KOMBINASI ALGORITMA ADVANCED ENCRYPTION STANDARD (AES) DAN HASH UNTUK MENGIDENTIFIKASI KEASLIAN IJAZAH

Henki Bayu Seta[1)], Moh. Mulki Ridho [2)], Theresiawati[3)]

[1), 2)] Teknik Informatika UPN "Veteran"Jakarta
[3)]Manajemen Informatika UPN "Veteran"Jakarta
Jl RS Fatmawati, Pondok Labu, Jakarta,12450
Email : henkiseta@upnvj.ac.id[1)], kii.ankii@gmail.com[2)], theresiawati@upnvj.ac.id[3)]

Abstrak

Ijazah dapat dikatakan sebagai ijzah palsu jika ijazah yang dikeluarkan oleh perguruan tinggi atau institut yang tidak mempunyai akreditasi atau ijazah yang diperoleh tanpa melalui proses pembelajaran dengan benar. Untuk menghindari penggunaan ijazah palsu maka harus diperlukan suatu teknologi untuk proses pengecekan. Proses pengecekan dilakukan dengan mengaplikasikan metode enkripsi dan dekripsi menggunakan algoritma Advanced Encryption Standard (AES) sebagai keamanan dalam basis data dan algoritma hash pada Message Digest 5 (MD5) sebagai nilai integritas dari ijazah. Dalam penelitian ini, MD5 dapat menghasilkan sidik jari digital dari informasi ijazah yang ada sehingga mempunyai nilai integritas digital untuk membuktikan keaslian suatu ijazah tersebut. Akan tetapi nilai integritas ini belum cukup untuk mengamankan data ijazah. Oleh karena itu penelitian ini menerapkan algoritma AES sebagai keamanan untuk menjaga keamanan nilai integritas ini.

Kata kunci: *algoritma Advanced Encryption Standard, Message Digest 5, ijazah, keamanan, enkripsi, dekripsi.*

1. Pendahuluan

Menurut Saiful Azad dan AL-Sakib Khan Pathan (2015, hlm.184) bahwa MD5 dibagi menjadi 4 pengolahan blok tahap analog, yang disebut sebagai putaran, di mana setiap putaran terdiri dari 16 operasi serupa berdasarkan fungsi F non-linear, penambahan modular, dan rotasi kiri. Oleh karena ini, ada tiga jenis operasi di MD5 yaitu operasi bit-wise Boolean, penambahan modular, dan operasi pergeseran sirklus. Semua operasi yang sangat cepat pada mesin 32-bit, yang membuat MD5 cukup cepat. Pada penelitian sebelumnya yang berjudul "Studi Algoritma Rijndeal Dalam Sistem Keamanan Data" oleh Eko Satrio menyimpulkan bahwa AES sangat aman untuk melindungi data, karena panjang kunci dan variatif pada kunci AES bisa mencegah dari segala macam ancaman.

Penelitian ini akan mengaplikasikan metode enkripsi dan dekripsi menggunakan algoritma *Advanced Encryption Standard* (AES) dan algoritma hash pada Message Digest 5 (MD5) untuk mengidentifikasi keaslian ijazah.

2. Pembahasan

Contoh dengan kasus entry data ijazah pada aplikasi dimana semua data dijadikan 1 string dalam program :
Data ijazah: 'DN -30 Ma 123456789''moh.mulki ridho''2011' Panjang data : 44 karakter (termasuk spasi) atau 352 bit (termasuk spasi) sehingga hanya terbagi menjadi 1 blok karena kurang dari sama dengan 512 bit. Maka 1 blok data ijazah direorganisasi kedalam bilangan hexadecimal, sebagai berikut :

M0	=	'	D	N	(spasi)		=	2	7	4	4	4	E	2	0
M1	=	-	3	0	(spasi)		=	2	D	3	3	3	0	2	0
M2	=	M	a	(spasi)	1		=	4	D	6	1	2	0	3	1
M3	=	2	3	4	5		=	3	2	3	3	3	4	3	5
M4	=	6	7	8	9		=	3	6	3	7	3	8	3	9
M5	=	'	'	m	o		=	2	7	2	7	6	D	6	F
M6	=	h	.	m	u		=	6	8	2	E	6	D	7	5
M7	=	l	k	i	(spasi)		=	6	C	6	B	6	9	2	0
M8	=	r	i	d	h		=	7	2	6	9	6	4	6	8
M9	=	o	'	'	2		=	6	F	2	7	2	7	3	2
M10	=	0	1	1	'		=	3	0	3	1	3	1	2	7

Kemudian lakukan padding untuk melengkapi blok 448bit pada MD5 ini.

M11 = €	=	8	0	0	0	0	0	0	0
M12 =	=	0	0	0	0	0	0	0	0
M13 =	=	0	0	0	0	0	0	0	0

Untuk 64bit terakhir diisikan sebagai panjang teks agar totol 512bit.

M14 = D	=	4	4	0	0	0	0	0	0
M15 =	=	0	0	0	0	0	0	0	0

Kemudian diubah menjadi binary :

M0	=	0010 0111	0100 0100	0100 1110	0010 0000
M1	=	0010 1101	0011 0011	0011 0000	0010 0000
……					
M14	=	0100 0100	0000 0000	0000 0000	0000 0000
M15	=	0000 0000	0000 0000	0000 0000	0000 0000

Kemudian MD5 membutuhkan penyanggah (buffer) yang masing-masing memiliki panjang 32 bit, maka total panjang 128 bit, atau 32 x 4 bit. Dimana ada 4 penyangga dalam MD5, yaitu :

A = 0 1 2 3 4 5 6 7
B = 8 9 A B C D E F
C = F E D C B A 9 8
D = 7 6 5 4 3 2 1 0

MD5 melakukan 4 putaran dimana setiap purataran berbeda-beda nilainya tergantung nilai dari penyanggah (buffer) yang ada. Berikut adalah putran pada MD5.

Putaran ke-1 : F(X, Y, Z) = (X ^ Y) ∨ (~X ^ Z)
Putaran ke-2 : G(X, Y, Z) = (X ^ Z) ∨ (Y ^ ~Z)
Putaran ke-3 : H(X, Y, Z) = X ⊕ Y ⊕ Z
Putaran ke-4 : I(X, Y, Z) = Y ⊕ (X ∨ ~Z)

Berikut ini adalah perhitungan dalam metode Message Digets 5 dibagi pada jumlah putaranya dimana diurai menjadi 4 bit setiap blok byte dengan rumus operasi dasar. $a \leftarrow b + CLS_s(a+g(b,c,d)+X[k]+T[i])$.

Iterasi ke-1

Putaran ke-1 : F(B,C,D) = (B ^ C) ∨ (~B ^ D).

A = 0000 0001 0010 0011 0100 0101 0110 0111
B = 1000 1001 1010 1011 1100 1101 1110 1111
C = 1111 1110 1101 1100 1011 1010 1001 1000
D = 0111 0110 0101 0100 0011 0010 0001 0000

B = 1000 1001 1010 1011 1100 1101 1110 1111
C = 1111 1110 1101 1100 1011 1010 1001 1000 and

B and C = 1000 1000 1000 1000 1000 1000 1000 1000

~B = 0111 0110 0101 0100 0011 0010 0001 0000
D = 0111 0110 0101 0100 0011 0010 0001 0000 and

~B and D = 1000 1000 1000 1000 1000 1000 1000 1000

B and C = 0111 0110 0101 0100 0011 0010 0001 0000
~B and D = 0111 0110 0101 0100 0011 0010 0001 0000 or

F = 1111 1110 1101 1100 1011 1010 1001 1000

F = 1111 1110 1101 1100 1011 1010 1001 1000
A = 0000 0001 0010 0011 0100 0101 0110 0111 +

F + A = 1111 1111 1111 1111 1111 1111 1111 1111

F + A = 1111 1111 1111 1111 1111 1111 1111 1111
M[0] = 0010 0111 0100 0100 0100 1110 0010 0000 +

F+A+M[0]= 0010 0111 0100 0100 0100 1110 0011 1111

F+A+ M[0] = 0010 0111 1100 0100 0100 1110 0011 1110

T = 1101 0111 0110 1010 1010 0100 0111 1000 +

F+A+M[0]+T = 1111 1111 0010 1111 1111 0010 1011 0110

F+A+M[0]+T = 1111 1111 0010 1111 1111 0010 1011 0110 CLS-7

CLS-7 = 1001 0111 1111 1001 0101 1011 0111 1111

CLS-7 = 1001 0111 1111 1001 0101 1011 0111 1111
B =1000 1001 1010 1011 1100 1101 1110 1111 +

A = 0011 1101 1010 0101 0010 1101 0100 1110

Untuk iterasi 2 sampai 16 atau M[i] dengan i=[1,…,15] lakukan langkah diatas hingga mendapatkan ronde ke 16. Untuk iterasi ke 17 sampai 32 gunakan operasi pada fungsi G(b,c,d) untuk mendapatkan ronde ke 32 dengan M[i]. Untuk iterasi ke 33 sampai 48 gunakan operasi pada fungsi H(b,c,d) untuk mendapatkan ronde ke 48 dengan M[i]. Untuk iterasi ke 49 sampai 64 untuk mendapatkan ronde ke 64 dengan M[i]. Semua nilai penyanggah harus digeser secara sikuler, dengan rumus :

temp ← d, d←c, c←b, b←a, a←temp

Nilai untuk Ronde ke 1 sebelum dilakukan penggeseran.
A = 0011 1101 1010 0101 0010 1101 0100 1110 = 3 d a 5 2 d 4 e
B = 1000 1001 1010 1011 1100 1101 1110 1111 = 8 9 a b c d e f
C = 1111 1110 1101 1100 1011 1010 1001 1000 = f e d c b a 9 8
D = 0111 0110 0101 0100 0011 0010 0001 0000 = 7 6 5 4 3 2 1 0

Nilai untuk Ronde ke 1 setelah dilakukan penggeseran.
A = 0111 0110 0101 0100 0011 0010 0001 0000 = 7 6 5 4 3 2 1 0
B = 0011 1101 1010 0101 0010 1101 0100 1110 = 3 d a 5 2 d 4 e
C = 1000 1001 1010 1011 1100 1101 1110 1111 = 8 9 a b c d e f
D = 1111 1110 1101 1100 1011 1010 1001 1000 = f e d c b a 9 8

Lakukan terus iterasi sampai ke 64, maka rounde yang dihasilkan sama dengan 64. Maka hasil dari hash dengan MD5 adalah hasil ronde ke 64 dari setiap iterasi dan putaran fungsi.

Pada iterasi ke 64 atau hasil ronde [64] sebagai berikut.
A = 0111 0110 0101 0100 0011 0010 0001 0000 = 6 2 e 1 c c 4 b
B = 0011 1101 1010 0101 0010 1101 0100 1110 = 3 a 3 6 d 8 c c
C = 1000 1001 1010 1011 1100 1101 1110 1111 = 3 b d 5 7 0 0 c
D = 1111 1110 1101 1100 1011 1010 1001 1000 = d 0 8 9 f d 1 e

Maka, Data ijazah : 'DN -30 Ma 123456789''moh.mulki rdho''2011'

Hasil hash : 62e1cc4b3a36d8cc3bd5700cd089fd1e

Enkripsi pada Metode AES

Pada algoritma AES, terdapat jumlah ronde tergantung pada panjang kunci yang digunakan. Untuk kunci 128bit berarti jumlah ronde hanya 10, untuk 192bit jumlah ronde 12 dan 256bit jumlah ronde 14. Setiap ronde terdapat 4 jenis transformasi yaitu SubByte, ShiftRow, MixColumns dan AddRoundKey. Contoh kunci dan teks (hasil hash) yang akan digunakan pada program identifikasi ijazah sebagai berikut :

Kunci : KunciAES12345678
Panjang kunci : 16 Karakter atau 128 bit
Teks asli : 62e1cc4b3a36d8cc-3bd5700cd089fd1e
Panjang teks : 32 karakter sama dengan 256 bit (maka menjadi 2 state)

Sebelum melakukan perhitungan atau transformasi pada AES, blok state yang panjangnya 16 karakter atau 16 bit atau 4 word itu harus direorganisasi kedalam bentuk hexcadecimal. Contoh pada state 1 diatas sebagai berikut

State 1
$S_{0,0}$ = 6 = 36 Hex = 0011 0110 $S_{0,2}$ = 1 = 31 Hex = 0011 0001
$S_{1,0}$ = a = 61 Hex = 0110 0001 $S_{1,2}$ = 4 = 34 Hex = 0011 0100
$S_{2,0}$ = 1 = 31 Hex = 0011 0001 $S_{2,2}$ = d = 64 Hex = 0110 0100
$S_{3,0}$ = 0 = 30 Hex = 0011 0000 $S_{3,3}$ = d = 64 Hex = 0110 0100
$S_{0,1}$ = 3 = 33 Hex = 0011 0011 $S_{0,3}$ = 8 = 38 Hex = 0011 1000
$S_{1,1}$ = f = 66 Hex = 0110 0110 $S_{1,3}$ = 7 = 37 Hex = 0011 0111
$S_{2,1}$ = d = 64 Hex = 0000 0000 $S_{2,3}$ = 0 = 30 Hex = 0011 0000
$S_{2,3}$ = 3 = 33 Hex = 0011 0011 $S_{3,3}$ = a = 61 Hex = 0110 0001

State 2
$S_{0,0}$ = 2 = 32 Hex = 0011 0010 $S_{0,2}$ = b = 62 Hex = 0110 0010
$S_{1,0}$ = 1 = 31 Hex = 0011 0001 $S_{1,2}$ = 4 = 34 Hex = 0011 0100
$S_{2,0}$ = 7 = 37 Hex = 0011 0111 $S_{2,2}$ = 9 = 39 Hex = 0011 1001
$S_{3,3}$ = 9 = 39 Hex = 0011 1001 $S_{3,2}$ = 8 = 38 Hex = 0011 1000
$S_{0,1}$ = a = 61 Hex = 0110 0001 $S_{0,3}$ = 9 = 39 Hex = 0011 1001
$S_{1,1}$ = b = 62 Hex = 0110 0010 $S_{1,3}$ = c = 63 Hex = 0110 0011
$S_{1,2}$ = 5 = 35 Hex = 0011 0101 $S_{2,3}$ = 0 = 30 Hex = 0011 0000
$S_{1,3}$ = 6 = 36 Hex = 0011 0110 $S_{3,3}$ = 6 = 36 Hex = 0011 0110

State 1

6	c	3	d
2	c	a	8
E	4	3	c
1	b	6	c

State 2

3	7	d	f
b	0	0	d
d	0	8	l
5	c	9	e

Pada blok state diatas dibagi menjadi 2 bagian state, dimana state ini disebut dengan state asli. Satu state

beriisi nilai dengan 128bit dimana pada setiap blok memuat 8 bit (contoh 6 = 0011 0110 byte). Berikut ini reorganisasi pada state 1 ke dalam hexcadecimal.

Contoh state 1 asli

6	c	3	d
2	c	a	8
E	4	3	c
1	b	6	c

Setelah direorganisasi ke hexcadecimal

36	63	33	64
32	63	61	38
65	34	33	63
31	62	36	63

Penyandian AES membutuhkan kunci untuk setiap ronde transformasi. Kunci dibangkitkan atau diekspansi dari kunci asli. Untuk kunci AES 128bit diorganisir terlebih dahulu kedalam hexadecimal kemudian menjadi 4 word dan disalin ke word keluaran w. Pada bagian pertama kunci tidak diubah (w[0], w[1], w[2], w[3]). Berikut reorganisir kunci AES 128bit dengan kunci **KunciAES12345678**.

K_0 = K = 4B hex = 01001011 K_8 = 1 = 31 hex = 00110001
K_1 = u = 75 hex = 01110101 K_9 = 2 = 32 hex = 00110010
K_2 = n = 6E hex = 01101110 K_{10} = 3 = 33 hex = 00110011
K_3 = c = 63 hex = 01100011 K_{11} = 4 = 34 hex = 00110100
K_4 = i = 69 hex = 01101001 K_{12} = 5 = 35 hex = 00110101
K_5 = A = 41 hex = 01000001 K_{13} = 6 = 36 hex = 00110110
K_6 = E = 45 hex = 01000101 K_{14} = 7 = 37 hex = 00110111
K_7 = S = 53 hex = 01010011 K_{15} = 8 = 38 hex = 00111000

w_0 = 4B 75 65 63
w_1 = 69 41 45 53
w_2 = 31 32 33 34
w_3 = 35 36 37 38

sama dengan

4B	69	31	35
75	41	32	36
65	45	33	37
63	53	34	38

Selanjutnya untuk elemen keluaran w[i] dengan i{4,…,43}. Sebagai contoh untuk menghitung w[4] dengan menggunakan rumus berikut :

w[i] = w[i-4] ⊕ SubWord(RotWord(w=[i-1])) ⊕ RCon [i/4]
w[4] = w[0] ⊕ SubWord(RotWord(w[3])) ⊕ RC [1]
karena w[3] = 35 36 37 38

SubWord(RotWord(w[3]) = SubWord(RotWord(3536 37 38)
= SubWord(36 37 38 35)
= 05 9A 07 96

Maka, w[4] = w[0] ⊕ SubWord(RotWord(w[3])) ⊕ RC[1]
= 4B 75 6E 63 ⊕ 05 9A 07 96 ⊕ 01 00 00 00
= **4F EE 67 D4**

Pada rumus diatas, untuk RotWord (35 36 37 38) hanya melakukan pertukaran, dimana nilai 35 digeser ke baris w paling akhir. Maka nilai RotWord(35 36 37 38) menjadi (36 37 38 35). Untuk SubWord(36 37 38 35) nilai 36 37 38 35 disubtitusikan dengan tabel subtitusi untuk transformasi SubBytes maka hasilnya 05 9A 07 96. Untuk perhitungan XOR dapat digambarkan sebagai berikut.

Tabel 1. Contoh Perhitungan 4B ⊕ 05 ⊕ 01 pada Ekspansi Kunci AES

4B	05	01	4B ⊕ 05 ⊕ 01
0	0	0	0
1	0	0	1
0	0	0	0
0	0	0	0
1	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

Jadi hasil dari 4B ⊕ 05 ⊕ 01 = 0100 1111 = 4F

Tabel 2. Contoh Perhitungan 75 ⊕ 9A ⊕ 00 pada Ekspansi Kunci AES

75	9A	00	75 ⊕ 9A ⊕ 00
0	1	0	1
1	0	0	1
1	0	0	1
1	1	0	0
0	1	0	1
1	0	0	1
0	1	0	1
1	0	1	0

Jadi hasil dari ⊕ 9A ⊕ 00 = 1110 1110 = EE

Tabel 3. Contoh Perhitungan 6E ⊕ 07 ⊕ 00 pada Ekspansi Kunci AES

6E	07	00	6E ⊕ 07 ⊕ 00
0	0	0	0
1	0	0	1
1	0	0	1
0	0	0	0
0	0	0	0
1	1	0	1
0	1	0	1
1	1	1	1

Jadi hasil dari 6E ⊕ 07 ⊕ 00 = 0110 0111 = 67

Tabel 4. Contoh Perhitungan 63 ⊕ 96 ⊕ 00 pada Ekspansi Kunci AES

63	96	00	63 ⊕ 96 ⊕ 00
0	1	0	1
1	0	0	1
1	0	0	0
0	1	0	1
0	0	0	0
0	1	0	1
1	1	0	0
1	0	1	0

Jadi hasil dari 63 ⊕ 96 ⊕ 00 = 1101 0100 = D4

Untuk w[i] dengan i {5,…,43} dapat dihitung dengan rumus yang sama dengan ekspandi kunci diatas, akan tetapi untuk menentukan RC dapat dilihat dari konstan RC dalam hexcadecimal. Ketika semua state asli telah direorganisasi kedalam bentuk hexadecimal (lampiran 1) dan kunci asli sudah diekspansi. Maka semua dimasukan dalam blok-blok state ukuran 4x4 untuk state. Untuk ekspansi kunci di bagi sesuai jumlah bit, untuk contoh kali ini menggunakan 128bit maka menjadi 4x4. Setelah proses ini selesai, lakukan langkah transformasi sebagai berikut:

a. SubBytes, Pada tahap ini transformasi dapat dilakukan dengan 2 cara, pada contoh kali ini hanya menggunakan tabel substitusi berikut transformasi SubBytes.

State 1 : hexcadecimal

36	63	33	64
32	63	61	38
65	34	33	63
31	64	36	63

Setelah transformasi SubBytes

05	FB	C3	43
23	FB	EF	07
4D	18	C3	FB
C7	43	05	FB

b. ShifRows, Pada tahap ini transformasi hanya menggeser nilai hexadecimal dari hasil transformasi SubBytes sebelumnya, berikut gambaran transformasi ShiftRows.

State 1 : transformasi SubByte

05	FB	C3	43
23	FB	EF	07
4D	18	C3	FB
C7	43	05	FB

Setelah transformasi ShiftRow

05	FB	C3	43
FB	EF	07	23
C3	FB	4D	18
FB	C7	43	05

c. MixColumns

Ubah blok state hasil dari transformasi ShiftRow kedalam bentuk matrik perkalian 4x4 dimana matrik tersebut diwakilkan atau sama dengan S.

$$S = \begin{pmatrix} 05 & FB & C3 & 43 \\ FB & EF & 07 & 23 \\ C3 & FB & 4D & 18 \\ FB & C7 & 43 & 05 \end{pmatrix}$$

Pada transformasi MixColumns untuk perhitungan melakukan perkalian antara dimana pada dibawah matrik pertama merupakan matrik ketentuan untuk kunci AES 128bit.

$$\begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \begin{pmatrix} 05 & FB & C3 & 43 \\ FB & EF & 07 & 23 \\ C3 & FB & 4D & 18 \\ FB & C7 & 43 & 05 \end{pmatrix} = \begin{pmatrix} S'_{0,0} & S'_{0,1} & S'_{0,2} & S'_{0,3} \\ S'_{1,0} & S'_{1,1} & S'_{1,2} & S'_{1,3} \\ S'_{2,0} & S'_{2,1} & S'_{2,2} & S'_{1,3} \\ S'_{3,0} & S'_{3,1} & S'_{3,2} & S'_{1,3} \end{pmatrix}$$

State untuk AES 128bit Hasil ShifRows Hasil MixColumns

Untuk proses perkalian matrik ini adalah :

1) Perkalian dengan 01 artinya tidak ada perubahan.
2) Perkalian dengan 02 artinya menggeser byte ke kiri.
3) Untuk perkalian dengan 03 artinya menggeser byte ke kiri dan melakukan XOR dengan nilai awal sebelum digeser. Setelah digeser, harus dilakukan XOR dengan 0x11B (0001 0001 1011) apabila hasil nilai yang digeser lebih besar daripada 0xFF (1111 1111).

Contoh perhitungan MixColumns :

$b_0 = 2_{a0} + 3_{a1} + 1_{a2} + 1_{a3}$
$b_1 = 1_{a0} + 2_{a1} + 3_{a2} + 1_{a3}$
$b_2 = 1_{a0} + 1_{a1} + 2_{a2} + 3_{a3}$
$b_3 = 3_{a0} + 1_{a1} + 1_{a2} + 2_{a3}$
dimana $a_0 = 0_5 = 0000\ 0101$
$a_1 = FB = 1111\ 1011$
$a_2 = C3 = 1100\ 0011$
$a_3 = 43 = 0100\ 0011$

$b_0 = 2_{a0} + 3_{a1} + 1_{a2} + 1_{a3}$
$b_0 = 2*0000\ 0101 + 3*1111\ 1011 + 1*1100\ 0011 + 1*0100\ 0011$
$b_0 = 2*0000\ 0101 + 3*1111\ 1011 + 1100\ 0011 + 0100\ 0011$
$b_0 = 0000\ 1010 + 0001\ 0110 + 1100\ 0011 + 0100\ 0011$
$b_0 = 0000\ 1010 \oplus 0001\ 0110 \oplus 1100\ 0011 \oplus 0100\ 0011$
$b_0 = 1001\ 1100 = 9C$

Jadi untuk nilai S'0,0 = 1001 1100 byte atau 9C bila diubah kedalam hexadecimal. Kemudian hitung semua materik yang ada, dengan perulangan yang sama hingga nilai S' atau hasil perkalian pada transformasi MixColumns terpenuhi semua.

$$\begin{pmatrix} 9C & S'_{0,1} & S'_{0,2} & S'_{0,3} \\ S'_{1,0} & S'_{1,1} & S'_{1,2} & S'_{1,3} \\ S'_{2,0} & S'_{2,1} & S'_{2,2} & S'_{1,3} \\ S'_{3,0} & S'_{3,1} & S'_{3,2} & S'_{1,3} \end{pmatrix}$$

d. AddRoundKey

Pada proses transformasi AddRoundKey telah dilakukan pada tahap pra-ronde atau sebelum masuk ronde ke-1. Dimana pra-ronde diisi dengan AddRoundKey operasi ekslusif OR dengan ekspansi kunci w[0], w[1], w[2], w[3]. Pada perhitungan AddRoundKey untuk ronde ke 1 sama dengan transformasi pada ronde ronde berikutnya. Perhitungan untuk transformasi AddRoundKey sangan simple karena hanya menggunakan XOR, berikut contoh perhitungannya pada ronde ke 1 :

$$\begin{pmatrix} 9C & X & X & X \\ X & X & X & X \\ X & X & X & X \\ X & X & X & X \end{pmatrix} \oplus \begin{pmatrix} 4F & X & X & X \\ EE & X & X & X \\ 67 & X & X & X \\ D4 & X & X & X \end{pmatrix} = \begin{pmatrix} S'_{0,0} & S'_{0,1} & S'_{0,2} & S'_{0,3} \\ S'_{1,0} & S'_{1,1} & S'_{1,2} & S'_{1,3} \\ S'_{2,0} & S'_{2,1} & S'_{2,2} & S'_{1,3} \\ S'_{3,0} & S'_{3,1} & S'_{3,2} & S'_{1,3} \end{pmatrix}$$

Hasil MixColumns Ekspansi Kunci Hasil XOR

Perhitungan AddRoundKey dengan XOR pada tabel dibawah ini dimana nilai hexadecimal diubah kedalam bentuk binary sebagai berikut.

9C	4F	9C ⊕ 4F
1	0	1
0	1	1
0	0	0
1	0	1
1	1	0
1	1	0
0	1	1
0	1	1

Contoh Perhitungan AddRoundKey

Maka nilai untuk $S'_{0,0}$ = 9C ⊕ 4F = 1101 0011 = D3 hex. Perhitungan pada AddRoundKey ini dihitung sampai semua state blok yang diubah ke dalam matrik 4x4 terpenuhi.

$$\begin{pmatrix} D3 & S'_{0,1} & S'_{0,2} & S'_{0,3} \\ S'_{1,0} & S'_{1,1} & S'_{1,2} & S'_{1,3} \\ S'_{2,0} & S'_{2,1} & S'_{2,2} & S'_{1,3} \\ S'_{3,0} & S'_{3,1} & S'_{3,2} & S'_{1,3} \end{pmatrix}$$

Ketika semua transformasi selesai melakukan perhitungan berarti untuk ronde pertama terpenuhi. Berikutnya untuk ronde ke 2 sampa dengan N_r-1 (untuk kunci 128bit hanya sampai ronde ke-10) perhitungan dan urutan untuk transformasi sama sampai teks asli menjadi chipper text.

Dekripsi Metode AES

Pada proses dekripsi dengan metode AES hampir sama dengan enkripsi metode AES. Persamaan pada jumlah ronde yang ada, jika enkripsi AES menggunakan kunci 128bit makan jumlah ronde adalah 10 maka jumlah ronde untuk dekripsi AES 10 ronde juga. Untuk transformasi yang ada pada tiap ronde dekripsi ada perbedaan urutan dengan transformasi pada enkripsi. Dimana pada proses enkripsi transformasi pada setiap ronde dimulai dari SubBytes, ShiftRows, MixColumns dan terakhir AddRoundKey.

Pada proses dekripsi urutan transformasi berbeda dimulai dengan InvShiftRows, InvSubBytes, AddRoundKey dan terakhir InvMixColumns. Untuk InvShiftRows, InvSubBytes dan InvMixColumns merupakan transformasi yang bersifat self-invers dengan syarat menggunakan kunci yang sama. Untuk state input merupakan chipper text dan hasil keluaran adalah state teks asli dengan ketentuan sebagai berikut, misal : Kunci : KunciAES12345678

Teks Sandi :
J/6mHMWZglZ8GgHHDpPMuMWeVmH+sRly
VD6E1AFUhNIltxVRhLEmcQ==

Untuk teks sandi harus direorganisasi kedalam bentuk hexadecimal kembali. Dan dibagi menjadi blok-blok state dengan panajang 128bit.

State sandi 1

$S_{0,0}$ = J = 4A Hex = 0100 1010 $S_{0,2}$ = g = 67 Hex = 0110 0111

$S_{1,0}$ = / = 2F Hex = 0010 1111 $S_{1,2}$ = l = 6C Hex = 0011 0100
$S_{2,0}$ = 6 = 36 Hex = 0011 0111 $S_{2,2}$ = Z = 5A Hex = 0101 1010
$S_{3,0}$ = m = 6D Hex = 0110 1101 $S_{3,3}$ = 8 = 38 Hex = 0011 1000
$S_{0,1}$ = H = 48 Hex = 0100 1000 $S_{0,3}$ = G = 47 Hex = 0100 0111
$S_{1,1}$ = M = 4D Hex = 0100 1101 $S_{1,3}$ = g = 67 Hex = 0110 0111
$S_{2,1}$ = W = 57 Hex = 0101 0111 $S_{2,3}$ = H = 48 Hex = 0100 1000
$S_{2,3}$ = Z = 5A Hex = 0101 1010 $S_{3,3}$ = H = 48 Hex = 0100 1000

Setelah dilakukan proses reorganisasi kemudia teks sandi dimasukan dalam blok state sandi seperti berikut ini :

State sandi

J	H	g	G
/	M	l	g
6	W	Z	H
m	Z	8	H

Setelah direorganisasi ke hexcadecimal

4A	48	67	47
2F	4D	6C	67
36	57	5A	48
6D	5A	38	48

Proses dekripsi AES juga membutuhkan kunci untuk setiap ronde transformasi. Untuk kunci pada dekripsi sama dengan kunci yang digunakan pada proses enkripsi, akan tetapi untuk pra-ronde ekspansi kunci yang digunakan adalah w[Nr*4…Nr*4+3] atau sama dengan w[40], w[41], w[42], w[43] untuk ekspansi sebanyak 10 ronde atau dengan kunci 128bit.

Misal,
w[40] = 12 AA 56 7B
w[41] = B3 27 E3 F0
w[42] = 9A 2C 49 8D
w[43] = 29 62 95 9F

Ketika state sandi telah direorganisasi kedalam bentuk hexadecimal (gambar 1, hlm.13) dan kunci asli sudah diekspansi. Maka semua dimaksudakn dalam blok-blok state ukuran 4x4 untuk state. Setelah proses ini selesai, maka lakukan langkah transformasi sebagai berikut :

a. InvShiftRows

Pada transformasi invers terhadap ShiftRows atau disebut dengan InvShiftRows hampir sama dengan transformasi ShiftRows dimana penggeseran berdasarkan indeks barisnya. Untuk InvShiftRows pergeseran berbeda arah dengan ShiftRows atau kebalikannya. Berikut ini contoh InvShiftRows

State sandi

4A	48	67	47
2F	4D	6C	67
36	57	5A	48
6D	5A	38	48

Setelah transformasi InvShiftRow

4A	48	67	47
67	2F	4D	6C
5A	48	36	57
5A	38	48	6D

b. InvSubByte

Pada transformasi ini sama dengan transformasi SubBytes dimana dilakukan proses subtitusi dengan tabel subtitusi. Akan tetapi pada InvSubBytes menggunakan tabel subtitusi pada transformasi. Berikut contoh transformasi InvSubBytes.

State sandi

4A	48	67	47
67	2F	4D	6C
5A	48	36	57
5A	38	48	6D

Setelah transformasi InvSubByte

5C	D4	0A	16
0A	4E	65	B4
46	D4	24	DA
46	76	D4	B3

c. AddRoundKey

Pada transformasi ini sama dengan transformasi yang ada pada proses enkripsi AES semua state blok dijadikan matrik sebagai berikut :

$$\begin{pmatrix} 5C & D4 & 0A & 16 \\ 0A & 4E & 65 & B4 \\ 46 & D4 & 24 & DA \\ 46 & 76 & D4 & B3 \end{pmatrix} \oplus \begin{pmatrix} 12 & B3 & 9A & 29 \\ AA & 27 & 2C & 62 \\ 56 & E3 & 49 & 95 \\ 7B & FO & 8D & 9F \end{pmatrix} = \begin{pmatrix} S'_{0,0} & S'_{0,1} & S'_{0,2} & S'_{0,3} \\ S'_{1,0} & S'_{1,1} & S'_{1,2} & S'_{1,3} \\ S'_{2,0} & S'_{2,1} & S'_{2,2} & S'_{1,3} \\ S'_{3,0} & S'_{3,1} & S'_{3,2} & S'_{1,3} \end{pmatrix}$$

Hasil InvShiftRows Ekpansi Kunci Hasil XOR

Maka nilai untuk $S'_{0,0}$ = 5C $\oplus$ 4F = 0100 1110 = 4E hex. Perhitungan pada AddRoundKey ini dihitung sampai semua state blok yang diubah ke dalam matrik 4x4 terpenuhi.

d. InvMixColumns

Pada transformasi ini InvMixColumns mengunakan perkalian matriks antara sebuah konstan dengan state. Kostan yang dipakai InvMixColumns merupakan invers matriks konstan pada transformasi MixColumns.

$$\begin{pmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{pmatrix} \begin{pmatrix} S'_{0,0} & S'_{0,1} & S'_{0,2} & S'_{0,3} \\ S'_{1,0} & S'_{1,1} & S'_{1,2} & S'_{1,3} \\ S'_{2,0} & S'_{2,1} & S'_{2,2} & S'_{1,3} \\ S'_{3,0} & S'_{3,1} & S'_{3,2} & S'_{1,3} \end{pmatrix} = \begin{pmatrix} 4E & FB & C3 & 43 \\ A1 & EF & 07 & 23 \\ 10 & FB & 4D & 18 \\ 3D & C7 & 43 & 05 \end{pmatrix}$$

Jadi

$$\begin{pmatrix} S'_{0,0} & S'_{0,1} & S'_{0,2} & S'_{0,3} \\ S'_{1,0} & S'_{1,1} & S'_{1,2} & S'_{1,3} \\ S'_{2,0} & S'_{2,1} & S'_{2,2} & S'_{1,3} \\ S'_{3,0} & S'_{3,1} & S'_{3,2} & S'_{1,3} \end{pmatrix} = \begin{pmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{pmatrix} \begin{pmatrix} 4E & FB & C3 & 43 \\ A1 & EF & 07 & 23 \\ 10 & FB & 4D & 18 \\ 3D & C7 & 43 & 05 \end{pmatrix}$$

Maka nilai dari,

$b_0 = 0E_{a0} + 0B_{a1} + 0D_{a2} + 09_{a3}$
$b_1 = 09_{a0} + 0E_{a1} + 0B_{a2} + 0D_{a3}$
$b_2 = 0D_{a0} + 09_{a1} + 0E_{a2} + 0B_{a3}$
$b_3 = 0B1_{a0} + 0D_{a1} + 09_{a2} + 0E_{a3}$

dimana a_0 = 4E = 0100 1110
a_1 = A1 = 1010 0001
a_2 = 10 = 0001 0000
a_3 = 3D = 0011 1101

$b_0 = 0E_{a0} \oplus 0B_{a1} \oplus 0D_{a2} \oplus 09_{a3}$
= 0E*4E $\oplus$ OB*A1 $\oplus$ OD*10 $\oplus$ 09*3D
= 0000 1110*0100 1110 $\oplus$ 0000 1011*1010 0001 $\oplus$ 0000 1101 *0001 0000 $\oplus$ 0000 1001*0011 1101

Untuk menghitung perkalian di Rijndeal's field dimana 0E*4E sebagai berikut.

1) Tahap pertama diubah menjadi polynomial

0E = 0000 1110 $= 0(x^7)+0(x^6)+0(x^5)+0(x^4)+1(x^3)+1(x^2)+1(x)+0$
$= x^3+x^2+x$

4E = 0100 1110 $= 0(x^7)+1(x^6)+0(x^5)+0(x^4)+1(x^3)+1(x^2)+1(x)+0$
$= x^6+x^3+x^2+x$

2) Kemudian kalikan 2 polynomial

$(x^3+x^2+x)*(x^6+x^3+x^2+x) = (x^9+x^6+x^5+x^4) + (x^8+x^5+x^4+x^3) + (x^7+x^4+x^3+x^2)$

$= x^9+x^6+x^8+x^3+x^7+x^4+x^2$
$= x^9+x^8+x^7+x^6+x^4+x^3+x^2$

3) Kemudian polynomial dengan modulus rijndeal yaitu

$x^9+x^8+x^7+x^6+x^4+x^3+x^2 \mod x^8+x^4+x^3+x+1$

untuk melakukan perhitungan mod pada polynomial diubah jadi binary $x^9+x^8+x^7+x^6+x^4+x^3+x^2$ = 111101110

$x^8+x^4+x^3+x+1$ = 100011011

111101110 mod 100011011

^10001100

11110110

Maka perkalian dari 0E*4E = 1111 0110 = **F6**

Lanjutkan semua perkalian yang ada, OB*A1, OD*10, 09*3D. Ketika semua nilai sudah dikalikan

maka harus di XOR kan. Jika semua nilai hasil dari perkalian telah di XORkan maka nilai tersebuat merukaan hasil dari $S'_{0,0.}$ Ketika semuan transformasi selesai melakukan perhitungan berarti untuk ronde pertama terpenuhi. Berikutnya untuk ronde ke 2 sampa dengan N_r-1 perhitungan dan urutan untuk transformasi sama sampai chipper text atau teks sandi menjadi menjadi teks asli (text hash).

Pengujian Waktu dan Akurasi

Table 10. Pengujian Waktu

No	Data Ijazah			Uji Ke	Waktu	Rata-Rata Waktu
	No Ijazah	Nama	Tahun			
1	DN -30 Ma 123456789	moh.mulki ridho	2011	1	0.195s	0.172s
				2	0.238s	
				3	0.082s	
2	AN -18 Na 123456789	reza pahlevi	2012	1	0.038s	0.273s
				2	0.067s	
				3	0.110s	
3	ZA -40 Fa 123456789	haris fadilah	2010	1	0.032s	0.168s
				2	0.341s	
				3	0.132s	
4	HN -60 Fa 123456789	hadi nur faisal	2011	1	0.126s	0.087s
				2	0.038s	
				3	0.098s	
5	DN -50 Za 123456789	aditiya nugraha	2011	1	0.115s	0.078s
				2	0.068s	
				3	0.052s	

Pada pengujian nomor 1, dilihat bahwa proses pengidentifikasian ijazah hanya memerlukan waktu 0.172 detik. Pengujian kedua, bahwa proses identifikasi ijazah hanya memerlukan waktu 0.273 detik. Pada pengujian waktu diatas, untuk setiap identifikasi ijazah membutuhkan rata-rata waktu 0.156 detik.

Table 11. Pengujian Akurasi

NO	No Ijazah	Nama	Tahun	Hasil Hash
1	DN -30 Ma 123456789	moh.mulki ridho	2011	62e1cc4b3a36d8cc3bd5700cd089fd1e
	DN -30 Ma 123456789	moh.mulki ridho	2011	62e1cc4b3a36d8cc3bd5700cd089fd1e
	DN -30 MA 123456789	moh.mulki ridho	2011	7c6f5424945753517b87ebcd222545bc
	DN -30 Ma 123456789	**moh. mulki ridho**	2011	cc1940dfb5ed5b0b8c56a729e4184767
	DN -30 Ma 123456789	moh.mulki ridho	**2001**	3fbd2675cdad8dcda75ffb4c8e829ef0
	DN -30 MA 123456789	**moh. mulki ridho**	**2001**	cff5fcaa641a162ef75e1bc176d12ad5
2	AN -18 Na 123456789	reza pahlevi	2012	6b75521ac188515420fba558d3968a94
	AN -18 Na 123456789	reza pahlevi	2012	6b75521ac188515420fba558d3968a94
	AN -18 Na 1234567891	reza pahlevi	2012	4b500526acd72a0feca28c644bc9fbb7
	AN -18 Na 123456789	**reza pahlevii**	2012	1156d92da77a6283287bb25c56e45c48
	AN -18 Na 123456789	reza pahlevi	**20012**	53a9edb1f50dd080d55f1252b1f9ef74
	AN -18 Na 1234567891	**reza pahlevii**	**20012**	52c9e6ccb873be6ff581c845218efeb1
3	ZA -40 Fa 123456789	haris fadilah	2010	8670d92473d8f8d3940ac3f9838cf968
	ZA -40 Fa 123456789	haris fadilah	2010	8670d92473d8f8d3940ac3f9838cf968
	ZA -40Fa 123456789	haris fadilah	2010	83c6904856fd322ce2d1a34d79c86dd7
	ZA -40 Fa 123456789	**haris fadila**	2010	d02167697f6229a2c6904e2182bf8499
	ZA -40 Fa 123456789	haris fadilah	**201**	bee70b7c5bbb42633316f8e8a70d891d
	ZA -40Fa 123456789	**haris fadila**	**201**	9e0314ba2e647cf984a53cb3011fdba3

Pada pengujian pertama, dimana jika data ijazah yang di-input dengan benar maka mendapatkan nilai hash **62e1cc4b3a36d8cc3bd5700cd089fd1e** dengan panjang 128-bit atau 32 karakter. Ketika dilakukan pengujian dengan data ijazah yang sama maka hasil tetap sama yaitu **62e1cc4b3a36d8cc3bd5700cd089fd1e**. Ketika dilakukan pengujian dengan penulisan nomor ijazah yang berbeda menjadi **DN -30 MA 123456789** (**Ma** menjadi **MA**) maka nilai hash yang dihasilkan menjadi **7c6f5424945753517b87ebcd222545bc**. Dilihat dari 2 hasil hash membuktikan bahwa perubahan yang kecil pada pesan akan (dengan probabilitas lebih) menghasilkan hash yang benar-benar berbeda. Jadi akurasi pada MD5 sangat sensitif terhadap nila pesan yang akan di hash, jika berbeda 1 karakter saja dapat merubah hasil hashing pada MD5 ini.

3. Kesimpulan

Dengan menggunakan algoritma Message Digets 5 (MD5) dapat membuat sidik jari digital dari sebuah ijazah sehingga mempercepat proses identifikasi keaslian suatu ijazah. Sedikit kesalahan pada penulisan data ijazah dapat merubah hasil sidik jari digital yang ada, tingkat akurasi sidik jari yang dihasilkan oleh MD5 sangat sensitif. Untuk melindungi sidik jari digital ijazah yang terdapat pada basis data dengan mengimplementasikan algoritma Advance Encryption Standard (AES).

Daftar Pustaka

[1] Ariyus, Dony, *Pengantar ilmu kriptografi, teoari, analisis dan implementasi*, Yogyakarta : Andi Publisher, 2008.

[2] Ariyus, Dony, *Computer Security*, Yogyakarta : Andi Publisher, 2006.

[3] Azad, Saiful dan Pathan, Al-Sakib Khan, *Practical Cyrptography algorithms and Implementations Using C++*, Baco Raton : CRC Press, 2015.

[4] Kromodimoeljo, Sentot, *Teori dan aplikasi kriptografi*, SPK IT Consulting, 2009.

[5] Kashogi, E.Z. Adnan. (2006) ITB, Bandung, *Algorithm MessageDigest5(MD5).*[online].available http://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2006-2007/Makalah-2006.html

[6] Laksono, Eko Puji. (2014), *Analisis Komparasi Algoritma Kriptografi Antara Metode DES (Data Encrytion System) Dan AES (Advanced Encrytion System)*, UIN Sunan Kalijaga, Jogyakarta, [online]. available http://digilib.uin-suka.ac.id/13257/

[7] McClure, Stuart, dkk., *Hacking Serangan dan Pertahanannya*, Yogyakarta : Andi Publisher, 2010,.

[8] Moh. Mulki Ridho. (2016), Implementasi Algoritma Advanced Encryption Standars (AES) dan Hash untuk Identifikasi Keaslian Ijazah, Skripsi, UPN "Veteran" Jakarta.

[9] 6] Sadikin, Rifki, *Kriptografi untuk keamanan jaringan*, Yogyakarta : Andi Publisher, 2011.

[10] Sugiarti, Yuni, *Analisis & perancangan UML (Unified Modeling Language) Generated VB.6*, Yogyakarta : Graha Ilmu, 2013.

[11] Satria, eko. (2009), *Studi algoritma rijndael dalam sistem keamanan data,* Departemen Matematika : Fakultas Matematika dan Ilmu Pengetahuan alam, Oktober, pp 60.

Biodata Penulis

Henki Bayu Seta, memperoleh gelar Sarjana Komputer (S.Kom), Jurusan Teknik Informatika UPN "Veteran"Jakarta, lulus tahun 2005. Memperoleh gelar Magister Informasi Teknologi (MTI) Program Pasca Sarjana Magister Teknologi Informasi Universitas Indonesia, Jakarta, lulus tahun 2013.Saat ini menjadi Dosen di UPN "Veteran" Jakarta.

Moh. Mulki Ridho, memperoleh gelar Sarjana Komputer (S.Kom), Jurusan Teknik Informatika UPN "Veteran"Jakarta, lulus tahun 2016.

Theresiawati, memperoleh gelar Sarjana Komputer (S.Kom), Jurusan Teknik Informatika UPN "Veteran"Jakarta, lulus tahun 2005. Memperoleh gelar Magister Informasi Teknologi (MTI) Program Pasca Sarjana Magister Teknologi Informasi Universitas Indonesia, Jakarta, lulus tahun 2013. Saat ini menjadi Dosen di UPN "Veteran" Jakarta.