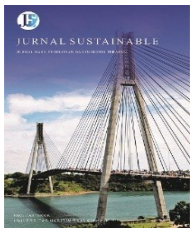


Jurnal Sustainable: Jurnal Hasil Penelitian dan Industri Terapan
Vol. 11, No. 01, hal. 34- 43, Mei 2022
Jurnal Sustainable: Jurnal Hasil Penelitian dan Industri Terapan
ISSN 2615-6334 (Online)
ISSN 2087-5347 (Print)

Penerapan Kombinasi Algoritma *Advanced Encryption Standard* (AES) dan Elgamal Dengan Fungsi *Secure Hash Algorithm* (SHA) Dalam Penyandian File Dokumen

Luqman Firdaus[1], Nurul Hayaty[2], Alena Uperiati[3]
[1,2]Jurusan Teknik Informatika, Fakultas Teknik, Universitas Maritim Raja Ali Haji
[1,2]Jl. Politeknik Senggarang, Tanjungpinang 29100
[*]Corresponding Author: fluqman803@gmail.com

Abstract— Documents or files are very important data because they are a source of information needed by anyone. The security of documents or files can be secured by cryptographic technique. The purpose of cryptographic is to disguise a message from another others and can only be opened by those who have the right to open the message. In this study, it combines a symmetric algorithm, namely the Advanced Encryption Standard (AES) algorithm and an asymmetric algorithm, namely the Elgamal algorithm with the addition of the SHA-256 hash function as data integrity, where the file will first be encrypted with the Elgamal algorithm then will be hashed with SHA-256 which aims for data integrity. Then the decryption process is carried out with the Elgamal algorithm and will then the be decrypted again by the Advanced Encryption Standard algorithm. The results of the test restore the file as before using the appropriate key and can also test data integrity if the data changes.
Keywords— *file, Advanced Encryption Standard, Elgamal, hash, SHA-256*

Intisari— Dokumen atau file merupakan data yang sangat penting karna merupakan sumber informasi yang dibutuhkan oleh siapa pun. Keamanan dokumen atau file bisa diamankan dengan teknik kriptografi. Tujuan dari kriptografi adalah menyamarkan sebuah pesan dari pihak lain dan hanya bisa dibuka oleh yang berhak membuka dari pesan tersebut. Pada penelitian ini mengkombinasikan antara algoritma simetri yaitu algoritma *Advanced Encryption Standard* (AES) dan algoritma asimetri yaitu algoritma Elgamal dengan ditambahkan dengan fungsi *hash* SHA-256 sebagai integritas data, dimana file akan terlebih dahulu di enkripsi dengan algoritma *Advanced Encryption Standard* (AES) dan selanjutnya akan dienkripsi dengan algoritma Elgamal lalu akan di *hash* dengan SHA-256 yang bertujuan untuk integritas data. Kemudian untuk proses dekripsi dilakukan dengan algoritma Elgamal dan selanjutnya akan didekripsi kembali oleh algoritma *Advanced Encryption Standard* (AES). Hasil dari pengujian mengembalikan file seperti semula dengan menggunakan kunci yang sesuai dan juga dapat menguji integritas data apabila data ada perubahan.
Kata kunci—File, Advanced Encryption Standard (AES), Elgamal, hash, SHA-256

ISSN 2615-6334 (Online)
ISSN 2087-5347 (Print)

I. PENDAHULUAN

Dokumen merupakan *file* atau data yang sangat penting karena merupakan sumber informasi yang diperlukan oleh suatu organisasi maupun perusahaan. Keamanan data sangatlah dibutuhkan pada saat ini mengingat teknologi yang semakin berkembang. Dampak negatif dalam perkembangan teknologi adalah adanya penyadapan dan pencurian data. Apabila tidak ada pengamanan yang kuat terhadap data-data tersebut, maka data tersebut bisa bocor ke publik sehingga bisa disalahgunakan oleh pihak yang tidak bertanggung jawab. Dari keamanan data-data tersebut, diperlukan sebuah sistem pengamanan data yang lebih baik agar terhindar dari ancaman yang ada. Ilmu yang mempelajari teknik-teknik pengamanan data adalah Kriptografi.

Kriptografi merupakan ilmu yang mempelajari teknik-teknik matematika yang berhubungan dengan aspek keamanan informasi [1]. Kriptografi menggunakan berbagai macam teknik dalam upaya untuk mengamankan data, untuk memenuhi hal tersebut dilakukan proses (enkripsi dan dekripsi) terhadap data yang akan dikirimkan [3] Salah satu algoritma kritografi adalah AES.

AES (*Advanced Encryption Standard*) merupakan algoritma kriptografi simetrik yang beroperasi pada *blok chipper* yang dapat mengenkripsi dan dekripsi informasi [1] Metode AES adalah algoritma *blok chipper* menggunakan teknik subtitusi, permutasi dan sejumlah putaran pada etiap blok yang akan dienkripsi dan dekripsi [4]. Kelemahan dari algoritma simetri adalah proses enkripsi dan dekripsi nya menggunakan kunci yang sama [2]. Oleh karena itu, digunakanlah algoritma asimetris yaitu Elgamal yang bertujuan untuk menutupi kelemahan dari AES dan juga untuk meningkatkan keamanan kunci.

Elgamal merupakan algoritma kriptografi asimetri yang berdasarkan konsep kunci publik [5]. Elgamal merupakan algoritma asimetris yang memerlukan waktu yang relatif lebih lama untuk melakukan proses enkripsi maupun dekripsi karena *chipertext* yang dihasilkannya berukuran dua kali dari ukuran semula [6]. Keamanan algoritma Elgamal terletak pada kesulitan perhitungan logaritma diskrit pada modulo prima yang besar, sehingga upaya untuk menyelesaikan masalah logaritma ini menjadi sulit dipecahkan [7]. Untuk mendukung beberapa aspek keamanan kriptografi yaitu autentikasi dan integritas data, penulis menambahkan satu metode yaitu menggunakan fungsi *hash* dengan SHA.

Ada banyak fungsi SHA, salah satu diantaranya adalah SHA-256. SHA-256 dibuat oleh NIST (*The National Institute of Standard and Technology*) pada tahun 2000. Keluaran dari SHA-256 adalah *hash value* yang memiliki panjang tetap sebesar 256 bit. Ada dua tahap penting dalam operasi SHA-256 yaitu *preprocessing* dan *hash computation* [8].

Berdasarkan uraian diatas, untuk meningkatkan keamanan data dalam penelitian ini, penulis mengambil judul Penerapan Kombinasi Algoritma AES dan Elgamal Dengan Fungsi Hash *Secure Hash Algorithm* (SHA) Dalam Penyandian *File* Dokumen.

II. LANDASAN TEORI

A. Kriptografi

Kriptografi adalah ilmu yang mempelajari teknik-teknik matematika yang berhubungan dengan aspek keamanan informasi, seperti kerahasiaan data, keabsahan data, integritas data, serta autentikasi data [2]. Pada tahun 80-an menyatakan kriptografi adalah ilmu dan seni untuk menjaga keamanan pesan. Kata seni dalam definisi ini berasal dari fakta sejarah bahwa pada awal sejarah kriptografi, setiap orang mempunyai cara yang unik untuk merahasiakan pesan. Kriptografi adalah suatu metode untuk melindungi suatu data atau informasi dengan menggunakan sandi, dimana sandi tersebut hanya bisa dimengerti oleh orang yang berhak menerima data atau informasi tersebut. Sedangkan tujuan kriptografi adalah melindungi data dari ancaman yang disengaja atau tidak disengaja.

B. Advanced Enryption Standard (AES)

Advanced Encryption Standard (AES) merupakan algoritma *cryptographic* yang dapat digunakan untuk mengamankan data. Algoritma AES adalah blok *chipertext* simetrik yang dapat mengenkripsi dan dekripsi informasi. Algoritma AES menggunakan kunci kriptografi 128, 192, dan 256 bit untuk mengenkrip dan dekrip data pada blok 128 bit. Pemilihan ukuran blok data dan kunci akan menentukan jumlah proses yang harus dilalui untuk proses enkripsi dan dekripsi [9].

Tabel 1. Jumlah Proses Enkripsi dan Dekripsi pada AES

Panjang Kunci Dalam Bit	Panjang Kunci (*Nk*) Dalam *words*	Ukuran Blok Dtaa (*Nb*) Dalam *words*	Jumlah proses (*Nr*)
128	4	4	10
192	6	4	12
256	8	4	14

1. Enkripsi AES

Enkripsi dilakukan bergantung pada sejumlah putaran dan didalam setiap putaran terdiri dari empat sub-proses [10] :

Proses pertama adalah *InitialRound* yaitu tahapan *AddRoundKey* yang melakukan XOR antara *state* plainteks dan *state key*. Lalu putaran sebanyak *Nr* – 1 (*Round Nr* – 1) proses yang dilakukan pada setiap putaran adalah :

a. Subbytes

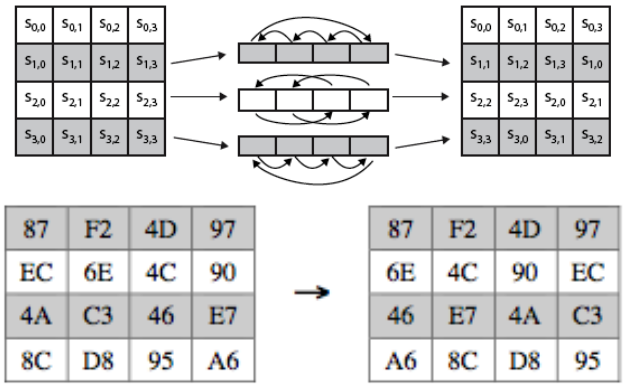
Proses subtitusi *bytes* dengan menggunakan table *S-Box*. Proses ini teragntung pada table *S-Box* untuk mengganti *byte* ke dalam *byte* lain [11].

Tabel 2. Tabel *S-Box*

hex	y=0	y=1	y=2	y=3	y=4	y=5	y=6	y=7	y=8	y=9	y=a	y=b	y=c	y=d	y=e	y=f
x=0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
x=1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
x=2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
x=3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
x=4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
x=5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
x=6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
x=7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
x=8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
x=9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
x=a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
x=b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
x=c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
x=d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
x=e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
x=f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

b. ShiftRow

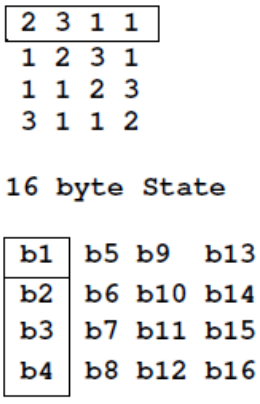
Proses menggeser *byte* secara ke kiri pada setiap baris [12].



Gambar 1. Transformasi *ShiftRow*

c. MixColumn

Setiap *byte* dari satu baris dalam transformasi matriks dikalikan dengan setiap nilai (*byte*) dari kolom tertentu [13].

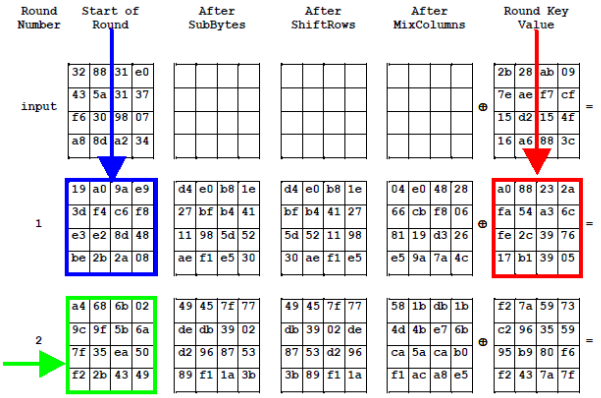


Gambar 2. Transformasi *MixColumn*

ISSN 2615-6334 (Online)
ISSN 2087-5347 (Print)

d. AddRoundKey

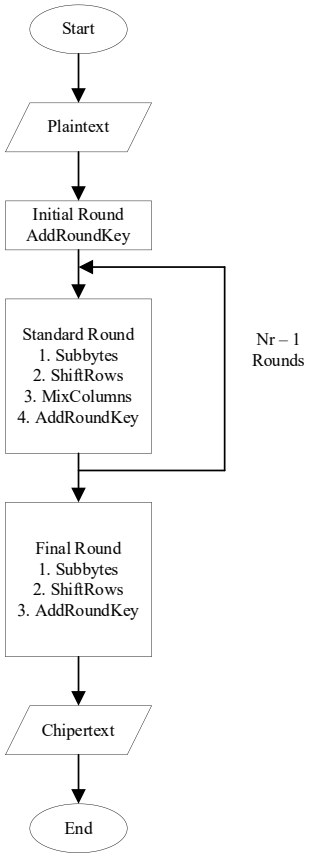
Pada proses ini melakukan proses XOR antara *state* sekarang dengan *state* dari kunci putaran [14].



Gambar 3. Transformasi *AddRoundKey*

Proses untuk putaran terakhir adalah :

a. Subbytes
b. ShiftRow
c. AddRoundKey



Gambar 4. Flowchart Enkripsi AES

a. Dekripsi AES

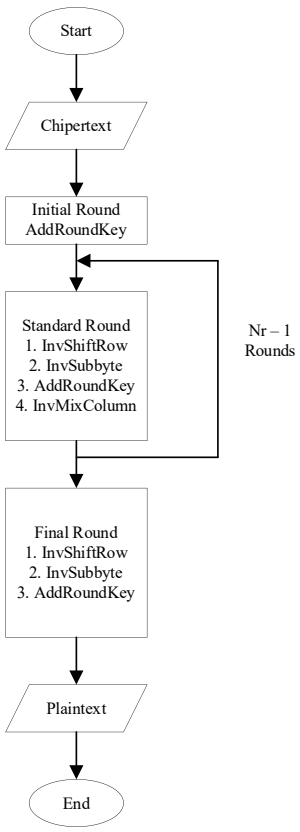
Proses dekripsi AES sama dengan urutan proses enkrispi AES dalam urutan terablik, pengirim dan penerima memiliki kunci yang sama untuk mengenkripsi dan dekripsi data [10] :

Proses pertama adalah *InitialRound* yaitu *AddRoundKey* yang melakukan XOR antara *state chipertext* dan *chiperkey*. Lalu putaran sebanyak *Nr – 1* proses yang dilakukan adalah :

a. InvShiftRow
b. InvSubbytes
c. AddRoundKey
d. InvMixColumn

Proses untuk putaran terakhir adalah :

a. InvShiftRow
b. InvSubbyte
c. AddRoundKey

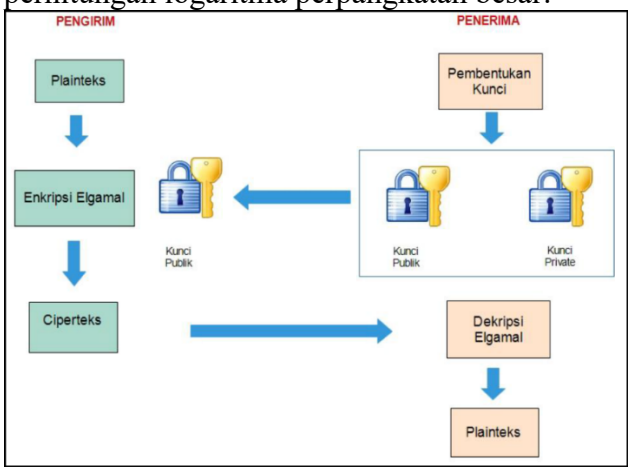


Gambar 5. Flowchart Dekripsi AES

C. Elgamal

Algoritma Elgamal ditemukan pada tahun 1985 oleh ilmuwan Mesir yaitu Taher Elgamal. Algoritma Elgamal merupakan algoritma berdasarkan konsep kunci publik. Algoritma ini pada umumnya digunakan untuk *digital signature,* namun kemudian dimodifikasi sehingga bisa digunakan untuk enkripsi dan dekripsi [5].

Keamanan algoritma Elgamal terletak pada kesulitan perhitungan logaritma diskrit pada modula prima yang besar, sehingga upaya untuk menyelesaikan masalah logaritma ini menjadi sulit untuk dipecahkan. Algoritma ini memiliki kelebihan yaitu pembangkitan kunci yang menggunakan logaritma diskrit dan metode enkripsi dan dekripsi yang menggunakan proses komputasi yang besar sehingga hasil enkripsi nya berukuran dua kali dari ukuran semula. Kekurangan algoritma ini adalah membutuhkan *resource* yang besar karena *chipertext* yang dihasilkan dua kali panjang *plaintext* serta membutuhkan *processor* yang mampu untuk melakukan komputasi yang besar untuk perhitungan logaritma perpangkatan besar.



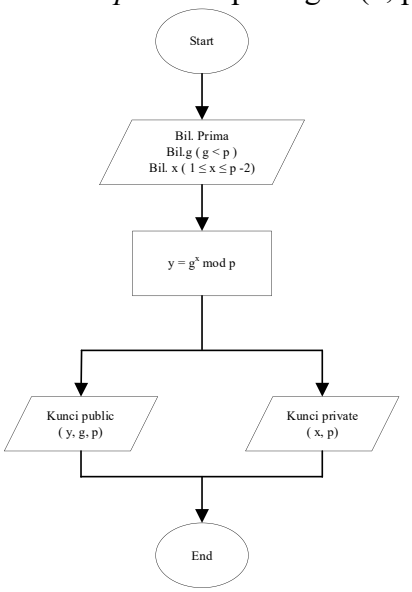
Gambar 6. Proses Algoritma Elgamal [2]

Algoritma Elgamal terdiri dari tiga buah proses, yaitu, proses pembentukan kunci, proses Enkripsi dan proses dekripsi [6].

1. *Proses Pembentukan Kunci*

Adapun proses enkripsi diawali dengan pembangkitan kunci secara acak yang menghasilkan beberapa bilangan diantaranya bilangan prima, bilangan acak pengirim, kunci *public* serta kunci *private.* Berikut merupakan algoritma pembangkitan kunci [15]:

a. Pilih sembarangan bilangan prima p
b. Pilih dua bilangan acak g dan x dengan syarat $g < p$ dan $1 \leq x \leq p - 2$
c. Hitung $y = g^x \bmod p$. Hasil dari perhitungan ini adalah :
   - Kunci *public* : *tripel* (y, g, p)
   - Kunci *private* : pasangan (x, p)



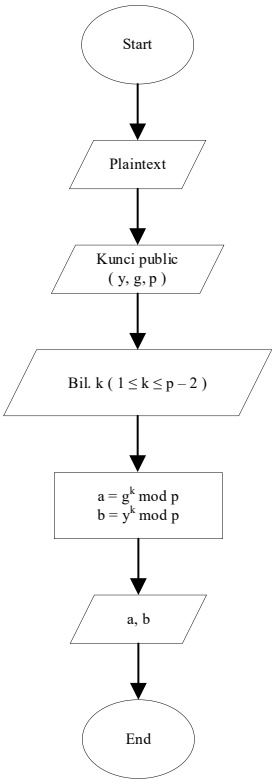
Gambar 7. Flowchart Pembentukan Kunci

2. *Proses Enkripsi*

Proses enkripsi Elgamal dilaksanakan menggunakan bilangan acak dan kunci *public* disertai dengan bilangan prima yang sudah ditentukan sebelumnya. Berikut merupakan algoritma Elgamal [15] :

ISSN 2615-6334 (Online)
ISSN 2087-5347 (Print)

1. Susun plaintext menjadi blok-blok *m1, m2, ...* (nilai setiap blok di dalam selang [0, p – 1].
2. Pilih bilangan acak k, yang dalam hal ini $1 \leq k \leq p - 2$.
3. Setiap blok m dienkripsi dengan rumus :
   $a = g^k \bmod p$; $b = y^k m \bmod p$
   Pasangan a dan b adalah *chipertext* untuk blok pesan m. Jadi, ukuran *chipertext* dua kali ukuran *plaintext* nya.

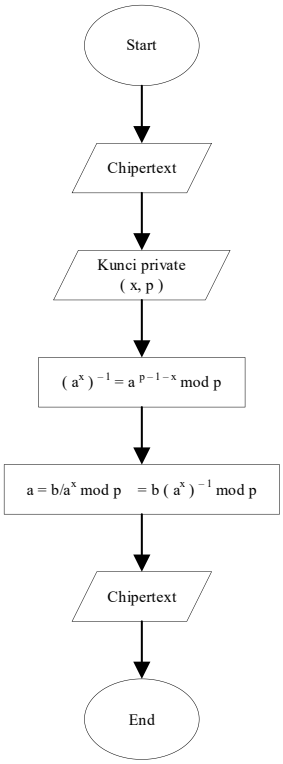


Gambar 8. Flowchart Enkripsi Elgamal

b. Proses Dekripsi Elgamal

Untuk melakukan proses dekripsi diawali dengan proses input *file* hasil enkripsi (*chipertext*), kunci *private* serta bilangan prima. Dengan beberapa variabel input tersebut, algoritma Elgamal akan memproses dekripsi *file* dengan sendirinya. Berikut merupakan algoritma dekripsi Elgamal [15] :

a. Gunakan kunci *private* x untuk menghitung $(a^x)^{-1} = a^{p-1-x} \bmod p$
b. Hitung *plaintext* dengan persamaan :
   $m = b/a^x \bmod p \quad = b\,(a^x)^{-1} \bmod p$



Gambar 9. Flowchart Dekripsi Elgamal

D. SHA-256

Fungsi *hash* SHA-256 merupakan versi SHA degan ukuran *digest* 256 pada versi SHA-2. SHA merupakan singkatan dari *Secure Hash Algorithm* yang merupakan fungsi *hash* satu arah yang dibuat oleh NIST (*National Institute of Standard and Technology*) [16].

SHA-256 menggunakan enam logika yang merupakan kombinasi dasar seperti AND, OR, XOR, pergeseran bit kekanan (*shift right*), dan rotasi bit kekanan (*rotate right*). Algoritma ini mengubah sebuah *message schedule* yang terdiri dari 64 element 32-bit *word,* delapan buah variabel 32-bit, dan variabel penyimpan nilai *hash* 8 buah *word* 32-bit [16].

1. Tahapan Proses SHA-256

SHA 256 mempunyai dua fungsi utama yaitu preprocessing dan hash computation. Preprocessing terdiri dari padding pesan, parsing pesan, set initial hash value. Hash computation terdiri dari persiapan message schedule, initialize variable, komputasi SHA 256, menjumlahkan variabel. Dari proses akan didapatkan keluaran berupa hash value yang panjangnya 256 bit [8].

Padding pesan dengan menambahkan bit-bit pengganjal sehingga total panjangnya 512 bit. *Padding* dilakukan dengan cara menambahkan bit 1 dan sisanya adalah bit 0 sejumlah *k.* Banyaknya tambahan bit pengganjal dapat dilihat dengan formula sebagai berikut [8] :

$$l + 1 + k = 448 \bmod 512$$

Dengan keterangan sebagai berikut :

l = Panjang pesan

k = Banyak bit 0 yang ditambahkan sebagai pengganjal

Diakhir pesan yang di *padding* tambahkan jumlah pesan yang telah dikonversi ke bit. Itulah akhir dari proses *padding* pesan. Proses berikutnya adalah melakukan *parsing* pesan.

Parsing pesan merupakan proses membentuk blok-blok pesan. Pesan yang di telah *padding* akan dikelompokkan ke dalam blok-blok pesan yang masing-masing memiliki 512 bit. Blok-blok pesan ini dikelompokkan ke dalam $M^{(0)}$, $M^{(1)}$, …, $M^{(n-1)}$. Kemudian masing-masing bloknya terdiri dari 32 bit, blok-blok pesan ini dinotasikan sebagai berikut, $M_0^{(i)}$, $M_1^{(i)}$, …, $M_{15}^{(i)}$. Proses *parsing* pesan selesai maka proses berikutnya adalah proses *set initial hash value.*

III. PENGUJIAN SISTEM

A. Perancangan Sistem

Pada pengujian ini menggunakan data yang terdiri dari beberapa file yaitu, .doc, .docx, .pdf, .xls, .xlsx dengan menggunakan kunci AES yang sama dan kunci Elgamal yang berpasangan. Pada pengujian ini akan menghasilkan berapa lama waktu proses yang dibutuhkan saat proses enkripsi dan dekripsi, besarnya ukuran file saat setelah proses enkripsi dan juga pengaruhnya berapa panjang dan besarnya kunci yang digunakan.

Tabel 2. Pengujian Sistem

File	Ukuran Awal	Panjang Kunci	Bil. Prima	Ukuran Akhir	Waktu Enkripsi	Waktu Dekripsi
.doc	100kb	128bit	257	936kb	00:17.95	00:12.45
.doc	500kb	128bit	257	4695kb	01:26.44	00:50.63
.docx	100kb	128bit	257	1039kb	00:18.95	00:18.05
.docx	500kb	128bit	257	5122kb	01:04.62	00:59.68
.xls	137kb	128bit	257	1296kb	00:17.33	00:02.49
.xls	657kb	128bit	257	6294kb	01:53.49	01:15.61
.xlsx	42kb	128bit	257	399kb	00:08.08	00:07.73
.xlsx	185kb	128bit	257	1708kb	00:35.60	00:07.98
.pdf	140kb	128bit	257	1332kb	00:24.03	00:08.17
.pdf	500kb	128bit	257	4376kb	01:16.31	01:07.72
.doc	100kb	128bit	547	997kb	01:18.48	00:04.00
.doc	500kb	128bit	547	4950kb	06:24.68	04:26.30
.docx	100kb	128bit	547	1104kb	00:59.53	01:00.74
.doc	500kb	128bit	547	5371k	07:36.3	05:36.3

ISSN 2615-6334 (Online)
ISSN 2087-5347 (Print)

x				b	5	4
.xls	137kb	128bit	547	1391kb	01:15.93	00:11.01
.xls	657kb	128bit	547	6662kb	06:57.29	04:28.71
.xlsx	42kb	128bit	547	423kb	00:26.97	00:14.72
.xlsx	185kb	128bit	547	1873kb	02:15.75	00:30.85
.pdf	140kb	128bit	547	1413kb	01:40.21	00:05.07
.pdf	500kb	128bit	547	4659kb	05:15.99	00:17.38
.doc	100kb	128bit	953	1016kb	02:45.14	01:14.09
.docx	100kb	128bit	953	1128kb	03:42.41	00:21.08
.xls	137kb	128bit	953	1417kb	04:27.22	02:51.87
.xlsx	42kb	128bit	953	433kb	01:08.62	00:29.04
.pdf	140kb	128bit	953	1446kb	04:23.23	04:25.00

Pada pengujian yang dilakukan diatas, dilihat pengujian dilakukan terhadap beberapa file. Pengujian dilakukan berdasarkan bilangan prima yang digunakan, mulai dari bilangan prima 257, 547, dan 953. Pada bilangan prima 257 terlihat bahwa waktu enkripsi terhadap file ukuran 100kb rata-rata waktu yang dibutuhkan adalah 20 detik, sedangkan waktu dekripsi yang dibutuhkan adalah 2 detik hingga 18 detik. Ukuran file .doc yang semua 100kb berubah menjadi 936kb, file .docx semula 100kb menjadi 1039kb, file .xls semula 137kb menjadi 1296kb, file .xlsx semula 42kb menjadi 399kb, file .pdf semula 140kb menjadi 1332kb. Pada file ukuran 500kb rata-rata waktu yang dibutuhkan adalah 1 menit, sedangkan waktu dekripsi yang dibutuhkan adalah 7 detik hingga 1 menit. Ukuran file .doc semula 500kb menjadi 4695kb, file .docx semula 500kb menjadi 5122kb, file .xls semua 657kb menjadi 6294kb, file .xlsx semula 185kb menjadi 1708kb, file .pdf semula 500kb menjadi 4376kb.

Pada bilangan prima 547 untuk ukuran file 100kb rata-rata waktu enkripsi yang dibutuhkan adalah 1 menit, sedangkan waktu dekripsi yang dibutuhkan adalah 4 detik hingga 1 menit. Ukuran file .doc semula 100kb menjadi 997kb, file .docx semula 100kb menjadi 1104kb, file .xls semula 137kb menjadi 1391kb, file .xlsx semula 42kb menjadi 423kb, file .pdf semula 140kb menjadi 1413kb. Pada file ukuran 500kb rata-rata waktu enkripsi yang dibutuhkan adalah 2 menit hingga 6 menit, sedangkan waktu dekripsi yang dibutuhkan adalah 17 detik hingga 5 menit. Ukuran file .doc semula 500kb menjadi 4950kb, file .docx semula 500kb menjadi 5371kb, file .xls semula 657kb menjadi 6662kb, file .xlsx semula 185kb menjad 1873kb, file .pdf semula 500kb menjadi 4659kb.

Pada bilangan prima 953 untuk ukuran file 100kb rata-rata waktu enkripsi yang dibutuhkan adalah 1 menit hingga 4 menit, sedangkan waktu dekripsi yang dibutuhkan adalah 29 detik hingga 4 menit. Ukuran file .doc semula 100kb menjadi 1016kb, file .docx semula 100kb menjadi 1182kb, file .xls semula 137kb menjadi 1417kb, file .xlsx semula 42kb menjadi 433kb, file .pdf semula 140kb menjadi 1446kb.

Dari pengujian yang telah dilakukan diatas dapat diambil kesimpulan bahwa semakin besarnya ukuran file dan besarnya bilangan prima, maka waktu yang dibutuhkan untuk proses enkripsi dan dekripsi lebih lama, dan juga ukuran file akan menjadi lebih besar.

B. Pengujian Kehandalan Sistem

Untuk menguji kehandalan sistem, penulis menggunakan metode *Brute Force* yang dimana pada metode ini mencari kemungkinan kunci yang ada. Pengujian dilakukan pada data dari kode ASCII yang *readable* sebanyak 95 data yang berupa alfanumerik dan tanda baca. Pengujian ini dilakukan pada kunci dengan

panjang 16 karakter [17] : $c = P_{16}^{95}$ (4)

Berdasarkan pada persamaan (4), pengujian juga menggunakan *key space* dari bilangan prima dengan batasan bilangan prima sampai dengan 1013. Total *key space* dari bilangan prima adalah 170.

$$C = \frac{95!+170!}{(95-16)!}$$

Berdasarkan persamaan diatas, maka :

$$c = \frac{265\ X\ 264\ X\ 263\ X\ 262\ X\ 261\ X\ 260\ X \ldots X\ 1}{79\ X\ 78\ X\ 77\ X\ 76\ X\ 75\ X\ 74\ X\ 73\ X \ldots X\ 1} = \frac{7{,}257E{+}306}{8{,}947E{+}116} = 8{,}112E + 189$$

Dari perhitungan diatas didapatkan bahwa kemungkinan kunci yang ada menggunakan *key space* bilangan prima berjumlah 170 adalah 8,112E+189 kombinasi.

Misalkan satu kombinasi kunci memerlukan waktu pemrosesan 0,14 detik, maka waktu yang dibutuhkan untuk memecahkan kunci adalah

t = 8,112E+189 x 0,14 detik = 1,135E+189 detik

Bila 1 hari = 24 jam, 24 jam = 1440 menit, 1440 menit = 86400 detik, dan 1 tahun = 365 hari, maka :

t = 1,135E+189 : 86400 detik = 1,314E+184 hari

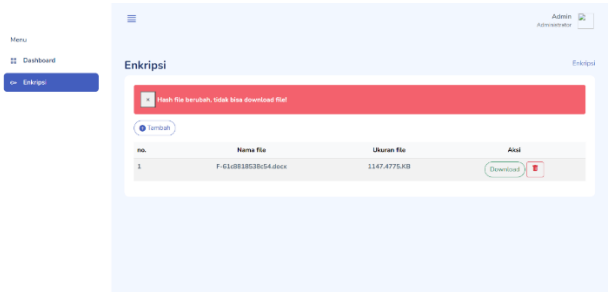
Sehingga,

t = 1,314E+184 x 365 hari = 4,797E+186 tahun

Dari perhitungan tersebut dapat disimpulkan bahwa waktu yang dibutuhkan oleh penyerang dengan menggunakan panjang kunci 16 dan *key space* bilangan prima 170 adalah 4,797E+186 tahun. Maka semakin besar bilangan prima yang digunakan, semakin lama juga waktu yang dibutuhkan oleh penyerang.

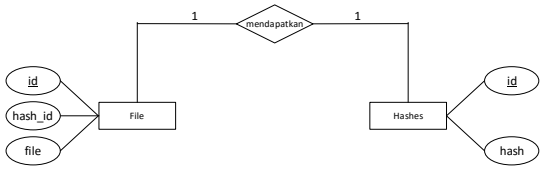
C. *Pengujian Fungsi Hash*

Pada pengujian ini dilakukan terhadap sistem yang dimana apabila file tersebut telah mengalami perubahan yang dilakukan terhadap pihak lain, maka file tersebut tidak bisa di download dan terdapat pesan bahwa file tersebut telah mengalami perubahan hasil *hash*. Ini menunjukkan bahwa file tersebut kemungkinan besar sudah berubah.



Gambar 10. Pengujian Hash

a. ERD



Gambar 10. ERD

IV. KESIMPULAN

Berdasarkan hasil penelitian dapat disimpulkan bahwa kombinasi algoritma *Advanced Encryption Standard* (AES) dan Elgamal Dengan Fungsi *Secure Hash Algorithm* (SHA) menghasilkan beberapa pengujian yaitu :

1. Pengujian Sistem
   Dengan dilakukannya percobaan terhadap beberapa file menghasilkan semakin besarnya ukuran file dan besarnya bilangan prima, maka proses enkripsi memakan waktu yang lama, dan juga ukuran file menjadi lebih besar dari ukuran semula.
2. Pengujian Kehandalan Sistem
   Dengan dilakukannya perhitungan menggunakan persamaan (4), menghasilkan kombinasi kunci sebesar 8,112E+189 kombinasi dengan 170 *key space* bilangan prima.
3. Pengujian *Hash*

ISSN 2615-6334 (Online)
ISSN 2087-5347 (Print)

Dengan dilakukannya percobaan terhadap file yang sudah dienkripsi, apabila terdapat perubahan atau penambahan 1 karakter didalam file, maka nilai hash akan berubah yang mendeteksi bahwa file sudah mengalami perubahan

REFERENSI

[1] Permana. A. A dan Nurnaningsih. D, Rancangan Aplikasi Pengamanan Data Dengan Algoritma Advanced Encryption Standard (AES), *Jurnal Teknik Informatika,* Vol. 11, No. 2, 2018.

[2] Widarma. A, Kombinasi Algoritma AES, RC4 dan Elgaml Dalam Skema Hybird Untuk Keamanan Data, *Journal Of Computer Engineering System And Science,* Vol. 1, No. 1, 2016

[3] Prasetyo. R dan Suryana. A, Aplikasi Pengamanan Data Dengan Teknik Algoritma Kriptografi AES dan Fungsi Hash SHA-1 Berbasis Desktop, *Jurnal SISFOKOM*, Vol. 5, No. 1, 2016.

[4] Nuari. R dan Ratama. N, Implementasi Algoritma Kriptografi AES (Advanced Encryption Standard) 128 Bit Untuk Pengamanan Dokumen Shipping, *Journal Of Artiffical Intelligence And Innovative Applications,* Vol. 1, No. 2, 2020.

[5] Fauzi. A, Maulita. Y, Sari. W, Analisa Algoritma Elgamal Dalam Penyandian Data Sebagai Keamanan Database, *Jurnal Informatika Kaputama*, Vol. 2, No. 1, 2018.

[6] Siddiq. F. R, Implementasi Sistem Kriptografi Hibrida Dengan Kombinasi Elgamal Cryptosystem Dan Spritz Cipher Untuk Pengamanan File Docx, *Skripsi*, 2019.

[7] Warnilah. A. I dan Nugraha. S. N, Komparasi Algoritma Kriptografi Elgamal Dan Caesar Cipher Untuk Enkripsi Dan Dekripsi Pesan, *Indonesian Journal On Computer And Information Technology,* Vol. 3, No. 2, 2018.

[8] Ginting. A. B. C, Perbandingan Algoritma Message Digest (MD5) Dan SHA-256 Pada Hashing File Dokumen, *Skripsi*, 2017.

[9] Kridalaksana. H. A, Astuti. F. I dan Pabokory. N. F, Implementasi Kriptografi Pengamanan Data Pada Pesan Teks, Isi File Dokumen, Dan File Dokumen Menggunakan Algoritma Advanced Encryption Standard (AES), *Jurnal Informatika Mulawarman,* 2015.

[10] Abdullah. M. A, Advanced Encryption Standard (AES) Algorithm To Encrypt And Decrypt Data, *Department Of Applied Mathematics And Computer Science*, 2017.

[11] Jain. R, Jejurkar. R, Chopade. S, Vaidya. S and Sanap. M, AES Algorithm Using 512Bit Key Implementation For Secure Communication, *Internation Journal Of Innovative Research In Computer And Communciation Engineering*, 2014.

[12] Selmane. N, Guilley. S and Danger. J. L, Practical Setup Time Violation Attacks On AES, *In Dependable Computing Conference,* 2008.

[13] Berent. A, Advanced Encryption Standard, *http://www.networkdls.com/Articles/AESbyExample.pdf*, 2013.

[14] Kretzschmar. U, AES128-AC Implementation For Ecncryption And Decryption, *TI-White Paper*, 2009.

[15] Saputro. A dan Karima. A, Pembangkitan Kunci Pada Algoritma Asimetri Elgamal Untuk Meningkatkan Keamanan Data Bertipe .docx, *Jurnal Ilmiah SISFOTENIKA*, Vol. 6, No. 2, 2016.

[16] Putri. M. D. R dan Sulastri. S, Implementasi Enkripsi Data Secure Hash Algorithm (SHA-256) Dan Message Digest Algorithm (MD5) Pada Proses Pengamanan Kata Sandi Sistem Penjadwalan Karyawan, *Jurnal Teknik Elektro*, Vol. 10, No. 2, 2018.

[17] Kriptosistem Kunci Asimetri Dan Pembangkitan Kunci Publik Menggunakan Jaringan Syaraf Tiruan Radial Basis Function, *Thesis*, 2016.