Implementasi Secure Hash Algorithm-1 Untuk Pengamanan Data Dalam Library Pada Pemrograman Java

Komang Aryasa*[1], Yesaya Tommy Paulus[2]
[1]Teknik Informatika STMIK Dipanegara Makassar
[2]Sistem Informasi STMIK Dipanegara Makassar
E-mail: *[1]**aryuh09@gmail.com**, [2]nov62003@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-1, 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, yang dapat dibuat dalam bentuk library pada pemrograman java sehingga dapat langsung digunakan untuk proses otentikasi dan keamanan data.

Kata Kunci *— SHA-1, data security, java, hash*

Abstract

Security problem is one of the important aspects of the information system. Once the importance of the value of information causes information is often desired by a particular person can then modify that information. To be able to certify that the information obtained, it must be done so that the authentication on the information known to the authenticity of the information. In this study implemented a method of secure hash algorithm-1, so that the recipient can know the information of changes to important data received from the other party. This method is able to generate a hash value from a string or file. With the slightest change to the information received, it is known authenticity. Secure Hash Algorithm is a hash function that works in one direction, this means that the message has been converted into a message digest can not be restored to its original message. Two different messages will always produce a different hash value. Software design results with different input string length would produce output with fixed string length is 160 bits, which can be made in the form of the Java programming library that can be directly used for authentication and data security.

Keywords *— SHA-1, data security, java, hash*

1. PENDAHULUAN

Masalah keamanan merupakan salah satu aspek penting dari sebuah sistem informasi. Sayang sekali masalah keamanan ini sering kali kurang mendapat perhatian dari para pemilik dan pengelola sistem informasi. Seringkali masalah keamanan berada di urutan kedua, atau bahkan di urutan terakhir dalam daftar hal-hal yang dianggap penting. Apabila mengganggu performansi dari sistem, seringkali keamanan dikurangi atau ditiadakan. Sangat pentingnya nilai sebuah informasi menyebabkan seringkali informasi diinginkan hanya boleh diakses oleh orang-orang tertentu. Jatuhnya informasi ke tangan pihak lain (misalnya pihak lawan bisnis) dapat menimbulkan kerugian bagi pemilik informasi. Sebagai contoh, banyak informasi dalam sebuah perusahaan, oleh karenanya untuk dapat mengesahkan informasi yang didapatkan, maka harus dilakukan *authentication* pada informasi sehingga perusahaan dapat terhindar dari penipuan mengenai informasi tentang produk yang sedang dalam *development*, algoritma-algoritma dan teknik-teknik yang digunakan untuk menghasilkan produk tersebut. Untuk itu keamanan dari sistem informasi yang digunakan harus terjamin dalam batas yang dapat diterima. Usaha-usaha yang dapat dilakukan untuk menjaga keamanan dari sebuah sistem antara lain dengan membangun teknologi kriptografi (dengan enkripsi dan deskripsi). 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 adalah fungsi *hash*[5,7].

Fungsi *hash* adalah sebuah fungsi yang 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*) adalah fungsi *hash* yang bekerja dalam satu arah, ini berarti pesan yang sudah diubah menjadi *message digest* tidak dapat lagi dikembalikan menjadi pesan semula. Dua pesan yang berbeda akan selalu menghasilkan nilai *hash* yang berbeda pula. Sehingga *message digest* dari sebuah *string* dapat dijadikan sebagai sebuah sidik jari.

Pokok permasalahan dalam penelitian ini adalah bagaimana mengimplementasikan algoritma SHA-1 kedalam bahasa pemrograman *Java* dan membentuk sebuah *library* dari kelas-kelas yang telah dibuat sebelumnya. Dan Ruang lingkup penelitian dibatasi pada:

1. Implementasi eksperimen menggunakan bahasa pemrograman *Java*
2. Algoritma enkripsi yang digunakan adalah SHA-1 (*Secure Hash Algorithm-1*)
3. *Library* yang dibentuk adalah ber-ekstensi *.jar.
4. *Input* yang digunakan adalah sebuah *string* dengan panjang maksimum adalah 2^{64} bit.
5. *Output* yang dihasilkan adalah sebuah *string* dengan panjang tetap yaitu 160 bit.
6. Untuk dapat menguji *library* yang dibentuk, dibuatkan sebuah interface yang mana digunakan untuk memasukkan *string* dan akan menghasilkan sebuah *message digest*.

Tujuan penelitian ini adalah membuat sebuah *library,* pada pemrograman java dalam membuat sebuah *message digest* dari suatu *string*. Sehingga dapat langsung digunakan untuk otentikasi dan keamanan data.

1.1. Fungsi Hash

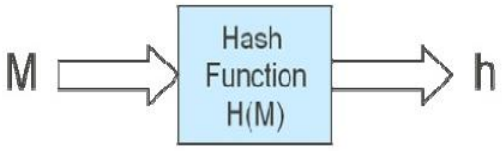
Sebuah fungsi *hash* (*hash function* atau*hash algorithm*) adalah suatu cara untuk menghasilkan sebuah *digital* "*fingerprint*" kecil dari sembarang data [1,2,3]. Fungsi ini memecahkan dan mencampurkan data untuk menghasilkan *fingerprint* yang sering disebut sebagai nilai *hash* (*hash value*). Nilai *hash* ini sering direpresentasikan dengan sebuah *string*pendek dari huruf-huruf dan angka-angka yang kelihatan acak (berbentuk heksadesimal). Sebuah fungsi *hash* yang baik adalah suatu fungsi yang tidak (jarang) memiliki output nilai *hash* yang sama untuk input yang berbeda.

1.2. Fungsi Hash Satu Arah

Fungsi *hash* satu arah (*One-way hash*) adalah fungsi *hash* yang bekerja dalam satu arah [1,2,3]. *String* yang telah diubah menjadi *message digest* tidak dapat lagi dikembalikan menjadi *string* semula. Dua *string* yang berbeda akan selalu menghasilkan nilai *hash* yang berbeda pula. Sifat-sifat dari fungsi *hash* satu arah adalah sebagai berikut :

1. Fungsi *H* dapat diterapkan pada blok data berukuran apa saja.
2. *H* menghasilkan nilai (*h*) dengan panjang tetap (*fixed-length output*).
3. *H*(*x*) mudah dihitung untuk setiap nilai *x* yang diberikan.
4. Untuk setiap *h* yang diberikan, tidak mungkin menemukan *x* sehingga *H*(*x*) = *h*.
5. Untuk setiap *x* yang diberikan, tidak mungkin mencari $y \neq x$ sedemikian sehingga *H*(*y*) = *H*(x).
6. Secara komputasi tidak mungkin mencari pasangan *x* dan *y* sedemikan sehingga *H*(*x*) = *H*(y)

Fungsi hash adalah publik (tidak dirahasiakan), dan keamanannya terletak pada sifat satu arahnya itu. Skema fungsi hash ditunjukkan pada gambar 1 di bawah ini:



Gambar 1. Fungsi *hash* satu arah

1.3. Secure Hash Algorithm (SHA)

Secure Hash Algorithm (SHA) adalah fungsi *hash* satu arah yang dibuat olehNIST (*National Institute of Standard and Technology*) [1,2,3]. SHA dinyatakan sebagai standar fungsi *hash* satu arah. SHA dapat dianggap sebagai kelanjutan pendahulunya MD5 dan dapat dikatakan aman karena dirancang sedemikian rupa sehingga secara komputasi tidak mungkin menemukan *string* yang berkoresponden dengan *message digest* yang diberikan.

1.4. Secure Hash Algorithm-1 (SHA-1)

SHA-1 menerima masukan berupa *string* dengan ukuran maksimum 2^{64} bit. Untuk setiap *string*, SHA-1 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. Berdasarkan cirinya SHA-1 dapat digunakan dengan algoritma kriptografi lainnya seperti *Digital Signature Algorithms* atau dalam generasi angka yang acak (*bits*). SHA-1 dikatakan aman karena proses SHA-1 dihitung secara infisibel untuk mencari *string* yang sesuai untuk menghasilkan *message digest* atau dapat juga digunakan untuk mencari dua *string* yang berbeda yang akan menghasilkan *message digest* yang sama. Untuk SHA-1 ukuran blok*string* -m bit- dapat ditentukan tergantung dari algoritmanya. Pada SHA-1 masing-masing blok*string* mempunyai 512 bit dimana dapat dilakukan dengan 16 urutan sebesar 32 bit. SHA-1 digunakan untuk menghitung *message digest* pada *string* atau *file* data yang diberikan sebagai *input*. Tujuan pengisian *string* adalah untuk menghasilkan total dari *string* yang diisi menjadi perkalian dari 512 bits. Beberapa hal yang dilakukan dalam pengisian *string*:

a. Panjang dari *string*, M adalah k bits dimana panjang $k < 2^{64}$. Tambahkan bit "1" pada akhir *string*. Misalkan *string* yang asli adalah "01010000" maka setelah diisi menjadi "010100001".

b. Tambahkan bit "0", angka bit "0" tergantung dari panjang *string*. Misalnya :*String* asli yang merupakan bit string: abcde
01100001 01100010 01100011 01100100 01100101.
Setelah langkah (a) dilakukan
01100001 01100010 01100011 01100100 01100101.

Panjang k = 40 dan angka bit di atas adalah 41 dan 407 ditambah bit "0" (448 – (40+1) = 407). Kemudian diubah dalam hex:
61626364 65800000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000

c. Untuk memperoleh 2 kata dari k, angka bit dalam *string* asli yaitu jika $k < 2^{32}$ maka kata pertama adalah semua bit "0". Maka gambaran dari 2 kata dari k = 40 dalam hex adalah 00000000 00000028.
61626364 65800000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000028

SHA-1 menggunakan urutan fungsi logika yang dilambangkan dengan f_0, f_1,…, f_{79}. Untuk masing-masing f_t, dimana $0 \leq t < 79$ akan menghasilkan output sebanyak 32 bit. Fungsinya adalah sebagai berikut:

$$f_t(B, C, D) = \begin{cases} (B \wedge C) \vee (\neg B \wedge D) & 0 \leq t \leq 19 \\ B \oplus C \oplus D & 20 \leq t \leq 39 \\ (B \wedge C) \vee (B \wedge D) \vee (C \wedge D) & 40 \leq t \leq 59 \\ B \oplus C \oplus D & 60 \leq t \leq 79 \end{cases}$$

⊕ = fungsi XOR (1)

Konstanta kata yang digunakan pada SHA-1 yang disimbolkan secara berurutan dari K(0), K(1),…, K(79) dalam bentuk *hex* seperti pada fungsi 02 berikut:

$$K_t = \begin{cases} 5A827999 & 0 \leq t \leq 19 \\ 6ED9EBA1 & 20 \leq t \leq 39 \\ 8F1BBCDC & 40 \leq t \leq 59 \\ CA62C1D6 & 60 \leq t \leq 79 \end{cases}$$

(2)

Algoritma SHA-1 dapat diringkas sebagai berikut[6,9]:

a. Penghitungan menggunakan dua *buffer* dimana masing-masing *buffer* terdiri dari lima sebesar 32 bit kata dan urutan 80 juga sebesar 32 bit kata. Lima kata pertama pada *buffer* kata diberi nama A, B, C, D, E sedangkan lima kata kedua diberi nama H_0, H_1, H_2, H_3, dan H_4. Kemudian pada 80 kata yang berurutan diberi nama W_0, W_1, …, W_{79} dan pada penghitungan ini juga memakai sebuah variabel sementara, TEMP.
b. Lakukan pengisian *string*, M dan kemudian kirimkan *string* tersebut ke dalam N 512 bit blok *string*, $M^{(1)}$, $M^{(2)}$, …, $M^{(n)}$. Caranya : 32 bit pertama dari blok *string* ditunjukkan ke $M_0^{(i)}$, lalu 32 bit berikutnya adalah $M_1^{(i)}$ dan selanjutnya berlaku hingga $M_{15}^{(i)}$.

c. Inisialisasi Nilai *Hash* (dalam bentuk hex) :
   H_0 = 67452301 H_3 = 10325476
   H_1 = EFCDAB89 H_4 = C3D2E1F0
   H_2 = 98BADCFE
d. Lakukan proses M_1, M_2, …, $M_n$dengan cara membagi M_i ke dalam 16 kata W_0, W_1, …, W_{15} dimana W_0 merupakan *left most*.
e. Hitung : For t = 16 to 79
   $W_t = S^1(W_{t-3} \square W_{t-8} \square W_{t-14} \square W_{t-16})$
f. Inisialisasi 5 variabel A, B, C, D, dan E dengan nilai *Hash* :
   $A = H_0$; $B = H_1$; $C = H_2$; $D = H_3$; $E = H_4$.
   Hitung: For t = 0 to 79
   $TEMP = S^5(A) + f_t(B,C,D) + E + Wt + Kt$
   $E = D$; $D = C$; $C = S^{30}(B)$; $B = A$; $A = TEMP$.
g. Hitung Nilai *Hash* :
   $H_0 = H_0 + A$; $H_1 = H_1 + B$;
   $H_2 = H_2 + C$; $H_3 = H_3 + D$;
   $H_4 = H_4 + E$.

Hasil dari *message digest* sebesar 160 bit dari *string*, M adalah: H_0 H_1 $H_2H_3H_4$**.**

1.5. Pemrograman Java

Implementasi algoritma SHA-1 ini di buat dari bahasa pemrograman java yang akan digenarate menjadi sebuah *library*. Javaadalah bahasa pemrograman yang dapat dijalankan di berbagai komputer termasuk telepon genggam. Bahasa ini awalnya dibuat oleh James Gosling saat masih bergabung di *Sun Microsystem* saat ini merupakan bagian dari *Oracle* dan dirilis tahun 1995. Bahasa ini banyak mengadopsi sintaks C dan C++ namun dengan sintaks model objek yang sederhana. *Java* merupakan bahasa pemrograman yang bersifat umum, dan secara khusus didesain untuk memanfaatkan dependensi implementasi seminimal mungkin. Karena fungsionaitasnya yang memungkinkan aplikasi *Java* mampu berjalan di beberapa *platform* sistem operasi yang berbeda [4,8].

2. METODE PENELITIAN

2.1. Jenis Penelitian

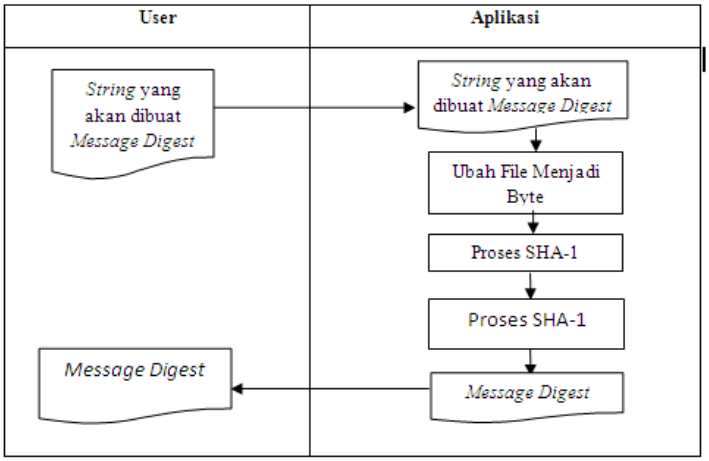
Jenis Penelitian dalam penelitian ini adalah penelitian eksperimental yaitu membuat sebuah *library*, yang dapat digunakan sebagai sarana bagi *programmer Java* dalam membentuk/ membuat sebuah *message digest* dari suatu *string*.

2.2. Rancangan Sistem

Untuk lebih memahami sistem yang akan dibentuk, maka dibuatkan blok diagram pada gambar 2 berikut:

Gambar 2. Blok Diagram Secara Umum

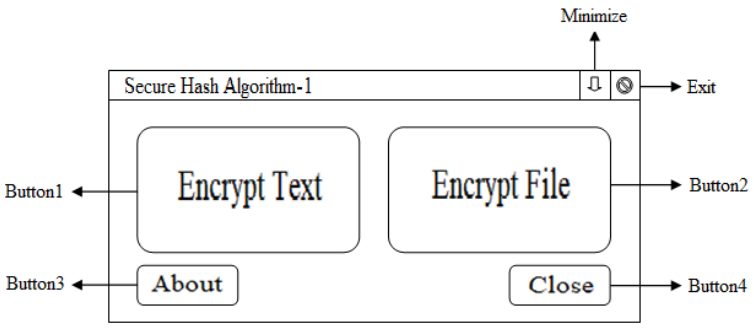
Pada aplikasi yang akan dibuat, data yang diinput berupa data teks atau file yang akan dibuatkan *message digest*-nya dimana pesan memiliki panjang maksimum adalah 2^{64} bit, kemudian pada awal pesan ditambahkan *padding* bit sehingga panjang pesan menjadi 448 bit, dan tambahkan 64 bit yang menyatakan panjang pesan. Setelah itu pesan dibagi ke dalam blok-blok dengan panjang 512 bit per blok dan pada masing-masing blok dilakukan proses SHA-1. Setelah proses SHA-1 dilakukan pada masing-masing blok maka didapatkan hasil sebuah string dengan panjang tetap yakni 160 bit. Untuk mengetahui alir dokumen dari sistem bisa dilihat pada gambar 3 berikut:



Gambar 3. Bagan Alir Dokumen SHA-1

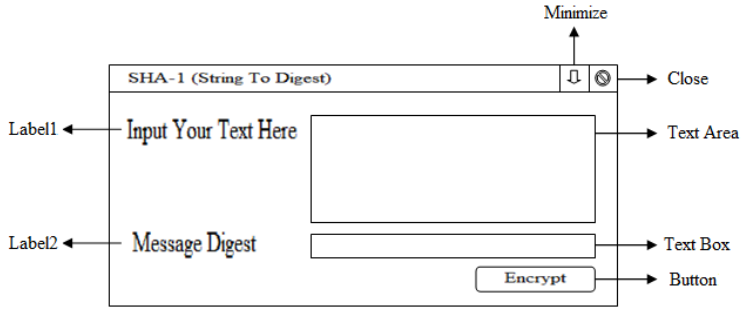
2.3. Rancangan Antarmuka Aplikasi

2.3.1. Form Utama



Gambar 4. Rancangan antarmuka Form Utama

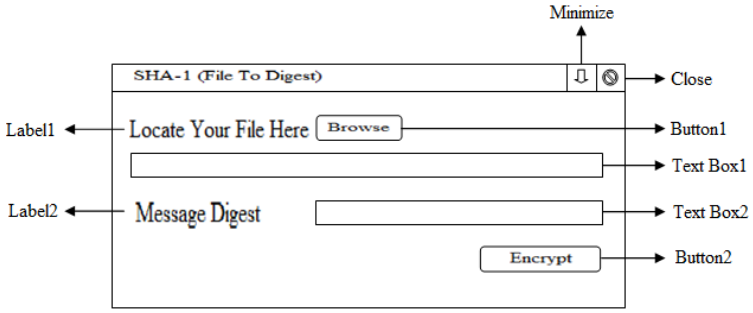
Form Utama seperti pada gambar 4 diatas merupakan antarmuka utama yang digunakan untuk menjalankan dua fungsi dari aplikasi *Secure Hash Algorithm*-1 (SHA-1) yaitu fungsi enkripsi teks dan enkripsi file.



Gambar 5. Rancangan antarmuka Form Enkrip Teks

Form Enkrip Teks pada gambar 5 merupakan antarmuka yang digunakan untuk menjalankan fungsi enkripsi teks, dari suatu pesan berupa teks yang diiput menjadi suatu pesan *digest*.

2.3.2. Form Enkrip File

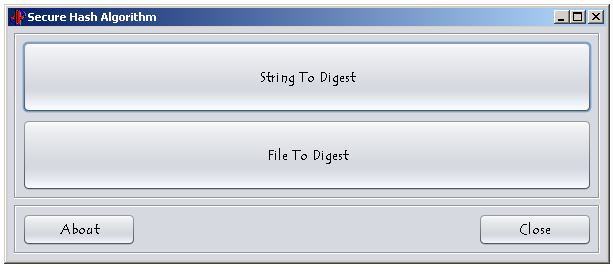


Gambar 6. Rancangan antarmuka Form Enkrip File

Form Enkrip File pada gambar 6 merupakan antarmuka yang digunakan untuk menjalankan fungsi enkripsi file, dari suatu pesan yang berada dalam sebuah file menjadi suatu pesan *digest*.

3. HASIL PENELITIAN

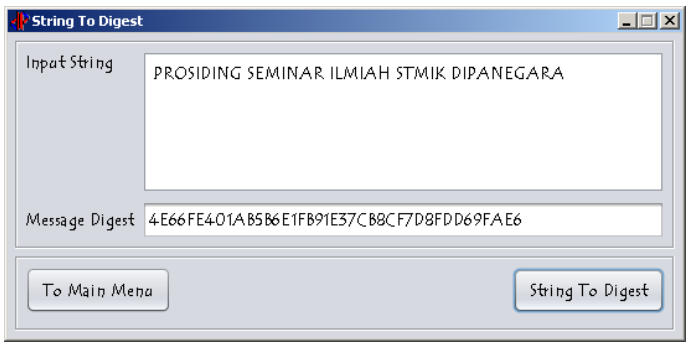
Secara umum proses enkripsi menggunakan algoritma SHA-1 dapat dijalankan atau diproses dengan dua proses input string yaitu *string to digest* dan *file to digest*, seperti yang terlihat pada gambar 7 berikut:



Gambar 7. Tampilan aplikasi utama

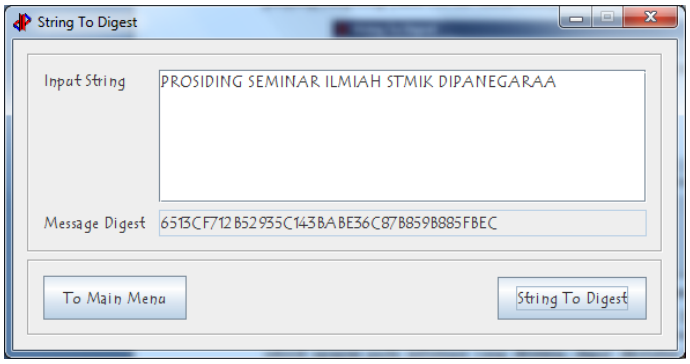
3.1. Hasil Enkripsi Teks (String to Digest)

Hasil yang diperoleh dari proses enkripsi teks, berasal dari suatu sting yang diinput yang kemudian diproses dengan menggunakan algoritma SHA-1 dan menghasilkan suatu pesan digest yang memiliki panjang yang tetap yaitu 160 bit. Untuk lebih jelasnya bisa dilihat pada gambar 8 berikut:



Gambar 8. Tampilan aplikasi utama

Pada gambar 8 diatas dengan input string PROSIDING SEMINAR ILMIAH STMIK DIPANEGARA akan menghasilkan panjang string 160 bit, dengan menggunakan metode SHA-1 apabila terjadi perubahan sekecil apapun pada informasi yang diterima, dapat diketahui keasliannya melalui message digest. Misalnya akan diubah dengan menambahkan 1 (satu) karakter A diakhir string menjadi PROSIDING SEMINAR ILMIAH STMIK DIPANEGARAA hasil message digest akan berubah secara signifikan seperti pada gambar 9 berikut:

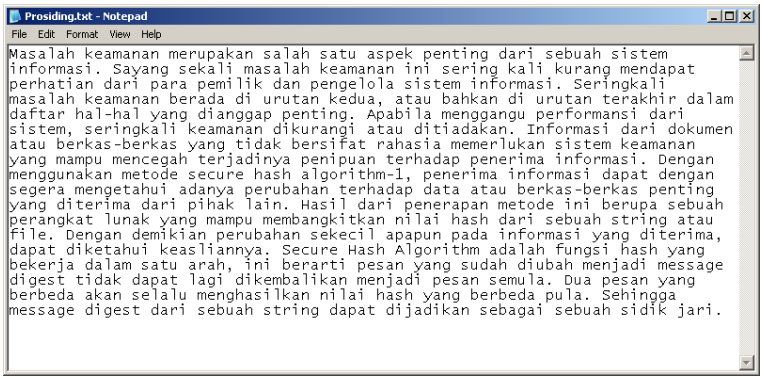


Gambar 9. Tampilan aplikasi utama

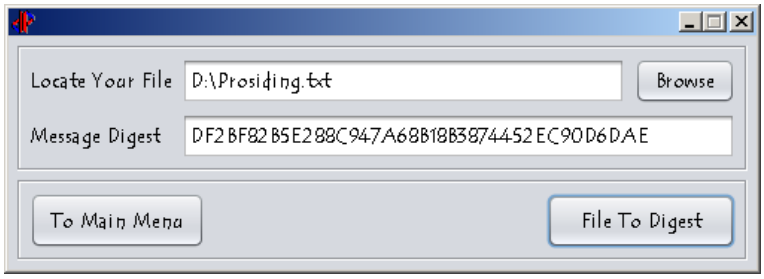
Dengan panjang string yang berbeda seperti pada gambar 8 dan gambar 9 akan tetap menghasilkan output string yang tetap yaitu 160 bit dan message digest akan berubah secara signifikan , hal ini akan berbeda dengan metode yang lain seperti md5 yang hanya menghasilkan panjang output 128 bit saja, ukuran *digest* yang dihasilkan MD5 lebih pendek ketimbang *digest* yang dihasilkan SHA-1. *Digest* MD5 juga cukup pendek untuk dipecahkan oleh serangan *birthday attack* dalam waktu relatif singkat [10].

3.2. Hasil Enkripsi File (File to Digest)

Hasil yang diperoleh dari proses enkripsi file, berasal dari suatu file yang dipilih dan kemudian diproses dengan menggunakan algoritma SHA-1 dan menghasilkan suatu pesan digest yang memiliki panjang yang tetap yaitu 160 bit. Untuk lebih jelasnya bisa dilihat pada gambar 10 dan gambar 11 berikut:



Gambar 10. Pesan yang berasal dari sebuah file



Gambar 11. Tampilan aplikasi utama

Pada gambar 10 adalah isi dari file ***Prosiding.txt*** yang dipilih melalui form seperti pada gambar 1. File to digest pada gambar 11 dengan file ***Prosiding.txt*** yang isi stringnya cukup panjang seperti pada gambar 11 akan tetap menghasilkan output 160 bit sama halnya dengan string to digest pada gambar 8 dan gambar 9 yang panjang stringnya lebih pendek dari file to digest.

4. KESIMPULAN

Hasil rancangan perangkat lunak dengan penerapan metode *secure hash algorithm*-1 dengan input panjang string yang berbeda akan menghasilkan output dengan panjang string tetap yaitu 160 bit. Penerapan metode *secure hash algorithm*-1 untuk keamanan data dalam rancangan ini dapat dibuat dalam bentuk *library* pada bahasa pemrograman java sehingga dapat langsung diimplementasikan untuk otentikasi atau pengamanan data maupun untuk pembuatan tanda tangan digital.

5. SARAN

Perangkat lunak dapat dikembangkan menjadi sebuah aplikasi text editor yang memberikan fasilitas tanda tangan digital (digital signature), sehingga diharapkan dapat digunakan oleh user dalam kehidupan sehari-hari seperti melakukan transaksi online. Perangkat lunak dapat dikembangkan dengan menambahkan kriptanalisis terhadap SHA-1 yang dibahas, sehingga dapat memberikan gambaran mengenai kemanan yang diberikan oleh SHA-1.

DAFTAR PUSTAKA

[1] Ariyus, D., 2008, *Pengantar Ilmu Kriptografi: Teori, Analisis, dan Implementasi*, Penerbit Andi, Yogyakarta.

[2] _____, 2005, *Computer Security*, Penerbit Andy, Yogyakarta

[3] _____, 2005, *Kriptografi Keamanan Data dan Komunikasi*, Graha Ilmu, Yogyakarta.

[4] Indrajani, Martin, 2007, *Pemrograman Berbasis Objek dengan Bahasa Java*, Elex Media Komputindo, Jakarta.

[5] Munir, R., 2006, *Kriptografi*, Penerbit Informatika, Bandung.

[6] Huda, M. 2009, Perkembangan Enkripsi Fungsi Hash pada SHA (Secure Hash Algorithm), *Jurnal Ilmu Komputer dan Teknologi Informasi*, Vol 3, No 2.

[7] Rahardjo, B., 2008, *Keamanan Sistem Informasi*, PT Insan Infonesia Bandung dan PT INDOCISC Jakarta.

[8] Fikri, R., Adam, I. F., Prakoso, I., 2005, *Pemrograman Java*, Andi, Yogyakarta.

[9] Alam, S., 2012, Otentikasi Nilai Hash Menggunakan Metode SHA, *Prosiding Seminar Ilmiah Sistem Informasi dan Teknologi Informasi*, Makassar.

[10] Wang W., Yu, H., 2005, How to Break MD5 and Other Hash Functions, *EUROCYPT*, hal 19-35.