Sains dan Komputer (SAINTIKOM)
Vol.18, No.2, Agustus 2019, pp. 119~129
P-ISSN: 1978-6603 E-ISSN : 2615-3475

Implementasi Keamanan Data Gaji Karyawan Pada PT. Capella Medan Menggunakan Metode *Advanced Encryption Standard* (AES)

Jaka Prayudha, Saniman, Ishak
* Program Studi Sistem Informasi, STMIK Triguna Dharma

Article Info
Article history:
Received May 31th, 2019
Revised June 12th, 2019
Accepted Augs 08th, 2019

Keyword:
Kriptografi
Advanced Encryption Standard
Data Gaji
Keamanan Data

ABSTRACT
Gaji merupakan suatu hal yang sudah sangat pokok pada kegiatan finansial pada sebuah instansi perusahaan karena hal tersebut berpengaruh terhadap kinerja para karyawan. Data gaji merupakan salah satu data yang bersifat rahasia yang hanya dapat dilihat oleh pihak-pihak tertentu, seperti bendahara dan kepala kantor. PT. Capella Medan adalah salah satu perusahaan yang harus menjaga data gaji karyawannya agar tidak disalahgunakan atau dimanipulasi oleh orang-orang yang tidak berhak dan akan menimbulkan kerugian bagi perusahaan.
Dalam hal ini diperlukan sebuah sistem dalam pengamanan data yang dapat melakukan penyandian dan pengacakan sebuah informasi yang berbasis komputer. Pengamanan ini dilakukan dengan cara menerapkan sebuah algoritma kriptografi yang bertujuan untuk mengenkripsi dan dekripsi sebuah pesan text. Algoritma kriptografi yang digunakan adalah algoritma Advanced Encryption Standard.
Hasil pengujian menunjukkan bahwa sistem keamanan data gaji karyawan dapat mengamankan data gaji dengan sangat baik dan menghindari terjadinya penyalahgunaan atau manipulasi data oleh orang-orang yang tidak memiliki wewenang atas data tersebut.

Copyright © 2019 STMIK Triguna Dharma.
All rights reserved.

First Author
Nama :Jaka Prayudha
Kantor :STMIK Triguna Dharma
Program Studi :SistemInformasi
E-Mail :jakaprayudha3@gmail.com

1. PENDAHULUAN

PT. Capella Medan adalah salah satu perusahaan swasta yang bergerak dibidang otomotif di wilayah Sumatera Utara. Dalam PT. Capella Medan, gaji yang diterima karyawan sesuai UMR setiap bulannya, dan pemberian bonus diadakan tiap tahunnya sesuai disiplin dan semangat kerja karyawan, staff, dan mekanik. Data gaji karyawan merupakan salah satu data rahasia yang hanya dapat dikelola oleh bendahara atau kepala kantor. Oleh karena itu, pihak perusahaan berusaha untuk mengamankan data tersebut agar terhindar dari penyalahgunaan atau manipulasi data oleh orang-orang yang tidak berhak yang dapat mengakibatkan kerugian pada perusahaan.

Data gaji pada perusahaan ini yaitu *file text* dari *Microsoft Excel*, isi *file* ini akan di input pada aplikasi yang akan digunakan, akan rawan terjadinya penyalahgunaan dan manipulasi data yang akan menyebabkan kerugian bagi pihak perusahaan.

Dalam hal ini dibutuhkan sistem keamanan untuk mengamankan data gaji karyawan PT. Capella Medan. Keamanan pada sistem ini menggunakan ilmu kriptografi *modern* yang menggunakan penyandian dalam sistemnya, dan metode yang digunakan adalah metode *Advanced Encryption Standard*.

Proses dalam pengamanan data gaji karyawan ini yaitu dengan melakukan enkripsi dengan merubah data *text* atau pesan (*plaintext)* menjadi data sandi (*plaintext*), dan dekripsi adalah merubah data sandi (*plaintext*) menjadi data *text* atau pesan (*ciphertext*).

2. METODE PENELITIAN

2.1 Gaji

Menurut Mulyadi (dalam Utami, 2014) 'gaji dapat didefinisikan secara global sebagai pembayaran atas penyerahan jasa yang dibayarkan kepada karyawan yang memiliki jenjang jabatan manajer dan pada umumnya merupakan pembayaran atas penyerahan jasa yang dilakukan oleh karyawan pelaksana (bagian produksi) dan dibayarkan berdasarkan pada hari kerja, jam kerja, atau jumlah satuan produk yang telah dihasilkan oleh karyawan'.

2.2 Kriptografi

Kriptografi berasal dari bahasa Yunani, menurut bahasa dibagi menjadi dua kripto dan graphia, kripto berarti *secret* (rahasia) dan graphia berarti *writing* (tulisan). Menurut terminologinya, kriptografi adalah ilmu dan seni untuk menjaga keamanan pesan ketika pesan dikirim dari suatu tempat ketempat yang lain (Ariyus, 2018 : 9). Kata "seni" tersebut berasal dari fakta sejarah bahwa pada masa-masa awal sejarah kriptografi setiap orang mungkin mempunyai cara yang unik untuk merahasiakan pesannya.

Pengamanan pada data dilakukan untuk menjaga kerahasiaan informasi dan agar aman dari orang-orang yang tidak bertanggung jawab, maka dilakukanlah pengamanan data dengan menggunakan algortima kriptografi.

2.3 Advanced Encryption Standard

Advanced Encryption Standard merupakan algoritma kriptografi simetrik yang beroperasi dalam mode penyandian blok *(block cipher)* yang memproses blok data 128-bit dengan panjang kunci 128-bit, 192-bit, dan 256-bit. Beberapa mode operasi yang dapat diterapkan pada algoritma kriptografi penyandi blok AES di antaranya adalah Electronic Code Book (ECB), Cipher Block Chaining (CBC), Cipher Feedback (CFB), dan Output Feedback (OFB). Implementasi AES dengan mode operasi ECB, CBC, CFB, dan OFB tentu saja memiliki kelebihan dan kekurangan tertentu dalam aspek tingkat keamanan data. Algoritma kriptografi bernama Rijndael yang didesain oleh oleh Vincent Rijmen dan John Daemen asal Belgia keluar sebagai pemenang kontes

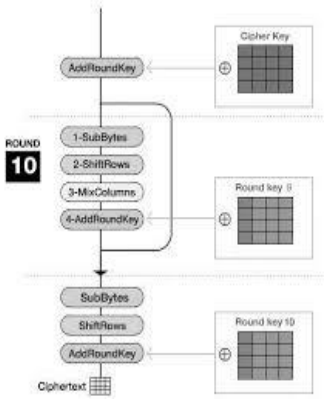
Algoritma kriptografi pengganti DES yang diadakan oleh NIST (National Institutes of Standards and Technology) milik pemerintah Amerika Serikat pada 26 November 2001. Algoritma Rijndael inilah yang kemudian dikenal dengan Advanced Encryption Standard (AES). Setelah mengalami beberapa proses standardisasi 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.

2.3.1 Proses Ekspansi Kunci

Proses Ekspansi Kunci Algoritma AES mengambil kunci cipher dan melakukan rutin ekspansi kunci untuk membentuk key schedule. Ekspansi kunci menghasilkan total Nb (Nr+1) word. Algoritma ini membutuhkan set awal key yang terdiri dari Nb word, dan setiap round Nr membutuhkan data kunci sebanyak Nb word. Hasil key schedule terdiri dari array 4 byte word linear yang dinotasikan dengan [wi]. SubWord adalah fungsi yang mengambil 4 byte word input dan mengaplikasikan SBox ke tiap-tiap data 4 byte untuk menghasilkan word output. Fungsi RotWord mengambil word [a0, a1, a2, a3] sebagai input, melakukan permutasi siklik, dan mengembalikan word [a1, a2, a3, a0]. Rcon[i] terdiri dari nilai-nilai yang diberikan oleh [xi-1, {00}, {00}, {00}], dengan xi-1 sebagai pangkat dari x (x dinotasikan sebagai {02}).

2.3.2 Proses Enkripsi

Proses enkripsi adalah proses penyandian *plaintext* menjadi *ciphertext*, atau pengubahan data menjadi bentuk rahasia. Proses enkripsi algoritma AES terdiri dari 4 jenis transformasi *bytes*, yaitu *SubBytes, ShiftRows, Mixcolumns*, dan *AddRoundKey*. 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*. Ilustrasi proses enkripsi AES dapat digambarkan seperti pada Gambar di bawah ini :



Gambar 1 Diagram enkripsi AES

Garis besar algoritma AES *Rijndael* yang beroperasi pada blok 128-bit dengan kunci 128bit (diluar proses pembangkitan *roundkey*) adalah sebagai berikut:

1. *AddRoundKey*, melakukan XOR antara awal (*plaintext*) dengan *cipher key*.
2. Putaran sebanyak Nr-1 kali. Proses yang dilakukan pada setiap putaran adalah :
   a. *SubBytes* adalah subsitusi *byte* menggunakan tabel subsitusi (S-Box).
   b. *ShiftRows* adalah pergeseran baris-baris *array state* secara *wrapping*.
   c. *MixColumns* adalah mengacak data di masing-masing kolom *array state*.
   d. *AddRoundKey* adalah melakukan XOR antara *state* sekarang *round key*.
3. *Final round*, proses untuk putaran terakhir :
   a. *SubBytes*
   b. *ShiftRows*
   c. *AddRoundKey*

Langkah kerja enkripsi adalah sebagai berikut:

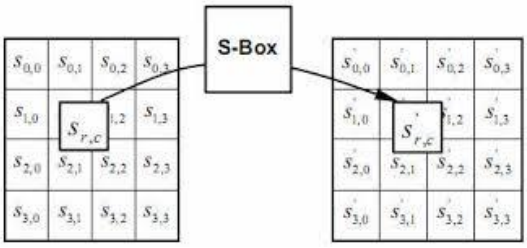
a. Transformasi *SubBytes*

SubBytes merupakan transformasi *byte* dimana setiap elemen pada state akan dipetakan dengan menggunakan sebuah tabel substitusi (S-Box). Tabel substitusi S-Box akan dipaparkan dalam Gambar 2.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

Gambar 2 S-Box Rijdael

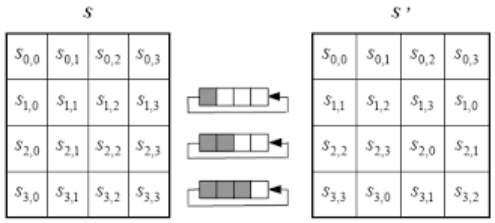
Untuk setiap *byte* pada *array state*, misalkan S[r, c] = xy, yang dalam hal ini xy adalah digit heksadesimal dari nilai S[r, c], maka nilai substitusinya, dinyatakan dengan S'[r, c], adalah elemen di dalam tabel subtitusi yang merupakan perpotongan baris x dengan kolom y. Gambar 3 mengilustrasikan pengaruh pemetaan *byte* pada setiap *byte* dalam *state*.



Gambar 3 Pengaruh pemetaan pada setiap *Byte* dalam *state*

b. *Shiftrows*

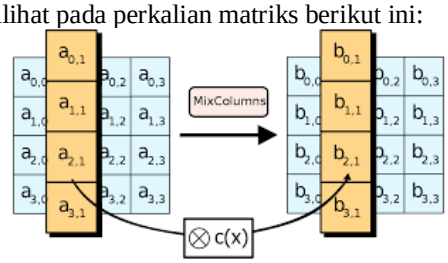
Transformasi *Shiftrows* pada dasarnya adalah proses pergeseran bit dimana bit paling kiri akan dipindahkan menjadi bit paling kanan (rotasi bit). Proses pergeseran *Shiftrows* ditunjukkan dalam Gambar 4 berikut:



Gambar 4 Proses shifrows

c. MixColumns

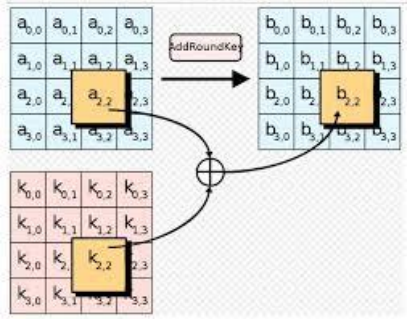
MixColumns mengoperasikan setiap elemen yang berada dalam satu kolom pada *state.* Secara lebih jelas, transformasi *mixcolumns* dapat dilihat pada perkalian matriks berikut ini:



Gambar 5 Proses *mixcolumns*

d. AddRoundKey

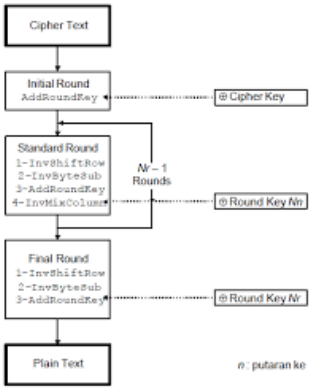
AddRoundKey melakukan XOR antara *state* sekarang dengan *round key*.



Gambar 6 Proses *addroundKey*

2.3.3 Proses Dekripsi

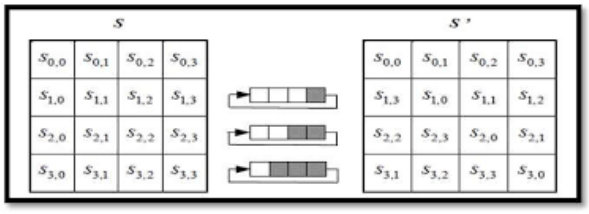
Transformasi *cipher* dapat dibalikkan dan diimplementasikan dalam arah yang berlawanan untuk menghasilkan *inverse cipher* yang mudah dipahami untuk algoritma AES. Transformasi *byte* yang digunakan pada *invers cipher* adalah *InvShiftRows*, *InvSubBytes*, *InvMixColumns*, dan *AddRoundKey*. Algoritma dekripsi dapat dilihat pada skema berikut ini :



Gambar 7 Proses dekripsi

a. *InvShiftRows*

InvShiftRows adalah transformasi *byte* yang berkebalikan dengan transformasi *ShiftRows*. Pada transformasi *InvShiftRows*, dilakukan pergeseran bit ke kanan sedangkan pada *ShiftRows* dilakukan pergeseran bit ke kiri. Ilustrasi transformasi *InvShiftRows* terdapat pada Gambar 8:



Gambar 8 Proses *invshiftrows*

b. *InvSubBytes*

InvSubBytes juga merupakan transformasi *bytes* yang berkebalikan dengan transformasi *SubBytes*. Pada *InvSubBytes*, tiap elemen pada *state* dipetakan dengan menggunakan tabel *Inverse* S-Box.

x \ y	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	52	09	6A	D5	30	36	A5	38	BF	40	A3	9E	81	F3	D7	FB
1	7C	E3	39	82	9B	2F	FF	87	34	8E	43	44	C4	DE	E9	CB
2	54	7B	94	32	A6	C2	23	3D	EE	4C	95	0B	42	FA	C3	4E
3	08	2E	A1	66	28	D9	24	B2	76	5B	A2	49	6D	8B	D1	25
4	72	F8	F6	64	86	68	98	16	D4	A4	5C	CC	5D	65	B6	92
5	6C	70	48	50	FD	ED	B9	DA	5E	15	46	57	A7	8D	9D	84
6	90	D8	AB	00	8C	BC	D3	0A	F7	E4	58	05	B8	B3	45	06
7	D0	2C	1E	8F	CA	3F	0F	02	C1	AF	BD	03	01	13	8A	6B
8	3A	91	11	41	4F	67	DC	EA	97	F2	CF	CE	F0	B4	E6	73
9	96	AC	74	22	E7	AD	35	85	E2	F9	37	E8	1C	75	DF	6E
a	47	F1	1A	71	1D	29	C5	89	6F	B7	62	0E	AA	18	BE	1B
b	FC	56	3E	4B	C6	D2	79	20	9A	DB	C0	FE	78	CD	5A	F4
c	1F	DD	A8	33	88	07	C7	31	B1	12	10	59	27	80	EC	5F
d	60	51	7F	A9	19	B5	4A	0D	2D	E5	7A	9F	93	C9	9C	EF
e	A0	E0	3B	4D	AE	2A	F5	B0	C8	EB	BB	3C	83	53	99	61
f	17	2B	04	7E	BA	77	D6	26	E1	69	14	63	55	21	0C	7D

Gambar 9 Proses *invsubbytes*

c. *InvMixColumns*

Setiap kolom dalam *state* dikalikan dengan matrik perkalian dalam AES. Perkalian dalam matrik dapat dituliskan:

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix}$$

d. *Inverse AddRoundKey*

Transformasi *Inverse AddRoundKey* tidak berbeda dengan transformasi *AddRoundkey* karna dalam transformasi ini hanya dilakukan operasi penambahan sederhana dengan operasi *bitwise* XOR.

3. ANALISA DAN HASIL

3.1 Algoritma Sistem

Adapun algoritma dalam metode *Advanced Encryption Standard* yang akan digunakan untuk menyelesaikan permasalahan adalah melakukan proses ekspansi kunci, enkripsi, dan dekripsi.

3.1.1 Proses Ekspansi Kunci

Kunci ronde (*round key*) dibutuhkan untuk proses enkripsi dan dekripsi pada algoritma *Advanced Encryption Standard*. Maksimal panjang kunci adalah sebanyak 16 digit dan jumlah kunci ronde yang diperlukankan adalah 10 kunci yang akan diperoleh dari proses ekspansi kunci. Pada kasus ini, kunci yang akan digunakan yaitu "Novithalis Lubis".

1. Urutkan kunci kedalam blok berukuran 128 bit (16 kode ASCII). Lalu ubah kunci kedalam bentuk *hexadecimal*.

N	O	v	i	t	h	a	l	i	s		L	u	b	i	s
4E	6F	76	69	74	68	61	6C	69	73	20	4C	75	62	69	73

2. Langkah selanjutnya yaitu susun kunci yang telah diubah kedalam bentuk hexadecimal ke dalam state berukuran 4 × 4 seperti dibawah ini :

4E	74	69	75
6F	68	73	62
76	61	20	69
69	6C	4C	73

Cipherkey / kunci ronde ke - 0

3. Setelah itu, untuk mendapatkan hasil kolom pertama pada sub kunci, langkah pertama yaitu dilakukan fungsi *RotWord* yaitu dengan menggeser setiap bit pada kolom ke-4 ke atas 1 kali dari kunci ronde ke-0.

75	62

62	*RotWord* →	69
69		73
73		75

4. Kemudian hasil dari *RotWord* disubtitusikan dengan nilai pada tabel S-*Box* (*SubBytes*).

62	→	AA
69	→	F9
73	→	8F
75	→	9D

5. Tahap akhir untuk mendapatkan kolom pertama yaitu proses XOR antara kolom pertama dari kunci ronde ke-0 dan hasil dari *SubBytes* lalu di-XOR-kan lagi dengan RC.

4E	⊕	AA	⊕	01	=	E5	}	kolom pertama (*wi*) pada kunci ronde selanjutnya (ronde ke-1) = K1
6F		F9		00		96		
76		8F		00		F9		
69		9D		00		F4		

6. Untuk mendapatkan kolom kedua, diperoleh dengan XOR antara *Wi* dengan kolom kedua dari kunci ronde ke-0. Untuk mendapatkan kolom ketiga dan keempat kunci ronde ke-1, dilakukan proses seperti memperoleh kolom kedua.

74	E5	91		69	91	F8		75	F8	8D						
68	96	FE		73	FE	8D		62	8D	EF						
61	⊕ F9	= 98		20	⊕ 98	= B8		69	⊕ B8	= D1						
6C	F4	98		4C	98	D4		73	D4	A7						
	Kolom ke-2				Kolom ke-3				Kolom ke-4							

7. Dari seluruh proses diatas, maka diperoleh kunci untuk ronde ke-1 yaitu :

E5	91	F8	8D
96	FE	8D	EF
F9	98	B8	D1
F4	98	D4	A7

Algoritma *Advanced Encryption Standard* 128 bit ini memiliki 10 ronde sehingga diperlukan 10 kunci ronde (*round key*). Untuk mendapatkan kunci ronde ke-2 sampai ke-10, proses diatas diulang sampai 10 kali. Kunci masing-masing ronde akan digunakan saat proses enkripsi dan dekripsi. Dibawah ini adalah hasil ekpansi kunci hingga ronde ke 10:

E5	91	F8	8D	38	A9	51	DC	…	1E	6C	F5	D3
96	FE	8D	EF	A8	56	D8	34		1F	89	AA	5B
F9	98	B8	D1	A5	3D	85	54		BB	85	9F	80
F4	98	D4	A7	A9	31	E5	42		B8	41	2A	71
Kunci Ronde Ke-1				Kunci Ronde Ke-2				…	Kunci Ronde Ke-10			

3.1.2 Proses Enkripsi

Plaintext yang akan digunakan yaitu "14750000". Urutkan kedalam blok dan ubah ke bilangan heksadesimal.

1	4	7	5	0	0	0	0								
31	34	37	35	30	30	30	30	20	20	20	20	20	20	20	20

Kemudian susun 16 *byte* pertama dari *plaintext* yang telah diubah ke bentuk heksadesimal kedalam *state* 4x4:

31	30	20	20
34	30	20	20
37	30	20	20
35	30	20	20

Lakukan XOR dengan kunci ronde ke-0. Proses ini dinamakan *AddRoundKey*.

31	30	20	20		4E	74	69	75		7F	44	49	55
34	30	20	20		6F	68	73	62		5B	58	53	42
37	30	20	20	⊕	76	61	20	69	=	41	51	00	49
35	30	20	20		69	6C	4C	73		5C	5C	6C	53

Proses diatas masih dalam *pra-round.* Hasil dari *AddRoundKey* diatas akan menjadi masukan untuk ronde ke-1 yang akan diproses dengan 4 transformasi yaitu *SubBytes*, *ShiftRows*, *MixColumns* dan *AddRoundKey*.

Round 1

Text

7F	44	49	55
5B	58	53	42
41	51	00	49
5C	5C	6C	53

SubBytes

D2	1B	3B	FC
39	6A	ED	2C
83	D1	63	3B
4A	4A	50	ED

ShiftRows

D2	1B	3B	FC
2C	39	6A	ED
63	3B	83	D1
ED	4A	4A	50

MixColumns

45	0C	01	4E
C2	6E	3B	05
14	8A	92	58
E3	BB	30	83

RoundKey 1

E5	91	F8	8D
96	FE	8D	EF
F9	98	B8	D1
F4	98	D4	A7

AddRoundKey

A0	9D	F9	C3
54	90	B6	EA
ED	12	2A	89
17	23	E4	24

Round 2

Text

A0	9D	F9	C3
54	90	B6	EA
ED	12	2A	89
17	23	E4	24

SubBytes

E0	5E	99	2E
20	60	4E	87
55	C9	E5	A7
F0	26	69	36

ShiftRows

E0	5E	99	2E
87	20	60	4E
E5	A7	55	C9
36	F0	26	69

MixColumns

A0	8B	FA	2E
F7	1C	80	9B
EC	20	39	52
0F	9E	C9	27

RoundKey 2

38	A9	51	DC
A8	56	D8	34
A5	3D	85	54
A9	31	E5	42

AddRoundKey

98	22	AB	F2
5F	4A	58	AF
49	1D	BC	06
A6	AF	2C	65

…..

Round 10

Text

29	08	F9	1E
57	24	A3	FB
64	18	2D	F2
93	4E	0B	B7

SubBytes

A5	30	99	72
5B	36	0A	0F
43	AD	D8	89
DC	2F	2B	A9

ShiftRows

A5	30	99	72
0F	5B	36	0A
D8	89	43	AD
A9	DC	2F	2B

RoundKey 10

1E	6C	F5	D3
1F	89	AA	5B
BB	85	9F	80
B8	41	2A	71

AddRoundKey

BB	5C	6C	A1
10	D2	9C	51
63	0C	DC	2D
11	9D	05	5A

CipherText

BB	5C	6C	A1
10	D2	9C	51
63	0C	DC	2D
11	9D	05	5A

Hasil dari proses *AddRoundKey* pada ronde ke-10 merupakan hasil akhir proses enkripsi yaitu: BB1063115CD20C9D6C9CDC05A1512D5A.

3.1.3 Proses Dekripsi

Proses-proses transformasi pada dekripsi dalam metode *Advanced Encryption Standard* yaitu *InvSubBytes, InvShiftRows, InvMixColumns* dan *AddRoundKey*. *AddRoundKey* merupakan transformasi yang bersifat *self-invers*. Kunci yang digunakan sama dengan yang digunakan pada proses enkripsi. Berikut adalah proses dekripsi dari hasil *ciphertext* yang telah diperoleh dari proses enkripsi sebelumnya.

Round 1

Chipertext

BB	5C	6C	A1
10	D2	9C	51
63	0C	DC	2D
11	9D	05	5A

Text

BB	5C	6C	A1
10	D2	9C	51
63	0C	DC	2D
11	9D	05	5A

RoundKey 10

1E	6C	F5	D3
1F	89	AA	5B
BB	85	9F	80
B8	41	2A	71

AddRoundKey

A5	30	99	72
0F	5B	36	0A
D8	89	43	AD
A9	DC	2F	2B

InvShiftRows

A5	30	99	72
5B	36	0A	0F
43	AD	D8	89
DC	2F	2B	A9

InvSubBytes

29	08	F9	1E
57	24	A3	FB
64	18	2D	F2
93	4E	0B	B7

Round 2

Text

29	08	F9	1E
57	24	A3	FB
64	18	2D	F2
93	4E	0B	B7

RoundKey 9

89	72	99	26
DF	96	23	F1
86	3E	1A	1F
47	F9	6B	5B

AddRoundKey

A0	7A	60	38
88	B2	80	0A
E2	26	37	ED
D4	B7	60	EC

InvMixColumns

0D	3D	84	95
D9	32	20	DA
73	D3	A6	26
B9	85	B5	5A

InvShiftRows

0D	3D	84	95
32	20	DA	D9
A6	26	73	D3
85	B5	5A	B9

InvSubBytes

F3	8B	4F	AD
A1	54	7A	E5
C5	23	8F	A9
67	D2	46	DB

…..

Round 10

Text

A0	9D	F9	C3
54	90	B6	EA
ED	12	2A	89
17	23	E4	24

RoundKey 1

E5	91	F8	8D
96	FE	8D	EF
F9	98	B8	D1
F4	98	D4	A7

AddRoundKey

45	0C	01	4E
C2	6E	3B	05
14	8A	92	58
E3	BB	30	83

InvMixColumns

D2	1B	3B	FC
2C	39	6A	ED
63	3B	83	D1
ED	4A	4A	50

InvShiftRows

D2	1B	3B	FC
39	6A	ED	2C
83	D1	63	3B
4A	4A	50	ED

InvSubBytes

7F	44	49	55
5B	58	53	42
5C	5C	6C	53
41	51	00	49

Kemudian melakukan 3 perhitungan terakhir, hasil dari *InvSubBytes* ronde ke-10 di-XOR-kan dengan *cipherkey* atau kunci ronde ke-0.

7F	44	49	55
5B	58	53	42
41	51	00	49
5C	5C	6C	53

$\oplus$

4E	74	69	75
6F	68	73	62
76	61	20	69
69	6C	4C	73

=

31	30	20	20
34	30	20	20
37	30	20	20
35	30	20	20

Selanjutnya adalah mengubah hasil dari *InvSubBytes* ronde ke-10 di-XOR-kan dengan *cipherkey* ke dalam bentuk bilangan desimal kemudian diubah lagi kedalam bentuk text berdasarkan kode ASCII.

31	34	37	35	30	30	30	30	20	20	20	20	20	20	20	20

PlainText

1	4	7	5	0	0	0	0								

4. IMPLEMENTASI DAN UJI COBA

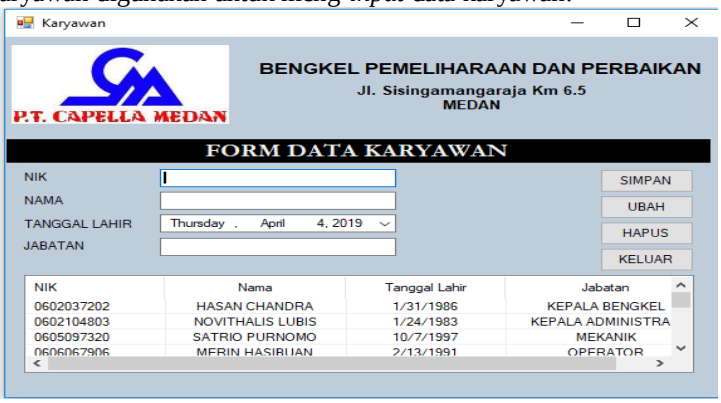
4.1 Form Menu Utama

Form Menu Utama merupakan form induk dari semua form yang ada pada sistem. Dimana terdapat 3 sub menu pada form Menu Utama yaitu File, Proses, Keluar. Berikut ini tampilan form Menu Utama:

Gambar 10 Tampilan *Form* Menu Utama

1. *Form* Data Karyawan

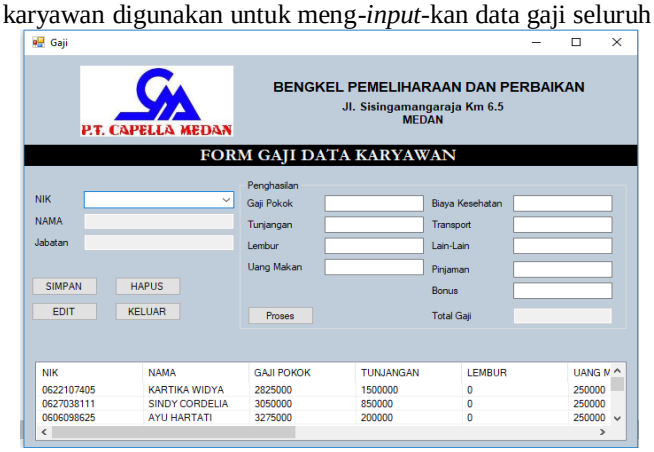
Form data karyawan digunakan untuk meng-*input* data karyawan.



Gambar 11 Tampilan *Form* Data Karyawan

2. *Form* Data Gaji Karyawan

Form data gaji karyawan digunakan untuk meng-*input*-kan data gaji seluruh karyawan.



Gambar 12 Tampilan *Form* Data Gaji karyawan

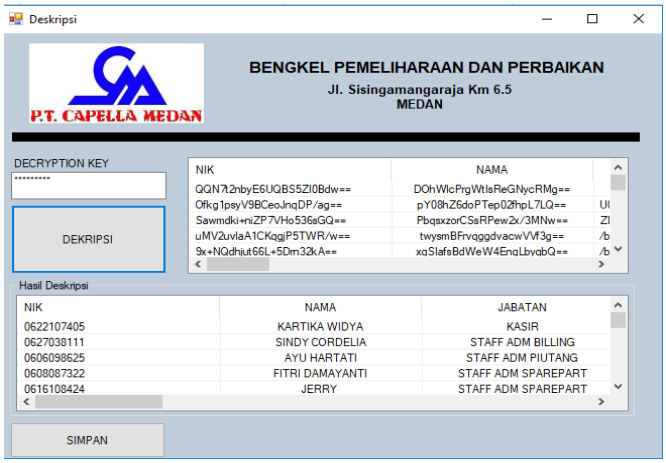
3. *Form* Enkripsi Data

Form enkripsi berisikan tentang file yang diproses ke algoritma AES dan komponen dalam program yang berbasis *desktop.*

Gambar 13 Tampilan *Form* Enkripsi Data

4. *Form* Dekripsi Data

Form dekripsi digunakan untuk mengembalikan data yang sebelumnya sudah dienkripsi agar dapat dibaca.



Gambar 14 Tampilan *Form* Dekripsi Data

5. KESIMPULAN

Kesimpulan yang dapat diambil dalam pembuatan sistem keamanan data gaji pada PT. Capella Medan dengan menggunakan metode *Advanced Encryption Standard* (AES) antara lain:

1. Perancangan sistem keamanan data gaji dapat dilakukan dengan menggunakan bahasa pemrograman berbasis *desktop* guna mempermudah proses mengamankan data gaji pada PT. Capella Medan.
2. Dalam mengamankan data gaji, metode *Advanced Encryption Standard* bekerja dengan sangat baik, data gaji mengalami beberapa tahap proses transformasi sehingga akhirnya menjadi teks yang tersandikan.
3. Penerapan metode *Advanced Encryption Standard* dalam mengamankan data gaji dapat mempermudah perusahaan dalam menjaga kerahasiaan data gaji karyawan dari orang yang tidak memiliki wewenang terhadap data tersebut.

DAFTAR PUSTAKA

[1] Candra, B., Wahyudi, J., & Hermawansyah. (2014). Pengembangan Sistem Keamanan Untuk Toko Online Pemrograman Php Dan Mysql. *Jurnal Media Infotama*, *11*(1), 31–39.

[2] Dony Ariyus. (2018). *Kriptografi Keamanan Data dan Komunikasi* (Pertama Ce). Yogyakarta: GRAHA ILMU.

[3] Dr.Suarga,M.Sc.,M.Math., P. . (2012). *Algoritma dan Pemrograman*. (ANDI OFFSET, Ed.). Yogyakarta.

[4] Hanafi, J. I., & Patombongi, A. (2016). Aplikasi Sms Kriptografi Menggunakan Metode Aes Berbasis Android, *1*(1), 69–75.

[5] Hendrayudi. (2011). *Dasar Dasar Pemrograman Microsoft Visual Basic 2008*. Bandung: CV YRAMA WIDYA.

[6] Informasi, S., Pada, P., Menengah, S., Pacitan, M., & Utami, R. (2014). Sistem Informasi Penggajian Pada Sekolah Menengah Atas (SMA) Muhammadiyah Pacitan Reny Utami, *6*(3), 32–35.

[7] Jiwandono, A., Bandung, I. T., & Bandung, J. G. (2011). Implementasi AES-ECB 128-bit untuk Komputasi Paralel pada GPU menggunakan Framework NVIDIA CUDA, 1–5.

[8] Keamanan, A. S. (2014). Pengembangan Sistem Keamanan Untuk Toko Online Pemrograman Php Dan Mysql, *11*(1).

[9] Lestari, D. (2014). Perancangan Sistem Informasi Penggajian Karyawan Pada PR. Tunas Mandiri Kabupaten Pacitan. *Perancangan Sistem Informasi Penggajian Karyawan Pada PR. Tunas Mandiri Kabupaten Pacitan*, *3*(4), 22–26.

[10] Mardiana, A. (2017). Jurnal Economix Volume 1 Nomor 1 Juni 2013, *1*(X), 11–22. https://doi.org/10.1016/j.neuroimage.2007.11.048

[11] Rifki Sadikin. (2012). *Kriptografi Untuk Keamanan Jaringn*. Yogyakarta: ANDI OFFSET.

[12] Rosa A.S M. Salahudin. (2013). *Rekayasa Perangkat Lunak*. (Informatika, Ed.). Bandung.

[13] Tulloh, A. R., Permanasari, Y., & Harahap, E. (2016). Kriptografi Advanced Encryption Standard (AES) Untuk Penyandian File Dokumen. *Jurnal Matematika UNISBA*, *2*(1), 118–125.

[14] Warkim, Lewenusa, I., & Karo, P. K. (2016). Kriptografi Algoritma Advanced Encryption Standard Dan Pengecekan Error Detection Cyclic Redundancy Check, (April).