ISSN 2548-9739

# PENGKODEAN PESAN TEKS DENGAN PROSES PENERAPAN ALGORITMA KRIPTOGRAFI SECURE HASH ALGORITHM (SHA)

**ROMANUS DAMANIK**

Fakultas Ilmu Komputer, Universitas Katolik Santo Thomas
Jl. Setia Budi No.479F, Tanjung Sari, Medan Selayang, Kota Medan, Sumatera Utara 20154
romanusdamanik@yahoo.com

**Abstrak**

Masalah Keamanan merupakan salah satu aspek penting dari sistem informasi. Begitu pentingnya nilai informasi menyebabkan seringkali informasi diinginkan oleh orang tertentu kemudian dapat memodifikasi informasi tersebut. Untuk dapat mengesahkan informasi yang didapatkan, maka harus dilakukan authentication pada informasi sehingga diketahui keaslian informasi. Dalam penelitian ini diimplementasikan suatu metode secure hash algorithm, sehingga penerima informasi dapat mengetahui adanya perubahan terhadap data penting yang diterima dari pihak lain. Metode ini mampu membangkitkan nilai hash dari sebuah string atau file. Dengan perubahan sekecil apapun pada informasi yang diterima, dapat diketahui keasliannya. Secure Hash Algorithm adalah fungsi hash yang bekerja satu arah, ini berarti pesan yang sudah diubah menjadi message digest tidak dapat dikembalikan menjadi pesan semula. Dua pesan yang berbeda akan selalu menghasilkan nilai hash yang berbeda pula. Hasil rancangan perangkat lunak ini dengan input panjang string yang berbeda akan menghasilkan output dengan panjang string tetap yaitu 160 bit.

**Kata Kunci** : Kriptografi, Secure Hash Algorithm, Data.

## I. PENDAHULUAN

Teknologi merupakan suatu kebutuhan yang sanggat penting dalam era teknologi saat ini, perkembangan teknologi komputer pada saat ini sangatlah penting bagi dunia jasa layanan publik, perindustrian, perkantoran, pendidikan, teknik dan dunia perdagangan. Untuk itu komputer sebagai pengolah data mempunyai peranan penting dari segi keamanan data.Masalah keamanan dan kerahasiaan data merupakan salah satu aspek penting dari suatu sistem informasi. Jika berbicara mengenai masalah keamanan yang berkaitan dengan penggunaan komputer, maka sulit memisahkannya dengan kriptografi.

Dalam kriptografi sering ditemukan istilah atau terminologi, seperti pesan (message) adalah data atau informasi yang dapat dibaca dan dimengerti maknanya. Nama lain untuk pesan adalah plainteks (plaintext) atau teks jelas (cleartext). Pesan dapat berupa data atau informasi yang dikirim (melalui kurir, saluran telekomunikasi, dsb) atau yang disimpan di dalam media perekaman (kertas, storage, dsb). Pesan yang tersimpan berupa teks, Supaya pesan tidak dapat dimengerti maknanya oleh pihak lain, maka pesan perlu disandikan ke bentuk lain yang tidak dapat dipahami. Bentuk yang tersandi disebut ciphertext atau kriptogram yang harus bisa ditransformasikan kembali menjadi plainteks semula agar pesan yang diterima bisa dibaca.

Adapun sebelumya telah di buat "***Implementasi Secure Hash Algorithm-1 Untuk Pengamanan Data Dalam Library Pada Pemrograman***

ISSN 2548-9739

***Java"*** oleh Komang Aryasa jurusan Tenik Informatika STMIK Dipanegara Makasar dan Yesaya Tommy Paulus jurusan Sitem Informasi STMIK Dipanegara Makasar dalam jurnalnya yang membahas tentang membuat sebuah library, pada pemgraman java dalam membuat sebuah message digest dari dari suatu string, sehingga dapat langsung digunakan untuk otentikasi dan keamanan data. Untuk itu keamanan dari sistem informasi yang digunakan harus terjamin dalam batas yang dapat diterima.Kriptografi merupakan sebuah teknik mengubah sebuah plaintext menjadi sebuah ciphertext Di dalam kriptografi terdapat sebuah fungsi yang sesuai untuk aplikasi keamanan seperti otentikasi dan integritas pesan. Fungsi tersebut merupakan fungsi hash, fungsi mampu menerima masukan string yang panjangnya sembarang dan mengkonversikannya menjadi string keluaran yang panjangnya tetap (fixed) dan umumnya jauh lebih kecil dibandingkan string semula. Fungsi hash satu arah (One way Hashing) bekerja dalam satu arah, seperti algoritma SHA(secure hash algorithm). SHA merupakan fungsi hash satu arah yang dibuat oleh NIST dan digunakan bersama DSS (Digital Signature Standard). Oleh NSA, SHA dinyatakan sebagai standar fungsi hash satu-arah. SHA didasarkan pada MD 4 yang dibuat oleh Rivest dari MIT. Algoritma ini menerima masukan berupa pesan dengan ukuran maksimum 264 bit dan menghasilkan digest yang panjangnya 160 bit yang lebih panjang dari MD5. SHA(Secure Hash Algorithm) 1 itu sejenis fungsi hash 160 bit yang katanya pengganti md5 (Message Digest) yang hash-nya cuma 128 bit.

### 1.2 Rumusan Masalah

Adapaun yang menjadi rumusan masalah yang didapat dari urain di atas adalah:

1. Bagaimana merancang aplikasi penerapan algoritma Secure Hash Algorithm(SHA) untuk menyadikan file teks?
2. Bagaimana analisis dalam menerapkan algoritma Secure Hash Algorithm(SHA) pada penyadian teks?

### 1.3 Tujuan

Adapun dari perumusan masalah diatas maka penulis membuat tunjuan yaitu:

1. Menerapkan algoritma Secure Hash Algorithm (SHA) dan Menguji dan menganalisis hasil penerapan algoritma Secure Hash Algorithm(SHA).
2. Untuk membuat sistem keamanan dengan menggunakan enkripsi pada file text.
3. Membuat dekripsi untuk pembacaan file yang telah di enkripsi.

## 2. LANDASAN TEORI

### 2.1 Definiasi Kriptografi

Menurut Dony Ariyus (2008, h. 13), kriptografi berasal dari bahasa Yunani, crypto dan graphia. Crypto berarti secret(rahasia) dan graphia berarti writing(tulisan). Menurut terminologynya, kriptografi adalah ilmu dan seni untuk menjaga keamanan pesan ketika pesan dikirim dari suatu tempat ke tempat lain Metode SHA.

Menurut Christof Paar, Jan Pelzl, dan Bart Preneel (2009, h. 309) SHA-1 menerima masukan berupa string dengan ukuran maksimum $2^{64}$ bit. Untuk setiap string, SHA akan menghasilkan keluaran sebanyak 160 bit dari string tersebut dan string keluaran itu disebut message digest. Panjang jarak message digest dapat berkisar antara 160 sampai 512 bit tergantung algoritmanya. Gambaran pembuatan message digest dengan algoritma SHA

Kriptografi adalah merupakan ilmu yang mempelajari teknik-teknik matematika yang berhubungan dengan aspek keamanan informasi seperti kerahasiaan, integritas data serta otentikasi. Kriptografi adalah proses penggunaan berbagai teknik dan atau ilmu dan seni untuk menjaga keamanan pesan.

ISSN 2548-9739

*Cryptographic algorithm* adalah fungsi matematika yang digunakan untuk enkripsi dan dekripsi. Terdapat dua fungsi yang saling berhubungan yaitu satu untuk enkripsi dan satu lagi untuk dekripsi. Enkripsi merupakan proses pengkodean sebuah pesan sehingga isi dari pesan tersebut tidak diketahui. Dekripsi adalah proses kebalikan dari enkripsi yaitu mentransformasi pesan yang dienkripsi kembali menjadi bentuk semula. Sebuah sistem enkripsi dan dekripsi disebut *cryptosystem*. Bentuk asli dari sebuah pesan disebut *plaintext* dan bentuk asli yang dienkripsi disebut *ciphertext*.

## 2.2 Algoritma Kriptografi

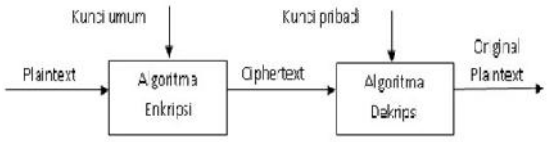
Algoritma kriptografi merupakan langkah-langkah logis bagaimana menyembunyikan pesan dari orang-orang yang tidak berhak atas pesan tersebut. Algoritma kriptografi terdiri dari tiga fungsi dasar yaitu : (Ariyus, 2008)

1. Enkripsi merupakan hal yang sangat penting dalam kriptografi, merupakan pengamanan data yang dikirimkan agar terjaga kerahasiannya. Pesan asli disebut *plaintext*, yang diubah menjadi kode-kode yang tidak dimengerti. Enkripsi bisa diartikan dengan cipher atau kode. Untuk mengubah teks asli ke bentuk teks kode digunakan algoritma yang dapat mengkodekan data.
2. Dekripsi merupakan kebalikan dari enkripsi. Pesan yang telah dienkripsi dikembalikan ke bentuk asalnya (teks asli/*plaintext*) disebut dengan dekripsi.
3. Kunci yang dipakai untuk melakukan enkripsi dan dekripsi. Kunci terbagi menjadi dua bagian yaitu kunci rahasia (*private key*) dan kunci umum (*public key*)

Biasanya algoritma kriptografi dapat dinotasikan sebagai berikut :

***Plaintext***(M)
***Ciphertext***(C)
**Enkripsi** (fungsi E)
**Dekripsi** (fungsi D)

Kriptografi itu sendiri terdiri dari dua proses utama yakni proses enkripsi dan proses dekripsi. Seperti yang telah dijelaskan di atas, proses enkripsi mengubah *plaintext* menjadi *ciphertext* (dengan menggunakan kunci tertentu) sehingga isi informasi pada pesan tersebut sukar dimengerti. Adapun alur dari proses enkripsi dan dekripsi pada kriptografi dapat dilihat pada gambar 2.1




Gambar 2.1 Konsep Proses Enkripsi dan Dekripsi
Sumber : Kriptografi, Dony Ariyus, Andi Publisher

## 2.3. Pengertian *Unified Modeling Language* (UML)

*Unified Modeling Language* (UML) adalah bahasa untuk menspesifikasi, memvisualisasi, serta mengonstruksi bangunan dasar sistem perangkat lunak, termasuk melibatkan pemodelan aturan-aturan bisnis. UML merupakan perkakas utama untuk analisis dan perancangan sistem berorientasi objek.

Menurut Sugiarti (2013, h. 34), menyatakan bahwa Unified Modeling Language (UML) adalah sebuah bahasa yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. Seperti bahasa-bahasa lainnya, UML mendefinisikan notasi dan *syntax*/semantik. Notasi UML merupakan sekumpulan bentuk khusus untuk menggambarkan berbagai diagram piranti lunak. Untuk membuat suatu model, UML memiliki diagram grafis sebagai berikut:

1. *Use case* diagram
2. *Class* diagram
3. *Behavior* diagram

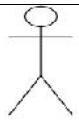
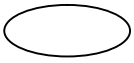
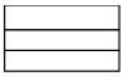
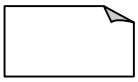
ISSN 2548-9739

a. *Statechart* diagram
b. *Activity* diagram
c. *Interaction* diagram
   1) *Sequence* diagram
   2) *Collaboration* diagram

4. Implementation diagram
   a. *Component* diagram
   b. *Deployment* diagram

Berikut ini adalah gambar dan fungsi simbol-simbol dalam *Unified Modeling Language* (UML):

**Tabel 1 Simbol-simbol *Unified Modeling Language* (UML)**

| No | Notasi | Gambar | Keterangan |
|---|---|---|---|
| 1 | *Actor* |  | Menggambarkan segala pengguna software aplikasi (user). Actor memberikan suatu gambaran jelas tentang apa yang harus dikerjakan software aplikasi. |
| 2 | *Use case* |  | Menjelaskan urutan kegiatan yang dilakukan actor dan sistem untuk mencapat suatu tujuan tertentu. |
| 3 | *Class* |  | Merupakan pembentuk utama dari sistem berorientasi objek. |
| 4 | *Interface* |  | Merupakan kumpulan operasi tanpa implementasi dari suatu class. |
| 5 | *Interaction* |  | Digunakan untuk menunjukkan baik aliran pesan atau informasi antar objek maupun hubungan antar objek. |
| 6 | *Note* |  | Digunakan untuk memberikan keterangan atau komentar tambahan dari suatu elemen sehingga bisa langsung terlampir dalam model. |
| 7 | *Dependency* |  | Merupakan relasi yang menunjukkan bahwa perubahan pada salah satu elemen memberi pengaruh pada elemen lain. |
| 8 | *Association* |  | Menggambarkan navigasi antar class. |
| 9 | *Generalization* |  | Menunjukkan hubungan antara elemen yang lebih umum ke elemen yang lebih spesifik. |

ISSN 2548-9739

### 2.3 Fungsi Hash

Fungsi hash disebut juga fungsi hash satu arah (*one-way function*), *message digest*, *fingerprint*, fungsi kompresi dan *message authentication code* (MAC), merupakan suatu fungsi matematika yang mengambil masukan panjang variabel dan mengubahnya ke dalam urutan biner dengan panjang yang tetap. Biasanya fungsi hash digunakan untuk membuat sidik jari yang bertujuan agar pesan yang dibuat benar adanya dari si pembuat atau yang diinginkan.

### 2.4. Secure Hash Algorithm (SHA)

Menurut Christof Paar, Jan Pelzl, dan Bart Preneel (2009, h. 309) SHA-1 menerima masukan berupa string dengan ukuran maksimum $2^{64}$ bit. Untuk setiap string, SHA akan menghasilkan keluaran sebanyak 160 bit dari string tersebut dan string keluaran itu disebut message digest. Panjang jarak message digest dapat berkisar antara 160 sampai 512 bit tergantung algoritmanya.

Berikut merupakan Langkah-langkah pembuatan message digest secara garis besar adalah sebagai berikut:

1. Penambahan Bit-bit Pengganjal
    1. Pesan ditambah dengan sejumlah bit pengganjal sedemikian sehingga panjang pesan (dalam satuan bit) kongruen dengan 448 modulo 512. Ini berarti panjang pesan setelah ditambahi bit- bit pengganjal adalah 64 bit kurang dari kelipatan 512. Angka 512 ini muncul karena SHA memperoses pesan dalam blok-blok yang berukuran 512.
    2. Pesan dengan panjang 448 bit pun tetap ditambah dengan bit-bit pengganjal. Jika panjang pesan
        448 bit, maka pesan tersebut ditambah dengan 512 bit menjadi 960 bit. Jadi, panjang bit-bit pengganjal adalah antara 1 sampai 512.
    3. Bit-bit pengganjal terdiri dari sebuah bit 1 diikuti dengan sisanya bit 0.
2. Penambahan Nilai Panjang Pesan Semula
    1. Pesan yang telah diberi bit-bit pengganjal selanjutnya ditambah lagi dengan 64 bit yang menyatakan panjang pesan semula.
    2. Setelah ditambah dengan 64 bit, panjang pesan sekarang menjadi 512 bit.
3. Inisialisai Penyangga MD
    1. SHA membutuhkan 5 buah penyangga (buffer) yang masing-masing panjangnya 32 bit (MD5 hanya mempunyai 4 buah penyangga). Total panjang penyangga adalah 5, maka N 32 = 160 bit, adalah 5 x 32 = 160 bit Keempat penyangga ini menampung hasil antara dan hasil akhir.
    2. Kelima penyangga ini diberi nama A, B, C, D, dan E. Setiap penyangga diinisialisasi dengan nilai-nilai (dalam notasi HEX) sebagai berikut:
        A = 67452301
        B = EFCDAB89
        C = 98BADCFE
        *D* = 10325476
        *E* = C3D2E1F0
    3. Proses $H_{SHA}$ terdiri dari 80 buah putaran (MD5 hanya 4 putaran), dan masing-masing putaran menggunakan bilangan penambah $K_t$, yaitu:

Putaran 0 t 19 $K_t$ = 5A827999
Putaran 20 t 39 $K_t$ = 6ED9EBA1
Putaran 40 t 59 $K_t$ = 8F1BBCDC
Putaran 60 t 79 $K_t$ = CA62C1D6

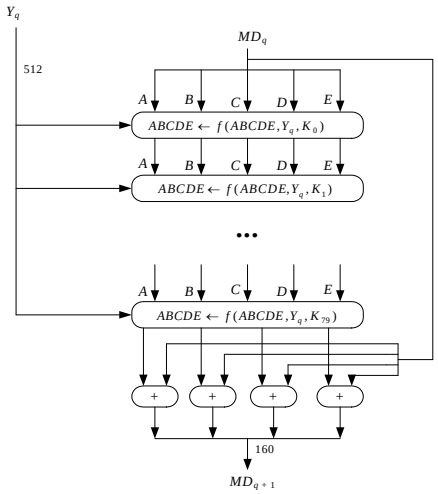
4. Pengolahan pesan dalam blok berukuran 512 bit.

### 2.5 Pengolahan Pesan dalam Blok Berukuran 512 bit.

1. Pesan dibagi menjadi *L* buah blok yang masing-masing panjangnya 512 bit ($Y_0$ sampai $Y_{L-1}$).

ISSN 2548-9739

2. Setiap blok 512-bit diproses bersama dengan penyangga MD menjadi keluaran 128-bit, dan ini disebut proses $H_{SHA}$. Gambaran proses $H_{SHA}$ diperlihatkan pada Gambar.




Gambar 2.2 Pengolahan blok 512 bit (Proses $H_{SHA}$)

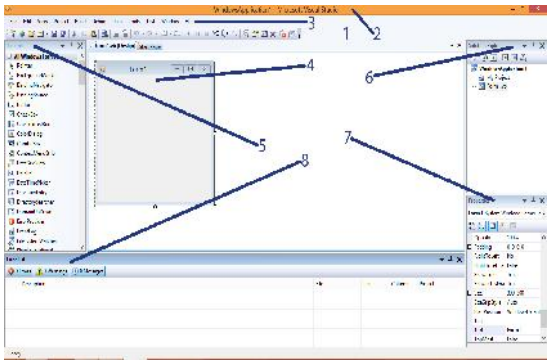
3. Proses $H_{SHA}$ terdiri dari 80 buah putaran (MD5 hanya 4 putaran), dan masing-masing putaran menggunakan bilangan penambah $K_t$, yaitu:
   Putaran $0 \le t \le 19$ $K_t$ = 5A827999
   Putaran $20 \le t \le 39$ $K_t$ = 6ED9EBA1
   Putaran $40 \le t \le 59$ $K_t$ = 8F1BBCDC
   Putaran $60 \le t \le 79$ $K_t$ = CA62C1D6
4. Pada Gambar II.4, $Y_q$ menyatakan blok 512-bit ke-$q$ dari pesan yang telah ditambah bit-bit pengganjal dan tambahan 64 bit nilai panjang pesan semula. $MD_q$ adalah nilai *message digest* 160-bit dari proses $H_{SHA}$ ke-$q$. Pada awal proses, $MD_q$ berisi nilai inisialisasi penyangga MD.
5. Setiap putaran menggunakan operasi dasar yang sama (dinyatakan sebagai fungsi *f*).

## 2.6 Pengertian Microsoft Visual Basic. Net

Menurut Rusmawan (2011, h.1), menyatakan bahwa *Visual Basic* 2008 merupakan salah satu paket bahasa pemrograman dari Visual Studio 2008. Microsoft Visual Basic atau lebih dikenal dengan VB, merupakan salah satu perangkat lunak (*software*) yang digunakan untuk pengembangan *software* (*software development*), dimana dengannya kita dapat membuat aplikasi-aplikasi sederhana bahkan profesional. Versi yang terbaru dari Microsoft Visual Basic adalah Microsoft Visual Studio 2008.

Pada versi terbaru ini kita dihadapkan pada tampilan grafis dan lingkungan IDE (*Integrated Development Environment*) yang sangat berbeda dari versi sebelumnya, yaitu suatu lingkungan kerja bagi programmer yang didukung oleh *compiler*, *editor* baik secara grafis maupun kode, dan mempermudah pengembang untuk mencari objek-objek pada *toolbox* yang diinginkan. Hal itu dapat langsung diletakkan pada objek form dengan meng-klik sebuah objek dan meletakkannya diatas form.

Berikut ini adalah lingkungan kerja *Microsoft Visual Basic*. *Net* 2008 secara umum sebagai berikut (Rusmawan ,2011):




Gambar 2.3 Tampilan Awal Visual Basic. Net 2008

1. Tampilan awal, berfungsi untuk memulai suatu program Microsoft Visual Basic. Net

ISSN 2548-9739

2008 yang akan dibuat terlihat seperti gambar berikut:

2. *Title Bar*, berfungsi untuk menampilkan nama *project* yang aktif.
3. *Menu Bar*, berfungsi untuk mengolah fasilitas yang dimiliki oleh VB. Net 2008, sedangkan *tool bar* berfungsi untuk melakukan perintah khusus secara cepat.
4. *Form* adalah objek utama yang berfungsi untuk meletakkan objek-objek yang terdapat pada *toolbox*.
5. *Toolbox*, berfungsi untuk menyediakan objek-objek atau komponen yang digunakan dalam merancang sebuah *form* pada program aplikasi.
6. *Solution Explorer*, berfungsi untuk menampilkan *project* beserta *file* pendukung yang terdapat pada sebuah program aplikasi.
7. *Properties Windows*, berfungsi mengatur properties pada objek yang diletakkan pada sebuah form.
8. *Error List*, suatu jendela yang berfungsi menampilkan setiap kesalahan dari pembuatan kode program.
9. *Code Editor*, suatu teks editor yang digunakan untuk merancang dan menata sebuah kode aplikasi dalam bentuk kode-kode baris program.
10. *Debugger*, digunakan sebagai monitor terhadap kesalahan yang terjadi, seperti *syntax error*, *runtime error*, *logic error*.

Adapun kelebihan dari pemrograman Visual Basic. Net 2008 dibandingkan *software* lain, diantaranya:

1. Kurva pembelajaran dan pengembangan yang lebih singkat dibandingkan bahasa pemrograman yang lain seperti C/C++, *Delphi* atau bahkan *PowerBuilder* sekalipun.
2. Menghilangkan kompleksitas pemanggilan fungsi *Windows* API, karena banyak fungsi-fungsi tersebut sudah di "*embeded*" ke dalam syntax Visual Basic.
3. Cocok digunakan untuk mengembangkan aplikasi/ program yang bersifat "*Rapid Application Development*".
4. Digunakan oleh hampir semua keluarga *Microsoft Office* sebagai bahasa Macronya.
5. Dapat membuat ActiveX Control.
6. Dapat menggunakan OCX/ komponen yang disediakan oleh pihak ketiga sebagai pengembangan.
7. Menyediakan wizard yang sangat berguna untuk mempersingkat/ mempermudah pengembangan aplikasi.
8. Mendekati OOP (Object Oriented Programming).
9. Dapat diintegerasikan dengan internet, baik pada sisi client maupun server.

## 3 METODOLOGI PENELITIAN

Subyek penelitian ini adalah Menganalisa pengkodean dalam penyandian data dengan algoritma kriptografi Secure Hash Algorithm (SHA), sehingga nantinya dari proses keamanan akan menghasilkan algoritma yang baru yang mempunyai tingkat kesulitan pengamanan data yang tinggi dan cepat dalam proses enkripsi maupun dekripsi.

Adapun metodologi yang digunakan pada penyusunan penelitian diatas antara lain adalah : Studi pustaka, pengumpulan jurnal ilmiah, pengumpulan ebook dan uji coba progam.

## 4 ANALISA DAN PERANCANGAN

### 4.1 Proses Enkripsi dan Dekripsi Algoritma SHA

Adapun pesan yang akan di proses enkripsi dan dekripsi pengkodean pesan dengan algoritma SHA adalah "ROMANUS" . Berikut merupakan langkah –langkah penyelesainnya :

| Merubah | kedalam bilangan b |
|---|---|
| R | 01010011 |
| O | 01010101 |
| M | 01001011 |
| A | 01000001 |
| N | 01010010 |
| U | 01001110 |
| S | 01001111 |

ISSN 2548-9739

1. Pajang pesan (L) = 184
01010011 01010101 01001011 01000001
00110101 00100000 01000001 01000111
00100000 00110001 00111001 00111000
00111001

2.Tambahkan “1” pada akhir pesan
01010011 01010101 01001011 01000001
01010010
00110101 00100000 01000001 01000111
00100000 00110001 00111001 00111000
00111001 1

3. Penambahan “0” pada akhir pesan
Karena L = 184, maka hasinya menjadi = 185 ,sehingga di pelukan “0” sebanyak 448-185 = 263 , maka menjadi
01010011 01010101 01001011
01000001 01010010 01001110
01001111 00100000 00110010
00110101 00100000 01000001 01000111
01010101 01010011 01010100
01010101 01010011
00100000 00110001 00111001 00111000
00111001 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

4. Tambakan 64 bit representasi dari panjang pesan asli, untuk L = 184 maka representasi dari 64 bit panjang pesan adalah 00000000 00000000 00000000 00000000 00000000 00000000 00000000 10111000

Maka setelah di tambahkan menjadi
01010011 01010101 01001011 01000001
01010010 01001110 01001111
00100000 00110010
00110101 00100000 01000001 01000111
01010101 01010011 01010100
01010101 01010011
00100000 00110001 00111001 00111000
00111001 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
10111000

Sehingga pesan yang dihasilkan panjangnya menjadi 512 bit

5. Inisialisasi penyangga (buffer) MD. SHA membutuhkan 5 buah penyangga (buffer) yang masing- masing panjangnya 32 bit. Total panjang penyangga adalah 5 x 32 = 160 bit. Kelima penyangga ini menampung hasil antara dan hasil akhir. Kelima penyangga tersebut diberi nama A, B, C, D, dan E. Setiap penyangga diinisialisasi dengan nilai-nilai (dalam notasi HEX) sebagai berikut :

A = 67452301
B = EFCDAB89
C = 98BADCFE
*D* = 10325476
*E* = C3D2E1F0

6. Pengolahan Pesan dalam Blok Berukuran 512 bit. Pesan dibagi menjadi L buah blok yang masingmasing panjangnya 512 bit (Y0 sampai YL-1). Setiap blok 512 diproses bersama dengan penyangga MD menjadi keluaran 128-bit, dan disebut proses HSHA . Proses HSHA terdiri dari 80 buah putaran, dan masing-masing putaran menggunakan bilangan penambah Kt, yaitu :

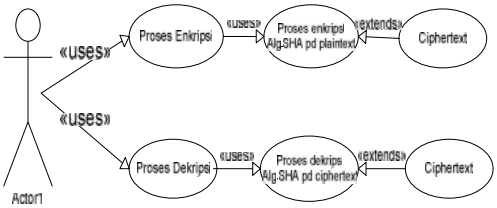
Putaran $0 \leq t \geq 19$ Kt = 5A827999

ISSN 2548-9739

Putaran 20 ≤t ≥39 Kt = 6ED9EBA1
Operasi dasar dari 80 putaran di atas dapat dijabarkan dalam algoritma berikut : For t ←0 to 79 do TEMP ←(a <<< 5) + ft(b,c,d) + e + Wt + Kt) dalam hal ini <<< menyatakan operasi pergeseran circular left shift (CLSs).

### 4.2 Proses Rancangan yang berjalan

Untuk memulai perancangan sistem maka dibuat terlebih dahulu desainnya menggunakan *Unifed Modeling Language* (UML), dimana use case dan activity diagram pada UML untuk mendiskripsikan apa yang harus dilakukan oleh sistem.

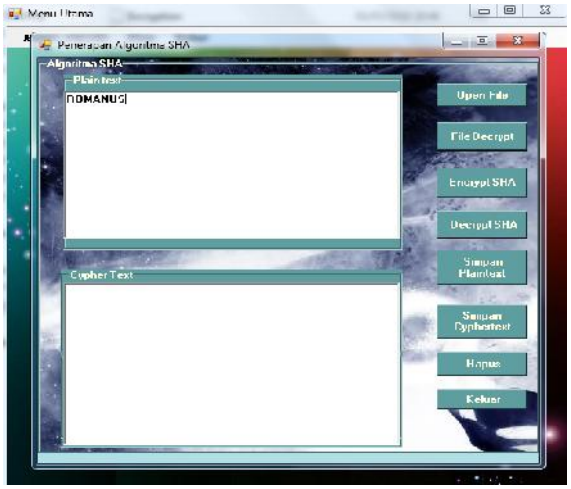



**Gambar 4.1 Use case diagram sistem**

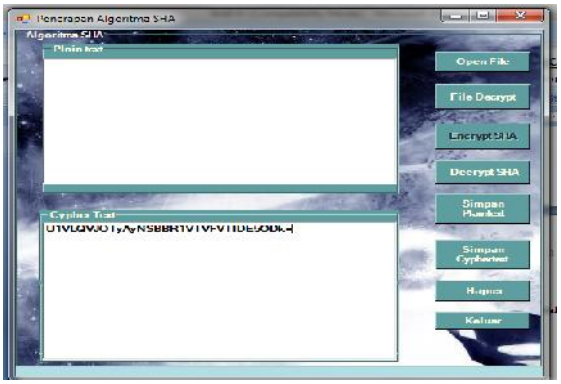
Gambar 4.1 menyatakan diagram use case sistem kriptografi untuk keamanan file text. Use case ini menjelaskan mengenai apa yang bisa dilakukan oleh pengguna. Pengguna dapat mengenkprisikan dan mendekripsi file text, kemudian di proses menggunakan Alogoritma SHA dengan membaca plaintext untuk melindungi charakter yang ada di dalam text.

## 5. IMPLEMENTASI SISTEM

Hasil yang diperoleh dari proses enkripsi teks, berasal dari suatu sting yang diinput yang kemudian diproses dengan menggunakan algoritma SHA dan menghasilkan suatu pesan digest yang memiliki panjang yang tetap yaitu 160 bit. Untuk lebih jelasnya bisa dilihat pada gambar berikut:




Gambar 5.1 Enkripsi Plainteks

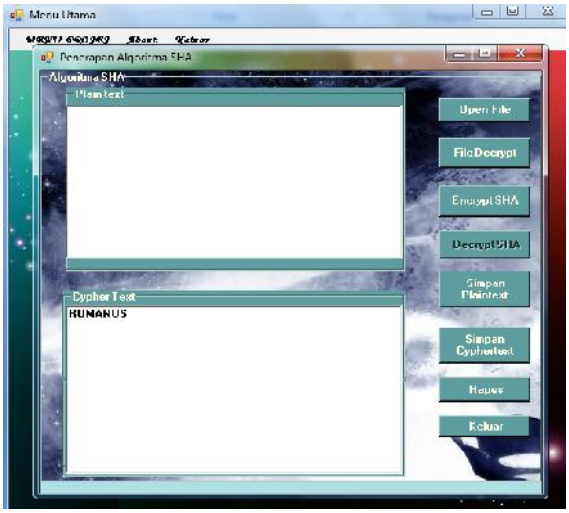



Gambar 5.2 Tampilan Cipherteks

Berikut merupakan pengambilan Cipherteks yang terdapat pada kompter :


Gambar 5.3 Proses Pengambilan Cipherteks

ISSN 2548-9739




Gambar 5.4 Plainteks Awal

## 6. KESIMPULAN DAN SARAN

### 6.1 Kesimpulan

Dari sistem yang telah dirancang dapat di simpulkan bahwa :

1. Aplikasi yang di bangun telah berhasil menyandikan file sehingga keamanan file dapat di wujudkan.
2. File yang telah di enkripsi tidak akan dapat diketahui isinya, kecuali menggunakan aplikasi yang telah di bangun.

### 6.2 Saran

Adapun saran yang telah disimpulkan dari penulisan diatas antara lain :

1. Perangkat lunak dapat dikembangkan menjadi sebuah aplikasi text editor yang memberikan fasilitas tanda tangan digital (digital signature),
2. Perangkat lunak dapat dikembangkan dengan menambahkan kriptanalisis terhadap SHA yang dibahas, sehingga dapat memberikan gambaran mengenai kemanan yang diberikan oleh SHA

## DAFTAR PUSTAKA

[1] Ariyus, Dony, 2010, Kriptografy Keamanan Data dan Komunikasi. Penerbit Graha Ilmu, Yogyakarta
[2] Kurniawan, Erick, 2010, Visual Studio 2011, Penerbit Andi, Yogyakarta.
[3] Munir, Rinaldi, 2006, Kriptografi, Penerbit Informatika, Bandung.
[4] Rusmawan, Uus, 2010, Aplikasi Database VB.6.0, Penerbit Andi, Yogyakarta.S
[5] adikin, Rifki, 2012, Kriptografi. Penerbit Andi, Yogyakarta.
[6] Sugiarti, Yuni, 2013, Analisis Perancangan UML, Penerbit Graha Ilmu, Yogyakarta