v0- = (((1503245912 XOR 29360271) + 939528695) XOR 683130215 + s[3])
v0- = (1482274519 + 939528695) XOR (683130215 + 859059510)
v0- = 2421803214 XOR 1542189725
v0- = 3417455187
v0 = 1998540607 – 3417455187 = -1418914580
maka dapat nilai akhir v1 dan v0 yaitu;
v0= 9574050378;v1= -1418919580

Ubah nilai integer dari hasil perhitungan menjadi karakter ASCII dengan melakukan pergeseran. Nilai desimal yang diambil berupa nilai byte (8 bit).
V0 shr 24
9574050378 shr 24 = 68
Vo shr 16
9574050378 shr 16 = 73
v1 shr 8
9574050378 shr 8 = 32
9574050378 shr = 68

BULLETIN OF INFORMATION TECHNOLOGY (BIT)
Volume 1, No. 3, November 2020, pp 122- 133
ISSN 2722-0524 (media online)

v1 shr 24
-1418919580 shr 24 = 65
V0 shr 16
-1418919580 shr 16 = 82
V0 shr 8
-1418919580 shr 8 = 77
-1418919580 = 65

Desimal (byte)	**68**	**73**	**32**	**68**
Biner	**0100 0100**	**0100 1001**	**0010 0000**	**0100 0100**
karakter	**D**	**I**	**Spasi**	**D**

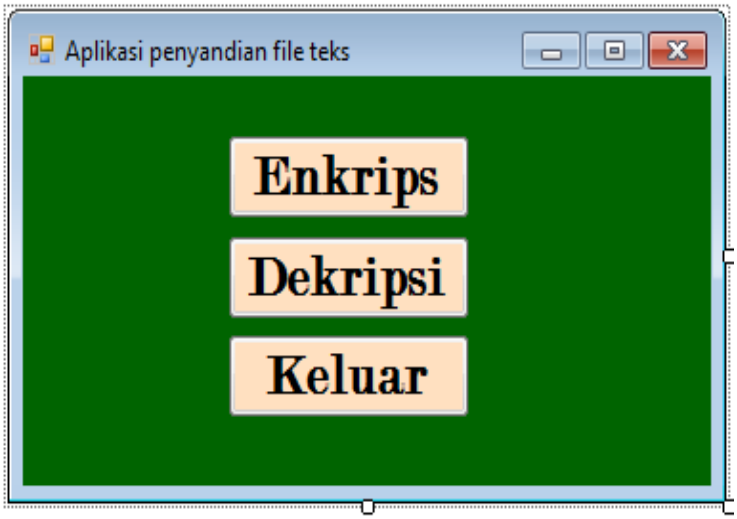
Desimal (byte)	**65**	**82**	**77**	**65**
Biner	**0100 0001**	**0101 0010**	**0100 1101**	**0100 0001**
karakter	**A**	**R**	**M**	**A**

5. Maka *Plaintext* dari *Chipertext :* **vN"<PfWO?8A ÿ**
   Plaintext : "**STMIK BUDI DARMA** ".

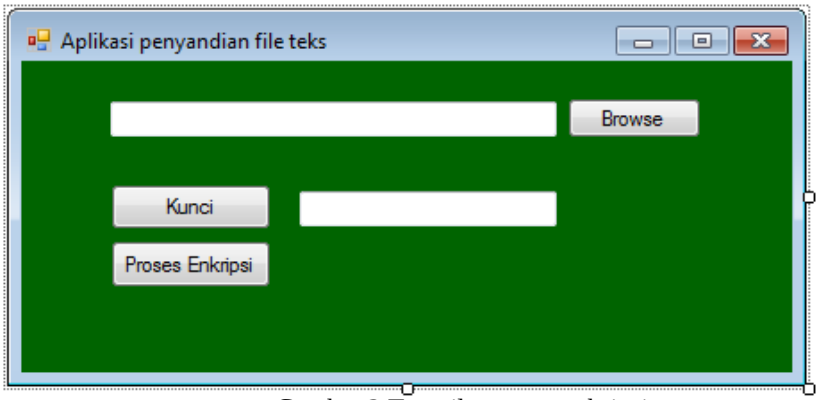
3. IMPLEMENTASI

4.1 Tampilan Menu Utama

Tampilan *menu* utama adalah tampilan yang membentuk atau menggambarkan rancangan halaman depan yang berisi beberapa *field* diantaranya yaitu Tombol Enkripsi. Tombol Dekripsi, dan tombol Keluar. Dan setiap *field* memiliki fungsi yang berbeda-beda.



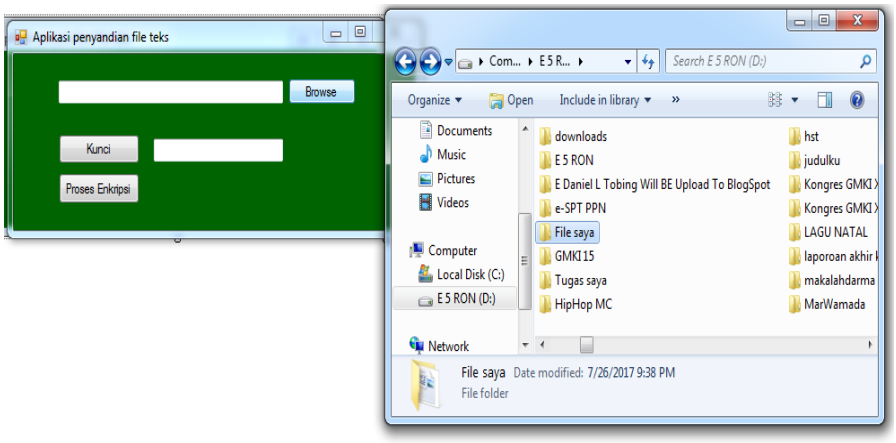
Gambar 1 Tampilan Menu Utama



Gambar 2 Tampilan menu enkripsi

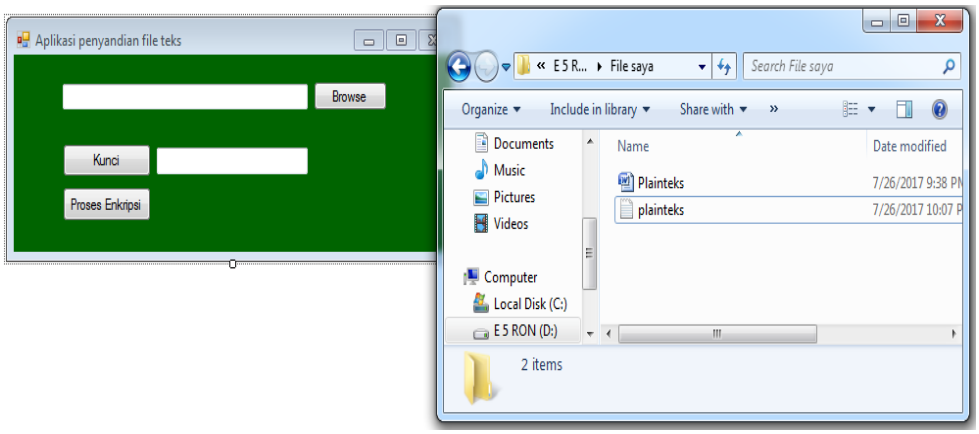
BULLETIN OF INFORMATION TECHNOLOGY (BIT)
Volume 1, No. 3, November 2020, pp 122- 133
ISSN 2722-0524 (media online)

Dengan menekan tombol enkripsi yang ada pada tampilan menu utama di atas maka akan melakukan proses enkripsi. Kemudian akan menampilakn tampilan hasil Enkripsi dari *ciphertek*s yang dipilih.

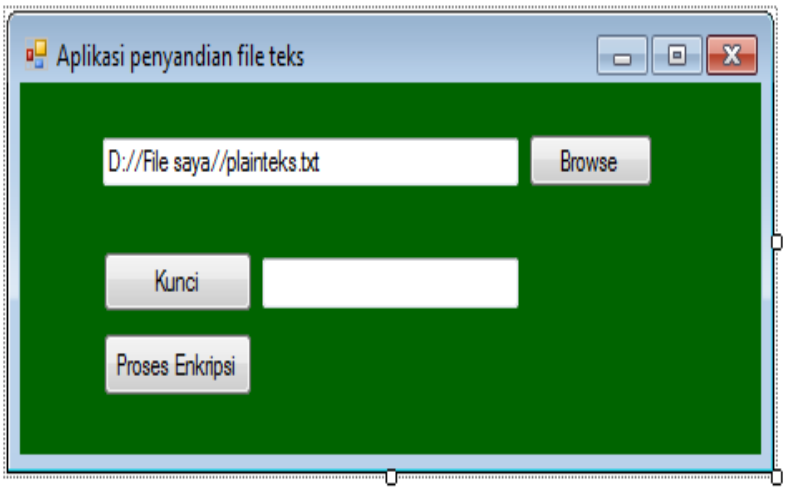


Gambar 3 Pencarian nama penyimpanan file yang akan di enkripsi

Untuk pertama kali adalah menekan tombol *browser* dan akan menampilkan alamat penyimpanan *file* yang akan di enkrip untuk pemilihan *file* teks yang berekstensi txt. Setelah *file* teks dipilih maka pada *teksbox* akan terisi alamat penyimpanan dan nama *file* yang akan di enkrip. Selanjutnya isi *teksbox* pada kunci misalnya jumlah kunci 4 lalu tekan tombol proses enkripsi. Setelah proses enkrip selesai kemuadian buka tempat penyimpanan data *file* teks tersebut kemudian buka *file* tersebut maka *file* yang dipilih akan terenkrip seperti gambar dibawah ini.

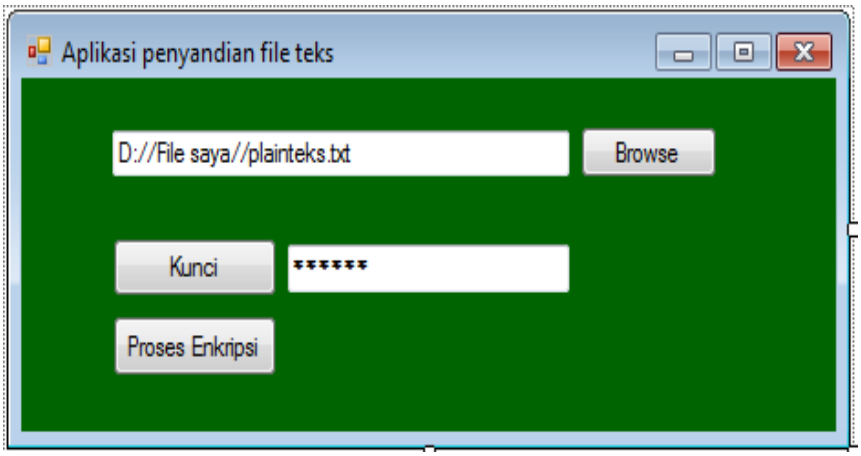


Gambar 4 Tampilanalamat file yang akan di enkripsi

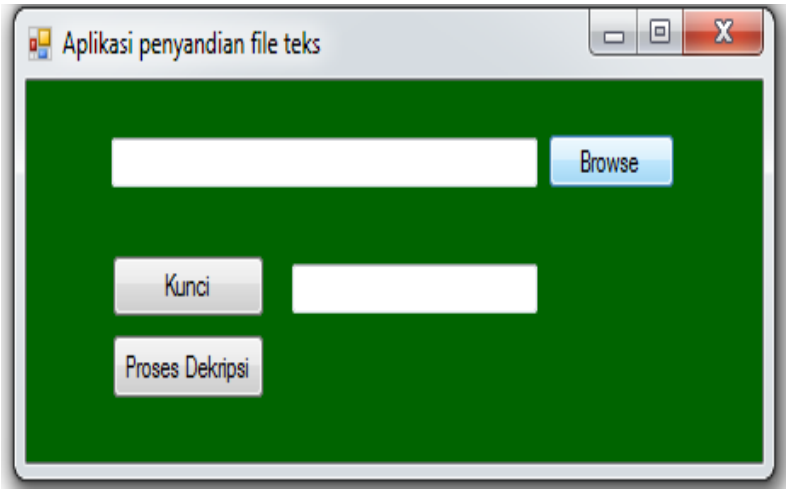


Gambar 5 Tampilan alamat file yang akan di enkripsi

BULLETIN OF INFORMATION TECHNOLOGY (BIT)
Volume 1, No. 3, November 2020, pp 122- 133
ISSN 2722-0524 (media online)

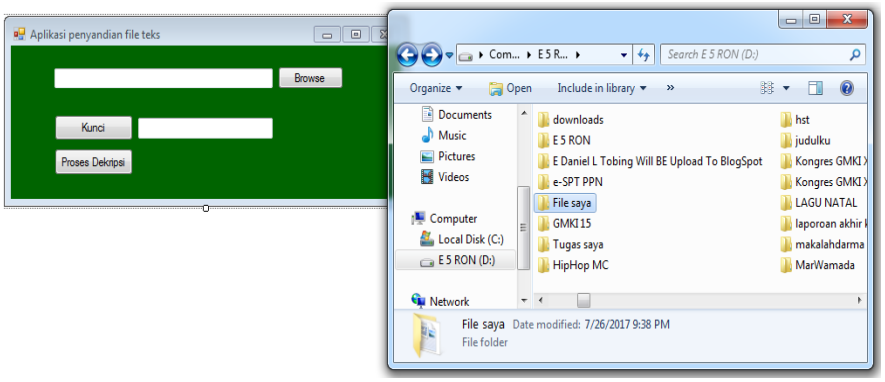


Gambar 6 Memasukkan kata kunci untuk enkripsi



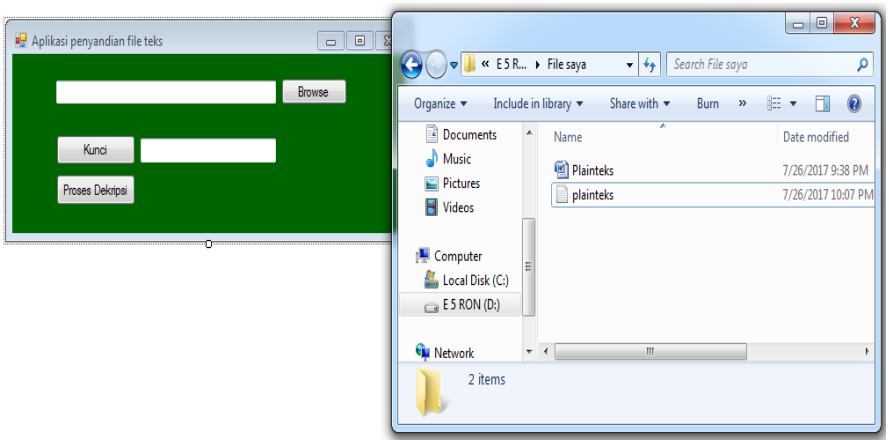
Gambar 7 Memasukkan kata kunci untuk enkripsi

Untuk proses dekripsi atau pengembalian data asli, pertama kali adalah menekan tombol *browser* dan akan menampilkan alamat penyimpanan *file* yang akan di dekrip untuk pemilihan *file* teks yang berekstensi txt. Setelah *file* teks dipilih maka pada *teksbox* akan terisi alamat penyimpanan dan nama *file* yang akan di dekrip. Selanjutnya isi *teksbox* pada kunci sesuai dengan kunci pada saat melakukan proses Enkripsi yaitu 4 lalu tekan tombol proses dekripsi. maka *file* teks yang terenkrip akan kembali menjadi *file* teks asli tanpa mengurangi nilai atau makna dari isi *plainteks* tersebut. Berikut proses pengembalian *plainteks.*

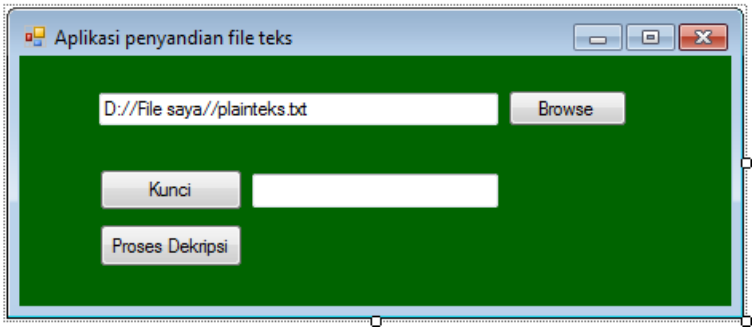


Gambar 8 Pencarian nama file yang akan di kembalikan atau di dekripsi

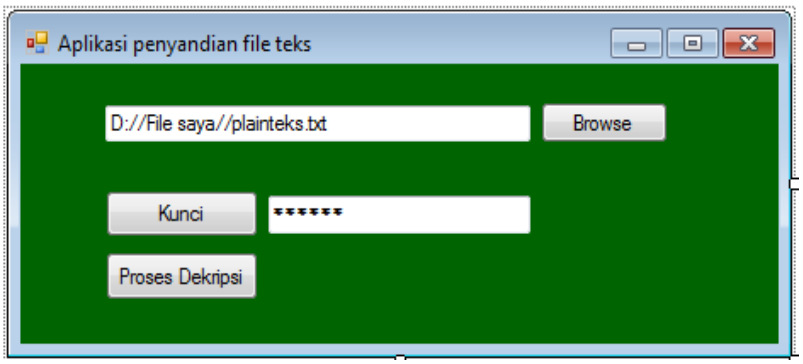
BULLETIN OF INFORMATION TECHNOLOGY (BIT)
Volume 1, No. 3, November 2020, pp 122- 133
ISSN 2722-0524 (media online)



Gambar 9 Tampilan nama file yang akan di dekripsi



Gambar 10 Tampilan alamat file yang akan di enkripsi



Gambar 11 Memasukkan kata kunci dekripsi untuk mengembalikan file yang telah di enkripsi

4. KESIMPULAN

Setelah penelitian dilakukan dan hasil pengujian diperoleh, maka penulis dapat menyimpulkan garis besar dari keseluruhan rangkuman skripsi ini :

1. Aspek kerahasiaan pada Pengamanan Data File Dokumen yang menggunakan Algoritma XTEA (*Extended Tiny Encrytion Algoritmh)* terletak pada pengamanan data yaitu “password”.
2. Aplikasi pengamanan File Dokumen ini dapat melakukan enkripsi dan deskripsi password.
3. Algoritma XTEA pada Kriptografi Modren dapat diimplementasikan pada sebuah siitem informasi.

REFERENCES

[1] Yuni, Marini, Perancangan Perangkat Lunak Pengamanan Data Dengan Metode XTEA, Sistem Informatika STMIK IBBI, 2010
[2] Setyaningsih, Emy, " Kriptografi dan Implementasi Menggunakan MATLAB", Andi Yogyakarta, 2015
[3] Munir, Rinaldi, "Kriptografi", Informatika Bandung, 2006
[4] Kadir, Abdul, " Konsep dan Implementasi Struktur Data dalam Pemrograman Delphi, Andi Yogyakarta, 2011
[5] Munir, Rinaldi, "Algoritma dan Pemrograman Dalam Bahasa Pascal dan C", Informatika Bandun, November, 2011.
[6] William, Kandar, Studi Mengenai Tyni Encryptin Algoritmh (TEA) dan Turunan-turunan (XTEA dan XXTEA, Instituti Teknologi Bandung, 2008
[7] http://www.theasciicode.com.ar/
[8] Dodit, Suprianto, Visual Basic 2008 untuk Berbagai Keperluan Pemrograman, Yogyakarta, 2010