IMPLEMENTASI METODE *ADVANCED ENCRYPTION STANDARD* (AES) DAN *MESSAGE DIGEST* 5 (MD5) PADA ENKRIPSI DOKUMEN (STUDI KASUS LPSE UNIB)

Gilang Gumira P.U.K.[1], Ernawati[2], Aan Erlanshari [3]

[12,3]Program Studi Teknik Infomatika, Fakultas Teknik, Universitas Bengkulu.
Jl. WR. Supratman Kandang Limun Bengkulu 38371A INDONESIA
(telp: 0736-341022; fax: 0736-341022)

[1]gilanggpuk@yahoo.co.id

Abstrak*:* Penelitian ini bertujuan untuk membangun suatu aplikasi pengamanan dokumen yang dapat membantu pengguna dalam mengamankan dokumen penting yang tidak bisa secara bebas dibaca oleh umum. Aplikasi ini akan mengenkripsi dan mendekripsikan kembali dokumen dengan menggunakan algoritma AES dan mengolah kunci menjadi nilai hash dengan menggunakan algoritma MD5. Aplikasi ini dikembangkan menggunakan analisis berorientasi objek *Unified Modeling Language* (UML) dan dibangun menggunakan *Adobe Dreamweaver CS6*. Hasil penelitian menunjukkan bahwa algoritma AES dengan panjang kunci 256 *bit* dapat menyandikan isi suatu file sehingga dapat mengamankan file tersebut. File yang dihasilkan oleh proses enkripsi bisa menjadi 2 kali lipat bahkan lebih dari ukuran dan jumlah karakter file aslinya, ini dikarenakan hasil enkripsi dibuat dalam hex. Sedangkan algoritma MD5 yang berfungsi sebagai penghasil nilai hash yang akan digunakan sebagai sandi untuk mengenkripsi dan mendekripsikan kembali dokumen, mampu menghasilkan dan menambah kerumitan dalam proses enkripsi dan dekripsi dokumen khususnya dari segi kerumitan sandi yang digunakan.

*Kata Kunci****:*** *Cryptogrhapy,* Algoritma AES-256, Algoritma MD5, Enkripsi, Pengamanan Dokumen

Abstract: **This study investigated to develop an application security documents that can assist users in securing important documents which can't be freely read by the public. This application will be encryption and back describe the document by using the AES algorithm and processing a key into a hash value by using the MD5 algorithm. The application was developed by using object analysis Unified Modeling Language (UML) and it is built by using Adobe Dreamweaver CS6. The results show that the AES algorithm with a key length of 256 bits can encode the contents of a file so that it can secure the file. The files generated by the encryption process can be a two-fold even more than the size of the original file and the number of character, this is because of the encrypted made in**

hexadecimal. While the MD5 algorithm that serves as a producer of hash values to result password encryption and describe the document, able to produce and able to add the complexity of the encryption process and able to describe of the document, especially in term of complexity of password use.

Keywords: Cryptograhpy, AES-256, MD5, encryption, document security

I. PENDAHULUAN

Pertukaran data melalui jaringan internet sangat mungkin dilakukan karena tentunya akan mempercepat dan memudahkan proses pertukaran data terutama dengan jarak yang jauh. Kemudahan ini tentunya berdampak pada munculnya resiko dan ancaman keamanan data, misalnya penduplikasian atau bahkan perusakan informasi itu sendiri. Untuk itu diperlukan suatu manajemen keamanan yang dapat melindungi atau paling tidak menahan suatu akses yang tidak berhak untuk durasi waktu tertentu.

Pengamanan file dengan menggunakan teknik kriptografi telah banyak dilakukan dalam berbagai penelitian. Implementasi dengan menggunakan salah satu algoritma kriptografi saja kiranya sudah mulai ditinggalkan dan beralih pada penggunaan gabungan antara asimetri dan simetri. Penelitian ini akan membuat sebuah aplikasi enkripsi dokumen menggunakan algoritma *Advanced Encryption Standard* (AES) dan *Message Digest* 5 (MD5). AES merupakan salah satu algoritma simetri dan akan digunakan sebagai algoritma enkripsi dan dekripsi pada dokumen, sedangkan MD5 merupakan fungsi hash dan akan digunakan sebagai algoritma penyandian kunci, karena MD5 merupakan algoritma satu arah.

Studi kasus yang akan diangkat pada penelitian ini adalah mengenai Layanan Pengadaan Secara Elektronik (LPSE) di Universitas Bengkulu. Aplikasi ini bersifat simulasi, karena aplikasi utama LPSE UNIB dibawah pengawasan LPSE pusat yang berada di Jakarta. Secara umum, alur proses aplikasi Layanan Pengadaan Secara Elektronik (LPSE) terbagi menjadi 3 bagian besar, yaitu: Pendaftaran Rekanan, Persiapan Lelang, dan Lelang. Untuk dapat mengikuti lelang melalui aplikasi LPSE UNIB, terlebih dahulu peseta lelang/perusahaan harus mendaftar untuk menjadi rekanan. Pada proses pendaftaran untuk menjadi rekanan ini peserta lelang diharuskan membuat akun di website LPSE UNIB. Kegunaan akun ini adalah sebagai sarana/media untuk mendapatkan informasi dan sekaligus sebagai tempat pengajuan proposal lelang secara online.

Saat mengunggah file dokumen, peserta lelang memerlukan sebuah sandi bebas berupa angka, huruf, atau perpaduan keduanya yang akan digunakan sistem sebagai kunci untuk mengenkripsi file peserta lelang dengan menggunakan algoritma MD5, yang akan diunggah ke database LPSE UNIB. Format file yang digunakan dalam proses lelang ini hanya file yang berformat .docx.

Setelah *user* mengunggah dokumen terenkripsi ke database LPSE UNIB, secara otomatis sistem akan memberikan ID yang berguna sebagai verifikasi apabila *user* tersebut mengikuti lelang lebih dari 1 kali dalam 1 periode lelang yang sama. Dan juga berguna sebagai informasi dokumen diunggah pada tanggal, bulan, dan tahun berapa.

File dokumen hanya bisa di unduh dan di lihat oleh pihak LPSE UNIB saja. Saat mengunduh file, operator LPSE UNIB memilih Nama Perusahaan

peserta lelang. Lalu sistem secara otomatis akan menampilkan ID User, file dokumen yang terenkripsi, dan sandi dalam bentuk hash. Klik dekripsi lalu file akan didekripsi oleh sistem dan langsung diunduh.

Mengingat akan pentingnya isi dokumen lelang dan untuk mendukung program pemerintah tentang meningkatkan transparansi dan akuntabilitas, dan memenuhi kebutuhan akses informasi yang *real time* guna mewujudkan *clean and good government* dalam pengadaan barang/jasa pemerintah (Asliana, 2012). Maka dari itu enkripsi dokumen perlu diterapkan di LPSE,

Berdasarkan permasalahan yang telah diuraikan di atas, peneliti akan Mengimplementasikan sebuah sistem pengenkripsian dokumen dengan judul **"Implementasi Metode Advanced Encryption Standard (AES) & Message Digest 5 (MD5) Pada Enkripsi Dokumen"**. Penggunaan dua algoritma ini diharapkan dapat meningkatkan keamanan dokumen dari penyalahgunaan oleh pihak yang tidak berhak.

II. LANDASAN TEORI

A. *Message Digest 5* (MD5)

MD5 di desain oleh Ronald Rivest pada tahun 1991 untuk menggantikan hash function sebelumnya, yaitu MD4 yang berhasil diserang oleh kriptanalis. Algoritma MD5 menerima masukan berupa pesan dengan ukuran sembarang dan menghasilkan *message digest* yang panjangnya 128 bit. MD5 merupakan fungsi hash kelima yang dirancang oleh Ron Rivest dan didefinisikan pada RFC 1321.

MD5 memproses teks masukan ke dalam blok - blok bit sebanyak 512 bit sebanyak 16 buah. Keluaran dari MD5 berupa 4 blok yang masing - masing 32 bit yang mana akan menjadi 128 bit yang biasa disebut dengan nilai hash. Simpul utama MD5 mempunyai blok pesan dengan panjang 512 bit yang kemudian dimasukan ke dalam 4 buah ronde. Dan hasil keluaran dari MD5 adalah berupa 128 bit dari byte terendah A yaitu ronde 1 dan byte tertinggi D yaitu ronde 2.

Setiap pesan yang akan dienkripsi terlebih dahulu dicari berapa banyak bit yang terdapat pada pesan. Kita dapat mengangap bahwa pesan yang akan kita enkripsi adalah pesan yang memiliki L bit. Disini L adalah *non negative integer*, L dapat pula NULL dan tidak harus kelipatan delapan.

Terdapat 4 langkah untuk mencari inti dari pesan dalam MD5, yaitu :

1. Menambah bit

Pesan akan ditambahkan bit - bit tambahan sehingga panjang bit akan kongruen dengan 448 mod 512. Hali ini berarti pesan akan mempunyai panjang yang hanya kurang 64 bit dari kelipatan 512 bit. Penambahan bit selalu dilakukan walaupun panjang dari pesan sudah kongruen dengan 448 mod 512 bit. Penambahan bit dilakukan dengan menambahkan "1" diawal dan diikuti "0" sebanyak yang diperlukan sehingga pesan akan kongruen dengan 448 mod 512.

2. Penambahan Panjang Pesan

Setelah penambahan bit, pesan masih membutuhkan 64 bit agar kongruen dengan kelipatan 512 bit. 64 bit tersebut merupakan perwakilan dari L (panjang pesan sebelum penambahan bit dilakukan). Jika panjang pesan > 2^{64} maka yang diambil adalah panjangnya dengan modulo $2^{64.}$ Dengan kata lain, jika panjang pesan semula adalah L bit, maka 64 bit yang ditambahkan menyatakan K modulo 2^{64}. Setelah

ditambah dengan 64 bit, panjang pesan sekarang adalah 512 bit. Penambahan pesan ini biasa disebut juga *MD Strengthening* atau Penguatan MD.

3. Inisialisasi penyangga MD5

Pada MD5 terdapat empat buah word atau penyangga *(buffer)* yang masing - masing panjangnya 32 bit. Total panjang penyangga adalah 4 x 32 = 128 bit. Empat buah penyangga berfungsi untuk menginisialisasi *message digest* untuk pertama kali dan menampung hasil antara dan hasil akhir MD5. Keempat penyangga ini diberi nama A, B, C , dan D. Setiap penyangga diinisialisasi dengan nilai - nilai *hexadecimal* dan bernilai tetap Word A : 01 23 45 67, Word B : 89 AB CD EF, Word C : FE DC BA 98, Word D: 76 54 32 10. Register - register ini biasa disebut dengan nama *chain variabel* atau variabel rantai.

4. Pengolahan Pesan dalam Blok Berukuran 512 bit

Pesan yang telah melewati 3 proses tinggal melewatkan proses terakhir yaitu pengolahan pesan. Pesan dibagi menjadi *L* buah blok yang masing - masing panjangnya 512 bit (Y_0 sampai Y_{L1}). Setiap blok 512 bit diproses bersama dengan penyangga MD menjadi keluaran 128-bit, dan ini disebut proses H_{MD5}.

Proses H_{MD5} terdiri dari 4 buah putaran, dan masing - masing putaran melakukan operasi dasar MD5 sebanyak 16 kali dan setiap operasi dasar memakai sebuah elemen *T*. Jadi setiap putaran memakai 16 elemen Tabel *T*.

Hasil MD ini lah yang nantinya akan disertakan dalam isi pesan, file yang akan didownload, dan kemudian tampilan layout website yang kemungkinan diubah oleh para hacker dan cracker untuk menjadi sebuah acuan bahwa pesan, file, dan website ini adalah benar keasliannya**.** (Duc H, Tan N, and Phong H. Pham, 2008).

B. *Advanced Encryption Standard* (AES)

Algoritma kriptografi bernama Rijndael yang didesain oleh Vincent Rijmen dan John Daemen asal Belgia keluar sebagai pemenang kontes algoritma kriptografi pengganti DES yang diadakan oleh NIST (National Institutes of Standards and Technology) milik pemerintah Amerika Serikat pada 26 November 2001.

Algoritma Rijndael inilah yang kemudian dikenal dengan Advanced Encryption Standard (AES). Setelah mengalami beberapa proses standardisasi oleh NIST, Rijndael kemudian diadopsi menjadi standard algoritma kriptografi secara resmi pada 22 Mei 2002. Pada 2006, AES merupakan salah satu algoritma terpopuler yang digunakan dalam kriptografi kunci simetrik (Rijmen, 1998).

Pada algoritma AES, jumlah blok *input*, blok *output*, dan *state* adalah 128 bit. Dengan besar data 128 bit, berarti Nb = 4 yang menunjukkan panjang data 22 tiap baris adalah 4 byte. Dengan blok input atau blok data sebesar 128 bit, key yang digunakan pada algoritma AES tidak harus mempunyai besar yang sama dengan blok input.

Cipher key pada algoritma AES bisa menggunakan kunci dengan panjang 128 bit, 192 bit atau 256 bit. Perbedaan panjang kunci akan mempengaruhi jumlah round yang akan di implementasikan pada algoritma AES ini. Tahapan enkripsi dan dekripsi AES, yaitu :

1. Enkripsi AES

Pada awal proses enkripsi, input yang telah dicopykan ke dalam *state* akan mengalami transformasi *byte AddRoundKey*. Setelah itu, state akan mengalami transformasi *SubBytes*,

ShiftRows, *MixColumns*, dan *AddRoundKey* secara berulang - ulang sebanyak Nr. Proses ini dalam algoritma AES disebut sebagai *round function. Round* yang terakhir agak berbeda dengan *round – round* sebelumnya, dimana pada *round* terakhir *state* tidak mengalami transformasi *MixColumns*.

2. Dekripsi AES

Transformasi chipher dapat dibalikkan dan diimplementasikan dalam arah yang berlawanan untuk menghasilkan *inverse cipher* yang mudah dipahami untuk algoritma AES. Transformasi *byte* yang digunakan pada invers cipher adalah *InvShiftRows*, *InvSubBytes*, *InvMixColumns*, dan *AddRoundKey*.

III. METODOLOGI PENELITIAN

A. *Teknik Pengumpulan Data*

Studi pustaka merupakan metode pengumpulan data yang diperoleh dari buku - buku dan/atau jurnal dalam pencarian referensi terkait pengumpulan data maupun perancangan aplikasi yang akan dibangun, yaitu referensi mengenai dokumen, kriptografi, algoritma AES dan MD5.

Analisis yang dilakukan adalah analisis terhadap pola dan karakteristik dari algoritma kriptografi terkait dan analisis terhadap *test-case* untuk validasi algoritma kriptografi yang dapat diterapkan.

B. *Metode Pengujian*

Setiap produk perangkat lunak dapat diuji melalui dua pendekatan pengujian, yang pertama disebut sebagai *black-box testing* dan kedua disebut sebagai *white-box testing*.

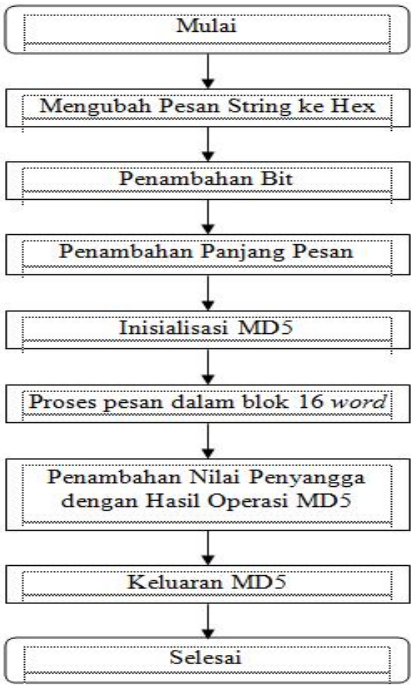
Pengujian *white box* dilakukan dengan menguji kode - kode progam yang dibuat pada aplikasi. Pengujian dilakukan dengan mengecek semua kode pada program telah dieksekusi paling tidak satu kali. Pengujian ini dilakukan pada proses pengembangan sistem, yakni pengujian kode program (*coding*).

Pada pengujian *black-box testing*, akan diamati hasil eksekuasi antarmuka melalui data uji dan memeriksa fungsional dari aplikasi yang telah dibuat.

IV. ANALISIS DAN PERANCANGAN

A. *Cara Kerja Sistem*

Pada sistem ini, MD5 digunakan sebagai algoritma penyandian kunci. Pertama, *user* menginputkan kunci dan file dokumen yang akan dienkripsi oleh sistem. Setelah *user* mengunggah kunci dan file dokumen, sistem menjalankan algoritma MD5 untuk mencari fungsi hash dari kunci tersebut dengan empat tahap proses, yaitu : Menambahkan bit, Penambahan panjang bit, Pengolahan pesan dalam blok 512 bit, dan Inisialisasi penyangga. Pada gambar 1 merupakan diagram alir algoritma MD5 :



Gambar 1 Diagram Alir Algoritma MD5

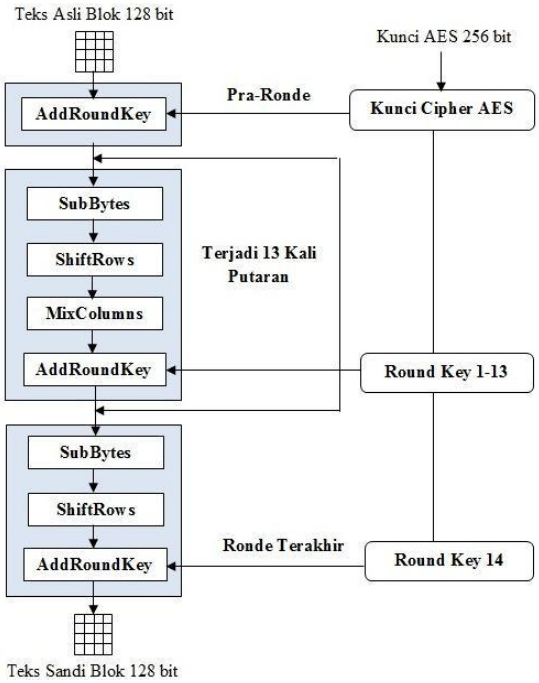
Pada sistem ini, AES digunakan sebagai algoritma enkripsi dan dekripsi dokumen.

a. Enkripsi

Setelah sistem mendapatkan fungsi hash,

sistem akan menggunakan fungsi hash tersebut sebagai kunci untuk mengenkripsikan dokumen yang *user* unggah. Setelah dokumen terenkripsi, sistem akan menyimpan file dokumen dan fungsi hash tersebut kedalam database.

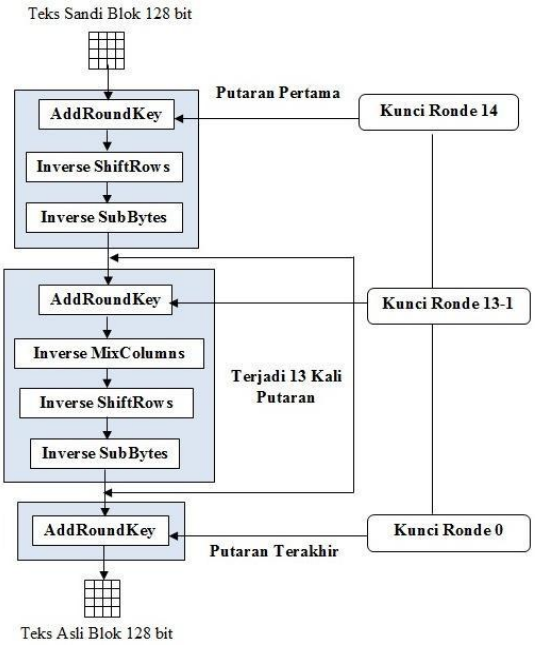
Pada awal proses enkripsi, input yang telah dicopykan ke dalam *state* akan mengalami transformasi *byte AddRoundKey*. Setelah itu, state akan mengalami transformasi *SubBytes*, *ShiftRows*, *MixColumns*, dan *AddRoundKey* secara berulang - ulang sebanyak Nr. Proses ini dalam algoritma AES disebut sebagai *round function. Round* yang terakhir agak berbeda dengan *round – round* sebelumnya, dimana pada *round* terakhir *state* tidak mengalami transformasi *MixColumns*. Pada gambar 2 merupakan diagram alir algoritma AES enkripsi.'



Gambar 2 Diagram Alir AES Enkripsi

Saat admin akan mengunduh file dokumen milik *user*, sistem akan mengambil file dokumen dan fungsi hash dari database dan mendekripsikan file dokumen tersebut.

Transformasi cipher dapat dibalikkan dan diimplementasikan dalam arah yang berlawanan untuk menghasilkan *inverse cipher* yang mudah dipahami untuk algoritma AES. Transformasi *byte* yang digunakan pada invers cipher adalah *InvShiftRows*, *InvSubBytes*, *InvMixColumns*, dan *AddRoundKey*. Pada gambar 3 merupakan diagram alir algoritma AES dekripsi.



Gambar 3 Diagram Alir Algoritma AES Enkripsi

V. PEMBAHASAN

A. Pengujian Sistem Secara Manual

Contoh sederhana dalam pengenkripsian menggunakan Algoritma AES dan MD5 dengan menggunakan permisalan kasus. Contoh kasus yang akan dicari adalah sebagai berikut :

Pengguna ingin mengenkripsi pesan yang berisi kata “unib”, dan kunci yang akan dipakai adalah “teknik”. Pengguna akan mengenkripsi pesan tersebut menggunakan Algoritma AES sebagai algoritma enkripsi dan dekripsi, dan menggunakan algoritma MD5 sebagai algoritma kunci agar pesan dapat diminimalisir dari

kemungkinan penyadapan dan tidak dapat terbaca oleh orang lain.

Sebelum mengenkripsi pesan, terlebih dulu harus dicari berapa fungsi hash dari “teknik” agar bisa digunakan sebagai kunci untuk mengenkripsi pesan “unib”. Perhitungan algoritma MD5 adalah sebagai berikut :

1. Tentukan pesan yang akan di *hashing*

   Pesan (String) : teknik

   Pesan (Hex) : **74 65 6b 6e 69 6b**

2. Penambahan bit

   Pesan ditambahkan dengan bit pengganjal hingga kongruen dengan 448 modulo 512 bit. Pesan : **74 65 6b 6e 69 6b**

   Pesan terdiri dari 6 karakter atau 6 byte = 6 x 8 = 48 bit, sehingga perlu ditambahkan sebesar 448 – 48 = 400 bit.

3. Pada bit terakhir sepanjang 64 bit diisi dengan nilai panjang pesan, dimana panjang pesan adalah 6 byte = 6 x 8 bit = 48 bit atau dalam hex = 30

4. Pengolahan pesan di dalam blok 16 word

   Pada MD-5 juga terdapat 4 (empat) buah fungsi nonlinear yang masing - masing digunakan pada tiap operasinya (satu fungsi untuk satu blok), yaitu:

   F(X,Y,Z) = XY v not(X) Z
   G(X,Y,Z) = XZ v Y not(Z)
   H(X,Y,Z) = X xor Y xor Z
   I(X,Y,Z) = Y xor (X v not(Z))

   Pengolahan pesan dilakukan sebanyak 4 putaran, tiap putaran dilakukan sebanyak 16 operasi dasar, yaitu sebagai berikut :

   Round (1,16)

   A = 2249092517 dalam Hex = 860E6DA5 B = 2931300306 dalam Hex = AEB817D2 C = 712877834 dalam Hex = 2A7DA70A D = 229885237 dalam Hex = 0DB3C535 Round (2,16)

   A = 3914487489 dalam Hex = E95256C1 B = 2441348034 dalam Hex = 918403C2 C = 3652998760 dalam Hex = D9BC5668 D = 28302740 dalam Hex = 01AFDD94 Round (3,16)

   A = 4131823715 dalam Hex = F646A063

   B = 3065820097 dalam Hex = B6BCB3C1

   C = 1815224134 dalam Hex = 6C321F46

   D = 73124886 dalam Hex = 045BCC16 Round (4,16)

   A = 1331224407 dalam Hex = **4F58DF57 (AA)**

   B = 2605068873 dalam Hex = **9B463249 (BB)**

   C = 3006515519 dalam Hex = **B333C93F(CC)**

   D = 2079023797 dalam Hex = **7BEB62B5 (DD)**

5. Inisialisasi penyangga/buffer MD5,

   Pada perhitungan ini menggunakan penyangga berikut (dalam hex)

   A = 67452301
   B = efcdab89
   C = 98badcfe
   D = 10325476

   Hasil akhir pengolahan pesan dengan 4 putaran yang masing-masing terdiri dari 16 operasi kemudian ditambahkan ke nilai penyangga MD5,

   A = A +AA
   B = B + BB
   C = C + CC
   D = D + DD

   Hasilnya adalah

   A = 3063808600 dalam Hex = B69E0258 B = 2333334994 dalam Hex = 8B13DDD2 C = 1273931325 dalam Hex = 4BEEA63D D = 2350757675 dalam Hex = 8C1DB72B

6. Output operasi MD5 diperoleh dari langkah nomor 6 dengan urutan seperti pada gambar 4

A	4	3	2	1
B	8	7	6	5
C	12	11	10	9
D	16	15	14	13

Gambar 4 Output Operasi MD5

A = B6 9E 02 58

B = 8B 13 DD D2

C = 4B EE A6 3D

D = 8C 1D B7 2B

Nilai digest pertama adalah 58, digest kedua 02, digest ketiga 9e dan seterusnya sesuai urutan pada tabel, hingga diperoleh 32 karakter digest MD5 yaitu

58029EB6D2DD138B3DA6EE4B2BB71D8C

Setelah mendapatkan nilai hash, langkah selanjutnya adalah mengenkripsi isi dokumen. perhitungan algoritma AES Enkripsi adalah sebagai berikut :

1. Input

   - Plainteks = ‘unib’
   - Password = md5(‘teknik’) = 58029eb6d2dd138b3da6ee4b2bb71d8c

2. Inisialisasi

   Konversi tiap karakter plainteks dari char ke hex. Plainteks dibagi perblok dengan kapasitas blok 16 karakter. Apabila plainteks tidak cukup 16 karakter atau juga blok terakhir tidak cukup 16 karakter maka sisanya dipadding atau ditambahkan karakter tertentu hingga cukup 16 karakter. Karakter tambahan dapat berupa bit kosong atau bit dengan nilai sisa bloknya. Konversi dari char ke hex dilakukan dengan mengambil nilai ASCII karakter dengan perintah ord(‘karakter’), hasilnya berupa nilai desimal yang kemudian dapat diubah ke hex. Sebagai contoh, karakter ‘u’ dengan perintah ord(‘u’) diperoleh nilai 117, nilai ini bila dikonversi ke hex menjadi 75. Nilai ASCII karakter juga dapat dilihat dari tabel ASCII.

   Untuk kata ‘unib’, tidak cukup 16 karakter, sehingga perlu ditambah bit dengan nilai sisa bloknya yaitu 12 (padding). Sehingga plainteks satu blok secara lengkap adalah seperti pada tabel C-3, Plainteks= ‘unib’+padding

   →756e69620c0c0c0c0c0c0c0c0c0c0c

   Blok dengan 16 karakter ini kemudian dibagi 4 sehingga menghasilkan 4 bagian dan masing - masing dikonversi ke tipe data word, seperti pada tabel 1.

Tabel 1Konversi Hex ke Word

Bagian	Hexadesimal	Word
1	756e6962	1970170210
2	0c0c0c0c	202116108
3	0c0c0c0c	202116108
4	0c0c0c0c	202116108

3. Penambahan *Round Key*

   Proses enkripsi AES 256 dilakukan sebanyak 14 *round*/putaran, setiap putaran membutuhkan Roundkey yang dihasilkan dari ekspansi kunci enkripsi.

   Roundkey pertama diperoleh dari konversi kunci/password ke hex. Kunci AES 256 terdiri dari 32 karakter atau 64 karakter setelah dikonversi menjadi hex, setelah dibagi 8 akan menghasilkan 8 bagian. Tiap bagian ini selanjutnya dikonversi ke word.

   Kunci = 58029eb6d2dd138b3da6ee4b2bb71d8c

   Kunci ini kemudian dikonversi ke hex seperti pada tabel 2

Tabel 2 Konversi ke Hex

No	Karakter	Desimal	Hexadesimal
1	5	53	35
2	8	56	38
3	0	48	30
4	2	50	32

Dan seterusnya sehingga menghasilkan :

Kunci = 35383032 39656236 64326464 31333862 33646136 65653462 32626237 31643863

Konversi ke word dapat dilihat pada Tabel 3 :

Tabel 3 Konversi ke Word

Bagian	Hexadesimal	Word
1	35383032	892874802
2	39656236	962945590
3	64326464	1681024100
4	31333862	825440354
5	33646136	862216502
6	65653462	1701131362
7	32626237	845308471
8	31643863	828651619

Kunci ini kemudian diekspansi sehingga dapat digunakan untuk 14 putaran AES 256. Adapun proses penambahan Roundkey adalah dengan menggunakan fungsi **XOR** antara plainteks dengan Roundkey yang bersesuaian. Hasilnya adalah seperti pada tabel 4.

Tabel 4 Round Key

Bagian	Word Plaintext	Word Roundkey	PT xor RK
1	1970170210	892874802	1079400784
2	202116108	962945590	896101946
3	202116108	1681024100	1748920424
4	202116108	825440354	1027552366

4. Proses Round AES 256

   Lakukan proses shiftRows + subWord + mixColumns + addRoundKey sebanyak 14 putaran, untuk 13 putaran dihasilkan:

   Round Ke-13
   Hasilnya
   adalah:

   Bagian 1 => -1356996027
   Bagian 2 => 676864778
   Bagian 3 => 60773198
   Bagian 4 => -473863477

5. SubWord

   Hasil dari ke-13 putaran kemudian di SubWord, yang hasilnya adalah:

   Bagian 1 => 2040830062
   Bagian 2 => 879371879
   Bagian 3 => 2078010671
   Bagian 4 => 293077535

6. ShiftRows dan AddRoundKey

   Hasil SubWord kemudian dilakukan shiftRowdan penambahan RoundKey kembali yang hasilnya adalah :

   Bagian 1 => -1561917226
   Bagian 2 => -201071109
   Bagian 3 => -749947920
   Bagian 4 => -644676683

7. Hasil Enkripsi AES

   Hasil Enkripsi adalah gabungan dari keempat bagian terakhir, yaitu :

   a2 e7 08 d6 f4 03 e5 fb d3 4c b3 f0 d9 93 03 b5

Setelah berhasil mengenkripsi pesan, selanjutnya pengguna akan mendekripsikan kembali pesan yang telah di enkripsi tersebut menjadi *plaintext* kembali agar bisa dibaca. Langkahnya sebagai berikut :

1. Input
   - Cipherteks
   = a2 e7 08 d6 f4 03 e5 fb d3 4c b3 f0 d9 93 03 b5
   - Password
   = md5('teknik')
   =58029eb6d2dd138b3da6ee4b2bb71d8c

2. Inisialisasi

Kemudian dibagi menjadi 4 bagian dan masing – masing dikonversi ke tipe data word, seperti pada Tabel 5.

Tabel 5 Konversi ke Word

Bagian	Hexadesimal	Word
1	a2e708d6	-1561917226
2	f403e5fb	-201071109
3	d34cb3f0	-749947920
4	d99303b5	-644676683

3. Penambahan Round Key

Proses dekripsi AES 256 juga dilakukan sebanyak 14 *round*/putaran, setiap putaran membutuhkan invRoundkey yang dihasilkan dari ekspansi kunci enkripsi.

InvRoundkey pertama diperoleh dari RoundKey pada proses enkripsi, dimana invRoundeKey pertama = RoundKey pertama, untuk invRoundKey berikutnya dilakukan invMixColumn.

Kunci ini kemudian diekspansi sehingga dapat digunakan untuk 14 putaran AES 256. Adapun proses penambahan invRoundkey juga dilakukan dengan menggunakan fungsi

XOR antara cipherteks dengan Roundkey yang bersesuaian. Pada proses dekripsi round dimulai dari 14, sehingga menggunakan invRoundKey yang ke-14. Hasilnya adalah seperti Tabel 5.

Tabel 5 InvRoundKey

Bagian	Word Cipherteks	Word invRoundkey	CT xor iRK
1	-1561917226	-611457591	2037050655
2	-201071109	-1059526763	886768238
3	-749947920	-1472975977	2071500903
4	-644676683	-935909990	295970351

4. Proses Round Dekripsi AES 256

Lakukan proses shiftRows invShiftRows + invSubBytes + invMixColumns + addRoundKey sebanyak 14 putaran, untuk 13 putaran dihasilkan :

Round Ke-1 Hasilnya adalah:

Bagian 1 => 167331231

Bagian 2 => -1766713261

Bagian 3 => 1165347712

Bagian 4 => 665952069

5. Lakukan invShiftRows + invSubWord + addRoundKey

Hasil round kemudian dilakukan invShiftRows + invSubWord + addRoundKey yang hasilnya :

Bagian 1 => 1970170210

Bagian 2 => 202116108

Bagian 3 => 202116108

Bagian 4 => 202116108

6. Hasil Dekripsi

Hasil Dekripsi adalah gabungan dari keempat bagian terakhir, dalam hexadesimal adalah

756e69620c0c0c0c0c0c0c0c0c0c0c0c,

dalam bentuk string adalah **unib.**

VI. KESIMPULAN

Berdasarkan hasil penelitian, pengujian, implementasi serta pembahasan mengenai Algoritma *Advanced Encryption Standard* dan Algoritma *Message Digest 5* dalam enkripsi dokumen berbasis *Website* sebagai berikut :

1. File yang dihasilkan oleh proses enkripsi bisa menjadi 2 kali lipat bahkan lebih dari ukuran dan jumlah karakter file aslinya, ini dikarenakan hasil enkripsi dibuat dalam hex. Satu karakter diwakili dua karakter hex, misalkan karakter “U” dalam hex menjadi “75” begitu juga karakter simbol dan spasi juga ada hex-nya. Jadi file hasil enkripsi bisa 2 kali bahkan lebih dari ukuran dan jumlah karakter file aslinya.

2. Dari hasil penelitian, telah dibuktikan bahwa ukuran file hasil enkripsi dan kebutuhan waktu proses dipengaruhi oleh ukuran file asli, namun tidak dipengaruhi oleh jenis format file. Semakin besar ukuran file asli maka semakin besar pula ukuran file hasil enkripsi dan kebutuhan waktu prosesnya.

VII. SARAN

Pada aplikasi ini ketika admin akan mendekripsikan dokumen, admin hanya menekan tombol dekripsi saja tanpa harus memasukkan *password* yang sesuai dengan kunci dokumen. Ada hal baik dan juga buruk dalam situasi seperti ini, hal baiknya adalah mengurangi manajemen kunci yang terlalu banyak untuk disimpan dan diingat oleh admin. Hal buruknya adalah ketika akun milik admin dibobol, maka semua data dokumen milik peserta akan dengan mudah diambil dan dibaca oleh pembobol tanpa harus mencari kunci yang sesuai dengan dokumen.

Maka dari itu, penelitian selanjutnya disarankan untuk mengatasi masalah manajemen kunci dokumen yang baik dan efisien untuk kedua belah pihak.

UCAPAN TERIMA KASIH

Terima kasih untuk Ibu Ernawati, S.T., M.Cs dan Bapak Aan Erlanshari, S.T., M.Eng yang telah meluangkan waktu untuk memberikan dukungan, bimbingan, motivasi, dan arahan dalam menyelesaikan jurnal rekrusif ini.

REFERENSI

[1] A. Kadir, "Pengenalan Sistem Informasi," 2006.
[2] A. Mulyanto, "Sistem Informasi : Konsep dan Aplikasinya," 2009.
[3] A,Nugroho, "Rational Rose untuk Permodelan Berorientasi Objek," 2005.
[4] A.S. A. Sampayya, "Keseimbangan Matematika Dalam Al Quran," 2007.
[5] Connolly, T., & Begg, C. "*Database Systems: A Practical Approach to Design, Implementation and Management"*. 1998.
[6] D.a. Rijmen, "AES submission document on Rijndael," 1998.
[7] E. Asliana, "The Procurement of Government Goods and Services in Indonesia," 2012.
[8] F.Widyaningrum, "Implementasi dan Analisis Aplikasi Transfer File Antar PC Menggunakan Algoritma RC4 128 BIT dan AES 128 BIT," 2008.
[9] Inayatullah, "Analisis Penerapan Algoritma MD5 Untuk Pengamanan Password," 2009.
[10] Jogiyanto, H. "*Analisis dan Perancangan Sistem Informasi"*. 2001.
[11] L.UNIB,"http://lpse.unib.ac.id/eproc/tentangkami : diakses pada tanggal 05 Desember 2015."
[12] Pender, T. A. (2002). *UML Weekend Crash Course.* Canada: Wiley Publishing, Inc.
[13] Rosa & Shalahuddin, "Metode Pengembangan Sistem," 2011.
[14] S.Bahri, "Implementasi Pengamanan Basis Data Menggunakan Metode Enkripsi MD5," 2012.
[15] T.T. N. Duc H. Nguyen, Tan N. Duong, Phong H. Pham, "Cryptanalysis of MD5 on GPU Cluster," 2008.
[16] V.Luisiana, "Implementasi Kriptografi Pada Fle Dokumen Menggunakan Algoritma AES-128," 2001.
[17] V.Yuniati, "Enkripsi Dan Dekripsi Dengan Algoritma AES 256 Untuk Semua Jenis File," 2009.