ALGORITMA MESSAGE DIGEST 5 (MD5) DALAM APLIKASI KRIPTOGRAFI

Rezza Mahyudin – NIM : 13505055

Program Studi Teknik Informatika, Institut Teknologi Bandung
Jl. Ganesha 10, Bandung
E-mail : if15055@students.if.itb.ac.id

Abstrak

Masalah keamanan dan kerahasiaan merupakan salah satu aspek penting dari suatu pesan, data, atau informasi. Dalam hal ini sangat terkait dengan betapa pentingnya pesan, data, atau informasi tersebut dikirim dan diterima oleh pihak atau orang yang berkepentingan, serta apakah pesan, data, atau informasi tersebut masih *authenticity*. Pesan, data, atau informasi akan menjadi kurang berguna lagi apabila di tengah jalan informasi itu disadap atau dibajak oleh orang yang tidak berhak atau berkepentingan.

Message Digest 5 (MD5) adalah salah satu alat untuk memberi garansi bahwa pesan yang dikirim akan sama dengan pesan yang diterima, hal ini dengan membandingkan 'sidik jari' atau 'intisari pesan' kedua pesan tersebut. MD5 merupakan pengembangan dari MD4 dimana terjadi penambahan satu ronde. MD5 memproses teks masukkan ke dalam blok-blok bit sebanyak 512 bit, kemudian dibagi ke dalam 32 bit sub blok sebanyak 16 buah. Keluaran dari MD5 berupa 4 buah blok yang masing-masing 32 bit yang mana akan menjadi 128 bit yang biasa disebut nilai hash. Makalah ini bertujuan untuk membahas proses prencanaan dan menganalisa proses keutuhan atau perubahan pesan dengan menggunakan MD5 dan juga dapat menganalisa hasil keluaran dari MD5 yang berupa kecepatan dari proses aplikasi yang dibuat.

Kata kunci : MD5, Kriptografi, hash, sidik-jari

1.Pendahuluan

Keamanan dan kerahasian data pada jaringan komputer saat ini menjadi isu yang sangat penting dan terus berkembang. Beberapa kasus menyangkut keamanan jaringan komputer saat ini menjadi suatu pekerjaan yang membutuhkan biaya penanganan dan pengamanan yang sedemikian besar. Sistem-sistem vital, seperti sistem pertahanan, sistem perbankan, sistem bandara udara dan sistem-sistem yang lain setingkat itu, membutuhkan tingkat keamanan yang sedemikian tinggi. Hal ini lebih disebabkan karena kemajuan bidang jaringan komputer dengan konsep *open system*-nya sehingga siapapun, di mananapun dan kapanpun, mempunyai kesempatan untuk mengakses kawasan-kawasan vital tersebut. Untuk menjaga keamanan dan kerahasiaan pesan, data, atau informasi dalam suatu jaringan komputer maka diperlukan beberapa enkripsi guna membuat pesan, data, atau informasi agar tidak dapat dibaca atau dimengerti oleh sembarang orang, kecuali oleh penerima yang berhak.

Pengamanan pesan, data, atau informasi tersebut selain bertujuan untuk meningkatkan keamanan, juga berfungsi untuk:

- Melindungi pesan, data, atau informasi agar tidak dapat dibaca oleh orang-orang yang tidak berhak.
- Mencegah agar orang-orang yang tidak berhak, menyisipkan atau menghapus pesan, data dan atau informasi.

Salah satu hal yang penting dalam komunikasi menggunakan komputer dan dalam jaringan komputer untuk menjamin kerahasiaan pesan, data, ataupun informasi adalah enkripsi (akan dijelaskan pada bagian berikutnya) . Disini enkripsi dapat diartikan sebagai kode atau *chiper* . Sebuah sistem pengkodean menggunakan suatu tabel atau kamus yang telah didefinisikan untuk kata dari informasi atau yang merupakan bagian dari pesan, data, atau informasi yang di kirim. Sebuah *chipper* menggunakan suatu algoritma yang dapat mengkodekan semua aliran data (*stream*) bit dari suatu pesan asli (*plaintext*) menjadi *cryptogram* yang tidak di mengerti. Karena sistem *chipper* merupakan suatu sistem yang telah siap untuk di outomasi, maka teknik ini digunakan dalam sistem keamanan jaringan komputer.

Salah satu dari bagian kriptografi adalah fungsi hash satu arah. Fungsi hash satu arah adalah dimana kita dengan mudah melakukan enkripsi untuk mendapatkan *cipher*-nya tetapi sangat sulit untuk mendapatkan *plaintext*-nya. Fungsi hash satu arah (*one-way hash function*) digunakan untuk membuat sidik jari (*fingerprint*) dari suatu dokumen atau pesan X. Pesan X (yang besarnya dapat bervariasi) yang akan di-*hash* disebut *pre-image*, sedangkan outputnya yang memiliki ukuran tetap, disebut *hash-value* (nilai hash). Fungsi hash dapat diketahui oleh siapapun, tak terkecuali, sehingga dapat memeriksa keutuhan dokumen atau pesan X tersebut. Tak ada algoritma rahasia dan umumnya tak ada pula kunci rahasia.

Salah satu fungsi hash yang paling banyak digunakan adalah Message Digest 5 (MD5). MD5 merupakan fungsi hash satu arah yang diciptakan oleh Ron Rivest pada tahun 1991 untuk menggantikan *hash-function* sebelumnya, MD4. MD5 adalah salah satu aplikasi yang digunakan untuk mengetahui bahwa pesan yang dikirim tidak ada perubahan sewaktu berada di jaringan.

Secara garis besar, Algoritma MD-5 adalah mengambil pesan yang mempunyai panjang variable, diubah menjadi 'sidik jari' atau 'intisari pesan' yang mempunyai panjang tetap yaitu 128 bit. 'Sidik jari' ini tidak dapat dibalik untuk mendapatkan pesan, dengan kata lain tidak ada orang yang dapat melihat pesan dari 'sidik jari' MD5.

Message Digest atau intisari pesan harus mempunyai tiga sifat penting, yaitu :

1. Bila P diketahui, maka MD5(P) akan dengan mudah dapat dihitung.
2. Bila MD5(P) diketahui, maka tidak mungkin menghitung P.
3. Tidak seorang pun dapat memberi dua pesan yang mempunyai intisari pesan yang sama. $H(M) \neq H(M')$.

2.Kriptografi dan Fungsi Hash Satu Arah

Bab ini akan menerangkan Kriptografi secara umum, prinsip dasar serta tujuan dari Kriptografi, prinsip dasar fungsi hash satu arah, prinsip dasar serta algoritma MD5.

2.1.Prinsip Dasar Kriptografi

Kriptografi (*cryptography*) merupakan ilmu dan seni untuk menjaga kerahasiaan pesan (data atau informasi) dengan cara menyamarkan (*to crypt* artinya menyamar) menjadi bentuk tersandi yang tidak mempunyai makna. Menurut sejarah, kriptografi sudah lama dikenal oleh tentara Sparta di Yunani pada permulaan tahun 400 SM. Mereka menggunakan alat yang disebut *scytale.* Teknik kriptografi seperti ini dikenal dengan nama transposisi chipper, yang merupakan metode enkripsi tertua.

Adapun tujuan dari system kriptografi adalah sebagai berikut :

- Confidentiality
  Memberikan kerahasiaan pesan dan menyimpan data dengan menyembunyikan informasi melalui teknik-teknik enkripsi.
- Message Integrity
  Memberikan jaminan untuk tiap bagian bahwa pesan tidak akan mengalami perubahan dari saat ia dibuat hingga saat ia dibuka.
- Non-repudiation
  Memberikan cara untuk membuktikan bahwa suatu dokumen dating dari seseorang apabila ia mencoba menyangkal memiliki dokumen tersebut.
- Authentication
  Memberikan dua layanan. Pertama, mengidentifikasi keaslian suatu pesan dan memberikan jaminan keautentikannya. Kedua, menguji identitas seseorang apabila ia akan memasuki sebuah system.

Pada dasarnya keamanan dan kerahasian suatu pesan, data, ataupun informasi adalah merupakan hal yang mutlak yang harus kita lakukan. Sedangkan alat untuk melakukan pengamanan data dalam sistem komunikasi jaringan komputer sering disebut *cryptography*. Kriptografi (*cryptography*) merupakan ilmu dan seni penyimpanan pesan, data, atau informasi secara aman. Kritptanalisis (*cryptanalysis)* merupakan ilmu dan seni pembongkaran pesan, data, atau informasi rahasia seperti di atas. Kriptologi (*cryptology*) adalah panduan dari kriptografi dan kriptanalist.

Kriptografi memiliki dua bagian yang penting, yaitu enkripsi dan dekripsi. Proses yang dilakukan untuk mengamankan sebuah pesan (yang disebut *plaintext*) menjadi pesan yang tersembunyi (disebut *ciphertext*) adalah **enkripsi** (*encryption*).

C = E (M), dimana

M = pesan asli,
E = proses enkripsi
C = pesan dalam bahasa sandi (untuk ringkasnya disebut sandi)

Ciphertext adalah pesan yang sudah tidak dapat dibaca dengan mudah. Menurut ISO 7498 – 2, terminologi yang lebih tepat digunakan adalah "*encipher*". Proses sebaliknya, untuk mengubah *ciphertext* menjadi *plaintext*, disebut **deskripsi** (*decryption*). Menurut ISO 7498 – 2, terminologi yang lebih tepat untuk proses ini adalah "*decipher*".

M = D (C), dengan
D = proses dekripsi

Untuk mengenkripsi dan mendekripsi data, kriptografi menggunakan suatu algoritma (cipher) dan kunci (key). Cipher adalah fungsi matematika yang digunakan untuk mengenkripsi dan mendekripsi data. Sedangkan kunci merupakan sederetan bit yang diperlukan untuk mengenkripsi dan mendekripsi data.

Algoritma kriptografi modern tidak lagi mengandalkan keamanannya pada kerahasiaan algoritma tetapi kerahasiaan kunci. Plaintext yang sama bila disandikan dengan kunci yang berbeda akan menghasilkan ciphertext yang berbeda pula. Dengan demikian algoritma kriptografi dapat bersifat umum dan boleh diketahui oleh siapa saja, akan tetapi tanpa pengetahuan tentang kunci, data tersandi tetap saja tidak dapat terpecahkan. Sistem kriptografi atau *Cryptosystem* adalah sebuah algoritma kriptografi ditambah semua kemungkinan plaintext, chypertext, dan kunci.

Misalnya pada *Caesar chipper*, teknik yang digunakan oleh Julius Caesar, kaisar Romawi, untuk menyandikan pesan yang ia kirim kepada para gubernurnya. Pada *Caesar chipper*, tiap huruf disubtitusi dengan huruf ketiga berikutnya dari susunan alphabet. Dalam hal ini kuncinya adalah jumlah pergeseran huruf (yaitu 3).

```
Plainteks     :  A B C D … Z
Chiperteks    :  D E F G … C
```

Dengan mengkodekan setiap huruf alphabet dengan integer : A = 0, B = 1, …, Z = 25, maka secara matematis *Caesar chipper* menyandikan plainteks p_i menjadi c_i dengan aturan

$$C_i = E (p_i) = (p_i + 3) \bmod 26$$

Sebagai contoh pesan

```
AWASI ASTERIX DAN OBELIX
```

Menjadi

```
DZDVL DVWHULA GDQ REHOLA
```

Penerima pesan mengembalikan lagi chiperteks dengan operasi yang berkebalikan dan menggunakan kunci yang sama, yang secara matematis dapat dinyatakan dengan persamaan

$$P_i = D (c_i) = (c_i - 3) \bmod 26$$

Sehingga chiperteks

```
DZDVL DVWHULA GDQ REHOLA
```

Dapat dikembalikan menjadi plainteks semula :

```
AWASI ASTERIX DAN OBELIX
```

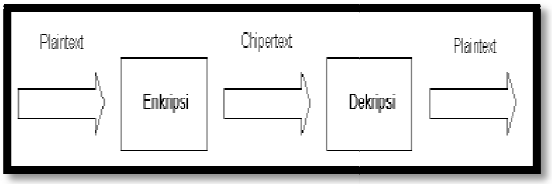
Secara matematis, pada system kriptografi yang menggunakan kunci K, maka fungsi enkripsi dan dekripsi menjadi

$E_{k1}(P) = C$
$D_{k2}(C) = P$

Dan kedua fungsi ini memenuhi

$$D_{k2}(E_{k1}(P)) = P$$

Pada gambar 2.1 dapat dilihat bahwa masukan berupa plaintext akan masuk ke dalam blok enkripsi dan keluarannya akan berupa ciphertext, kemudian ciphertext akan masuk ke dalam blok dekripsi dan keluarannya akan kembali menjadi plaintext semula.



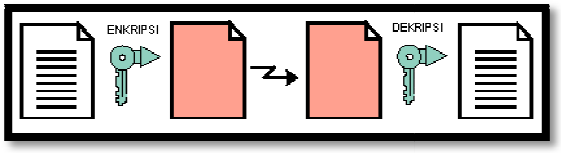
Gambar 2.1 Proses Enkripsi dan Dekripsi

Ada 2 (dua) model algoritma enkripsi yang menggunakan kunci, yaitu kunci simetrik dan kunci asimetrik.

2.2.Kunci Simetris

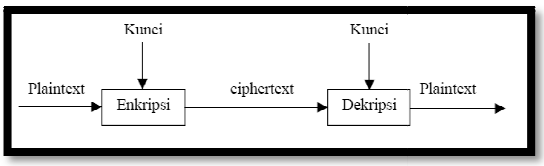
Kunci Simetris adalah jenis kriptografi yang paling umum digunakan. Kunci untuk membuat pesan yang di sandikan sama dengan kunci untuk membuka pesan yang disandikan itu. Jadi pembuat pesan dan penerimanya harus memiliki kunci yang sama persis. Siapapun yang memiliki kunci tersebut termasuk pihak-pihak yang tidak diinginkan dapat membuat dan membongkar rahasia ciphertext. Contoh algoritma kunci simetris yang terkenal adalah DES

(*Data Encryption Standard*). Dibawah ini adalah penggambaran sebuah kunci simetris :



Gambar 2.2 Kunci Simetris

Proses enkripsi – dekripsi algoritma kriptografi kunci simetris dapat dilihat pada gambar di bawah ini :



Gambar 2.3 Proses enkripsi – dekripsi kunci simetris

Enkripsi kunci simetrik dapat dibagi ke dalam 2 (dua) kelompok yaitu metode *stream cipher* dan metode *block cipher.*

Pada metode *stream cipher*, sebuah *chipper* menggunakan suatu algoritma yang dapat mengkodekan semua aliran data (*stream*) bit dari suatu pesan asli (*plaintext*) menjadi *cryptogram* yang tidak dimengerti. Karena system *chipper* merupakan suatu system yang telah siap untuk di outomasi, maka teknik ini digunakan dalam system keamanan jaringan computer. Ada banyak model dan metode enkripsi, salah satu di antaranya adalah enkripsi dengan algoritma *Rivest Code 4* (RC4). Model ini merupakan salah satu algoritma kunci simetris yang berbentuk *stream chipper*. Algoritma ini ditemukan pada tahun 1987 oleh Ronald Rivest dan menjadi simbol keamanan RSA (merupakan singkatan dari tiga nama penemu: Rivest Shamir Adleman) RC4 menggunakan panjang kunci dari 1 sampai 256 bit yang digunakan untuk menginisialisasikan tabel sepanjang 256 bit. Tabel ini digunakan untuk generasi yang berikut dari *peudo random* yang menggunakan XOR dengan *plaintext* untuk menghasilkan *chipertext*. Masing-masing elemen dalam tabel saling ditukarkan minimal sekali.

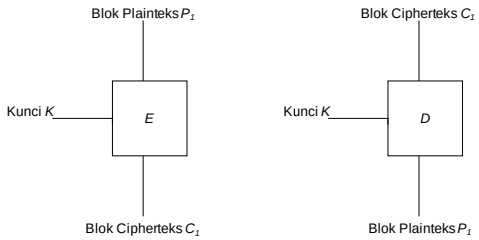
Metode lainnya adalah metode *block chipper* yang memproses sekaligus sejumlah tertentu data (biasanya 64 bit atau 128 bit blok), contohnya : Blowfish, DES, Gost, Idea, RC5, Safer, Square, Twofish, RC6, Loki97, dan lain-lain. Pada metode ini dimaksudkan agar blok tidak terlalu kecil maupun panjang. Blok yang terlalu kecil akan rentan terhadap kebocoran kriptografi dengan hanya pengetahuan berupa pasangan plaintext dan ciphertext, sedangkan blok yang terlalu panjang akan semakin mengurangi efisiensi proses komputasi. Metode *block chipper* mengadopsi konsep one-to-one mapping yang diperlukan agar saat proses dekripsi dari chippertext, hanya ada satu plaintext. Perubahan pada satu bit pada plaintext akan mengubah bit – bit pada chipertext dengan probabilitas 50%.

Ide dasar dari *block chipper* adalah melakukan subtitusi pada blok yang kecil dan permutasi pada blok yang lebar. Ini disebut juga dengan ‘product chiper’. Satu langkah proses ini disebut satu round dan kemudian diulang-ulang. Subtitusi sendiri adalah suatu proses dimana jika kita memiliki input k-bit dengan kemungkinan 2^k, kita harus menentukan pasangan setiap k-bit tersebut yang lebarnya juga k-bit. Subtitusi agak tidak praktis untuk 64-bit, tapi masih praktis untuk 8-bit (misalnya). Sementara permutasi mempunyai definisi yaitu untuk setiap bit dari input k-bit, tiap bit ditukar posisinya ke tempat lain. Misalnya bit ke-1 dari input menjadi bit ke-13. Lalu bit ke-2 dari input jadi bit ke 61, dst.

Ide dasar dari suatu multiple rounds adalah misalnya product chipper dengan S-box 8-bit dan P-box 64-bit. Jika ada 1 bit dari input yang diubah, maka dalam 1 round akan menghasilkan 8-bit perubahan pada output. Pada round kedua: 8-bit yang berubah tersebut tersebar merata pada bit-bit lainnya, sehingga sehabis round ke-2, sudah ‘mengacak’ output dengan cukup baik.

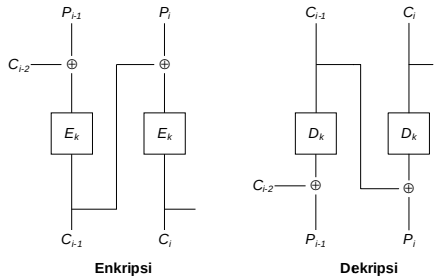
Ada empat (4) macam modus operandi block chipper yaitu :

1. Electronic Code Book



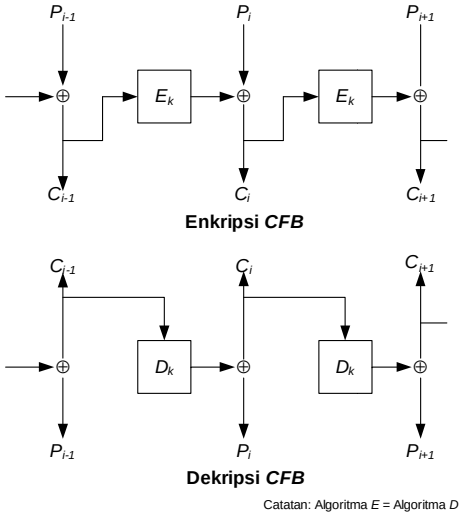
Gambar 2.4 Skema Enkripsi dan Dekripsi dengan Mode *ECB*

2. Cipher Block Chaining



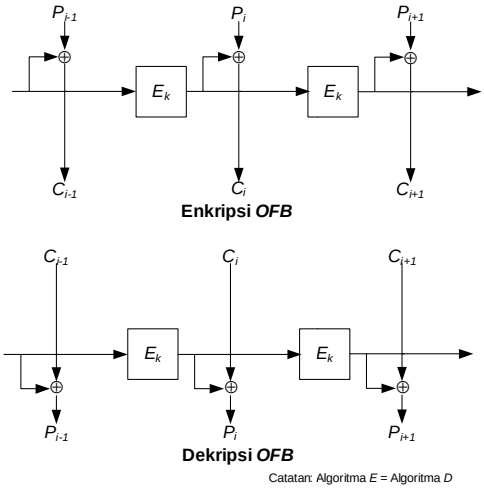
Gambar 2.5 Enkripsi dan Dekripsi dengan Mode *CBC*

3. K-bit Cipher Feedback Mode (CFB)



Gambar 2.6 Enkripsi dan Dekripsi Mode *CFB n*-bit untuk blok *n*-bit

4. K-bit Output Feedback Mode (OFB)



Gambar 2.7 Enkripsi dan Dekripsi *OFB n*-bit untuk blok n-bit

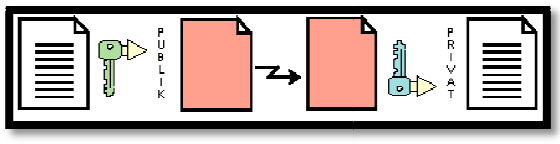
2.3.Kunci Asimetris

Enkripsi kunci asimetrik (biasa disebut enkripsi kuncu public) dibuat sedemikian rupa sehingga kunci yang dipakai untik enkripsi berbeda dengan kunci yang dipakai untuk dekripsi. Enkripsi kunci public disebut demikian karena kunci untuk enkripsi boleh disebarluaskan kepada umum sedangkan kunci untuk mendekripsi hanya disimpan oleh orang yang bersangkutan. Enkripsi asimetik dapat ditulis seperti berikut :

$E_k (P) = C$

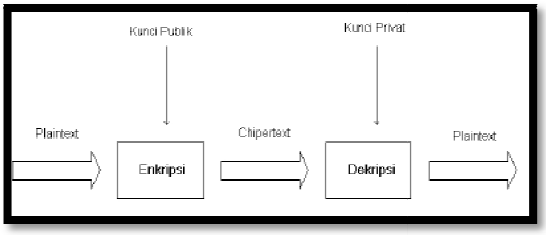
$D_k (C) = P$

Di bawah ini adalah penggambaran suatu kunci asimetris,



Gambar 2.8 Kunci Asimetris

Contohnya seperti pada gambar 2.9, bila seseorang ingin mengirim pesan kepada orang lain maka orang tersebut menggunakan kunci public orang tersebut untuk mengenkripsi pesan yang kita kirim kepadanya lalu orang tersebut akan mendekripsi pesan tersebut dengan kunci privat miliknya.



Gambar 2.9 Enkripsi kunci asimetris

Teknik enkripsi asimetris ini jauh lebih lambat bila dibandingkan dengan enkripsi kunci simetris. Oleh karena itu, biasanya bukanlah pesan itu sendiri yang disandikan dengan kunci asimetris, namun hanya kunci simetrislah yang disandikan dengan kunci asimetris. Sedangkan pesannya dikirim setelah disandikan dengan kunci simetris tadi. Contoh algoritma terkenal yang menggunakan kunci asimetris adalah RSA (merupakan singkatan penemunya yakni Rivest, Shamir dan Adleman).

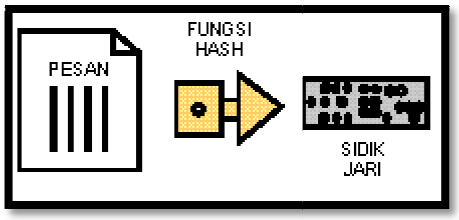
2.4.Tujuan Kriptografi

Seperti juga perkembangan ilmu kriptografi, tujuan-tujuan dari kriptografi teruslah berkembang. Bila pertama kali dibuat hanya untuk keamanan data saja, tetapi sekarang semakin banyak tujuan-tujuan yang ingin dicapai, yaitu :

1. *Privasi*, musuh tidak dapat membongkar tulisan yang kita kirim.
2. *Autentifikasi*, Penerima pesan dapat meyakinkan dirinya sendiri bahwa pesan yang diterima tidak terjadi perubahan dan berasal dari orang yang diinginkan.
3. *Tanda Tangan*, penerima pesan dapat meyakinkan pihak ketiga bahwa pesan yang diterima berasal dari orang yang diinginkan.
4. *Minimal*, tidak ada yang dapat berkomunikasi dengan pihak lain kecuali berkomunikasi dengan pihak yang diinginkan.
5. *Pertukaran bersama*, suatu nilai (misalnya tanda tangan sebuah kontrak) tidak akan dikeluarkan sebelum nilai lainnya (misalnya tanda tangan pihak lain) diterima.
6. *Koordinasi*, di dalam komunikasi dengan banyak pihak, setiap pihak dapat berkoordinasi untuk tujuan yang sama walaupun terdapat kehadiran musuh.

2.5.Prinsip Dasar Fungsi Hash Satu Arah

Fungsi hash satu arah memiliki banyak nama : fungsi pembanding, fungsi penyusutan, intisari pesan, sidik jari, *message integrity check* (MIC) atau pemeriksa keutuhan pesan dan *manipulation detection code* (MDC) atau pendeteksi penyelewengan kode.



Gambar 2.10 Membuat Sidik Jari Pesan

Fungsi hash satu arah dibuat berdasarkan ide tentang fungsi pemampatan. Fungsi hash adalah sebuah fungsi atau persamaan matematika yang mengambil input dengan panjang variable (pre-image) dan merubahnya menjadi panjang yang tetap (biasanya lebih pendek), keluarannya biasa disebut nilai hash.

Data yang disimpan di dalam memori computer perlu ditempatkan suatu cara sedemikian sehingga pencariannya dapat dilakukan dengan cepat. Setiap data yang berupa *record* mempunyai *field* **kunci** yang unik yang membedakan suatu *record* dengan *record* lainnya. Fungsi hash (*hash function*) digunakan untuk menempatkan suatu *record* yang mempunyai nilai kunci *k.* fungsi hash yang paling umum berbentuk

$$h(k) = k \bmod m$$

yang dalam hal ini m adalah jumlah lokasi memori yang tersedia (misalkan memori berbentuk sel-sel yang diberi indeks 0 sampai $m - 1$). Fungsi h di atas menempatkan *record* dengan kunci k pada suatu lokasi memori yang beralamat $h(k)$.

Fungsi hash satu arah adalah sebuah fungsi hash yang berjalan hanya satu arah. Adalah mudah untuk menghitung nilai hash dari *pre-image*, tetapi sangat sulit untuk membangkitkan *pre-image* dari nilai hash-nya.

Metode fungsi hash satu arah adalah berfungsi melindungi data dari modifikasi. Apabila ingin melindungi data dari modifikasi yang tidak terdeteksi, dapat dihitung hasil fungsi hash dari data tersebut, selanjutnya dapat menghitung hasil fungsi hash lagi dan membandingkannya dengan hasil hasil yang pertama, apabila berbeda maka terjadi perubahanselama pengiriman.

Sebagai contohnya, adalah bila si pengirim (A) akan mengirim pesan kepada temannya (B). sebelum mengirim, A melakukan hash pada pesannya untuk

mendapatkan nilai hash, kemudian dia mengirim pesan itu beserta nilai hash-nya. Lalu B melakukan hash untuk mencari nilai hash dari pesan itu, bila terjadi perbedaan maka sewaktu pengiriman telah terjadi perubahan pada pesan tersebut.

Masukan dari fungsi hash satu arah adalah blok pesan dan keluaran dari blok text atau nilai hash sebelumnya ini dapat dilihat pada gambar berikut

Gambar 2.11 Fungsi Hash Satu Arah

Sehingga secara garis besar, hash dari blok M_i adalah:

$$h_i = (M_i , h_{i-1})$$

nilai hash ini bersama blok pesan berikutnya menjadi masukan berikutnya bagi fungsi pemampatan. Nilai hash keseluruhan adalah nilai hash dari blok paling akhir. *Pre-image* sedapatnya mengandung beberapa binary yang menggambarkan panjang dari masukan pesan. Teknik ini digunakan untuk mengatasi masalah yang dapat terjadi bila pesan yang memiliki pesan yang tidak sama memiliki nilai hash yang sama. Metode ini biasa disebut **MD-*strengthening*** atau **penguatan MD.**

2.6.Sistem Kriptografi MD5

Pada bagian ini dijelaskan mengenai system kriptografi MD-5 secara spesifik, yaitu system kriptografi algoritma MD5 yang menjelaskan dari awal masukan hingga keluarannya.

2.6.1.Prinsip Dasar MD5

Message Digest 5 (MD5) adalah salah satu penggunaan fungsi hash satu arah yang paling banyak digunakan. MD5 merupakan fungsi hash kelima yang dirancang oleh Ron Rivest dan didefinisikan pada RFC 1321. MD5 merupakan pengembangan dari MD4 dimana terjadi penambahan satu ronde. MD5 memproses teks masukan ke dalam blok-blok bit sebanyak 512 bit, kemudian dibagi ke dalam 32 bit sub blok sebanyak 16 buah. Keluaran dari MD5 berupa 4 buah blok yang masing-masing 32 bit yang mana akan menjadi 128 bit yang biasa disebut nilai hash.

Pada gambar 2.12 terlihat simpul utama dari MD5. Simpul utama MD5 mempunyai blok pesan dengan panjang 512 bit yang masuk ke dalam 4 buah ronde. Hasil keluaran dari mD5 adalah berupa 128 bit dari byte terendah A dan tertinggi byte D.

2.6.2.Penjelasan Algoritma MD-5

Setiap pesan yang akan dienkripsi, terlebih dahulu dicari berapa banyak bit yang terdapat pada pesan. Kita anggap sebanyak b bit. Di sini b adalah bit non-negatif integer, b bisa saja 0 dan tidak harus selalu berupa bilangan kelipatan delapan (8). Pesan dengan panjang b bit dapat digambarkan sebagai berikut :

m_0_m_1 …..m_(b-1)

Terdapat lima (5) langkah yang dibutuhkan untuk menghitung inti sari pesan. Adapun langkah-langkah tersebut dijelaskan pada subbab-subbab berikut

2.6.2.1.Menambahkan bit

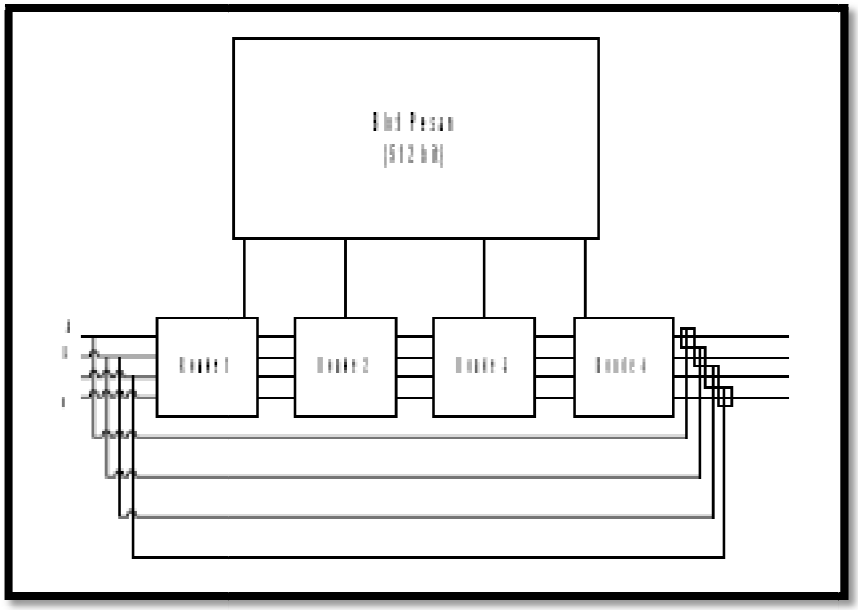
Pesan akan ditambahkan bit-bit tambahan sehingga panjang bit akan kongruen dengan 448 mod 512. Hal ini berarti pesan akan mempunyai panjang yang hanya kurang dari 64 bit dari kelipatan 512 bit. Penambahan bit selalu dilakukan walaupun panjang dari pesan sudah kongruen dengan 448 mod 512 bit. Penambahan bit dilakukan dengan menambahkan “1” di awal dan dengan diikuti penambahan “0” sebanyak yang diperlukan sehingga panjang pesan akan kongruen dengan 448 mod 512.

2.6.2.2.Penambahan Panjang Pesan

Setelah penambahan bit, pesan masih membutuhkan 64 bit agar kongruen dengan kelipatan 512 bit. 64 bit tersebut merupakan perwakilan dari b (panjang pesan sebelum penambahan bit dilakukan). Bit-bit ini ditambahkan ke dalam dua word (32 bit) dan ditambahkan dengan *low-order* terlebih dahulu. Penambahan pesan ini biasa disebut juga ***MD strengthening*** atau **Penguatan MD.**

2.6.2.3.Inisialisasi MD-5

Pada MD5 terdapat empat buah *word* 32 bit register yang berguna untuk menginisialisasi *message digest* pertama kali. Register-register ini di inisialisasikan dengan bilangan hexadecimal.



Gambar 2.12 Simpul utama MD-5

Word A: 01 23 45 67
Word B: 89 AB CD EF
Word C: FE DC BA 98
Word D: 76 54 32 10

Register-register ini biasa disebut dengan nama **chain variable** atau **variabel rantai.**

2.6.2.4.Proses Pesan di Dalam Blok 16 Word

Pada MD5 juga terdapat empat buah fungsi non-linear yang masing-masing digunakan pada tiap operasinya (satu fungsi untuk satu blok), yaitu :

F (X, Y, Z) = (X Y) ((X) Z)

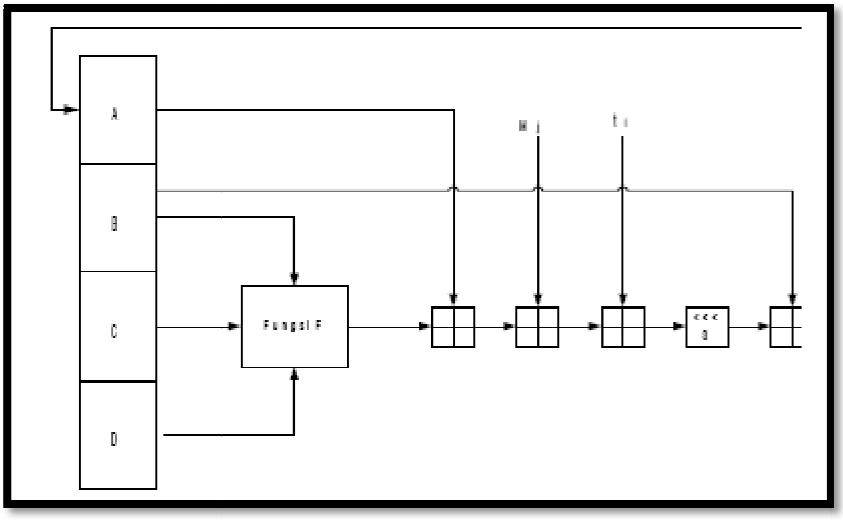
G (X, Y, Z) = (X Z) (Y (Z))

H (X, Y, Z) = X Y Z

I (X, Y, Z) = Y (X (Z))

(untuk XOR, untuk AND, untuk OR, untuk NOT).

Pada gambar 2.13 dapat dilihat satu buah operasi dari MD5 dengan operasi yang dipakai sebagai contoh adalah FF(a,b,c,d,M_j,s,t_i) menunjukan a = b + ((a + F(b,c,d) + M_j + t_i) <<< s)



Gambar 2.13 Satu Buah Operasi MD5

Bila M_j menggambarkan pesan ke-j dari sub blok (dari 0 sampai 15) dan <<<s menggambarkan bit akan digeser ke kiri sebanyak s bit, maka keempat operasi dari masing-masing ronde adalah :

FF(a,b,c,d,M_j,s,t_i) menunjukan a = b + ((a + F(b,c,d) + M_j +t_i) <<<s)

GG(a,b,c,d,M_j,s,t_i) menunjukan a = b + ((a + G(b,c,d) + M_j +t_i) <<<s)

HH(a,b,c,d,M_j,s,t_i) menunjukan a = b + ((a + H(b,c,d) + M_j +t_i) <<<s)

II(a,b,c,d,M_j,s,t_i) menunjukan a = b + ((a + I(b,c,d) + M_j +t_i) <<<s)

2.6.2.5.Keluaran MD-5

Keluaran dari MD-5 adalah 128 bit dari word terendah A dan tertinggi word D masing-masing 32 bit.

3.Perancangan Aplikasi

Spesifikasi :

Aplikasi yang digunakan adalah aplikasi 32-bit yang berjalan pada system operasi *windows 98* ke atas.

Aplikasi berbentuk sebuah file *executable* dengan nama **md5_test.exe**

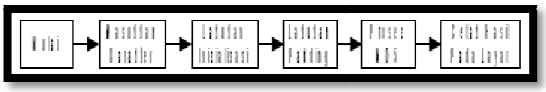
Aplikasi **md5_string** → mencari nilai hash dari masukan berupa file yang dipilih oleh pemakai.

Aplikasi **test suite** untuk memeriksa bahwa program yang sudah dibuat sudah sesuai dengan RFC 1321.

Aplikasi ini juga terdapat contoh penggunaan dari MD5.

3.1.Proses MD5 Dengan Masukan Berupa String

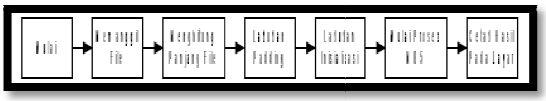
Proses MD5 dengan masukan berupa string adalah proses yang masukannya berupa karakter-karakter yang dimasukan melalui *keyboard*. Hal ini dapat dilihat pada Gambar 3.1



Gambar 3.1 Proses MD5 dengan masukan string

3.2.Proses MD5 Dengan Masukan Berupa File

Proses MD5 dengan masukan berupa file adalah proses MD5 yang masukannya memanggil file yang kenmudian dihitung berapa panjang bitnya, dalam keadaan ini file diperlakukan sebagai bit memori sehingga masukannya tidak terpengaruh pada ekstensinya. Kemudian dilakukan proses MD5. Hal ini dapat dilihat pada gambar 3.2.



Gambar 3.2 Proses MD5 dengan masukan file

3.3.Proses MD5 Sebagai Test Suite

Test suite dilakukan untuk mengetahui apakah program yang dibuat ini sudah benar ataukah masih terdapat kesalahan-kesalahan. Sebagai perbandingannya digunakan hasil yang sudah didapatkan oleh Ron Rivest yang sudah didefinisikan pada RFC 1321. Pada gambar 3.3 dapat dilihat bahwa masukan dari MD5 sudah ditentukan sehinnga hanya membandingkan hasil pada layar dengan yang tercantum pada RFC 1321.

Gambar 3.3 Proses MD5 sebagai Test Suite

3.4.Contoh Aplikasi

Pada contoh aplikasi ini terdapat dua buah aplikasi yang dipakai yaitu simpan dan tampil. Simpan akan menyimpan data yaitu berupa Nama, nilai ujian dan nilai hash-nya. Akhir akan melakukan perhitungan MD5 dari data yang disimpan untuk mendapatkan nilai hash-nya yang kemudian membandingkan nilai hash-nya dengan nilai hash dari data semula, apabila nilai hash yang didapatkan sama, maka data akan ditampilkan. Tetapi bila nilai hash yang didapat berbeda maka pada form akhir akan ditampilkan pesan bahwa data terdapat kesalahan atau perubahan.

3.5.Pengukuran Kecepatan Aplikasi

Pengukuran kecepatan aplikasi merupakan sebuah analisa yang akan dipakai untuk mengukur tingkat kecepatan dari proses mencari nilai hash dari file dengan menggunakan aplikasi MD5.

Adapun rumus yang dipakai dalam aplikasi untuk menghitung kecepatan mencari nilai hash tersebut adalah :

$$\text{Kecepatan} = \frac{\qquad\qquad}{\qquad\qquad}$$

Satuan dari kecepatan ini enkripsi ini adalah Mbytes/detik

Dalam analisa kecepatan ini, akan dilakukan sebanyak 5 (lima) kali pengambilan waktu terbaik yang diperlukan untuk enkripsi untuk setiap filenya kemudian dicari waktu rata-ratanya.

Besar file dan table perbandingan kecepatan maksimum dengan kecepatan rerata terhadap besar file.

4.Analisis Kecepatan MD5

Analisi kecepatan disini adalah analisis tentang kecepatan aplikasi dalam mengenkrip file untuk mencari nilai hash. Analisis dilakukan untuk mencari kecepatan apliksi dengan masukan yang file yang mempunyai perbedaan dalam hal ukuran.

Pengujian dilakukan dengan cara mengenkrip file sebanyak 31 (tiga puluh satu) buah file dengan besar file yang berbeda-beda. Setiap file dilakukan pengambilan waktu eksekusi sebanyak5 kali kemudian mencari waktu rata-ratanya.

4.1.Hasil Pengujian

Hasil pengujian digambarkan dengan tabel hasil pengujian, yang kemudian dijabarkan dengan grafik hasil uji coba terhadap file yaitu Grafik kecepatan aplikasi terhadap besar file, Grafik rata-rata waktu eksekusi terhadap besar file dan tabel perbandingan kecepatan maksimum dengan kecepatan rata-rata terhadap besar file.

5.Kesimpulan

Beberapa kesimpulan yang dapat diperoleh dari makalah ini adalah :

1. *Message Digest 5* (MD5) adalah sebuah fungsi hash satu arah yang mengubah masukan dengan panjang variabel menjadi keluaran dengan panjang tetap yaitu 128 bit.
2. Rata-rata kecepatan dari program aplikasi MD-5 adalah 7,1633 Mbytes/detik
3. Aplikasi yang dibuat hanya efektif digunakan untuk ukuran file kurang dari 40 Mbytes.
4. Sumber daya computer berpengaruh terhadap kecepatan enkripsi.

6.Saran

Adapun saran-saran dari penulis tentang apliksi ini adalah :

Aplikasi MD5 dapat digabungkan dengan algoritma kriptografi lainnya sehingga akan didapatkan algoritma kriptografi yang lebih handal.

DAFTAR PUSTAKA

[1] Munir, Rinaldi. (2003). Bahan Kuliah IF2153 Matematika Diskrit. Departemen Teknik Informatika, Institut Teknologi Bandung.

[2] Pfleeger, Charles P. (1997). Security in Computing Second Edition. Prentice-hall Intenational, inc.

[3] Budi Raharjo. (2002). Keamanan Sistem Informasi Berbasis Internet. PT Insan Indonesia

[4] Kriptografi. (2006). *http://id.wikipedia.org/wiki/Kriptografi*. Tanggal akses : 25 Desember 2006 pukul 21:00

[5] MD5. (2006). *http://id.wikipedia.org/wiki/MD5*. Tanggal akses : 25 Desember 2006 pukul 21:15

[6] Aplikasi Kriptografi dengan Algoritma Message Digest 5. (2006). *www.elektro.undip.ac.id/transmisi/jun06/5_aghus_abp.pdf* . Tanggal akses : 1 Januari 2007 pukul 10:00

[7] Kriptografi. (2006). *http://www.geocities.com/amwibowo/resource/komparasi/bab3.html* . Tanggal akses : 26 Desember 2006 pukul 20:00