Implementasi Algoritma *Advanced Encryption Standard* (AES) 128 Untuk Enkripsi dan Dekripsi *File* Dokumen

Asri Prameshwari[1], Nyoman Putra Sastra[2]
Fakultas Teknik Elektro, Universitas Udayana
Jln. Jalan Kampus Bukit Jimbaran 80361 Denpasar Bali, Telp: 0361-703315
e-mail: [1]asri.prameshwari@yahoo.com, [2]putra.sastra@unud.ac.id

Abstrak

Keamanan data atau informasi adalah hal yang sangat penting bagi pengguna jaringan internet saat ini. Kasus penyadapan akan pesan atau informasi merupakan salah satu hal yang sangat merugikan, dengan adanya kemungkinan terjadinya kejadian ini, maka perlunya peningkatan dalam hal keamanan pertukaran informasi menjadi penting. Pada saat ini, keamanan pertukaran informasi ini perlu mendapatkan perhatian khusus, maka penelitian ini akan membuat suatu implementasi kriptografi algoritma AES-128 untuk enkripsi dan dekripsi data yang berupa file dokumen (PDF, DOC, TXT). Algoritma Advanced Encryption Standard (AES) dipilih karena memiliki suatu tingkat keamanan pertukaran informasi yang cukup bagus, dan pada penelitian ini diuji coba file dokumen untuk melihat kecepatan waktu yang dibutuhkan selama proses enkripsi dan dekripsi.

Kata kunci: *AES-128, Keamanan, Enkripsi, Dekripsi.*

Abstract

Data information security is important for the internet users lately. Many cases regarding wiretapping message and information is very detrimental, with the possibility of the occurrence of the events, the need for improvement in terms of exchange data information security becomes important. At this time data information security needs to get special attention, this research will make an implementation of cryptographic algorithm AES-128 to encrypt and decrypt data specifically in file document such as pdf, doc and txt. The Algorithm Encryption Standard (AES) was chosen because it has a level of security that is a pretty good in exchange of data information and this study the document files was tested to see the speed of time needed during the encryption and decryption process.

Keywords*: AES-128, Security, Encryption, Decryption.*

1. Pendahuluan

Kerahasiaan dari data atau informasi merupakan suatu kelengkapan pelayanan yang dibuat untuk menjaga agar informasi yang tersimpan tidak dapat dibaca atau dibuka oleh pihak yang tidak berhak. Upaya dalam menjaga kerahasiaan dari data informasi tersebut sudah tercetus sejak jaman dahulu tepatnya pada jaman romawi dengan metode pergeseran huruf atau karakter dengan dasar nilai tertentu.

Pada jaman modern berbasis teknologi komputer, upaya-upaya tersebut berkembang dengan menggunakan algoritma yang diciptakan oleh banyak ahli, namun hal tersebut masih saja dapat dipecahkan oleh pihak-pihak yang tidak bertanggung jawab, maka dari itu perkembangan algoritma kriptografi semakin pesat demi keamanan data.

Konsep perlindungan akan data informasi dapat dilakukan dengan sistem enkripsi dan dekripsi menggunakan algoritma yang telah ditetapkan sebelumnya. Proses enkripsi di sini diartikan sebagai proses perubahan dari suatu pesan asli (*plain text)* menjadi suatu pesan yang terlindungi dalam hal ini pesan yang tersandi (*chiper text)*, sedangkan untuk proses dekripsi adalah suatu proses pengembalian pesan tersandi yang terlindungi menjadi bentuk data asli pesan tersebut [3].

Pada kedua proses tersebut dibutuhkan suatu pengaman yang menjamin bahwa pesan tersebut terlindungi pada prosesnya, pengaman tersebut dinamakan *key*. Fungsi dari *key* ini adalah kunci untuk membuka atau mengawali tiap proses, pada penelitian ini *key* yang digunakan adalah algoritma kunci simetri (*Simmetric-key algorithm)* yaitu algoritma kunci yang sama pada saat melakukan proses enkripsi dan dekripsi.

p-ISSN: 2089-1814; e-ISSN: 2460-3694; DOI: 10.30864/eksplora.v8i1.139

Pada penelitian ini proses enkripsi dan dekripsi yang dilakukan dalam perlindungan data informasi berbasiskan algoritma kriptografi yaitu Algoritma AES-128 dengan Kunci Simetri. Penerapan algoritma ini akan dilakukan pada pengamanan jenis data berjenis dokumen dengan tipe *pdf*, *doc*, *txt*. Pada proses pembuatan aplikasi perlindungan enkripsi dekripsi ini akan menggunakan *software Microsoft Visual Studio* 2012 sebagai bahasa pemrograman.

2. Tinjauan Pustaka (*State of the Art*)

Algoritma *Advanced Encryption Standard (*AES) adalah suatu algoritma *block chiper* dan mempunyai sifat simetri yang menggunakan kunci simetri pada waktu proses enkripsi dan dekripsi [2]. Pada tahun 2001, AES digunakan sebagai standar algoritma kriptografi terbaru yang dipublikasikan oleh NIST (National Institute of Standard and Technology) sebagai pengganti algoritma DES (Data Encryption Standard) yang sudah berakhir masa penggunaannya. Algoritma AES adalah algoritma kriptografi yang dapat mengenkripsi dan mendekripsi data dengan panjang kunci yang bervariasi, yaitu 128 bit, 192 bit, dan 256 bit [1]. Perbedaan dari ketiga urutan tersebut adalah panjang kunci yang mempengaruhi jumlah *round* (perputaran) yang dapat digambarkan dalam bentuk tabel [6]:

Tabel 1. Tabel urutan data algoritma AES.

	Panjang Kunci	Panjang Blok	Jumlah Putaran
AES-128	4	4	10
AES-129	6	4	12
AES-256	8	4	14

Pada Tabel 1 di atas dijelaskan mengenai tipe dari algoritma AES dengan panjang kunci, panjang blok dan jumlah putaran yang berbeda-beda. Untuk penelitian ini digunakan AES-128 bit dengan jumlah putaran enkripsi sebanyak 10 kali.

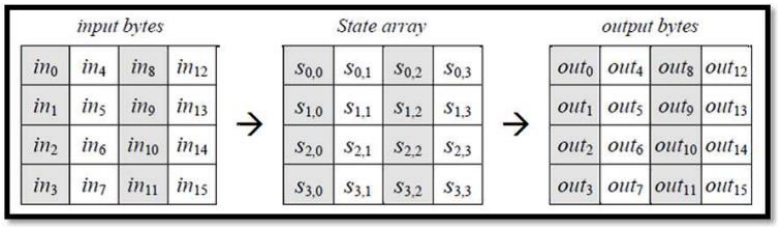
Terdapat 4 transformasi putaran pada proses enkripsi dan dekripsi [7]:

1. *SubBytes*
   Berfungsi untuk menukar isi dari *byte* dengan menggunakan tabel substitusi.
2. *ShiftRows*
   Proses pergeseran blok per baris pada *state array*.
3. *MixColumn*
   Proses mengalikan blok data (pengacakan) di masing-masing *state array* dengan rumus sebagai berikut:
   $$A(x) = \{03\}x^2 + \{01\}x^2 + \{01\}x + \{02\}$$
4. *AddRoundKey*
   Mengombinasikan *state array* dan *round key* dengan hubungan XOR.

Pada proses dekripsi algoritma AES:

1. *InvShiftRows*
   Melakukan pergeseran bit ke kanan pada setiap blok baris.
2. *InvSubBytes*
   Setiap elemen pada *state* dipetakan dengan tabel *Inverse S-Box*.
3. *InvMixColumn*
   Setiap kolom dalam *state* dikalikan dengan matriks AES.
4. *AddRoundKey*
   Mengombinasikan *state array* dan *round key* dengan hubungan XOR.

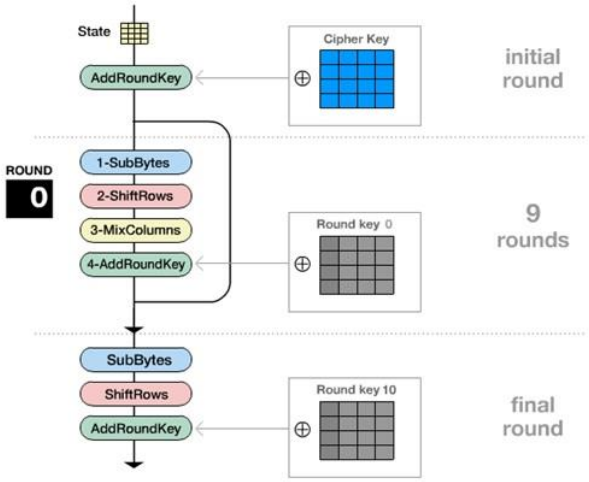
Penggambaran proses transformasi putaran dapat dilihat dari Gambar 1 [6] :



Gambar 1. Proses *input bytes, state array, output bytes*.

Dari Gambar 1 dapat dijelaskan bahwa algoritma AES ini ada dasarnya, algoritma AES ini merupakan *array of bytes* dengan dua dimensi yang disebut dengan *state*. Rumus ukuran dari *state* adalah $NROWS\ x\ NCOLS$, dari *state* ini akan diproses enkripsi dan dekripsi yang hasilnya akan dimasukkan ke dalam *array of state*. Pada awal proses enkripsi data dimasukkan ke dalam *input bytes* yang kemudian akan di salin kedalam *array state*, pada proses ini nantinya akan dilakukan enkripsi dan dekripsi, hasil keluarannya akan ditampung dalam *output bytes*.

Pada awal proses enkripsi, *input* yang telah disalin ke dalam *state* akan mengalami transformasi *AddRoundKey*. Setelah itu *state* akan mengalami transformasi *SubBytes*, *ShiftRows*, *MixColumns*, dan *AddRoundKey* secara berulang-ulang sebanyak *round*/putaran (Nr). Proses ini dalam algoritma AES disebut sebagai *round function*. *Round* yang terakhir, *state* tidak diberikan transformasi *MixColumns* [2]. Ilustrasi proses awal enkripsi dengan menggunakan algoritma AES -128 dijelaskan pada Gambar 2 [6]:



Gambar 2. Proses enkripsi dengan menggunakan algoritma AES-128.

3. Metode Penelitian

3.1. Deskripsi Singkat Aplikasi

Pada penelitian ini akan dibuat suatu aplikasi enkripsi dekripsi *file* dokumen dengan menggunakan algoritma AES-128 berbasiskan *desktop application*.

Aplikasi ini mempunyai dua tujuan yang pertama adalah untuk melakukan proses enkripsi *file* dokumen yang mempunyai ekstensi *.pdf*, *.doc* dan *.txt* dengan menggunakan *symmetric key* yang di-*input*-kan ketika akan dimulai proses enkripsi dengan keluaran hasil yaitu *file* enkripsi dengan ukuran *file* yang lebih kecil dan juga waktu proses yang dibutuhkan untuk proses enkripsi.

Tujuan kedua dari aplikasi ini adalah untuk melakukan proses dekripsi terhadap *file* dokumen yang telah di proses enkripsi sebelumnya. Pada proses ini juga diperlukan *key* yang sama dengan *key* yang di-*input*-kan pada proses enkripsi. Hasil keluaran dari proses dekripsi ini adalah ukuran *file* yang kembali seperti semula dan waktu proses yang dibutuhkan untuk proses dekripsi.

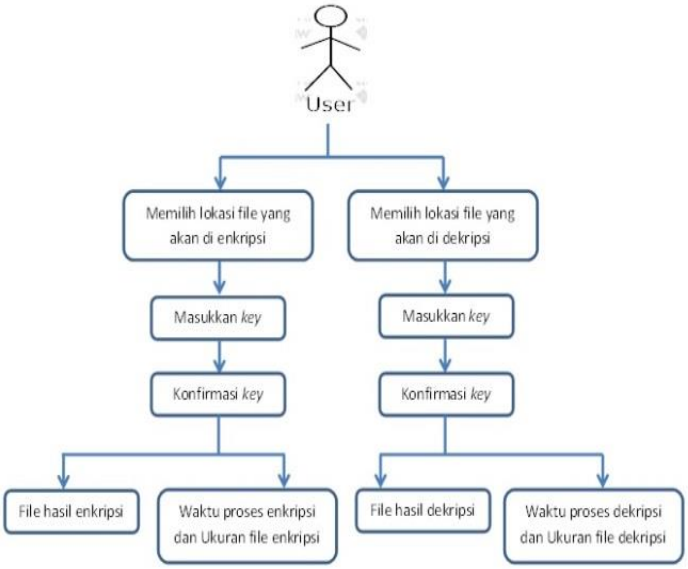
3.2. Perancangan Sistem

Pada perancangan sistem yang akan diterapkan pada penelitian ini akan dijelaskan melalui *activity diagram*. Dari diagram tersebut permodelan interaksi dari keseluruhan alur kerja sistem dapat digambarkan dengan baik.

Proses pertama adalah proses enkripsi yang dimulai dengan memilih lokasi *file* yang akan di enkripsi, setelah itu dilanjutkan dengan memasukkan dan mengkonfirmasi kunci enkripsi untuk kemudian dilakukan proses enkripsi yang akan menghasilkan *file* hasil enkripsi dan informasi mengenai waktu beserta besar ukuran *file* hasil tersebut.

Proses kedua adalah dekripsi, pada proses ini *file* yang sebelumnya sudah melalui proses enkripsi dimasukkan lagi sebagai *file input*-an yang kemudian ditambahkan dengan kunci yang sama, hal ini diperlukan karena peran kunci adalah validasi untuk dapat melanjutkan proses dekripsi. *Output* dari proses ini adalah *file* dokumen yang telah terdekripsi beserta waktu dan besar ukuran *file* dekripsi.

Dari kedua proses tersebut, dapat dijelaskan pada Gambar 3 dan dalam bentuk *activity diagram* sebagai berikut:



Gambar 3. *Activity diagram* aplikasi.

4. Implementasi

Implementasi merupakan bagian yang menjelaskan tentang pembuatan tampilan aplikasi. Penjelasan mengenai implementasi sebagai berikut.

4.1. Implementasi Tampilan Menu Utama

Pada proses awal berjalannya aplikasi, aplikasi akan menampilkan menu utama yang menunjukkan berbagai menu yang ada. Sebagai *input*-an *file* dokumen yang akan di enkripsi adalah *file* dokumen yang berjenis *.pdf*, *.doc*, dan *.txt*.

Gambar 4. Tampilan awal aplikasi.

Pada Gambar 4 di atas menunjukkan tentang tampilan awal dan garis besar dari aplikasi ini. Terdapat menu *file* sumber yakni asal lokasi *file* yang akan di enkripsi, masukkan kata kunci yang akan dijadikan *key*, kata kunci ini gabungan dari huruf dan angka dan konfirmasi kata kunci untuk mengulang kata kunci yang sama (*symmetric key*).

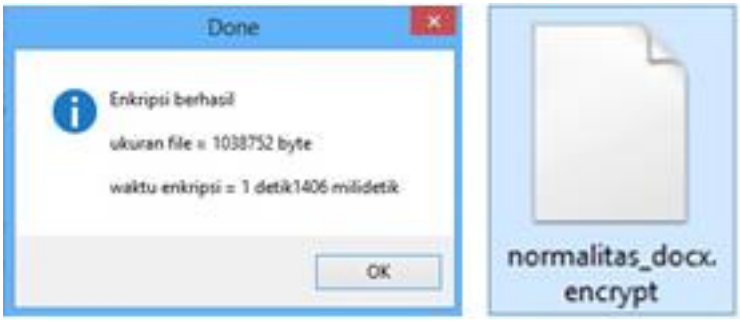
4.2. Implementasi Tampilan *Input* Enkripsi *File*

Pada proses ini pengguna memilih *file* yang akan dienkripsi dan memasukkan *encryption key*. Pada Gambar 5 dijelaskan mengenai *file* dokumen yang akan dienkripsi disertai dengan memasukkan *key* dan konfirmasi *key*.

Gambar 5. Tampilan enkripsi i*nput*, enkripsi *file*, dan kata kunci.

4.3. Implementasi Tampilan *File* Enkripsi, Waktu yang dibutuhkan dan Ukuran *File*

Hasil keluaran dari program ini adalah menampilkan ukuran *file* yang telah di enkripsi, waktu lama enkripsi dan *file* yang telah dienkripsi.

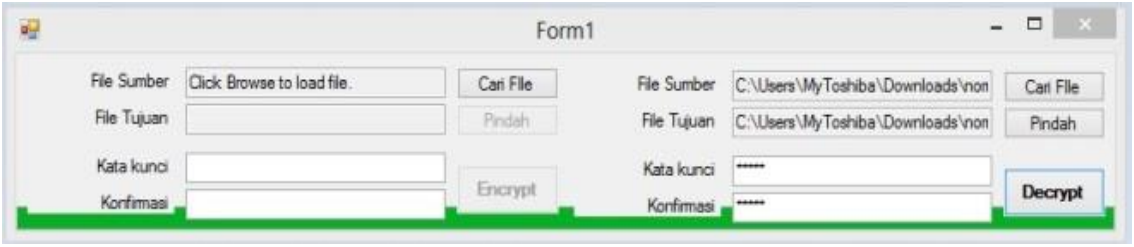


Gambar 6. Tampilan *file* hasil enkripsi, waktu dan ukuran *file*.

Pada Gambar 6 di atas menunjukkan tentang informasi yang didapat oleh pengguna sebagai penjelasan akhir dari proses enkripsi. Setelah melakukan proses enkripsi akan keluar pesan berhasil dengan keterangan ukuran *file* dan waktu yang dibutuhkan. *File* hasil enkripsi terdapat pada folder yang sama dengan *file* awal. Hasil *file* enkripsi ditandai dengan ditambahkannya ekstensi *.encrypt* untuk hasil keluaran *file*.

4.4. Implementasi Tampilan Proses Dekripsi

Pada proses ini pengguna memilih *file* yang akan didekripsi dan *decryption key*.

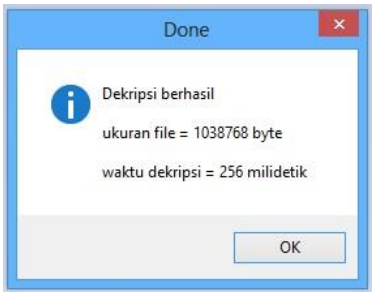


Gambar 7. Tampilan dari proses dekripsi.

Pada proses dekripsi dimulai dengan memasukkan *file* hasil enkripsi dan *decryption key* yang sesuai dengan proses enkripsi awal.

4.5. Implementasi Tampilan Hasil Dekripsi

Pada proses ini akan ditampilkan hasil dekripsi dimana hasil dari proses dekripsi adalah *file* awal yang sama sebelum diproses. Di samping hasil *file*, juga tercantum informasi ukuran *file* hasil dekripsi dan waktu yang dibutuhkan untuk dekripsi.



Gambar 8. Tampilan hasil dekripsi.

5. Uji Coba

Fase uji coba terhadap aplikasi ini dilakukan pada beberapa *file* dokumen yang terdiri dari 2 *file* dokumen berjenis *.doc*, 2 *file* dokumen berjenis *.txt* dan 2 *file* dokumen berjenis *.pdf* dengan besar ukuran file berbeda-beda.

Dari *file-file* di atas dapat dijabarkan proses enkripsi dalam bentuk tabel seperti di bawah ini:

Tabel 2. Tabel uji coba enkripsi.

No	Nama File	Ukuran File awal	Ukuran File enkripsi	Waktu enkripsi
1.	Example 1.pdf (12 Hal)	7178000 byte	7349271 btye	3,389 dtk
2.	Example 2.pdf (16 Hal)	818000 byte	836961 byte	0,156 dtk
3.	Example 3.docx (17 Hal)	1015000 byte	1038752 byte	0,312 dtk
4.	Example 4.doc (17 Hal)	133000 byte	136192 byte	0,156 dtk
5.	Example 5.txt	3000 byte	2990 byte	0 dtk
6.	Example 6.txt	8000 byte	8062 byte	0 dtk

Dari tabel di atas dapat dilihat untuk ukuran *file* awal dalam bentuk *byte*. Untuk tiap contoh yang digunakan juga dicantumkan banyaknya halaman sebagai referensi ukuran *file* awal, dari parameter-parameter tersebut dapat di hitung berapa hasil ukuran *file* enkripsi dan waktu enkripsi yang dibutuhkan untuk masing-masing contoh.

Dari tabel hasil enkripsi di atas, proses kemudian dilanjutkan dengan proses dekripsi yang kemudian hasilnya dapat dilihat pada tabel di bawah ini:

Tabel 3. Tabel uji coba dekripsi.

No	Nama File	Ukuran File enkripsi	Ukuran File dekripsi	Waktu dekripsi
1.	Example 1_pdf.encrypt	7349271 btye	7349280 byte	2,558 dtk
2.	Example 2_pdf.encrypt	836961 byte	836976 byte	0,312 dtk
3.	Example 3_docx.encrypt	1038752 byte	1038768 byte	0,312 dtk
4.	Example 4_doc.encrypt	136192 byte	136208 byte	0,06 dtk
5.	Example 5_txt.encrypt	2990 byte	2992 byte	0,01 dtk
6.	Example 6_txt.encrypt	8062 byte	8064 byte	0,156 dtk

Dari tabel hasil dekripsi di atas dapat dilihat ukuran *file* enkripsi yang lebih kecil dari *file* awal dan kemudian di dekripsi menjadi ukuran *file* semula, waktu yang dibutuhkan tergantung dari masing-masing *file* yang akan didekripsi. Beberapa *file* membutuhkan waktu yang relatif lebih singkat dari proses enkripsi, ada beberapa yang membutuhkan tambahan waktu lebih lama sepersekian detik.

6. Kesimpulan

Dari hasil uji coba enkripsi dan dekripsi di atas dapat disimpulkan bahwa:

1. Algoritma AES-128 dapat dijadikan salah satu alternatif untuk proses keamanan data dalam hal ini enkripsi dan dekripsi *file* dokumen.
2. Ukuran *file* merupakan salah satu variabel yang cukup penting karena berpengaruh terhadap waktu proses enkripsi dan dekripsi. Pada variabel hasil, waktu merupakan tolak ukur dari proses, apakah terhitung cepat atau lambat dari ukuran file yang harus diproses.
3. Hasil dari enkripsi ini bisa dijamin keamanannya selama *symmetry key encryption* tidak bocor ke pihak yang tidak bertanggung jawab.

4. Dari hasil penelitian telah dibuktikan bahwa isi *file* awal yang mengalami proses enkripsi, kemudian mengalami proses dekripsi, maka akan kembali seperti file awal semula.
5. Dari hasil enkripsi dan dekripsi *file*, pada penelitian bisa disimpulkan bahwa waktu yang dibutuhkan relatif lebih cepat dibanding dengan penelitian sebelumnya (Yulius Rio, 2016) . Proses enkripsi untuk 7,1 MB dibutuhkan waktu 3,3 detik, dibandingkan dengan 1,8 MB dibutuhkan waktu 1 detik. Proses dekripsi untuk 7,2 MB dibutuhkan waktu 2.5 detik dibandingkan dengan 1,8 MB dibutuhkan waktu 0.4 detik.

Daftar Pustaka

[1] Soni Harza Putra, dkk., "Implementasi Algoritma Kriptografi *ADVANCED ENCRYPTION STANDARD* (AES) Pada Kompresi Data Teks". Jurnal Teknologi Informasi, Universitas Brawijaya Malang. 2013.
[2] Voni Yuniati, dkk. "Enkripsi dan Dekripsi Dengan Algoritma AES-256 Untuk Semua Jenis File". Jurnal nformatika Volume 5 No. 1 April 2009.
[3] Veronica Lusiana. "Implementasi Kriptografi Pada File Dokumen Menggunakan Algoritma AES-128".2013.
[4] Rifkie Primartha. "Penerapan Enkripsi dan Dekripsi File Menggunakan Algoritma *Advanced Encryption Standard (AES)*". *Journal Research in Computer Science and Applications Vol.2 No.1 Januari 2013* ISSN: 2301-8488. 2013.
[5] Yulius Rio Pujianto, dkk. "Perancangan dan Impelementasi Aplikasi Kriptografi Algoritma AES-128 Pada File Dokumen".Artikel Ilmiah Program Studi Teknik Informartika, Fakultas Teknologi Informasi, Universitas Kristen Satya Wacana, Salatiga. 2016.
[6] Munir, Rinaldi. Kriptografi. Bandung:Penerbit Informatika. 2006.
[7] R.Kristoforus JB, dkk. "Implementasi Algoritma *Rijndael* Untuk Enkripsi dan Dekripsi Pada Citra Digital".Seminar Nasional Aplikasi Teknologi Informasi 2012 (SNATI 2012). ISSN: 1907-5022. 2012.