

# Analisis Beberapa Fungsi Hash dan Implementasi Fungsi Hash pada Protokol SMTP

Erdiansyah Fajar Nugraha / 13508055<sup>1</sup>

Program Studi Teknik Informatika

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung, Jl. Ganessa 10 Bandung 40132, Indonesia

<sup>1</sup>if18055@students.if.itb.ac.id

**Abstrak**—Fungsi Hash adalah fungsi yang menerima masukan *string* yang panjangnya sembarang, lalu mentransformasikannya menjadi *string* keluaran yang panjangnya tetap (*fixed*) (umumnya berukuran jauh lebih kecil daripada ukuran *string* semula). Fungsi hash adalah fungsi satu-arah (*one-way function*), satu arah di sini yaitu pesan yang sudah diubah tidak dapat dikembalikan lagi menjadi pesan semula (*irreversible*). Beberapa fungsi hash yang sudah ada diantaranya MD2, MD4, MD5, Secure Hash Function (SHA), *snefru*, N-hash, RIPE-MD dan lain-lain.

Aplikasi dari fungsi hash itu sendiri diantaranya untuk menjaga integritas data, menghemat waktu pengiriman, dan menormalkan panjang data yang beraneka ragam.

Pada makalah ini akan dibahas khusus tentang aplikasi fungsi hash untuk menjaga integritas data. Salah satunya yaitu MAC, message authentication code, yaitu fungsi satu arah yang menggunakan kunci untuk membangkitkan nilai hashnya. MAC dilekatkan pada (*embed*) pesan, MAC ini digunakan untuk otentikasi pesan tanpa perlu merahasiakan pesan. MAC ini bukan tanda tangan digital, namun hanya menyediakan otentikasi pengirim dan menjaga integritas pesan.

Aplikasi dari MAC diantaranya otentikasi arsip yang digunakan oleh dua atau lebih pengguna, dan menjaga integritas (keaslian) isi arsip terhadap perubahan. Dalam implementasinya algoritma MAC ini dapat menggunakan fungsi hash yang akan dijelaskan dalam makalah ini.

Sebagai mana kita ketahui ada beberapa protokol pengiriman email diantaranya POP3, IMAP, SMTP, dan HTTP. Implementasi MAC ini akan digunakan pada protokol SMTP. Protokol SMTP (*simple mail transfer*) adalah salah satu protokol standar email. SMTP ini dispesifikasikan untuk outgoing email, yang menggunakan TCP pada port 25. Untuk keamanan email yang menggunakan protokol SMTP, koneksi SMTP ini cukup aman karena menggunakan SSL. Namun protokol tersebut belum ada fitur yang menunjang untuk otentikasi pengirim dan fitur yang menjaga integritas pesan yang dikirim. Oleh karena itu, dirancanglah MAC yang dapat digunakan pada protokol SMTP.

**Kata Kunci**—fungsi hash, MAC, SMTP.

## I. PENDAHULUAN

Fungsi *hash* adalah fungsi yang menerima masukan *string* yang panjangnya sembarang dan mengkonversinya menjadi *string* keluaran yang panjangnya tetap atau *fixed*. Jika *string* menyatakan pesan atau *message*, maka sembarang *message*  $M$  berukuran bebas dikompresi oleh

fungsi *hash*  $H$  dan menghasilkan nilai *hash* (*hash-value*).

Jika fungsi *hash* dituliskan dalam notasi persamaan maka penulisannya adalah sebagai berikut :

$$h = H(M)$$

$h$  merupakan keluaran hash dari fungsi  $H$  untuk masukan  $M$ . Keluaran fungsi hash disebut juga nilai *hash* (*hash-value*) atau pesan ringkas (*message digest*). Fungsi *hash* dapat mengkompresi sembarang pesan yang berukuran beberapa saja menjadi *message digest* yang ukurannya selalu tetap (dan biasanya lebih pendek dari panjang semula).

Fungsi *hash* satu arah (*one-way hash*) adalah fungsi *hash* yang bekerja dalam satu arah, artinya pesan yang diubah dalam *message digest* tidak dapat dikembalikan lagi menjadi pesan semula. Hingga saat ini, fungsi *hash* satu-arah sudah banyak dibuat orang. Contoh fungsi *hash* satu arah yang sudah dibuat adalah MD2, MD4, MD5, SHA, *Snefru*, N-hash, RIPE-MD.

Fungsi *hash* dapat diaplikasikan untuk kehidupan sehari-hari, biasanya fungsi *hash* digunakan untuk mendeteksi atau memverifikasi keaslian suatu salinan dokumen. Pendeteksian keaslian dapat dilakukan dengan cara melakukan *hashing* terhadap dokumen asli dan salinan dokumennya. Jika *message digest* dokumen asli berbeda dengan *message digest* salinan dokumen, maka salinan dokumen tersebut disimpulkan sudah mengalami perubahan atau sudah tidak asli lagi. Fungsi hash ini dapat digunakan sebagai salah satu algoritma MAC (*message authentication code*).

MAC (*message authentication code*) adalah fungsi satu-arah yang menggunakan kunci rahasia (*secret key*) dalam pembangkitan nilai *hash*-nya. MAC biasanya dilekatkan (*embed*) pada pesan. Salah satu penggunaan MAC ini adalah sebagai otentikasi arsip yang digunakan oleh dua atau lebih pengguna, menjaga integritas isi arsip terhadap perubahan. Algoritma yang digunakan dalam MAC ini salah satunya adalah menggunakan fungsi *hash*. Pada makalah ini akan dipilih salah satu fungsi *hash*, yang akan digunakan sebagai algoritma MAC.

Sebagaimana kita ketahui MAC ini membutuhkan sebuah kunci rahasia untuk membangkitkan nilai *hash*-

nya, pada konteks ini kunci rahasia tersebut akan digenerate secara random.

MAC ini akan diimplementasikan pada protokol SMTP. SMTP adalah salah satu protokol standar email. SMTP ini dispesifikasikan untuk outgoing email, yang menggunakan TCP pada port 25. Untuk keamanan email yang menggunakan protokol SMTP, koneksi SMTP ini cukup aman karena menggunakan SSL.

Implementasinya pada protokol SMTP ini, pertama akan dibangkitkan sebuah kunci rahasia yang digenerate secara random, kemudian kunci tersebut akan di gunakan untuk pembangkitan nilai hash pada MAC, untuk mengamankan kunci tersebut maka kunci tersebut akan di enkripsi dengan algoritma kunci public yaitu algoritma RSA yang akan mengenkripsi kunci rahasia tersebut dengan menggunakan kunci public pengguna email, sebelum berkirim email maka akan si pengirim dan si penerima akan menerima sebuah kunci rahasia yang telah dienkripsi yang akan digunakan sebagai otentikasi pengirim pesan dan integritas pesan.

Pada saat pengiriman email, akan disertakan pula MAC pada pesan yang dikirimkan, sehingga penerima pesan dapat mengecek pesan tersebut benar-benar dikirim oleh orang yang bersangkutan dengan alamat email pengirim.

## II. DASAR TEORI

### A. Fungsi Hash

Fungsi *hash* adalah fungsi yang menerima masukkan *string* yang panjangnya sembarang dan mengkonversinya menjadi *string* keluaran yang panjangnya tetap atau *fixed*. Jika *string* menyatakan pesan atau *message*, maka sembarang *message* *M* berukuran bebas dikompresi oleh fungsi *hash* *H* dan menghasilkan nilai *hash* (*hash-value*).

Jika fungsi *hash* dituliskan dalam notasi persamaan maka penulisannya adalah sebagai berikut :

$$h = H(M)$$

*h* merupakan keluaran hash dari fungsi *H* untuk masukkan *M*. Keluaran fungsi hash disebut juga nilai *hash* (*hash-value*) atau pesan ringkas (*message digest*). Fungsi *hash* dapat mengkompresi sembarang pesan yang berukuran beberapa saja menjadi *message digest* yang ukurannya selalu tetap (dan biasanya lebih pendek dari panjang semula).

Fungsi *hash* satu arah adalah fungsi *hash* yang bekerja dalam satu arah, artinya tidak dapat dikembalikan lagi menjadi pesan semula.

Sifat-sifat fungsi *hash* satu arah adalah:

1. Fungsi *H* dapat diterapkan pada blok data berukuran berapa saja.
2. *H* menghasilkan nilai *hash* *h* dengan panjang tetap atau *fixed*.
3. *H(x)* mudah dihitung untuk setiap nilai *x* yang diberikan.
4. Untuk setiap *h* yang dihasilkan, tidak mungkin dikembalikan nilai *x* sedemikian sehingga *H(x) = h*.

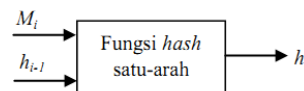
*h*. Itulah sebabnya fungsi *H* dikatakan sebagai fungsi *hash* satu arah (*one-way hash*).

5. Untuk setiap *x* yang diberikan, tidak mungkin mencari *y*  $\neq x$  sedemikian sehingga *H(y) = H(x)*.
6. Tidak mungkin mencari pasangan *x* dan *y* sedemikian sehingga *H(y) = H(x)*.

Masukkan fungsi *hash* adalah blok pesan (*M*) dan keluaran dari *hashing* blok pesan sebelumnya.

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

Skema fungsi *hash* satu arah adalah sebagai berikut:



Gambar skema fungsi hash satu arah

Fungsi *hash* adalah public (tidak dirahasiakan), dan keamanannya terletak pada sifat satu arah.

Ada beberapa fungsi *hash* satu-arah yang sudah pernah dibuat oleh orang, antara lain.

1. MD2, MD4, MD5
2. *Secure Hash Function (SHA)*
3. *Snefru*,
4. *N-hash*,
5. *RIPE-MD*, dan lain-lain

Dalam makalah ini hanya akan dibahas secara ringkas tentang MD4, MD5 dan SHA.

#### 1. Algoritma MD4

*MD4* adalah fungsi *hash* satu arah yang dibuat oleh Ronal Rivest. *MD4* menerima masukkan berupa pesan atau *message* dengan ukuran sembarang dan menghasilkan *message digest* yang panjangnya 128 bit. *MD4* digunakan juga dalam tanda tangan digital (*Digital Signature Standard*).

Langkah-langkah pembuatan *message digest* dengan algoritma *MD4* adalah sebagai berikut:

- a. Penambahan bit-bit pengganjal (*padding bits*)
- b. Penambahan nilai panjang pesan semula
- c. Inisialisasi 4 buah penyangga (*buffer*) MD.
- d. Pengolahan pesan dalam blok berukuran 512 bit

Pesan dibagi menjadi *L* buah blok yang masing-masing panjangnya 512 bit (*Y<sub>0</sub>* sampai *Y<sub>L-1</sub>*) dan setiap blok melewati proses *H<sub>MD4</sub>* yaitu setiap blok 512-bit diproses bersama dengan penyangga *MD4* menjadi keluaran 128-bit.

Proses *H<sub>MD4</sub>* terdiri dari 3 putaran dan masing-masing putaran melakukan operasi dasar *MD4* sebanyak 16 kali. Operasi dasar *MD4* berbeda-beda pada setiap putarannya. Perbedaannya adalah penambahan bilangan heksadesimal pada operasi dasar putaran kedua dan ketiga.

Operasi dasar *MD4* dapat ditulis sebagai berikut (terurut mulai dari putaran 1, 2, dan 3):

$$a = a + CLS_s(a + g(b, c, d) + X[k])$$

$$a = a + CLS_s(a + g(b, c, d) + X[k] + 5A827999)$$

$$a = a + CLS_s(a + g(b, c, d) + X[k] + 6ED9EBA1)$$

Keterangan:

$a, b, c, d$  = empat buah penyangga 32-bit (berisi nilai penyangga A, B, C, dan D).

$g$  = salah satu fungsi F, G, H

$CLS_s$  = *circular left shift* sebanyak  $s$  bit

$X[k]$  = kelompok 32-bit ke- $k$  dari 512 bit *message ke- $q$*

Fungsi  $F, G, H$  merupakan fungsi yang dipakai pada setiap putaran. Fungsi  $F, G$ , dan  $H$  dituliskan sebagai berikut:

$$F(x, y, z) = (x \wedge y) \vee (\sim x \wedge z)$$

$$G(x, y, z) = (x \wedge y) \vee (x \wedge z) \vee (y \wedge z)$$

$$H(x, y, z) = x \oplus y \oplus z$$

Setelah putaran ketiga,  $a, b, c$ , dan  $d$  ditambahkan ke  $A, B, C$ , dan  $D$ , selanjutnya algoritma memproses untuk blok data selanjutnya ( $Y_{q+1}$ ). Keluaran akhir dari algoritma MD4 adalah hasil penyambungan dari bit-bit di  $A, B, C$ , dan  $D$ .

## 2. Algoritma MD5

MD5 adalah fungsi *hash* satu arah yang dibuat oleh Ronal Rivest. Algoritma MD5 merupakan perbaikan dari algoritma MD4 setelah algoritma MD4 diketahui memiliki kemungkinan terjadinya kolisi dan dapat diserang oleh kriptanalis Algoritma MD5 menerima masukan berupa pesan atau *message* dengan ukuran sembarang dan menghasilkan *message digest* yang panjangnya 128 bit.

Langkah-langkah pembuatan *message digest* dengan algoritma MD5 adalah sebagai berikut:

- Penambahan bit-bit pengganjal (*padding bits*)
- Penambahan nilai panjang pesan semula
- Inisialisasi 4 buah penyangga (*buffer*) MD.
- Pengolahan pesan dalam blok berukuran 512 bit

Pesan dibagi menjadi  $L$  buah blok yang masing-masing panjangnya 512 bit ( $Y_0$  sampai  $Y_{L-1}$ ) dan setiap blok melewati proses  $H_{MD5}$  yaitu setiap blok 512-bit diproses bersama dengan penyangga MD5 menjadi keluaran 128-bit.

Proses  $H_{MD5}$  terdiri dari 4 putaran dan masing-masing putaran melakukan operasi dasar MD5 sebanyak 16 kali dalam setiap operasi dasar memakai sebuah element T. Jadi setiap putaran memiliki 16 elemen tabel T.

Operasi dasar MD5 dapat ditulis sebagai berikut:

$$a = b + CLS_s(a + g(b, c, d) + X[k] + T[i])$$

Keterangan:

$a, b, c, d$  = empat buah penyangga 32-bit (berisi nilai penyangga A, B, C, dan D).

$g$  = salah satu fungsi F, G, H, I

$CLS_s$  = *circular left shift* sebanyak  $s$  bit

$X[k]$  = kelompok 32-bit ke- $k$  dari 512 bit *message ke- $q$*

$T[i]$  = elemen tabel  $T$  ke- $i$  (32-bit)

Fungsi  $F, G, H, I$  merupakan fungsi untuk memanipulasi masukkan  $a, b, c$ , dan  $d$  dengan ukuran 32-bit. Fungsi  $F, G, H$ , dan  $I$  dituliskan sebagai berikut:

$$F(b, c, d) = (b \wedge c) \vee (\sim b \wedge d)$$

$$G(b, c, d) = (b \wedge d) \vee (c \wedge \sim d)$$

$$H(b, c, d) = b \oplus c \oplus d$$

$$I(b, c, d) = c \oplus (b \wedge \sim d)$$

Sedangkan nilai  $T[i]$  dibuat tetap dan ditulis dalam sebuah tabel. Tabel tersebut disusun dengan fungsi  $2^{32} \times \text{abs}(\sin(i))$  ( $i$  dalam radian).

Setelah putaran ketiga,  $a, b, c$ , dan  $d$  ditambahkan ke  $A, B, C$ , dan  $D$ , selanjutnya algoritma memproses untuk blok data selanjutnya ( $Y_{q+1}$ ). Keluaran akhir dari algoritma MD4 adalah hasil penyambungan dari bit-bit di  $A, B, C$ , dan  $D$ .

## 3. Secure Hash Algorithm (SHA)

SHA adalah fungsi *hash* satu-arah yang dibuat oleh NIST dan digunakan bersama dengan DSS (*Digital Signature Standard*). SHA didasarkan pada MD4 yang dibuat oleh Ronald L. Rivest. SHA disebut aman atau *secure* karena SHA dirancang sedemikian rupa sehingga cara komputasi tidak mungkin menemukan pesan yang berkoresponden dengan *message digest* yang diberikan.

Algoritma SHA menerima masukan berupa pesan atau *message* dengan ukuran maksimum  $2^{64}$  bit dan menghasilkan *message digest* yang panjangnya 160 bit. Hasil keluaran atau *message digest* dari SHA lebih panjang daripada *message digest* yang dihasilkan dengan MD5.

Langkah-langkah pembuatan *message digest* dengan algoritma SHA adalah sebagai berikut:

- Penambahan bit-bit pengganjal (*padding bits*).
- Penambahan nilai panjang pesan semula
- Inisialisasi 5 penyangga (*buffer*) MD
- Pengolahan pesan dalam blok berukuran 512 bit

Pesan dibagi menjadi  $L$  buah blok yang masing-masing panjangnya 512 bit ( $Y_0$  sampai  $Y_{L-1}$ ) dan setiap blok melewati proses  $H_{SHA}$  yaitu setiap blok 512-bit diproses bersama dengan penyangga SHA menjadi keluaran 160-bit.

Proses  $H_{SHA}$  terdiri dari 80 putaran dan masing-masing putaran menggunakan bilangan penambah  $K_n$ , yaitu:

Untuk  $0 \leq n \leq 19, K_n = 5A827999$

Untuk  $20 \leq n \leq 39, K_n = 6ED9EBA1$

Untuk  $40 \leq n \leq 59, K_n = 8F1BBCDC$

Untuk  $60 \leq n \leq 79, K_n = CA62C1D6$

Setiap putaran menggunakan operasi dasar yang sama (dinyatakan dalam fungsi  $f$ ).

Operasi dasar SHA dapat ditulis sebagai berikut:

$$a = (CLS_5(a) + f_n(b, c, d) + e + W_n + K_n)$$

$$b = a$$

$$c = CLS_{30}(b)$$

$$d = c$$

$$e = d$$

Keterangan:

$a, b, c, d, e$  = lima buah peyangga 32-bit (berisi nilai peyangga  $A, B, C, D$ , dan  $E$ )

$n$  = putaran,  $0 \leq n \leq 79$

$f_n$  = fungsi logika

$CLS_s$  = circular left shift sebanyak  $s$  bit

$W_n$  = word 32-bit ke- $k$  dari 512 bit message ke- $q$

$K_n$  = bilangan penambah

Fungsi  $f$  yang digunakan pada operasi dasar berbeda pada putaran tertentu. Fungsi  $f$  dapat dituliskan sebagai berikut:

Untuk  $0 \leq n \leq 19, f_n = (b \wedge c) \vee (\sim b \wedge d)$

Untuk  $20 \leq n \leq 39, f_n = b \oplus c \oplus d$

Untuk  $40 \leq n \leq 59, f_n = (b \wedge c) \vee (b \wedge d) \vee (c \wedge d)$

Untuk  $60 \leq n \leq 79, f_n = b \oplus c \oplus d$

Sedangkan nilai  $W$  yang terdiri dari  $W_1$  sampai  $W_{16}$  diinisialisasi dari 16 word pada blok yang sedang diproses, sedangkan nilai  $W_n$  pada putaran berikutnya diperoleh dengan persamaan:

$$W_n = W_{n-16} \oplus W_{n-14} \oplus W_{n-8} \oplus W_{n-3}$$

Setelah putaran ke-79,  $a, b, c, d$ , dan  $e$  ditambahkan ke  $A, B, C, D$ , dan  $E$ , selanjutnya algoritma memproses untuk blok data berikutnya ( $Y_{q+1}$ ). Keluaran akhir dari algoritma *SHA* adalah hasil penyambungan bit-bit di  $A, B, C, D$ , dan  $E$ .

### B. MAC(Message Authentication Code)

MAC adalah fungsi satu arah yang menggunakan kunci rahasia (*secret key*) dalam pembangkitan nilai *hash*. Sebagaimana telah dijelaskan diatas *MD5* dan *SHA* tidak memerlukan kunci untuk menghasilkan nilai *hash*. MAC juga menghasilkan nilai *hash* yang selalu berukuran tetap (*fixed*) untuk ukuran pesan berapa saja. MAC biasanya dilekatkan (*embed*) pada pesan. MAC digunakan untuk otentikasi tanpa perlu merahasiakan pesan. MAC bukanlah tanda-tangan digital. MAC hanya menyediakan otentikasi pengirim dan integritas pesan saja.

MAC secara matematis :

$$MAC = C_K(M)$$

Keterangan:

MAC = nilai *hash*

$C$  = fungsi *hash* ( $()$ )

$K$  = kunci rahasia

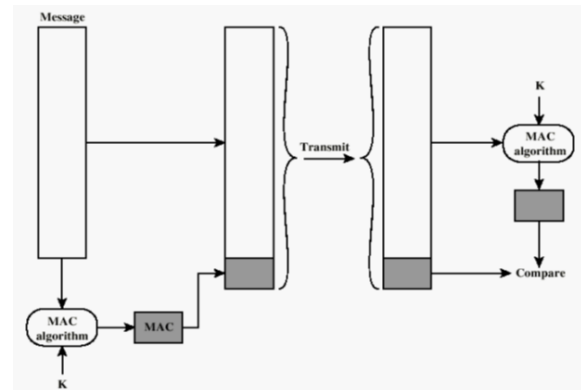
### Beberapa Aplikasi MAC

- Otentikasi arsip yang digunakan oleh dua atau lebih pengguna
- Menjaga integritas (keaslian) isi arsip terhadap perubahan, misalnya karena serangan virus.

### Beberapa Algoritma MAC

- Algoritma MAC berbasis cipher blok  
MAC dibangkitkan dengan menggunakan algoritma *cipher* blok dengan mode *CBC* atau *CFB*. Nilai *hash*-nya adalah hasil enkripsi blok terakhir.
- Algoritma MAC berbasis fungsi *hash* satu arah

### Skema MAC



### C. Simple Mail Transfer Protocol (SMTP)

Simple Mail Transfer Protocol (SMTP) adalah transmisi standar pada email melalui jaringan. SMTP ini dispesifikasikan untuk outgoing email, yang menggunakan TCP pada port 25. Untuk keamanan email yang menggunakan protokol SMTP, koneksi SMTP ini cukup aman karena menggunakan SSL.

SMTP ini digunakan oleh *mail server* dan *mail transfer agent* untuk mengirim dan menerima pesan, namun untuk level *client* hanya digunakan untuk mengirim pesan ke *mail server*. Untuk menerima pesan *client* biasanya menggunakan *Post Office Protocol (POP)* atau *Internet Message Access Protocol (IMAP)*.

Salah satu bahasa pemrograman internet yang mendukung SMTP yaitu PHP (*Personal Home Page*).

## III. ANALISIS

Saya melakukan analisis terhadap fungsi *hash* satu-arah yang sudah saya pilih sebelumnya. Fungsi *hash* yang saya analisis adalah *MD4*, *MD5*, dan *SHA*. Saya melakukan analisis terhadap masing-masing fungsi *hash* dalam hal algoritma yang dipakai dan kemungkinan terjadinya kolisi pada fungsi *hash* tersebut. Analisis yang saya lakukan akan saya deskripsikan dalam bentuk keunggulan dan kelemahan fungsi *hash* satu-arah yang bersangkutan.

### A. Analisis Algoritma MD4

Keunggulan algoritma *MD4* adalah:

1. Algoritma *MD4* didesain sedemikian rupa sehingga lebih cepat dikomputasi pada mesin 32-bit dan mudah untuk diimplementasikan.
2. *MD4* tidak memerlukan tabel substitusi. *MD4* berbeda dengan *MD5* yang memerlukan tabel substitusi. Tabel substitusi pada *MD5* diperlukan untuk menyimpan nilai  $T$  yang digunakan pada setiap putarannya.

3. Komputasi *MD4* sangat cepat karena *MD4* hanya terdiri dari 3 putaran dan setiap putaran dilakukan 16 kali. Untuk kasus *hashing* terhadap satu blok pesan berukuran 512 bit, total jumlah perulangan yang dilakukan adalah 48 kali.
4. Panjang pesan yang dapat di-*hash* lebih dari  $2^{64}$  bit. Sedangkan *SHA* hanya dapat memproses pesan dengan panjang maksimal  $2^{64}$  bit.

Kelemahan algoritma *MD4* adalah:

1. Algoritma *MD4* sudah dibuktikan memiliki kemungkinan terjadinya kolisi [1]. Kolisi yang dimaksudkan adalah kolisi *hash* (*hash colision*). Suatu fungsi hash  $H$  dikatakan memiliki hash colision apabila terdapat message  $x$  dan message  $y$ ,  $x \neq y$  dan  $H(x) = H(y)$ .
2. Algoritma *MD4* dapat diserang oleh kriptanalisis.

### B. Analisis Algoritma *MD5*

Keunggulan algoritma *MD5* adalah:

1. Panjang pesan yang dapat di-*hash* lebih dari  $2^{64}$  bit. Sedangkan *SHA* hanya dapat memproses pesan dengan panjang maksimal  $2^{64}$  bit.
2. Algoritma *MD5* lebih menjamin untuk tidak terjadinya kolisi *hash*. Karena *MD5* menggunakan tabel substitusi  $T$  yang digunakan setiap putarannya.
3. Tingkat keamanan algoritma *MD5* masih tinggi. Hal tersebut dapat didukung karena *MD5* memiliki tabel substitusi  $T$  untuk digunakan dalam operasi dasar pada setiap putarannya.

Kelemahan algoritma *MD5* adalah:

1. Komputasi *MD5* memakan waktu yang lebih lama karena *MD5* memproses setiap blok pesan berukuran 512 bit dalam 4 putaran dan setiap putarannya diulangi selama 16 kali.

### C. Analisis Algoritma *SHA*

Keunggulan algoritma *SHA* adalah:

1. Panjang nilai *hash* (*hash-value*) atau *message digest* yang dikeluarkan berukuran 160 bit. Menurut saya, semakin panjang *message digest* yang dikeluarkan semakin sulit terjadinya kolisi dari suatu fungsi *hash* tersebut.
2. Algoritma *SHA* diklaim aman atau *secure*. Keamanan algoritma *SHA* dapat disebabkan karena proses komputasi dalam algoritma *SHA* rumit dan berbeda-beda untuk setiap putarannya.

Kelemahan algoritma *SHA* adalah:

1. Panjang pesan yang dapat di-*hash* maksimal berukuran  $2^{64}$  bit. Sedangkan *MD4* dan *MD5* dapat memproses pesan dengan panjang berapapun.
2. Kecepatan algoritma *SHA* lebih lambat daripada kecepatan komputasi algoritma *MD4*.

## IV. PERANCANGAN

Berdasarkan hasil analisis fungsi *hash* *MD4*, *MD5*, dan *SHA*, teranalisis bahwa algoritma *MD5* memiliki tiga

keunggulan yaitu panjang pesan yang dapat di-*hash* lebih dari  $2^{64}$  bit. Sedangkan *SHA* hanya dapat memproses pesan dengan panjang maksimal  $2^{64}$  bit. Yang kedua, algoritma *MD5* lebih menjamin untuk tidak terjadinya kolisi *hash*. Karena *MD5* menggunakan tabel substitusi  $T$  yang digunakan setiap putarannya, dan yang ketiga tingkat keamanan algoritma *MD5* masih tinggi. Hal tersebut dapat didukung karena *MD5* memiliki tabel substitusi  $T$  untuk digunakan dalam operasi dasar pada setiap putarannya.

Namun algoritma *MD5* ini memiliki kekurangan yaitu komputasi *MD5* memakan waktu yang lebih lama karena *MD5* memproses setiap blok pesan berukuran 512 bit dalam 4 putaran dan setiap putarannya diulangi selama 16 kali.

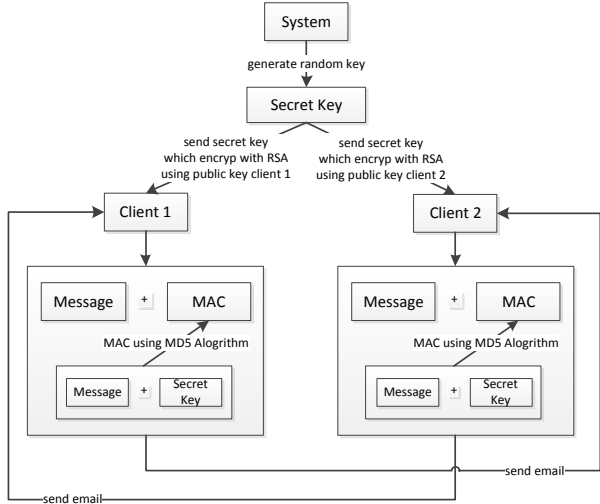
Saya memilih algoritma *MD5* ini sebagai fungsi *hash* yang akan digunakan dalam MAC ini karena terlihat jelas kelebihan algoritma *MD5* ini dibanding dua algoritma lainnya yaitu *MD4* dan *SHA*.

Untuk selanjutnya MAC yang menggunakan algoritma *MD5* ini akan diimplementasi pada protokol SMTP. Protokol SMTP ini adalah protokol pengiriman email *outgoing* pada level *client*. Sebagaimana kita ketahui ukuran email yang dikirim biasanya tidak terlalu besar, hal ini akan sedikit mengabaikan kekurangan *MD5* yang membutuhkan komputasi yang lebih lama.

Implementasinya pada protokol SMTP ini, pertama akan dibangkitkan sebuah kunci rahasia yang digenerate secara random, kemudian kunci tersebut akan di gunakan untuk pembangkitan nilai *hash* pada MAC, untuk mengamankan kunci tersebut maka kunci tersebut akan di enkripsi dengan algoritma kunci public yaitu algoritma RSA yang akan mengenkripsi kunci rahasia tersebut dengan menggunakan kunci public pengguna email, sebelum berkirim email maka akan si pengirim dan si penerima akan menerima sebuah kunci rahasia yang telah dienkripsi yang akan digunakan sebagai otentikasi pengirim pesan dan integritas pesan.

Pada saat pengiriman email, akan disertakan pula MAC pada pesan yang dikirimkan, sehingga penerima pesan dapat mengecek pesan tersebut benar-benar dikirim oleh orang yang bersangkutan dengan alamat email pengirim.

## Skema Rancangan sistem



```

//embed mac to message
$message = $message.'<mac>'.$mac.'</mac>';

//sent message via SMTP
mail($to, $subject, $sendmessage,$from);

?>
  
```

## Pesan yang akan dikirim:

From:

To:

Subject:

**Analisis Beberapa Fungsi Hash dan Implementasi MAC dengan Fungsi Hash pada protokol SMTP**

SMTP adalah salah satu protokol standar email. SMTP ini dispesifikasikan untuk outgoing email, yang menggunakan TCP pada port 25. Untuk keamanan email yang menggunakan protokol SMTP, koneksi SMTP ini cukup aman karena menggunakan SSL. MAC adalah satu fungsi satu arah yang menggunakan kunci rahasia dalam proses pembangkitan nilai hashnya. MAC bukan tanda tangan digital, namun fungsi dari MAC ini sendiri adalah sebagai otentikasi pengirim dan integritas pesan. Pertama-tama akan dibandingkan beberapa fungsi hash, dianalisis kelebihan dan kekurangannya, dan dipilih salah fungsi hash yang akan digunakan pada MAC. Implementasinya pada protokol SMTP ini, pertama akan dibangkitkan sebuah kunci rahasia yang digenerate secara random, kemudian kunci tersebut akan di gunakan untuk pembangkitan nilai hash pada MAC, untuk mengamankan kunci tersebut maka kunci tersebut akan di enkripsi dengan algoritma kunci public yaitu algoritma RSA yang akan mengenkripsi kunci rahasia tersebut dengan menggunakan kunci public pengguna email, sebelum ber kirim email maka akan si pengirim dan si penerima akan menerima sebuah kunci rahasia yang telah dienkripsi yang akan digunakan sebagai otentikasi pengirim dan integritas pesan. Pada saat pengiriman email, akan disertakan pula MAC pada pesan yang dikirimkan, sehingga penerima pesan dapat mengecek pesan tersebut benar-benar dikirim oleh orang yang bersangkutan dengan alamat email pengirim.

Referensi  
Slide kuliah Kriptografi IF3058, MAC, RSA, Fungsi Hash.

## V. IMPLEMENTASI

Implementasinya akan menggunakan bahasa pemrograman internet yaitu *PHP(Personal Home Page)*.

## Form pengiriman pesan:

localhost/MACinSMTP/sendemail.php

From:

To:

Subject:

## Pesan yang dikirim

Message:

**Analisis Beberapa Fungsi Hash dan Implementasi MAC dengan Fungsi Hash pada protokol SMTP**

SMTP adalah salah satu protokol standar email. SMTP ini dispesifikasikan untuk outgoing email, yang menggunakan TCP pada port 25. Untuk keamanan email yang menggunakan protokol SMTP, koneksi SMTP ini cukup aman karena menggunakan SSL. MAC adalah satu fungsi satu arah yang menggunakan kunci rahasia dalam proses pembangkitan nilai hashnya. MAC bukan tanda tangan digital, namun fungsi dari MAC ini sendiri adalah sebagai otentikasi pengirim dan integritas pesan saja. Pertama-tama akan dibandingkan beberapa fungsi hash, dianalisis kelebihan dan kekurangannya, dan dipilih salah fungsi hash yang akan digunakan pada MAC. Implementasinya pada protokol SMTP ini, pertama akan dibangkitkan sebuah kunci rahasia yang digenerate secara random, kemudian kunci tersebut akan di gunakan untuk pembangkitan nilai hash pada MAC, untuk mengamankan kunci tersebut maka kunci tersebut akan di enkripsi dengan algoritma kunci public yaitu algoritma RSA yang akan mengenkripsi kunci rahasia tersebut dengan menggunakan kunci public pengguna email, sebelum ber kirim email maka akan si pengirim dan si penerima akan menerima sebuah kunci rahasia yang telah dienkripsi yang akan digunakan sebagai otentikasi pengirim dan integritas pesan. Pada saat pengiriman email, akan disertakan pula MAC pada pesan yang dikirimkan, sehingga penerima pesan dapat mengecek pesan tersebut benar-benar dikirim oleh orang yang bersangkutan dengan alamat email pengirim.

Referensi  
Slide kuliah Kriptografi IF3058, MAC, RSA, Fungsi Hash.  
<mac>0275a4b597e6e94e175737ddb3a239df</mac>

Mail Sent

## MAChandler.php

```

<?php
include 'keygenerator.php';
include 'makeMAC.php';

$sender = $_POST['fsender'];
$from = 'From: '.$sender;
$to = $_POST['freciever'];
$message = $_POST['fmessage'];
$subject = $_POST['fsubject'];

//get secret key
$key = generatekey($sender,$to);

//make mac for message
$mac = makeMAC($message, $key);
  
```

## VI. SIMPULAN

Kesimpulan dari hasil analisis fungsi hash:

1. Kompleksitas atau tingkat kerumitan suatu algoritma pada fungsi hash sebanding atau berbanding lurus dengan tingkat keamanan fungsi hash tersebut.
2. Kompleksitas atau tingkat kerumitan suatu algoritma pada fungsi hash berbanding terbalik dengan kecepatan dari algoritma fungsi hash tersebut.

Kesimpulan dari hasil implementasi MAC pada protokol SMTP yaitu sistem ini akan menjaga integritas pesan yang dikirim dan dapat mengecek otentikasi pengirim pesan.

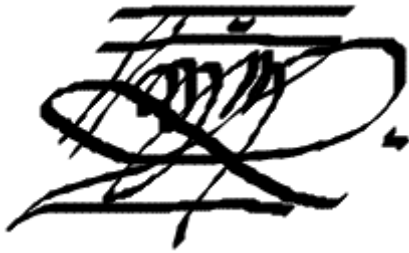
#### REFERENSI

- [1] Slide kuliah kriptografi if3058, fungsi hash, MAC.
- [2] <http://tools.ietf.org/html/rfc2554> RFC, SMTP Service extension for authentication.
- [3] [http://www.w3schools.com/php/php\\_mail.asp](http://www.w3schools.com/php/php_mail.asp) PHP mail
- [4] [http://en.wikipedia.org/wiki/Message\\_authentication\\_code](http://en.wikipedia.org/wiki/Message_authentication_code) Message Authentication Code.
- [5] [http://en.wikipedia.org/wiki/Simple\\_Mail\\_Transfer\\_Protocol](http://en.wikipedia.org/wiki/Simple_Mail_Transfer_Protocol) Simple Mail Transfer Protocol.

#### PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 9 Mei 2010



Erdiansyah Fajar Nugraha / 13508055