JURNAL TIMES
TECHNOLOGY INFORMATICS & COMPUTER SYSTEM

Volume X No 2, Desember 2021
*p*ISSN : 2337 – 3601
*e*ISSN : 2549 – 015X

Tersedia *online* di http://ejournal.stmik-time.ac.id

IMPLEMENTASI ALGORITMA AES-128 DAN SHA-256 DALAM PERANCANGAN APLIKASI PENGAMANAN FILE DOKUMEN

Herman[1], Robby Wijaya[2], Kenner Farandi[3], Satriya Miharja[4], Wilson[5]
Program Studi Teknik Informatika
STMIK TIME Medan
Jl. Merbabu No.32 AA-BB Medan 20212, Telp:061-4561932
email: herman@stmik-time.ac.id[1], robbywijaya@stmik-time.ac.id[2], satria@marcotania.com[4], wu95.wilson@gmail.com[5]

Abstrak

Keamanan file dokumen merupakan sebuah hal yang sangat penting bagi setiap orang ataupun instansi perusahaan pada saat ini. File dokumen biasanya berisikan informasi-informasi penting seseorang ataupun perusahaan baik itu laporan rahasia perusahaan, strategi bisnis rahasia perusahaan, ataupun riwayat perusahaan. Proses penjagaan dokumen hanya dijaga menggunakan proteksi kata sandi. Proteksi kata sandi bertujuan mencegah agar hanya user yang mengetahui kata sandi yang dapat masuk dan mengakses informasi di dalamnya. Namun penjagaan dokumen melalui autentikasi kata sandi memiliki kekurangan seperti mudah diserang dengan teknik Brute Force yaitu serangan yang dilakukan untuk membobol kata sandi dengan cara mencoba berbagai macam kombinasi kata sandi sampai akhirnya menemukan kata sandi yang tepat. Permasalahan tersebut perlu dibangun sebuah aplikasi pengamanan informasi dokumen dengan menerapkan algoritma kriptografi. Pada penelitian ini, akan dikombinasikan 2 algoritma kriptografi yaitu Advance Encryption Standard (AES) yang bernilai 128 dan Secure Hash Algorithm (SHA) yang bernilai 256 bit. Hasil penelitian menunjukkan bahwa proses enkripsi dan dekripsi file menggunakan algoritma AES-128 dan SHA-256 dapat dilakukan dengan cepat tergantung dari besar size dari file dokumen yang dienkripsi.

Kata Kunci: Keamanan File Dokumen, Kriptografi, Algortima Advance Encryption Standard, Algoritma Secure Hash Algorithm

1. Pendahuluan

Keamanan *file* dokumen merupakan sebuah hal yang sangat penting bagi setiap orang ataupun instansi perusahaan pada saat ini [1]. *File* dokumen biasanya berisikan informasi-informasi penting seseorang ataupun perusahaan baik itu laporan rahasia perusahaan, strategi bisnis rahasia perusahaan, ataupun riwayat perusahaan. Sebuah *file* dokumen yang rentan dicuri serta dibaca oleh orang yang tidak berkepentingan tentunya dapat berpotensi merugikan perusahaan ataupun kemungkinan informasi dokumen tersebut dapat diubah dan kemudian disalahgunakan. Proses penjagaan dokumen hanya dijaga menggunakan proteksi kata sandi. Proteksi kata sandi bertujuan mencegah agar hanya *user* yang mengetahui kata sandi yang dapat masuk dan mengakses informasi di dalamnya. Namun penjagaan dokumen melalui autentikasi kata sandi memiliki kekurangan seperti mudah diserang dengan teknik *Brute Force* yaitu serangan yang dilakukan untuk membobol kata sandi dengan cara mencoba berbagai macam kombinasi kata sandi sampai akhirnya menemukan kata sandi yang tepat. Selain serangan *Brute Force*, tentunya pengamanan *file* dengan metode kata sandi rentan dilihat oleh orang di sekitar sehingga dapat berpotensi kata sandi diketahui oleh orang yang tidak berkepentingan.

Permasalahan tersebut perlu dibangun sebuah aplikasi pengamanan informasi dokumen dengan menerapkan algoritma kriptografi. Kriptografi merupakan salah satu teknik yang digunakan untuk meningkatkan aspek keamanan suatu informasi. Kriptografi digunakan untuk mengubah pesan rahasia yang dapat dimengerti menjadi sebuah pesan yang tidak dapat dimengerti. Pesan yang akan dirahasiakan sebelum disamarkan disebut *plaintext*, sedangkan pesan setelah disamarkan disebut *chipertext*. Proses penyamaran *plaintext* ke *chipertext* disebut enkripsi, sedangkan pengembalian *chipertext* menjadi *plaintext* semula disebut dekripsi. Pada penelitian ini, akan dikombinasikan 2 algoritma kriptografi yaitu *Advance Encryption Standard* (AES) yang bernilai 128 bit dan *Secure Hash Algorithm* (SHA) yang bernilai 256 bit yang diterapkan pada aplikasi pengamanan informasi dokumen yang akan dibangun. Algoritma AES-218 akan mengenkripsi *file* dokumen dan algoritma SHA-256 akan mengenkripsi informasi catatan penting terkait *file* tersebut agar tidak dapat diubah.

Penelitian yang membahas pengamanan informasi dokumen dengan algoritma AES-128 dan SHA256 yaitu penelitian pertama yang membahas tentang penerapan algoritma AES untuk mengenkripsi dan mendekripsi dokumen. Hasil penelitian menyimpulkan bahwa algoritma AES dapat mengekripsi semua jenis karakter berupa *string*, huruf, angka, dan simbol. Pada saat mendekripsi QR *Code* aplikasi akan mengaktifkan fungsi kamera dan melakukan *scanning* QR *Code* yang akan menjadi *plaintext* kembali [1]. Penelitian kedua membahas tentang pengamanan informasi karya ilmiah dengan menerapkan *Blockchain Technology* yang dikombinasikan dengan

algoritma SHA-256 dan tanda tangan digital *Elliptic Curve Digital Signature Algorithm* (ECDSA). Hasil penelitian menunjukkan bahwa mekanisme model yang diusulkan sangat aman dalam mengamankan informasi karya ilmiah dan tidak mungkin informasi karya ilmiah dapat dicuri ataupun diubah oleh siapapun [2]. Kedua penelitian tersebut sudah sangat baik, namun belum mencoba kombinasi algoritma AES-218 dan SHA-256 dalam mengamankan lebih banyak jenis *file* dokumen.

2. Landasan Teori

Perancangan

Terdapat beberapa definisi yang menjelaskan pengertian perancangan berdasarkan beberapa sumber antara lain:

a. Perancangan dapat diartikan perencanaan dari pembuatan suatu sistem yang menyangkut berbagai komponen sehingga akan menghasilkan sistem yang sesuai dengan hasil dari tahap analisa sistem [3].
b. Perancangan adalah langkah pertama dalam fase pengembangan rekayasa produk atau sistem. Perancangan itu adalah proses penerapan berbagai teknik dan prinsip yang bertujuan untuk mendefinisikan sebuah peralatan, satu proses atau satu sistem secara detail yang membolehkan dilakukan realisasi fisik [4].

Aplikasi

Perangkat lunak aplikasi yaitu perangkat lunak yang digunakan untuk membantu pemakai komputer untuk melaksanakan pekerjaannya. Jika ingin mengembangkan program aplikasi sendiri, maka untuk menulis program aplikasi tersebut, dibutuhkan suatu bahasa pemrograman, yaitu *language software*, yang dapat berbentuk *assembler*, *compiler* ataupun *interpreter*. Jadi *language software* merupakan bahasanya dan program yang ditulis merupakan program aplikasinya [5].

Aplikasi juga dapat diklasifikasikan menjadi 2 bagian yaitu: [6]

a. Aplikasi *software* spesialis yaitu program dengan dokumentasi tergabung yang dirancang untuk menjalankan tugas tertentu.
b. Aplikasi paket yaitu program dengan dokumentasi tergabung yang dirancang untuk jenis masalah tertentu.

Kriptografi

Kriptografi memiliki berbagai macam pengertian, secara etimologis kata kriptografi berasal dari bahasa Yunani yang terdiri atas dua kata yaitu *krupto* yang berarti tersembunyi atau rahasia dan *grafh* yang berarti tulisan [7].

Algoritma Kriptografi

Algoritma kriptografi adalah aturan untuk *enchipering* dan *dechipering* atau fungsi matematika yang digunakan untuk enkripsi dan dekripsi pesan [7]..

Algoritma AES-128

Algoritma *Advanced Encryption Standard (*AES) adalah suatu algoritma *block chiper* dan mempunyai sifat simetri yang menggunakan kunci simetri pada waktu proses enkripsi dan dekripsi [8]. Terdapat 4 transformasi putaran pada proses enkripsi dan dekripsi yaitu: [8]

a. *SubBytes*
   Berfungsi untuk menukar isi dari byte dengan menggunakan tabel substitusi.
b. *ShiftRows*
   Proses pergeseran blok per baris pada *state array*.
c. *MixColumn*
   Proses mengalikan blok data (pengacakan) di masing-masing *state array* dengan rumus sebagai berikut:

$$A(x)=\{03\}x2+\{01\}x2+\{01\}x+\{02\}$$

d. *AddRoundKey*
   Mengombinasikan state *array* dan *round key* dengan hubungan XOR.

Pada proses dekripsi algoritma AES: [8]

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

Algoritma SHA-256

Fungsi *hash* dalam kriptografi adalah fungsi *hash* yang berupa sebuah algoritma yang mengambil sejumlah blok data dan mengembalikan bit *string* berukuran tetap [9].

Operasi algoritma *hashing* SHA-256 dapat dibagi menjadi tiga operasi yang berbeda [10].

a. *Pre-Processing*: Operasi yang melakukan logika *padding* dan *parses* dari *input message*.

b. *Message Scheduler*: Fungsi yang menghasilkan 64 *words* dari 16 *words input message block*.
c. Fungsi Kompresi: Fungsi yang melakukan operasi *hashing* aktual dari *message* bergantung pada kata yang keluar dari *scheduler* pesan di setiap putarannya.

Use Case Diagram

Use Case Diagram adalah suatu diagram yang dikelompokkan ke dalam aspek perilaku. Deskripsi perilaku dari setiap *Use Case* dijelaskan secara detail, terperinci dan terpisah dengan menggunakan arsip yang secara tekstual, yaitu *Use Case Scenario* atau *Use Case Specification* atau *Use Case Description* [11].

Balsamiq Mockup 3

Prototipe merupakan gambaran kasar mengenai perangkat lunak yang ingin dibangun. Desain prototipe dalam langkah awal setelah analisis terhadap permasalahan. Prototipe berguna untuk memberi informasi tampilan perangkat lunak yang ingin dirancang. Dengan adanya prototipe maka seorang programer dapat mudah dan cepat dalam membagun perangkat lunak tersebut. Perlu diingat prototipe bukan design persis perangkat lunak yang dirancang melainkan sebuah sketsa secara umum [12].

***Entity Relationship Diagram* (ERD)**

Entity Relationship Diagram adalah sebuah pendekatan *top-bottom* dalam merancang sebuah basis data, dimulai dengan mengidentifikasikan data yang penting dan digambarkan dalam suatu model. *Entity Relationship Diagram* merupakan pemodelan yang berguna untuk digunakan agar mendapatkan pemahaman yang tepat terhadap data dan penggunaannya di dalam suatu perusahaan [11].

***Hypertext Preprocessor* (PHP)**

PHP adalah salah suatu bahasa *scripting* khususnya digunakan untuk *web development.* Karena sifatnya yang *server side scripting* maka untuk menjalankan PHP harus menggunakan *web server* [13].

MySQL

MySQL adalah sebuah perangkat lunak pembuat *database* yang bersifat terbuka atau *open source* dan berjalan disemua platform baik Linux maupun sistem operasi Windows. MySQL merupakan program pengakses *database* yang bersifat *network* sehingga dapat digunakan untuk aplikasi *Multi User*. MySQL *(My Structure Query Language)* merupakan sebuah *database server* yang awalnya berjalan pada sistem Unix dan Linux [14].

3. Metode Penelitian

Analisis Aplikasi

Analisis aplikasi pada penelitian ini terbagi menjadi 3 tahapan proses yaitu:

a. Analisis masalah yaitu melakukan analisis lebih mendetail terkait permasalahan dalam penelitian ini [15].
b. Analisis algoritma yang diusulkan yaitu algoritma AES-128 dan SHA-256. Pada tahapan ini akan ditunjukkan contoh kasus sederhana bagaimana algoritma AES-128 dalam mengenkripsi informasi dokumen dan mendekripsinya serta contoh kasus sederhana bagaimana algoritma SHA-256 mengenkripsi informasi catatan dokumen agar tidak dapat diubah. Penelitian ini menggunakan kombinasi 2 buah algoritma dalam proses pengamanan *file* yaitu AES-128 dan SHA-256. Proses penerapan algoritma akan dijelaskan dalam contoh kasus sederhana.

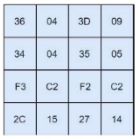
   Tahap pertama, pemilik *file* harus melakukan *upload file* yang akan dienkripsi, kemudian selanjutnya *file* akan dienkripsi dengan menggunakan algoritma AES-128.

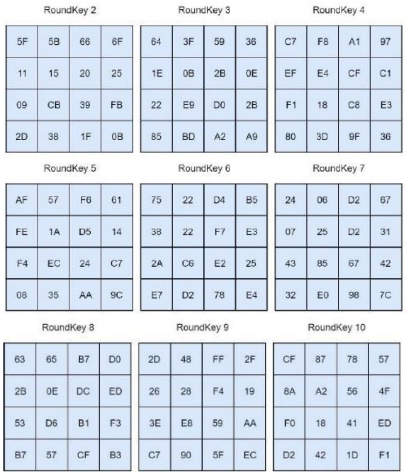
Proses Enkripsi

Contoh kasus asumsikan sebuah *plain text* berupa data dari nama *file* dan *key* menggunakan tanggal ditambah tanggal terbalik:

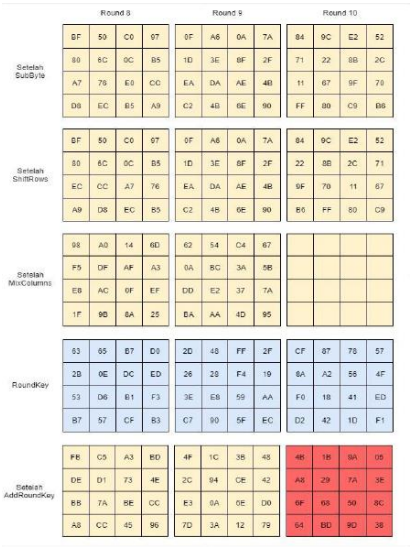
Plaintext : -0.006028093
Dalam HEX : 2d 30 2e 30 30 36 30 32 38 30 39 33 30 30 30 30
Key : 3004201991024003
Dalam HEX : 33 30 30 34 32 30 31 39 39 31 30 32 34 30 30 33

Maka didapatkan hasil *roundkey*-1 *sampai* roundkey-10.



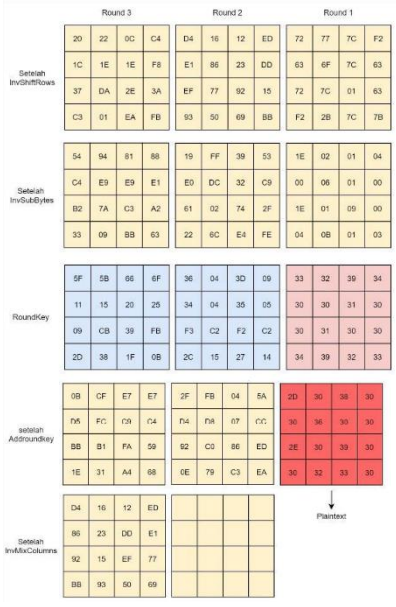


Setelah melalui beberapa tahapan didalam proses enkripsi maka *plain text* berubah menjadi *chipper text*. Berikut ini adalah hasil dari proses enkripsi.



Proses Dekripsi

Proses dekripsi adalah mengubah *chipper text* kembali kedalam bentuk *plain text*. Berikut ini adalah hasil akhir dari proses dekripsi.



Berikutnya setelah *file* berhasil dienkripsi, maka proses selanjutnya adalah melakukan enkripsi terhadap informasi catatan *file* tersebut dengan SHA-256. Berikut ini tahapan algoritma SHA-256 dalam mengenkripsi catatan *file*.

1) Asumsikan bahwa informasi catatan *file* disimpan dalam sebuah variabel M. Dengan nilai M = "abc".
2) Konversikan isi variabel M ke dalam bentuk binary sehingga akan menjadi M = 011000010110001001100011.
3) Lakukan padding dengan langkah pertama menambahkan bit '1' pada pesan asli sehingga akan menjadi M = 011000010110001001100011 1.
4) Selanjutnya tambahkan bit '0' sebanyak nilai k, dimana k didapat dari perhitungan rumus $l + 1 + k \equiv 448 \bmod 512$. Maka dalam contoh kasus ini, nilai k = 423. Sehingga pesan yang sudah dipadding akan menjadi seperti berikut
   01100001 01100010 01100011 10000000 00000000 00000000 00000000 00000000 00000000 00000000
   00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
   00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
   00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
   00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
   00000000 00000000 00000000 00000000 00000000 00000000
5) Selanjutnya tambahkan lagi panjang pesan asli dalam bentuk biner 64 bit. Panjang pesan M adalah 24 bit, dalam bentuk biner adalah 00011000. Maka pesan yang dipadding akan menjadi seperti berikut ini:
   01100001 01100010 01100011 10000000 00000000 00000000 00000000 00000000 00000000 00000000
   00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
   00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
   00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
   00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
   00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00011000
6) Kemudian pesan diparsing menjadi 16 buat word 32-bit untuk masing-masing blok pesan M. Sehingga menjadi seperti ini:

$M_0^{(0)}$: 0110 0001 0110 0010 0110 0011 1000 0000
$M_1^{(0)}$: 0000 0000 0000 0000 0000 0000 0000 0000
$M_2^{(0)}$: 0000 0000 0000 0000 0000 0000 0000 0000
$M_3^{(0)}$: 0000 0000 0000 0000 0000 0000 0000 0000
$M_4^{(0)}$: 0000 0000 0000 0000 0000 0000 0000 0000
$M_5^{(0)}$: 0000 0000 0000 0000 0000 0000 0000 0000
$M_6^{(0)}$: 0000 0000 0000 0000 0000 0000 0000 0000
$M_7^{(0)}$: 0000 0000 0000 0000 0000 0000 0000 0000
$M_8^{(0)}$: 0000 0000 0000 0000 0000 0000 0000 0000
$M_9^{(0)}$: 0000 0000 0000 0000 0000 0000 0000 0000
$M_{10}^{(0)}$: 0000 0000 0000 0000 0000 0000 0000 0000
$M_{11}^{(0)}$: 0000 0000 0000 0000 0000 0000 0000 0000
$M_{12}^{(0)}$: 0000 0000 0000 0000 0000 0000 0000 0000
$M_{13}^{(0)}$: 0000 0000 0000 0000 0000 0000 0000 0000
$M_{14}^{(0)}$: 0000 0000 0000 0000 0000 0000 0000 0000
$M_{15}^{(0)}$: 0000 0000 0000 0000 0000 0000 0001 1000

7) Langkah selanjutnya adalah *set initial hash value*, yakni nilai awal dari *hash*.
   $H_0^{(0)}$ = 6a09e667
   $H_1^{(0)}$ = bb67ae85
   $H_2^{(0)}$ = 3c6ef372
   $H_3^{(0)}$ = a54ff53a
   $H_4^{(0)}$ = 510e527f
   $H_5^{(0)}$ = 9b05688c
   $H_6^{(0)}$ = 1f83d9ab
   $H_7^{(0)}$ = 5be0cd10
8) Selanjutnya menyiapkan *message schedule* dari 16 buah *word* hasil *parsing*. Kemudian siapkan 8 *working variabel* yang nilainya diambil dari *initial hash value* tadi.
   a = $H_0^{(0)}$ = 6a09e667
   b = $H_1^{(0)}$ = bb67ae85
   c = $H_2^{(0)}$ = 3c6ef372
   d = $H_3^{(0)}$ = a54ff53a
   e = $H_4^{(0)}$ = 510e527f
   f = $H_5^{(0)}$ = 9b05688c
   g = $H_6^{(0)}$ = 1f83d9ab
   h = $H_7^{(0)}$ = 5be0cd10
9) Siapkan juga koefisien SHA-256 yang sudah ditetapkan pada standar SHA-2. Ke-64 koefisien tersebut adalah sebagai berikut ini yang dibaca urut dari kiri ke kanan:

```
428a2f98 71374491 b5c0fbcf e9b5dba5 3956c25b 59f111f1 923f82a4 ab1c5ed5
d807aa98 12835b01 243185be 550c7dc3 72be5d74 80deb1fe 9bdc06a7 c19bf174
e49b69c1 efbe4786 0fc19dc6 240ca1cc 2de92c6f 4a7484aa 5cb0a9dc 76f988da
983e5152 a831c66d b00327c8 bf597fc7 c6e00bf3 d5a79147 06ca6351 14292967
27b70a85 2e1b2138 4d2c6dfc 53380d13 650a7354 766a0abb 81c2c92e 92722c85
a2bfe8a1 a81a664b c24b8b70 c76c51a3 d192e819 d6990624 f40e3585 106aa070
19a4c116 1e376c08 2748774c 34b0bcb5 391c0cb3 4ed8aa4a 5b9cca4f 682e6ff3
748f82ee 78a5636f 84c87814 8cc70208 90befffa a4506ceb bef9a3f7 c67178f2
```

10) Kini yang dibutuhkan dalam komputasi SHA-256 sudah siap semua, saatnya melakukan komputasi SHA-256. Proses komputasi akan dilakukan dalam bentuk iterasi.

	a	b	c	d	e	f	g	h
init:	6a09e667	bb67ae85	3c6ef372	a54ff53a	510e527f	9b05688c	1f83d9ab	5be0cd19
t = 0	5d6aebcd	6a09e667	bb67ae85	3c6ef372	fa2a4622	510e527f	9b05688c	1f83d9ab
t = 1	5a6ad9ad	5d6aebcd	6a09e667	bb67ae85	78ce7989	fa2a4622	510e527f	9b05688c
t = 2	c8c347a7	5a6ad9ad	5d6aebcd	6a09e667	f92939eb	78ce7989	fa2a4622	510e527f
t = 3	d550f666	c8c347a7	5a6ad9ad	5d6aebcd	24e00850	f92939eb	78ce7989	fa2a4622
t = 4	04409a6a	d550f666	c8c347a7	5a6ad9ad	43ada245	24e00850	f92939eb	78ce7989
t = 5	2b4209f5	04409a6a	d550f666	c8c347a7	714260ad	43ada245	24e00850	f92939eb
t = 6	e5030380	2b4209f5	04409a6a	d550f666	9b27a401	714260ad	43ada245	24e00850
t = 7	85a07b5f	e5030380	2b4209f5	04409a6a	0c657a79	9b27a401	714260ad	43ada245
t = 8	8e04ecb9	85a07b5f	e5030380	2b4209f5	32ca2d8c	0c657a79	9b27a401	714260ad
t = 9	8c87346b	8e04ecb9	85a07b5f	e5030380	1cc92596	32ca2d8c	0c657a79	9b27a401
t = 10	4798a3f4	8c87346b	8e04ecb9	85a07b5f	436b23e8	1cc92596	32ca2d8c	0c657a79
t = 11	f71fc5a9	4798a3f4	8c87346b	8e04ecb9	816fd6e9	436b23e8	1cc92596	32ca2d8c
t = 12	87912990	f71fc5a9	4798a3f4	8c87346b	1e578218	816fd6e9	436b23e8	1cc92596
t = 13	d932eb16	87912990	f71fc5a9	4798a3f4	745a48de	1e578218	816fd6e9	436b23e8
t = 14	c0645fde	d932eb16	87912990	f71fc5a9	0b92f20c	745a48de	1e578218	816fd6e9
t = 15	b0fa238e	c0645fde	d932eb16	87912990	07590dcd	0b92f20c	745a48de	1e578218
t = 16	21da9a9b	b0fa238e	c0645fde	d932eb16	8034229c	07590dcd	0b92f20c	745a48de
t = 17	c2fbd9d1	21da9a9b	b0fa238e	c0645fde	846ee454	8034229c	07590dcd	0b92f20c
t = 18	fe777bbf	c2fbd9d1	21da9a9b	b0fa238e	cc899961	846ee454	8034229c	07590dcd
t = 18	fe777bbf	c2fbd9d1	21da9a9b	b0fa238e	cc899961	846ee454	8034229c	07590dcd
t = 19	e1f20c33	fe777bbf	c2fbd9d1	21da9a9b	b0638179	cc899961	846ee454	8034229c
t = 20	9dc68b63	e1f20c33	fe777bbf	c2fbd9d1	8ada8930	b0638179	cc899961	846ee454
t = 21	c2606d6d	9dc68b63	e1f20c33	fe777bbf	e1257970	8ada8930	b0638179	cc899961
t = 22	a7a3623f	c2606d6d	9dc68b63	e1f20c33	49f5114a	e1257970	8ada8930	b0638179
t = 23	c5d53d8d	a7a3623f	c2606d6d	9dc68b63	aa47c347	49f5114a	e1257970	8ada8930
t = 24	1c2c2838	c5d53d8d	a7a3623f	c2606d6d	2823ef91	aa47c347	49f5114a	e1257970
t = 25	cde8037d	1c2c2838	c5d53d8d	a7a3623f	14383d8e	2823ef91	aa47c347	49f5114a
t = 26	b62ec4bc	cde8037d	1c2c2838	c5d53d8d	c74c6516	14383d8e	2823ef91	aa47c347
t = 27	77d37528	b62ec4bc	cde8037d	1c2c2838	edffbff8	c74c6516	14383d8e	2823ef91
t = 28	363482c9	77d37528	b62ec4bc	cde8037d	6112a3b7	edffbff8	c74c6516	14383d8e
t = 29	a0060b30	363482c9	77d37528	b62ec4bc	ade79437	6112a3b7	edffbff8	c74c6516
t = 30	ea992a22	a0060b30	363482c9	77d37528	0109ab3a	ade79437	6112a3b7	edffbff8
t = 31	73b33bf5	ea992a22	a0060b30	363482c9	ba591112	0109ab3a	ade79437	6112a3b7
t = 32	98e12507	73b33bf5	ea992a22	a0060b30	9cd9f5f6	ba591112	0109ab3a	ade79437
t = 33	fe604df5	98e12507	73b33bf5	ea992a22	59249dd3	9cd9f5f6	ba591112	0109ab3a
t = 34	a9a7738c	fe604df5	98e12507	73b33bf5	085f3833	59249dd3	9cd9f5f6	ba591112
t = 35	65a0cfe4	a9a7738c	fe604df5	98e12507	f4b002d6	085f3833	59249dd3	9cd9f5f6
t = 36	41a65cb1	65a0cfe4	a9a7738c	fe604df5	0772a26b	f4b002d6	085f3833	59249dd3
t = 37	34df1604	41a65cb1	65a0cfe4	a9a7738c	a507a53d	0772a26b	f4b002d6	085f3833
t = 38	6dc57a8a	34df1604	41a65cb1	65a0cfe4	f0781bc8	a507a53d	0772a26b	f4b002d6
t = 39	79ea687a	6dc57a8a	34df1604	41a65cb1	1efbc0a0	f0781bc8	a507a53d	0772a26b
t = 40	d6670766	79ea687a	6dc57a8a	34df1604	26352d63	1efbc0a0	f0781bc8	a507a53d
t = 41	df46652f	d6670766	79ea687a	6dc57a8a	838b2711	26352d63	1efbc0a0	f0781bc8
t = 42	17aa0dfe	df46652f	d6670766	79ea687a	decd4715	838b2711	26352d63	1efbc0a0
t = 43	9d4baf93	17aa0dfe	df46652f	d6670766	fda24c2e	decd4715	838b2711	26352d63
t = 44	26628815	9d4baf93	17aa0dfe	df46652f	a80f11f0	fda24c2e	decd4715	838b2711
t = 45	72ab4b91	26628815	9d4baf93	17aa0dfe	b7755da1	a80f11f0	fda24c2e	decd4715
t = 46	a14c14b0	72ab4b91	26628815	9d4baf93	d57b94a9	b7755da1	a80f11f0	fda24c2e
t = 47	4172328d	a14c14b0	72ab4b91	26628815	fecf0bc6	d57b94a9	b7755da1	a80f11f0
t = 48	05757ceb	4172328d	a14c14b0	72ab4b91	bd714038	fecf0bc6	d57b94a9	b7755da1
t = 49	f11bfaa8	05757ceb	4172328d	a14c14b0	6e5c390c	bd714038	fecf0bc6	d57b94a9
t = 50	7a0508a1	f11bfaa8	05757ceb	4172328d	52f1ccf7	6e5c390c	bd714038	fecf0bc6
t = 51	886e7a22	7a0508a1	f11bfaa8	05757ceb	49231c1e	52f1ccf7	6e5c390c	bd714038
t = 52	101fd28f	886e7a22	7a0508a1	f11bfaa8	529e7d00	49231c1e	52f1ccf7	6e5c390c
t = 53	f5702fdb	101fd28f	886e7a22	7a0508a1	9f4787c3	529e7d00	49231c1e	52f1ccf7
t = 54	3ec45cdb	f5702fdb	101fd28f	886e7a22	e50e1b4f	9f4787c3	529e7d00	49231c1e
t = 55	38cc9913	3ec45cdb	f5702fdb	101fd28f	54cb266b	e50e1b4f	9f4787c3	529e7d00
t = 56	fcd1887b	38cc9913	3ec45cdb	f5702fdb	9b5e906c	54cb266b	e50e1b4f	9f4787c3
t = 57	c062d46f	fcd1887b	38cc9913	3ec45cdb	7e44008e	9b5e906c	54cb266b	e50e1b4f
t = 58	ffb70472	c062d46f	fcd1887b	38cc9913	6d83bfc6	7e44008e	9b5e906c	54cb266b
t = 59	b6ae8fff	ffb70472	c062d46f	fcd1887b	b21bad3d	6d83bfc6	7e44008e	9b5e906c
t = 60	b85e2ce9	b6ae8fff	ffb70472	c062d46f	961f4894	b21bad3d	6d83bfc6	7e44008e
t = 61	04d24d6c	b85e2ce9	b6ae8fff	ffb70472	948d25b6	961f4894	b21bad3d	6d83bfc6
t = 62	d39a2165	04d24d6c	b85e2ce9	b6ae8fff	fb121210	948d25b6	961f4894	b21bad3d
t = 63	506e3058	d39a2165	04d24d6c	b85e2ce9	5ef50f24	fb121210	948d25b6	961f4894

11) Hasil dari iterasi terakhir nanti kemudian dijumlah dengan *initial hash valuenya* tadi. Sehingga akan menghasilkan sebagai berikut:
$H_0^{(0)}$ = 506e3058 + 6a09e667 = ba7816bf
$H_1^{(0)}$ = d39a2165 + bb67ae85 = 8f01cfea
$H_2^{(0)}$ = 04d24d6c + 3c6ef372 = 414140de
$H_3^{(0)}$ = b85e2ce9 + a54ff53a = 5dae2223
$H_4^{(0)}$ = 5ef50f24 + 510e527f = b00361a3
$H_5^{(0)}$ = fb121210 + 9b05688c = 96177a9c
$H_6^{(0)}$ = 948d25b6 + 1f83d9ab = b410ff61
$H_7^{(0)}$ = 961f4894 + 5be0cd10 = f20015ad

Hasil penjumlahan antara iterasi terakhir dengan initial hash valuenya inilah yang nantinya digabungkan dan menjadi hasil akhir dari SHA-256 dan terbentuklah sebuah message digest dari pesan M yang berisi "abc". Maka nilai hash dari pesan M adalah **ba7816bf8f01cfea414140de5dae2223b00361a396177a9cb410ff61f20015ad**.

c. Analisis sistem yang dibangun yaitu menggambarkan fitur-fitur yang tersedia pada sistem/aplikasi yang akan dibangun dengan memodelkannya menggunakan *Use Case Diagram*.

Perancangan dan Pembangunan Aplikasi

Perancangan pada penelitian ini dibagi menjadi 2 proses yaitu:

a. Perancangan tampilan aplikasi yang digambarkan dengan menggunakan *software Balsamiq Mockup 3*.
b. Perancangan basis data yang digambarkan dengan menggunakan *tools Entity Relationship Diagram* (ERD).

Pembangunan aplikasi pada penelitian ini menggunakan beberapa bahasa pemograman yaitu:

a. Bahasa pemograman PHP.

MySQL sebagai bahasa *query database*.

4. Hasil Penelitian

Berikut ini keseluruhan hasil tampilan aplikasi yang telah selesai dibangun antara lain:

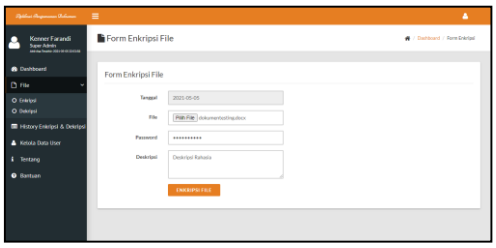
a. Tampilan Awal

Tampilan awal berupa sebuah *form login* yang harus diisikan oleh pengguna agar dapat menggunakan aplikasi. Aplikasi dapat digunakan oleh dua *user* yaitu *user* Administrator dan Super Admin. Perbedaannya adalah Super Admin dapat menambahkan Administrator agar dapat menggunakan aplikasi. Apabila ingin melakukan *login*, maka dapat dilakukan dengan mengisi *textbox username* dan *password* kemudian menekan tombol *login*.

Gambar 1. Tampilan Awal

b. Tampilan *Form* Enkripsi *File*

Tampilan yang berisikan *form* untuk melakukan proses enkripsi *file* dokumen dengan algoritma AES-128 dan SHA-256. AES-128 akan melakukan enkripsi *file* dokumen sedangkan SHA-256 akan mengenkripsi informasi catatan deskripsi agar tidak dapat diubah.



Gambar 2. Tampilan *Form* Enkripsi *File*

Pengisian *password* harus memenuhi syarat memiliki 1 huruf besar, huruf minimal 6 huruf dan maksimal 12 huruf, serta harus memiliki minimal 1 simbol dan 1 angka. Jika tidak memenuhi syarat tersebut maka akan muncul pesan *error* seperti gambar 4.4 di bawah ini

Gambar 3. Tampilan Pesan *Error* Jika Tidak Memenuhi Syarat

5. Kesimpulan

Kesimpulan dari hasil penelitian yang telah dilakukan yaitu:

a. Aplikasi pengamanan *file* dokumen yang dibangun menerapkan algoritma AES-128 dan SHA-256 sehingga dapat mengenkripsi *file* dokumen secara aman karena proses enkrip dan dekrip *file* hanya dapat dilakukan pada aplikasi yang dibangun.
b. Hasil penelitian menunjukkan bahwa proses enkripsi dan dekripsi *file* menggunakan algoritma AES-128 dan SHA-256 dapat dilakukan dengan cepat tergantung dari besar *size* dari *file* dokumen yang dienkripsi.

6. Daftar Pustaka

[1] D. Q. P. A. Paramarta, A. Kusyanti and M. Data, "Implementasi Algoritme Advance Encryption Standard (AES) pada Enkripsi dan Dekripsi QR-Code," Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, vol. II, no. 12, pp. 6729-6736, 2018.

[2] Andi, R. Purba and R. Yunis, "Application of Blockchain Technology to Prevent The Potential Of Plagiarism in Scientific Publication," in 2019 Fourth International Conference on Informatics and Computing (ICIC), Semarang, 2019.

[3] D. A. Rianto, S. Assegaf and E. Fernando, "Perancangan Aplikasi Sistem Informasi Geografis (SIG) Lokasi," Jurnal Ilmiah Media SISFO, vol. 9, no. 2, 2015.

[4] B. Nadeak, A. Parulian, Pristiwanto and S. R. Siregar, "Perancangan Aplikasi Pembelajaran Internet Dengan Menggunakan Metode Computer Based Instruction," Jurnal Riset Komputer (JURIKOM), vol. 3, no. 4, 2016.

[5] T. Sutabri, Pengantar Teknologi Informasi, Yogyakarta: Andi, 2015.

[6] Y. Iskandar, Buku Ajar Pengantar Ilmu Komputer, Yogyakarta: Penerbit Deepublish, 2018.

[7] J. Simarmata, Sriadhi and R. Rahim, Kriptografi, Teknik Keamanan Data Dan Informasi, Yogyakarta: Andi Offset, 2020.

[8] A. Prameshwari and N. P. Sastra, "Implementasi Algoritma Advanced Encryption Standard (AES) 128 Untuk Enkripsi dan Dekripsi File Dokumen," Eksplora Informatika, vol. VIII, no. 2, pp. 52-58, 2018.

[9] A. Y. Mulyadi and R. Rahman, "Implementasi Algoritma AES 128 dan SHA – 256 Dalam Pengkodean pada Sebagian Frame Video CCTV MPEG-2," Jurnal Aplikasi dan Teori Ilmu Komputer (JATIKOM), vol. I, no. 1, pp. 30-36, 2018.

[10] S. Sulastri and R. D. M. Putri, "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. X, no. 2, pp. 70-74, 2018.

[11] R. A. Sukamto and M. Shalahuddin, Rekayasa Perangkat Lunak, Bandung: Informatika, 2015.

[12] Didik, "Membuat Prototipe menggunakan Balsamiq Mockup," Kodingin, 22 July 2018. [Online]. Available: https://kodingin.com/membuat-prototipe-menggunakan-balsamiq-mockup/. [Accessed 20 January 2020].

[13] M. Masrur, Pemrograman PHP dan MySQL untuk Pemula, Yogyakarta: Andi, 2016.

[14] J. Enterprise, Pengenalan HTML dan CSS, Jakarta: PT. Elex Media Komputindo, 2016.

[15] K. A. Prasetyo, M. Idhom and H. E. Wahanani, "Sistem Pencegahan Serangan Brute Force Pada Multiple Server Dengan Menggunakan Fail2Ban," Jurnal Informatika dan Sistem Informasi, vol. I, no. 3, pp. 709-715, 2020.