Implementasi Kombinasi Algoritma Enkripsi Aes 128 Dan Algoritma Kompresi Shannon-Fano

Heri Haryanto, Romi Wiryadinata , Muhammad Afif
Jurusan Teknik Elektro, Universitas Sultan Ageng Tirtayasa Cilegon, Indonesia
elektrojos@yahoo.com, romi_wiryadinata@yahoo.com, muhammad.afif.israr@gmail.com

Abstrak *– Data dan informasi menjadi suatu hal yang tidak dapat dipisahkan dari setiap aspek kehidupan manusia. Perkembangan IT (Information Technology) yang semakin cepat juga memberikan tantangan terhadap masalah keamanan data, informasi, dan media penyimpanannya. Kriptografi dapat menjaga agar data atau informasi tetap aman, tanpa mengalami gangguan dari pihak ketiga dan kompresi dapat membuat ukuran sebuah data menjadi lebih kecil. Tujuan penelitian ini adalah untuk membandingkan antara kombinasi algoritma enkripsi-kompresi dengan algoritma kompresi-enkripsi. Penggabungan algoritma enkripsi AES 128 dan algoritma kompresi Shannon-Fano pada file txt, docx, dan pdf dengan kombinasi enkripsi-kompresi tidak cocok untuk digunakan karena membuat ukuran file bertambah dan memiliki waktu proses yang lebih lama apabila dibandingkan proses kompresi-enkripsi. Rata-rata nilai rasio kompresi untuk proses enkripsi-kompresi pada file txt, docx, dan pdf masing-masing sebesar 1.01 sedangkan rata-rata nilai rasio kompresi untuk proses kompresi-enkripsi pada file txt adalah sebesar 0.64 dan untuk file docx dan pdf masing-masing sebesar 1.00. Rata-rata nilai penghematan ruang untuk proses enkripsi-kompresi pada file txt, docx, dan pdf masing-masing adalah sebesar -1.074%, -1.040%, dan -1.025% sedangkan rata-rata penghematan ruang untuk proses kompresi-enkripsi adalah sebesar 35.896%, -0.759%, dan -0.0291%. Rata-rata waktu proses yang didapat untuk kombinasi enkripsi-kompresi pada file txt, docx, dan pdf masing-masing adalah 2.22 s, 2.179 s, dan 2.204 s sedangkan rata-rata waktu proses untuk kombinasi kompresi-enkripsi 2.05 s, 2.114 s, dan 2.122 s.*

Kata Kunci: *Kriptografi, AES 128, Shannon-fano.*

Abstract *- The data and information into something that can not be separated from every aspect of human life. The development of IT (Information Technology) accelerated also provide a challenge to the security issues of data, information, and media storage. Cryptography can keep the data or information secure, without the interference of a third party and compression can make the size of the data becomes smaller. The purpose of this study was to compare the combination of encryption and compression algorithms with compression-encryption algorithm. Surviving AES 128 encryption algorithm and the Shannon-Fano compression algorithms on a file txt, docx, and pdf with a combination of encryption and compression are not suitable for use because it makes the file size increases and has a longer processing time when compared to compression-encryption process. The average value of the compression ratio for encryption-compression on the file txt, docx, pdf and 1:01 respectively, while the average value of the compression ratio for compression-encryption process on the txt file is of 0.64 and docx and pdf file for each 1.00 respectively. The average value of saving space for the encryption-compression on the file txt, docx, and pdf respectively amounted to -1 074% -1040% and -1025% while the average saving space for compression-encryption process is equal to 35 896 % - 0759% and -0.0291%. Average processing time is obtained for the combination of encryption and compression in a txt file, docx, and pdf respectively is 2:22 s, 2179 s and 2204 s, while the average time to process a combination of compression-encryption 2:05 s, 2114 s, and 2122 s.*

Keywords*: Cryptography, AES 128, Shannon-fano.*

I. PENDAHULUAN

Data dan informasi menjadi suatu hal yang tidak dapat dipisahkan dari setiap aspek kehidupan. Data merupakan sebuah bahan baku untuk menghasilkan sebuah informasi, sedangkan informasi sangat dibutuhkan dalam setiap pengambilan keputusan. Perkembangan IT (*Information Technology*) yang semakin cepat juga memberikan tantangan terhadap masalah keamanan data, informasi, dan media penyimpanannya.

Masalah keamanan dan media penyimpanan merupakan aspek-aspek penting dari sebuah sistem informasi. Pada kenyataannya masalah keamanan kurang mendapat perhatian dari perancang dan pengelola sistem informasi. Media penyimpanan HDD (*Hard Disk Drive*) yang semakin besar juga tidak menjawab kebutuhan teknologi informasi jika data berkas (*file*) yang ada semakin besar pula. Pertukaran data dan informasi akan selalu membutuhkan koneksi yang cepat dan stabil, tetapi koneksi yang cepat tidak akan berpengaruh jika data yang dikirim tidak dapat

dijamin kerahasiaannya dan memiliki ukuran lebih besar daripada *bandwidth* koneksinya.

Kriptografi adalah suatu ilmu yang mempelajari bagaimana cara menjaga agar data atau informasi tetap aman, tanpa mengalami gangguan dari pihak ketiga sedangkan kompresi data merupakan cabang ilmu komputer yang bersumber dari teori informasi. Teori informasi memfokuskan pada berbagai metode tentang informasi termasuk penyimpanan dan pemrosesan pesan.

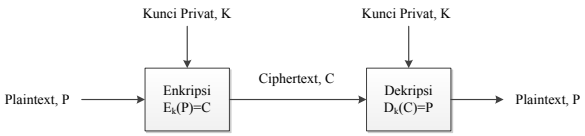
Berdasarkan permasalahan yang ada, maka penelitian ini akan mencoba untuk mengkombinasikan dua algoritma, yaitu algoritma enkripsi AES 128 dan algoritma kompresi Shannon-Fano yang digunakan sebagai metode kriptografi. Diharapkan dengan penggabungan dua algoritma dari hasil penelitian ini pengiriman data dapat dilakukan dengan lebih aman dan efisien karena data hasil proses dapat menghindari gangguan dari pihak yang tidak berhak. Algoritma Shannon-Fano yang digunakan sebagai metode kriptografi ini juga akan menghasilkan ukuran data yang lebih kecil setelah diproses.

II. DASAR TEORI

A. Jenis-Jenis Kriptografi

1. Kriptografi Simetris

Kunci untuk enkripsi dan dekripsi yang sama merupakan konsep dasar dari kriptografi simetris. Istilah lainnya adalah *private-key cryptography*, *secret-key cryptography*, atau *conventional cryptography*. Dalam kriptografi kunci simetris, dapat diasumsikan bahwa penerima dan pengirim pesan telah terlebih dahulu berbagi kunci sebelum pesan dikirim. Keamanan dari sistem ini terletak pada kerahasiaan kuncinya.

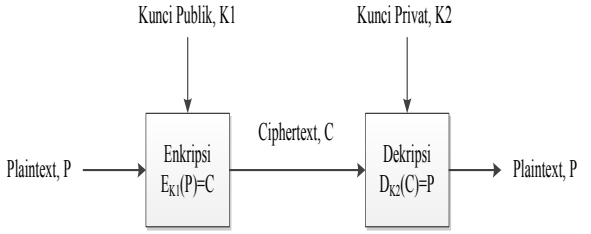


Gambar 1. Skema Proses Kriptografi Simetris

Plaintext P dienkripsi dengan kunci privat K menghasilkan fungsi $E_k(P) = C$ dimana C merupakan *ciphertext* hasil enkripsi *plaintext* P. Proses dekripsi *ciphertext* C memerlukan kembali kunci privat K untuk mendapatkan kembali *plaintext* dengan fungsi $D_k(C) = P$.

2. Kriptografi Asimetris

Kriptografi kunci asimetris atau kunci publik memiliki dua buah kunci yang berbeda pada proses enkripsi dan dekripsinya. Nama lainnya adalah kriptografi kunci publik (*public-key cryptography*). Kunci untuk enkripsi pada kriptografi asimetris ini tidak rahasia (diketahui oleh publik), sedangkan kunci untuk dekripsinya bersifat rahasia (kunci privat). Entitas pengirim akan mengenkripsi dengan menggunakan kunci publik, sedangkan entitas penerima mendekripsi pesan dengan menggunakan kunci privat.



Gambar 2.Skema Proses Kriptografi Asimetris

Plaintext P dienkripsi dengan kunci publik K1 menghasilkan fungsi $E_{K1}(P) = C$ dimana C merupakan *ciphertext* hasil enkripsi *plaintext* P. Proses dekripsi *ciphertext* C memerlukan kunci privat K2 tidak seperti proses kriptografi simetris. Kunci privat K2 menjadi kunci untuk mendekripsi *ciphertext* C dengan fungsi $D_{K2}(C) = P$ sehingga kembali menghasilkan *plaintext* P.

B. Stream Cipher Dan Block Cipher

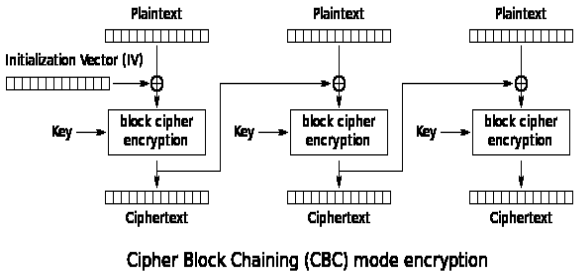
Tipe pengerjaan algoritma kriptogafi simetris terbagi menjadi dua yaitu *stream cipher* dan *block cipher*. *Stream cipher* merupakan suatu pengerjaan algoritma kriptografi simetris yang beroperasi dalam suatu pesan berupa *bit* tunggal atau terkadang dalam suatu *byte*. *Block cipher* beroperasi dengan rangkaian *bit-bit plaintext* yang dibagi menjadi blok-blok *bit* dengan besar yang sama. Algoritma enkripsi menghasilkan blok *ciphertext* yang berukuran sama dengan blok *plaintext*-nya. Blok *plaintext* akan menghasilkan blok *ciphertext* yang sama apabila dienkripsi menggunakan kunci yang sama. Hal ini berbeda dengan stream cipher dimana *bit-bit plaintext* akan menghasilkan *bit-bit ciphertext* yang berbeda setiap kali dienkripsi.

C. Cipher-block Chaining (CBC)

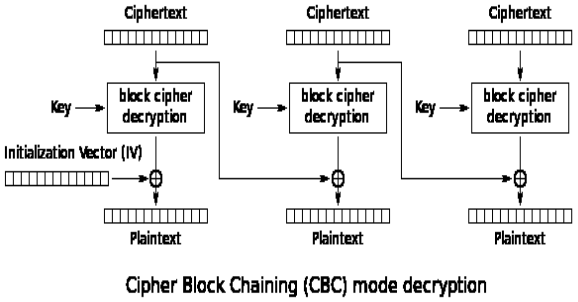
Mode CBC menerapkan mekanisme umpan balik (feedback) pada prosesnya. Hasil enkripsi blok C_{i-1} diumpanbalikan ke dalam blok *plaintext* P_i yang akan digunakan untuk proses enkripsi. Berikut adalah persamaannya:

$$C_i = E_K(P_i \oplus C_{i-1}) \quad (1)$$
$$P_i = D_K(C_i \oplus C_{i-1}) \quad (2)$$

Persamaan (2.1) digunakan untuk enkripsi dan persamaan (2.2) digunakan untuk dekripsi. Enkripsi pada blok pertama C_0 menggunakan IV (Initialization Vector). IV diberikan oleh *user* atau dibangkitkan secara acak oleh program dan tidak bersifat rahasia. Proses enkripsi pada *plaintext* pertama menggunakan IV sebagai C_0. Gambar 2.4 adalah skema proses enkripsi mode CBC dan Gambar 2.5 adalah skema proses dekripsi mode CBC.



Gambar 3. Skema Proses Enkripsi Mode CBC



Gambar 4. Skema Proses Dekripsi Mode CBC

D. Algoritma Rijndael

Algoritma Rijndael atau yang biasa disebut AES (*Advanced Encryption Standard*) merupakan algoritma kriptografi yang beroperasi dalam bentuk blok *ciphertext* simetris untuk mengenkripsi (*encipher*) dan dekripsi (*decipher*) informasi. Algoritma AES menggunakan kunci kriptografi 128, 192, dan 256 *bit* untuk mengenkrip dan mendekrip data pada blok 128 *bit*. Blok *plaintext* sebesar 128 *bit* dimasukkan ke dalam *state* yang berbentuk bujur sangkar berukuran 4×4 *byte*. *State* ini di-XOR dengan *key* dan selanjutnya diolah 10 kali dengan subtitusi-transformasi *linear-addkey* yang di akhir akan diperoleh *ciphertext*. Secara garis besar, AES memiliki empat proses utama yaitu:

1. *AddRoundKey*
2. *SubBytes*
3. *ShifRows*
4. *MixColumns*

E. Kompresi

Kompresi data merupakan cabang ilmu komputer yang bersumber dari teori informasi. Teori informasi sendiri adalah salah satu cabang matematika yang berkembang sekitar dekade 1940-an. Tokoh utama dari teori informasi adalah Claude Shannon dari Bell Laboratory. Teori informasi memfokuskan pada berbagai metode tentang informasi termasuk penyimpanan dan pemrosesan pesan. Teori informasi mempelajari pula tentang *redundancy* (informasi tidak berguna) pada pesan. Semakin banyak *redundancy* semakin besar pula ukuran pesan dan upaya mengurangi *redundancy* inilah yang akhirnya melahirkan subjek ilmu tentang kompresi data. Teknik kompresi data dibagi menjadi dua kategori besar yaitu *lossy compression* dan *lossless compression*.

F. Algoritma Shannon-Fano

Teknik *coding* ini dikembangkan oleh dua orang dalam dua buah proses yang berbeda, yaitu Claude Shannon di Bell Laboratory dan R. M. Fano di MIT, namun karena memiliki kemiripan akhirnya teknik ini dinamai dengan manggabungkan nama keduanya. Pada dasarnya proses *coding* dengan algoritma ini membutuhkan data akan frekuensi jumlah kemunculan suatu karakter pada sebuah pesan.

Pada dasarnya cara kerja dari algoritma Shannon-Fano ini sama persis dengan algoritma Huffman. Algoritma ini membentuk sebuah pohon, kemudian meng-*encoding*-nya, dan yang terakhir adalah mengembalikannya dalam bentuk karakter teks atau *decoding*. Hanya saja perbedaan yang fundamental terdapat pada pembuatan pohon. Pembuatan pohon pada Shannon-Fano dibuat berdasarkan proses dari atas ke bawah berbeda dengan Huffman yang membuat pohon dari bawah ke atas. Panjang *codeword* Shannon-Fano dapat dihitung dengan persamaan berikut:

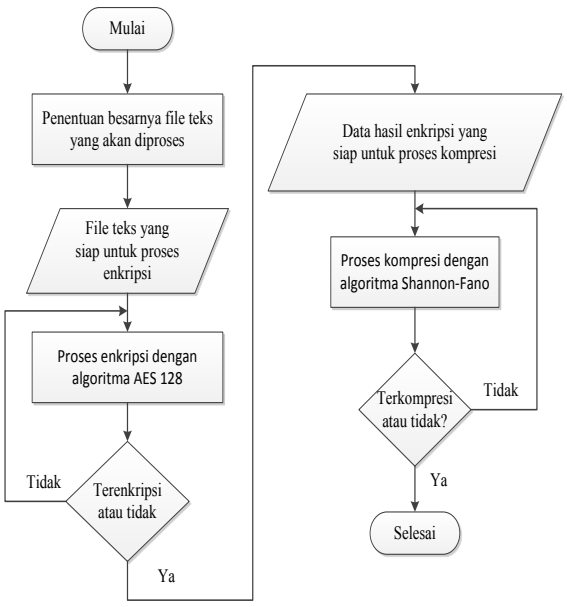
$$\log_2\left(\frac{1}{p_i}\right) \leq I_i \log_2\left(\frac{1}{p_i}\right) + 1 \qquad (5)$$

dimana p_i merupakan probabiltas kemunculan simbol dan I_i merupakan panjang *codeword*. Hal ini juga penting untuk dicatat bahwa kode Shannon-Fano memenuhi kondisi kode awalan yang berarti bahwa tidak ada *codeword* yang sama untuk *codeword* setelahnya.

III. METODOLOGI PENELITIAN

A. Proses Enkripsi-Kompresi

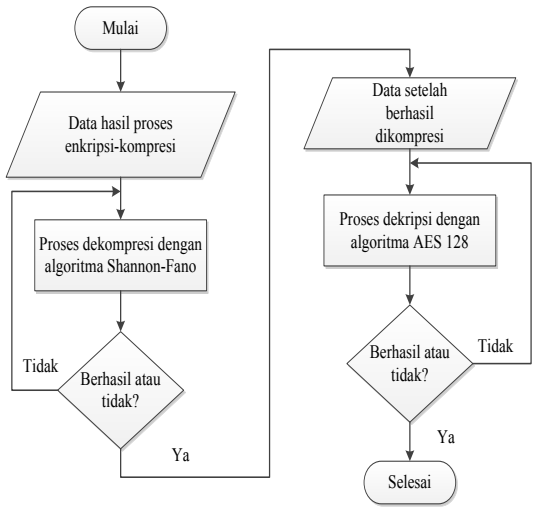
Tahap proses enkripsi-kompresi merancang program penggabungan algoritma enkripsi AES 128 dengan algoritma kompresi Shannon-Fano. Diagram alir untuk proses enkripsi-kompresi data diperlihatkan pada Gambar 3.2.



Gambar 5. Diagam Alir Proses Enkripsi-Kompresi Data

B. Proses Dekompresi-Dekripsi

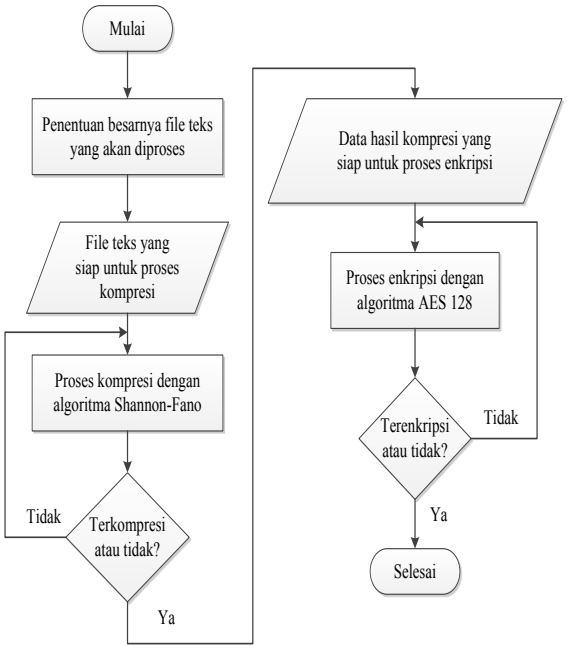
Tahap proses dekompresi-dekripsi merancang program untuk merubah data hasil proses enkripsi-kompresi menjadi bentuk data awal. Diagram alir untuk proses dekompresi-dekripsi dapat dilihat pada Gambar 6.



Gambar 6. Diagram Alir Proses Dekompresi-Dekripsi

C. Proses Kompresi-Enkripsi

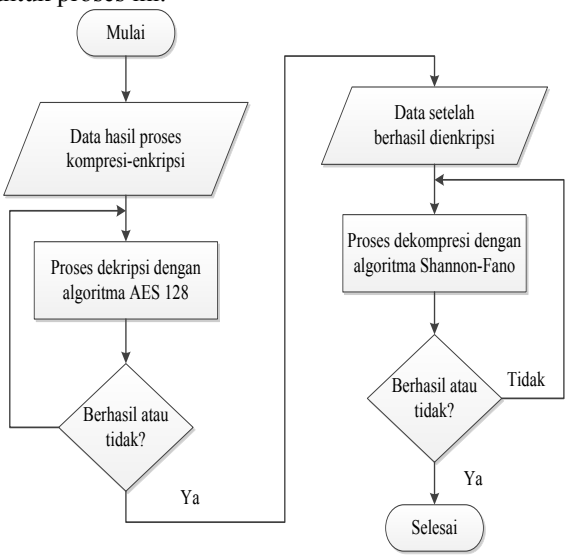
Tahap proses kompresi-enkripsi merancang program penggabungan algoritma kompresi algoritma Shannon-Fano dengan algortima enkripsi algoritma AES 128. Gambar 7 merupakan digram alir untuk proses kompresi-enkripsi data.



Gambar 7. Diagram Alir Proses Kompresi-Enkripsi Data

D. Perancangan Proses Dekripsi-Dekompresi

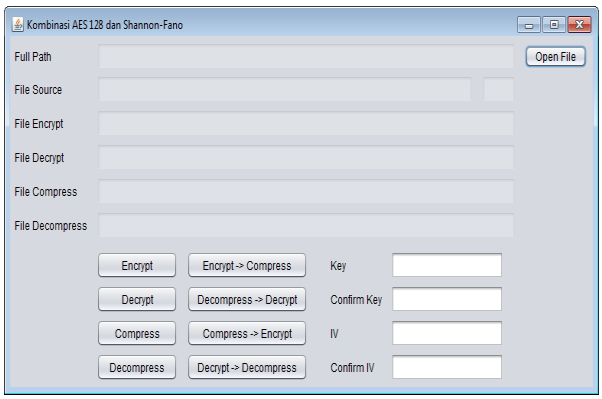
Tahap penelitian ini merancang program untuk merubah data hasil proses kompresi-enkripsi menjadi bentuk data awal. Gambar 8. merupakan diagram alir untuk proses ini.



Gambar 8. Diagram Alir Proses Dekripsi-Dekompresi

IV. HASIL DAN PEMBAHASAN

Pengujian program dilakukan dalam beberapa tahap seperti pengujian program enkripsi-kompresi, pengujian program dekompresi-dekripsi, pengujian program kompresi-enkripsi, dan pengujian progam dekripsi-dekompresi. Pengujian juga dilakukan membandingkan kecepatan proses enkripsi-kompresi dengan proses kompresi-enkripsi. Gambar 4.1 merupakan tampilan antarmuka program yang dibuat pada penelitian ini.



Gambar 9. Tampilan Antarmuka Program

A. Pengujian Enkripsi-Kompresi

File plaintext pertama dienkripsi dengan algoritma AES 128 kemudian *ciphertext* hasil enkripsi menjadi *input* untuk dikompresi dengan algortima Shannon-Fano. Format *file* yang dipakai dalam pengujian ini adalah txt, docx, dan pdf yang masing-masing berjumlah sepuluh *file* dengan besar yang berbeda. Hasil enkripsi akan menghasilkan *file* berformat *encrypt* dan hasil kompresi akan menghasilkan *file* berformat *compress*. Tabel 1, Tabel 2, dan Tabel 3 menunjukan hasil pengujian penggabungan algoritma enkripsi-kompresi pada *file* txt, docx, dan pdf.

Tabel 1. Hasil Pengujian Enkripsi-Kompresi Pada *File* Txt

Nomor *File*	Ukuran *File* (KB)		
	File Awal	Setelah Enkripsi	Setelah Kompresi
1	112	112	116
2	225	225	229
3	337	337	341
4	421	421	425
5	535	535	539
6	626	626	630
7	727	727	731
8	823	823	827
9	919	919	923
10	1011	1011	1015

Tabel 2. Hasil Pengujian Enkripsi-Kompresi Pada *File* Docx

Nomor *File*	Ukuran *File* (KB)		
	File Awal	Setelah Enkripsi	Setelah Kompresi
1	115	115	119
2	225	225	228
3	312	312	316
4	406	406	410
5	504	504	508
6	605	605	609
7	701	701	705
8	840	840	844
9	921	921	925
10	1025	1025	1029

Tabel 3. Hasil Pengujian Enkripsi-Kompresi Pada *File* Pdf

Nomor *File*	Ukuran *File* (KB)		
	File Awal	Setelah Enkripsi	Setelah Kompresi
1	116	116	120
2	201	201	205
3	321	321	324
4	417	417	420
5	528	528	532
6	622	622	626
7	713	713	717
8	823	823	827
9	910	910	913
10	1043	1043	1047

B. Pengujian Dekompresi-Dekripsi

File output proses enkripsi-kompresi menjadi *input* proses ini. *File* berformat *compress* akan didekompresi dan diubah menjadi format *file* awal karena diprogram sesuai dengan *output* proses dekompresi. Hasil dekompresi akan didekripsi sehingga *file* dapat diakses kembali. Tabel 4, Tabel 5, dan Tabel 6 menunjukan hasil pengujian dekompresi-dekripsi pada *file* txt, docx, dan pdf.

Tabel 4 Hasil Pengujian Dekompresi-Dekripsi Pada *File* Txt

Nomor *File*	Ukuran *File* (KB)		
	File Awal	Setelah Dekompresi	Setelah Dekripsi
1	112	112	112
2	225	225	225
3	337	337	337
4	421	421	421
5	535	535	535
6	626	626	626
7	727	727	727
8	823	823	823
9	919	919	919
10	1011	1011	1011

Tabel 5 Hasil Pengujian Dekompresi-Dekripsi Pada *File* Docx

Nomor *File*	Ukuran *File* (KB)		
	File Awal	Setelah Dekompresi	Setelah Dekripsi
1	115	115	115
2	225	225	225
3	312	312	312
4	406	406	406
5	504	504	504
6	605	605	605
7	701	701	701
8	840	840	840
9	921	921	921
10	1025	1025	1025

Tabel 6. Hasil Pengujian Dekompresi-Dekripsi Pada *File* Pdf

Nomor *File*	Ukuran *File* (KB)		
	File Awal	Setelah Dekompresi	Setelah Dekripsi
1	116	116	116
2	201	201	201
3	321	321	321
4	417	417	417
5	528	528	528
6	622	622	622
7	713	713	713
8	823	823	823
9	910	910	910
10	1043	1043	1043

C. Pengujian Kompresi-Enkripsi

File pertama dikompres dengan algoritma Shannon-Fano dan hasil kompresi akan menjadi *input* untuk proses enkripsi AES 128. Format *file* yang

dipakai dalam pengujian ini adalah txt, docx, dan pdf yang masing-masing berjumlah sepuluh *file* dengan besar yang berbeda. Hasil kompresi akan menghasilkan *file* berformat *compress* dan hasil enkripsi akan menghasilkan *file* berformat *encrypt*. Tabel 7, Tabel 8, dan Tabel 9 menunjukan hasil pengujian penggabungan algoritma kompresi-enkripsi pada *file* txt, docx, dan pdf.

Tabel 7. Hasil Pengujian Kompresi-Enkripsi Pada *File* Txt

Nomor *File*	Ukuran *File* (KB)		
	File Awal	Setelah Kompresi	Setelah Enkripsi
1	112	74	74
2	225	145	145
3	337	216	216
4	421	269	269
5	535	342	342
6	626	400	400
7	727	462	462
8	823	525	525
9	919	586	586
10	1011	643	643

8. Hasil Pengujian Kompresi-Enkripsi Pada *File* Docx

Nomor *File*	Ukuran *File* (KB)		
	File Awal	Setelah Kompresi	Setelah Enkripsi
1	115	118	118
2	225	228	228
3	312	314	314
4	406	409	409
5	504	501	501
6	605	608	608
7	701	707	707
8	840	845	845
9	921	925	925
10	1025	1030	1030

Tabel 9. Hasil Pengujian Kompresi-Enkripsi Pada *File* Pdf

Nomor *File*	Ukuran *File* (KB)		
	File Awal	Setelah Kompresi	Setelah Enkripsi
1	116	120	120
2	201	204	204
3	321	324	324
4	417	420	420
5	528	533	533
6	622	627	627
7	713	718	718
8	823	825	825
9	910	911	911
10	1043	948	948

D. Pengujian Dekripsi-Dekompresi

File output proses kompresi-enkripsi menjadi *input* ini. *File* berformat *encrypt* akan didekripsi dan diubah menjadi format *file* awal karena diprogram sesuai dengan *output* proses dekripsi. Hasil dekripsi akan didekompresi sehingga *file* dapat diakses kembali. Tabel 10, Tabel 11, dan Tabel 12 menunjukan hasil pengujian dekripsi-dekompresi pada *file* txt, docx, dan pdf.

Tabel 10. Hasil Pengujian Dekripsi-Dekompresi Pada *File* Txt

Nomor *File*	Ukuran *File* (KB)		
	File Awal	Setelah Dekripsi	Setelah Dekompresi
1	112	74	112
2	225	145	225
3	337	216	337
4	421	269	421
5	535	342	535
6	626	400	626
7	727	462	727
8	823	525	823
9	919	586	919
10	1011	643	1011

Tabel 11. Hasil Pengujian Dekripsi-Dekompresi Pada *File* Docx

Nomor *File*	Ukuran *File* (KB)		
	File Awal	Setelah Dekripsi	Setelah Dekompresi
1	115	118	115
2	225	228	225
3	312	314	312
4	406	409	406
5	504	501	504
6	605	608	605
7	701	707	701
8	840	845	840
9	921	925	921
10	1025	1030	1025

Tabel 12. Hasil Pengujian Dekripsi-Dekompresi Pada *File* Pdf

Nomor *File*	Ukuran *File* (KB)		
	File Awal	Setelah Dekripsi	Setelah Dekompresi
1	116	120	116
2	201	204	201
3	321	324	321
4	417	420	417
5	528	533	528
6	622	627	622
7	713	718	713
8	823	825	823
9	910	911	910
10	1043	948	1043

E. Pengujian Rasio Kompresi Dan Penghematan Ruang

Pengujian ini dilakukan untuk mengetahui seberapa besar rasio kompresi dan penghematan ruang yang dihasilkan dari *file* hasil proses enkripsi-kompresi dan kompresi-enkripsi kemudian dibandingkan hasilnya. Hal ini dilakukan untuk mengetahui kombinasi algortima mana yang lebih efisien untuk digunakan. Tabel 13 dan Tabel 14 menunjukan perbandingan rasio kompresi dan space savings pada file txt.

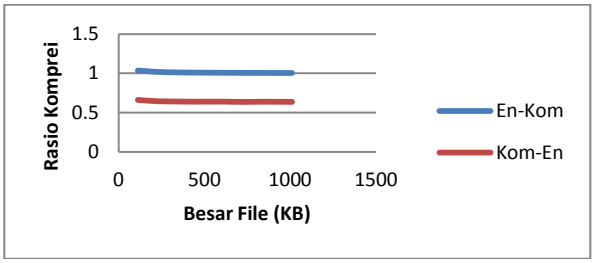
Tabel 13. Perbandingan Rasio Kompresi Kombinasi Algoritma Pada *File* Txt

Nomor File	Rasio Kompresi	
	Enkripsi-Kompresi	Kompresi-Enkripsi
1	1,035714286	0,660714286
2	1,017777778	0,644444444
3	1,011869436	0,640949555
4	1,009501188	0,638954869
5	1,007476636	0,639252336
6	1,006389776	0,638977636
7	1,005502063	0,635488308
8	1,004860267	0,637910085
9	1,004352557	0,637649619
10	1,003956479	0,636003956

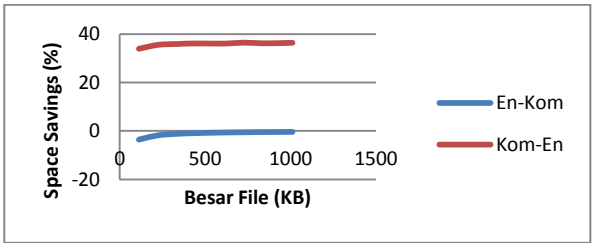
Tabel 14. Perbandingan Space Savings Kombinasi Algoritma Pada *File* Txt

Nomor File	Space Savings (%)	
	Enkripsi-Kompresi	Kompresi-Enkripsi
1	-3,571428571	33,92857143
2	-1,777777778	35,55555556
3	-1,18694362	35,90504451
4	-0,950118765	36,10451306
5	-0,747663551	36,07476636
6	-0,638977636	36,10223642
7	-0,550206327	36,45116919
8	-0,486026731	36,20899149
9	-0,435255713	36,23503808
10	-0,395647873	36,39960435

Dapat dilihat pada tabel 13 nilai rasio kompresi pada *file* txt untuk gabungan algoritma dengan kombinasi enkripsi-kompesi mempunyai rata-rata 1.01 sedangkan untuk kombinasi kompresi-enkripsi mempunyai rata-rata 0.64 dan pada Tabel 4.14 persentase penghematan ruang untuk kombinasi enkripsi-kompesi mempunyai rata-rata -1.07% sedangkan untuk kombinasi kompresi-enkripsi mempunyai rata-rata 35.90%. Nilai space savings untuk proses enkripsi-kompresi bernilai negatif dikarenakan *file* yang diproses menjadi bertambah ukuran *file*nya.



Gambar 10. Grafik Perbandingan Rasio Kompresi *File* Txt



Gambar 11. Grafik Perbandingan Space Savings *File* Txt

Tabel 15 dan Tabel 16 menunjukan perbandingan rasio kompresi dan space savings pada file docx.

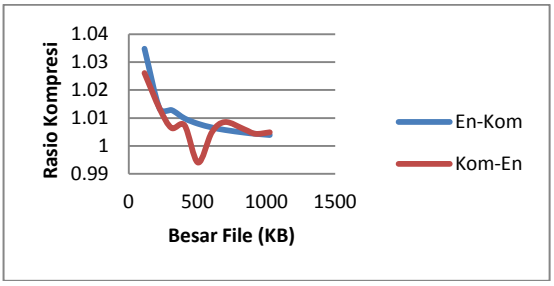
Tabel 15. Perbandingan Rasio Kompresi Kombinasi Algoritma Pada *File* Docx

Nomor File	Rasio Kompresi	
	Enkripsi-Kompresi	Kompresi-Enkripsi
1	1,034782609	1,026086957
2	1,013333333	1,013333333
3	1,012820513	1,006410256
4	1,009852217	1,007389163
5	1,007936508	0,994047619
6	1,00661157	1,004958678
7	1,005706134	1,008559201
8	1,004761905	1,005952381
9	1,004343105	1,004343105
10	1,003902439	1,004878049

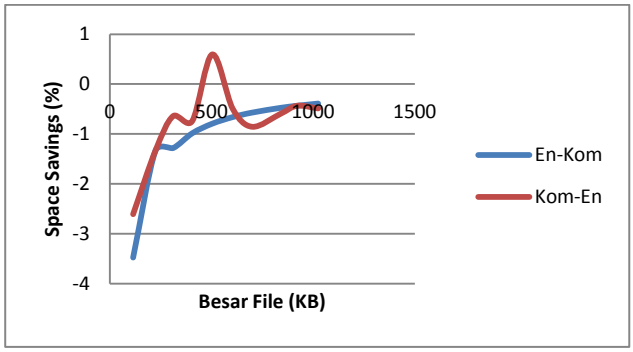
Tabel 16. Perbandingan Space Savings Kombinasi Algoritma Pada *File* Docx

Nomor File	Space Savings (%)	
	Enkripsi-Kompresi	Kompresi-Enkripsi
1	-3,47826087	-2,608695652
2	-1,333333333	-1,333333333
3	-1,282051282	-0,641025641
4	-0,985221675	-0,738916256
5	-0,793650794	0,595238095
6	-0,661157025	-0,495867769
7	-0,570613409	-0,855920114
8	-0,476190476	-0,595238095
9	-0,434310532	-0,434310532
10	-0,390243902	-0,487804878

Dapat dilihat pada tabel 15 nilai rasio kompresi pada *file* txt untuk gabungan algoritma dengan kombinasi enkripsi-kompesi mempunyai rata-rata 1.01 sedangkan untuk kombinasi kompresi-enkripsi mempunyai rata-rata 1.00 dan pada Tabel 16 persentase penghematan ruang untuk kombinasi enkripsi-kompesi mempunyai rata-rata -1.04% sedangkan untuk kombinasi kompresi-enkripsi mempunyai rata-rata -0.75%. Nilai space savings bernilai negatif dikarenakan *file* yang diproses menjadi bertambah ukuran *file*nya.



Gambar 12. Grafik Perbandingan Rasio Kompresi *File* Docx



Gambar 13. Grafik Perbandingan Space Savings *File* Docx

Tabel 17 dan Tabel 18 menunjukan perbandingan rasio kompresi dan space savings pada file pdf.

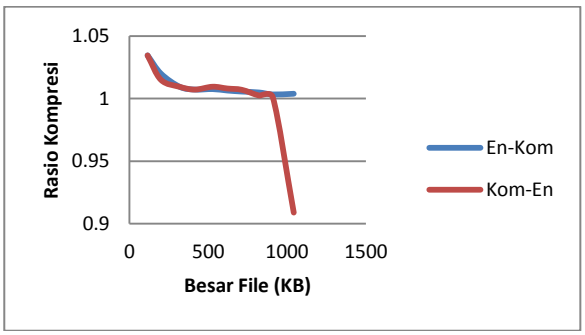
Tabel 17. Perbandingan Rasio Kompresi Kombinasi Algoritma Pada *File* Pdf

Nomor File	Rasio Kompresi	
	Enkripsi-Kompresi	Kompresi-Enkripsi
1	1,034482759	1,034482759
2	1,019900498	1,014925373
3	1,009345794	1,009345794
4	1,007194245	1,007194245
5	1,007575758	1,009469697
6	1,006430868	1,008038585
7	1,005610098	1,007012623
8	1,004860267	1,002430134
9	1,003296703	1,001098901
10	1,003835091	0,908916587

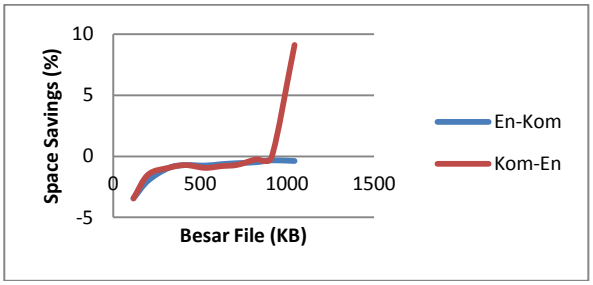
Tabel 18. Perbandingan Space Savings Kombinasi Algoritma Pada *File* Pdf

Nomor File	Space Savings (%)	
	Enkripsi-Kompresi	Kompresi-Enkripsi
1	-3,448275862	-3,448275862
2	-1,990049751	-1,492537313
3	-0,934579439	-0,934579439
4	-0,71942446	-0,71942446
5	-0,757575758	-0,946969697
6	-0,643086817	-0,803858521
7	-0,561009818	-0,701262272
8	-0,486026731	-0,243013366
9	-0,32967033	-0,10989011
10	-0,383509108	9,108341323

Dapat dilihat pada tabel 17 nilai rasio kompresi pada *file* txt untuk gabungan algoritma dengan kombinasi enkripsi-kompesi mempunyai rata-rata 1.01 sedangkan untuk kombinasi kompresi-enkripsi mempunyai rata-rata 1.00 dan pada Tabel 4.18 persentase penghematan ruang untuk kombinasi enkripsi-kompesi mempunyai rata-rata -1.02% sedangkan untuk kombinasi kompresi-enkripsi mempunyai rata-rata -0.02%. Nilai space savings bernilai negatif dikarenakan *file* yang diproses menjadi bertambah ukuran *file*nya.



Gambar 14 Grafik Perbandingan Rasio Kompresi *File* Pdf



Gambar 15. Grafik Perbandingan Space Savings *File* Pdf

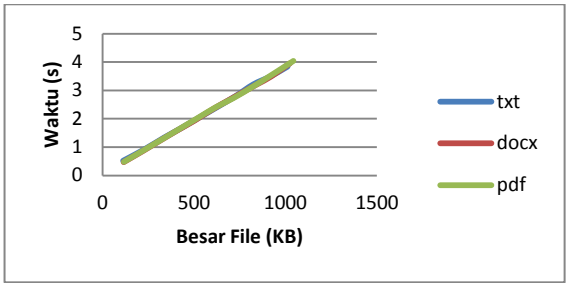
F. Pengujian Waktu Proses

Pengujian ini dilakukan untuk mengetahui berapa lama waktu yang dibutuhkan untuk melakukan sebuah proses untuk setiap tipe *file* yang dipakai dalam penelitian ini. Tabel 19. menunjukan waktu yang

dibutuhkan untuk proses penggabungan algoritma enkripsi-kompresi pada setiap tipe *file*.

Tabel 19. Waktu Proses Enkripsi-Kompresi

Nomor *File*	Waktu Proses (s)		
	Txt	Docx	Pdf
1	0.52	0.47	0.48
2	0.91	0.88	0.8
3	1.34	1.22	1.26
4	1.64	1.58	1.63
5	2.07	1.94	2.06
6	2.41	2.35	2.42
7	2.8	2.7	2.72
8	3.21	3.2	3.14
9	3.5	3.49	3.49
10	3.84	3.96	4.04



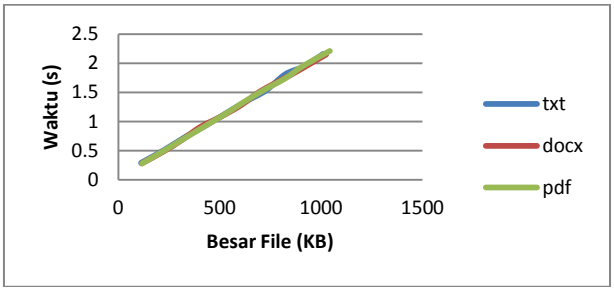
Gambar 16. Grafik Perbandingan Waktu Untuk Proses Enkripsi-Kompresi

Dapat dilihat pada Gambar 16 waktu proses enkripsi-kompresi untuk *file* txt, docx, dan pdf masing-masing mempunyai rata-rata 2.22 s, 2.179 s, 2.204 s.

Tabel 20 menunjukan waktu yang dibutuhkan untuk proses penggabungan algoritma dekompresi-dekripsi pada setiap tipe *file*.

Tabel 20. Waktu Proses Dekompresi-Dekripsi

Nomor *File*	Waktu Proses (s)		
	Txt	Docx	Pdf
1	0.29	0.28	0.28
2	0.51	0.49	0.45
3	0.75	0.68	0.7
4	0.91	0.91	0.9
5	1.15	1.08	1.13
6	1.34	1.28	1.34
7	1.53	1.52	1.53
8	1.81	1.78	1.74
9	1.95	1.94	1.95
10	2.16	2.15	2.21



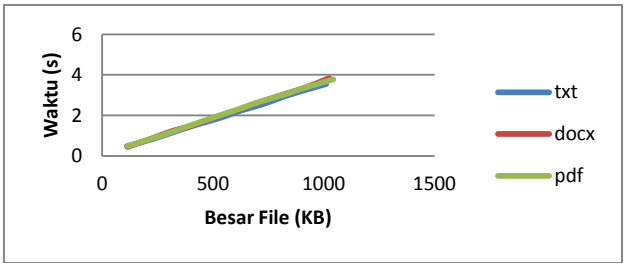
Gambar 17. Grafik Perbandingan Waktu Untuk Proses Dekompresi-Dekripsi

Gambar 17 menunjukan grafik perbandingan waktu untuk proses dekompresi-dekripsi dari Tabel. Dapat dilihat pada Gambar waktu proses dekompresi-dekripsi untuk *file* txt, docx, dan pdf masing-masing mempunyai rata-rata 1.24 s, 1.211 s, 1.223 s. Dilihat dari rata-rata waktu prosesnya, proses dekompresi-dekripsi memiliki waktu proses yang lebih cepat daripada waktu proses enkripsi-kompresi. Hal tersebut dikarenakan proses dekompresi hanya mengurut kembali nilai biner dari *ciphertext* untuk kemudian disamakan dengan *codeword* yang sesuai pada *codemap*. Proses dekripsi juga tidak memakan waktu proses yang banyak

Tabel 21. menunjukan waktu yang dibutuhkan untuk proses penggabungan algoritma kompresi-enkripsi pada setiap tipe *file*.

Tabel 21. Waktu Proses Kompresi-Enkripsi

Nomor *File*	Waktu Proses (s)		
	Txt	Docx	Pdf
1	0.49	0.45	0.49
2	0.83	0.87	0.78
3	1.23	1.22	1.21
4	1.53	1.5	1.58
5	1.89	1.89	1.99
6	2.24	2.25	2.33
7	2.56	2.62	2.68
8	2.94	3.11	3.03
9	3.26	3.39	3.35
10	3.55	3.84	3.78



Gambar 18. Grafik Perbandingan Waktu Untuk Proses Kompresi-Enkripsi

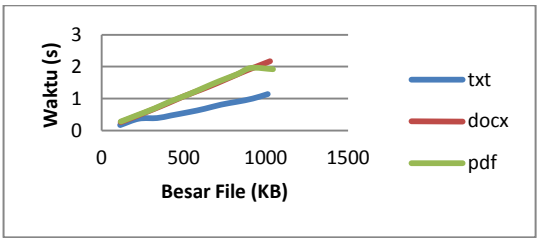
Dapat dilihat pada Gambar 18 waktu proses kompresi-enkripsi untuk *file* txt, docx, dan pdf masing-masing mempunyai rata-rata 2.05 s, 2.114 s, 2.122 s.

Waktu proses kompresi-enkripsi lebih cepat apabila dibandingkan dengan waktu proses enkripsi-kompresi. Hal tersebut disebabkan karena proses kompresi diawal proses membuat ukuran *file* menjadi lebih kecil dan membuat *byte* hasil kompresi menjadi sederhana.

Tabel 22 menunjukan waktu yang dibutuhkan untuk proses penggabungan algoritma dekripsi-dekompresi pada setiap tipe *file*.

Tabel 22. Waktu Proses Dekripsi-Dekompresi

Nomor *File*	Waktu Proses (s)		
	Txt	Docx	Pdf
1	0.17	0.26	0.28
2	0.37	0.48	0.45
3	0.39	0.66	0.69
4	0.47	0.86	0.91
5	0.58	1.08	1.13
6	0.68	1.28	1.34
7	0.81	1.48	1.54
8	0.9	1.79	1.76
9	1	1.96	1.96
10	1.14	2.17	1.92



Gambar 19. Grafik Perbandingan Waktu Untuk Proses Dekripsi-Dekompresi

Dapat dilihat pada Gambar 19 waktu proses dekripsi-dekompresi untuk *file* txt, docx, dan pdf masing-masing mempunyai rata-rata 0.65 s, 1.202 s, 1.198 s. Hal tersebut disebabkan karena proses dekompresi dilakukan di akhir tidak seperti pada proses dekompresi-dekripsi. Proses dekripsi membuat *file* kembali menjadi *ciphertext* hasil kompresi sehingga proses dekompresi menjadi lebih cepat.

V. KESIMPULAN DAN SARAN

A. Kesimpulan

Berdasarkan hasil penelitian simulasi kriptografi menggunakan Shannon-Fano sebagai algoritma kompresi dan AES sebagai enkripsi data didapat kesimpulan sebagai berikut:

1. Algoritma Shannon-Fano tidak cocok digunakan untuk *file* teks berformat docx dan pdf karena format *file* tersebut lebih kompleks dibandingkan dengan *file* dengan format txt.
2. Hasil pengujian kombinasi enkripsi-kompresi pada *file* txt, docx, dan pdf membuat ukuran *file* menjadi lebih besar sedangkan kombinasi kompresi-enkripsi dapat membuat ukuran *file* menjadi lebih kecil untuk *file* txt tapi tidak untuk *file* docx dan pdf. Rata-rata nilai rasio kompresi untuk proses enkripsi-kompresi pada *file* txt, docx, dan pdf masing-masing sebesar 1.01 sedangkan rata-rata nilai rasio kompresi untuk proses kompresi-enkripsi pada *file* txt adalah sebesar 0.64 dan untuk *file* docx dan pdf masing-masing sebesar 1.00. Rata-rata nilai penghematan ruang untuk proses enkripsi-kompresi pada *file* txt, docx, dan pdf masing-masing adalah sebesar -1.074%, -1.040%, dan -1.025% sedangkan rata-rata penghematan ruang untuk proses kompresi-enkripsi adalah sebesar 35.896%, -0.759%, dan -0.0291%.
3. Proses kompresi-enkripsi memiliki waktu proses yang lebih cepat dibandingkan dengan waktu proses enkripsi-kompresi begitu pula dengan proses pembaliknya. Rata-rata waktu proses yang didapat untuk kombinasi enkripsi-kompresi pada *file* txt, docx, dan pdf masing-masing adalah 2.22 s, 2.179 s, dan 2.204 s sedangkan rata-rata waktu proses yang didapat untuk kombinasi kompresi-enkripsi adalah 2.05 s, 2.114 s, dan 2.122 s.
4. Penggabungan algoritma enkripsi AES 128 dan algoritma kompresi Shannon-Fano dengan kombinasi enkripsi-kompresi tidak cocok untuk digunakan karena membuat ukuran *file* bertambah dan memiliki waktu proses yang lebih lama apabila dibandingkan proses kompresi-enkripsi.

B. Saran

Masih terdapat kekurangan dalam penelitian ini sehingga perlu pengembangan agar menjadi lebih baik lagi. Saran dari skripsi ini untuk penelitian selanjutnya yaitu:

1. Pembuatan aplikasi *executable* untuk penggabungan algoritma enkripsi dan kompresi sehingga tidak hanya dapat digunakan pada komputer yang memiliki *software* NetBeans.
2. Penggunaan algoritma kompresi lain yang lebih baik untuk dikombinasikan dengan algoritma enkripsi AES 128.
3. Penggunaan algoritma Rijndael dengan kunci 192 bit atau 256 bit untuk dikombinasikan dengan algoritma kompresi.

DAFTAR PUSTAKA

Adrisatria, Yogi. *Penerapan Algoritma Huffman Dalam Dunia Kriptografi*. Bandung: Institut Teknologi Bandung

Andri, M. Yuli. 2009. *Implemetasi Algoritma Kriptografi DES, RSA dan Algoritma Kompresi LZW pada berkas digital*. Medan: Fakultas Matematika dan Ilmu Pengetahuan Alam USU

Katti, dkk. *Using an innovative coding algorithm for data encryption*. North Dakota: Department of Electrical and Computer Engineering

Munir, Rinaldi. 2005. *Bahan Kuliah IF5054 Kriptografi*. Bandung: Institut Teknologi Bandung

Sobe, dkk. *Combining Compression, Encryption and Fault-tolerant Coding for Distributed Storage*. Luebec: Institute of Computer Engineering Luebec

Wiryadinata, Romi.2007. *Data Compression Coding Using Static And Dynamic Method of Shannon-Fano*. Media Informatika, Vol. 5 No. 2