STUDI DAN MODIFIKASI MD5 UNTUK MENGATASI KOLISI DAN IMPLEMENTASINYA DALAM SITUS JEJARING SOSIAL

Arief Latu Suseno – NIM: 13505019

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika Institut Teknologi Bandung
Jl. Ganesha 10, Bandung
E-mail : if15019@students.if.itb.ac.id

Abstrak

Makalah ini menjelaskan bagaimana algoritma MD5 sebagai autentikator situs jejaring sosial diminimalkan untuk didapatkan kolisinya.Algoritma MD5 yang digunakan adalah MD5 yang telah dimodifikasi dengan melipatgandakan insialisasi penyangga dan menambahkan 10 karakter di akhir hasil fungsi *hash* MD5 sebagai penanda jumlah karakter yang akan dienkripsi. Dengan fungsi *hash* seperti ini, kemungkinan terjadinya kolisi akan semakin kecil sehingga pengguna yang terotentikasi dalam situs jejaring sosial adalah benar-benar pengguna sesungguhnya.

Kata kunci: fungsi *hash*, kolisi, situs jejaring sosial, *social community network*

1. Latar Belakang

Web site Situs jejaring sosial sekarang ini sedang menjamur. Sebut saja Friendster, Hi5, Tagged, Plurk, atau bahkan Facebook. Situs terakhir yang disebutkan ini memang luar biasa perkembangannya. Facebook telah 'meracuni' sebagian besar penduduk dunia yang melek internet. Beberapa jam dari tiap keseharian pecandu Facebook pasti diluangkan untuk mengakses situs yang dibuat oleh Mark Zuckerberg, seorang lulusan Universitas Harvard. Baik itu melalui internet, BlackBerry, atau bahkan melalui GPRS biasa. Hal inilah yang membuat peringkat Facebook naik pesat. September tahun 2006 Facebook berada di peringkat ke-60 dan 1 tahun kemudian naik pesat hingga berada di peringkat ke-7. [MAE07]

Fenomena ini membuat sebagian orang bertanya apakah situs-situs jejaring sosial tersebut aman. Selama ini situs yang menggunakan kata kunci untuk bisa mengakses situs tersebut biasanyamenggunakan fungsi hash satu arah untuk mengenkripsi kata kunci, yaitu MD5. MD5 memang cukup populer untuk mengenkripsi suatu pesan. Bahkan bahasa pemrograman PHP, Java dan C# telah menyertakan fungsi MD5 di dalam *library* nya. Namun seiring dengan semakin majunya teknologi yang juga semakin pintarnya kriptanalis, MD5 mulai ditemukan kelemahannya, yaitu terjadinya kolisi. Hans Dobbertin, Xiaoyun Wang, Philip Hawks, Michael Paddon, Gregory G. Rose, Vlastimil Klima, dan Stefan Lucks, Magnus Daum adalah contoh orang-orang yang menemukan kolisi di MD5. [AND08]

Pengamanan suatu akun seharusnya tidak boleh ada cacat sama sekali. Kolisi pada MD5 yang telah ditemukan sebenarnya cukup berbahaya. Walaupun peluangnya kecil, tapi tetap saja ada kemungkinan 2 orang yang memiliki fungsi *hash* yang sama sehingga ada kemungkinan orang yang terotentikasi oleh sistem bukanlah pengguna sebenarnya. Kasus seperti ini mungkin tidak terlalu parah jika terjadi pada situs jejaring sosial karena mungkin yang akan bisa dirusak atau dicuri hanyalah data privasi. Namun kasus seperti ini akan sangat berbahaya jika terjadi pada situs-situs lelang seperti *eBay*. 'Pengguna terotentikasi' ini akan bisa sesuka hati menggunakan uang yang tersedia di akun tersebut.

Untuk itu diperlukan suatu fungsi *hash* yang bisa mencegah terjadinya kolisi sekecil mungkin. Makalah ini akan membahas metode modifikasi yang

digunakan dan menerapkannya dalam situs jejaring sosial. Modifikasi dilakukan dengan melipatgandakan inisialisasi penyangga [AND08] dan menambahkan 10 karakater di akhir hasil fungsi *hash* tersebut.

2. Rumusan Masalah

Dari latar belakang yang telah diuraikan maka dapat dirumuskan beberapa permasalahan pada makalah ini yaitu :

1. Bagaimana mendefiniskan serangan-serangan yang biasa dilakukan terhadap algoritma MD5
2. Bagaimana memodifikasi algoritma MD5 sehingga akan sulit dicari kolisinya tanpa harus mengorbankan performansi algoritma MD5 tersebut

3. Tujuan

Dari permasalahan yang ada pada rumusan masalah maka makalah ini bertujuan :

1. Memodifikasi algoritma MD5 agar aman digunakan untuk autentikasi pengguna dalam sistem, khususnya situs jejaring sosial

4. Batasan Masalah

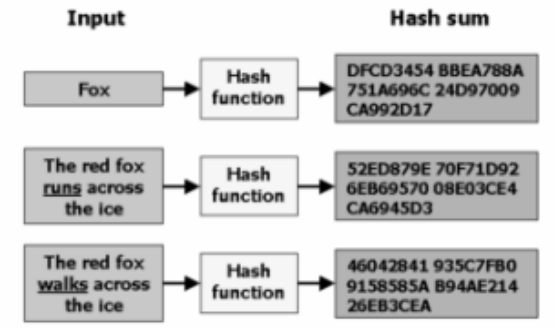
1. Makalah ini tidak membicarakan modifikasi algoritma MD5 secara *advance* untuk menjaga performansi dari algoritma MD5

5. Dasar Teori

5.1. Fungsi Hash

Di dalam kriptografi terdapat sebuah fungsi yang sesuai untuk aplikasi keamanan seperti otentikasi dan integritas pesan. Fungsi tersebut adalah fungsi *hash*, yaitu fungsi yang menerima masukan string yang panjangnya sembarang dan mengonversinya menjadi string keluaran yang panjangnya tetap (umumnya berukuran jauh lebih kecil daripada ukuran string semula). [RIN06]

Keluaran dari fungsi *hash* disebut nilai *hash* atau *message digest*. Keluaran fungsi hash pasti tidak sama untuk setiap pesan yang berbeda. Gambar 5.1 mencontohkan beberapa buah pesan dengan panjang yang berbeda-beda selalu di hash menhasilkan pesan ringkas dengan panjang yang tetap.



Gambar 5. 1 Contoh hashing beberapa buah pesan dengan panjang yang berbeda-beda

5.1.1. Fungsi *Hash* Satu Arah

Fungsi *hash* yang dimaksudkan dalam makalah ini adalah fungsi hash satu arah. Fungsi hash satu arah (One-way hash) adalah fungsi hash yang bekerja dalam satu arah: pesan yang sudah diubah menjadi message digest tidak dapat dikembalikan lagi menjadi pesan semula. Dua pesan yang berbeda akan selalu menghasilkan nilai hash yang berbeda pula.

Properti-properti yang harus dimiliki fungsi hash satu arah adalah:

a. *Preimage Resistant*, yaitu untuk sebuah nilai hash h, tidak mungkin menemukan pesan masukan m yang memenuhi h = hash (m).
b. *Second Preimage Resistant*, yaitu untuk sebuah nilai pesan masukan m1, tidak mungkin menemukan pesan masukan m2 yang memenuhi *hash (m1) = hash (m2)* dimana m1 tidak sama dengan m2.
c. *Collision Resistant*, yaitu tidak mungkin (secara komputasi) menemukan pasangan pesan masukan m1 dan m2 yang memiliki nilai hash yang sama (*hash (m1) = hash(m2)*).
d. *Hash(m)* mudah dihitung untuk setiap nilai m yang diberikan

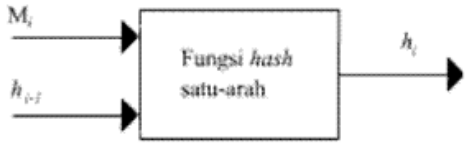
e. *Hash* menghasilkan nilai hash dengan panjang tetap

f. Fungsi *hash* dapat diterapkan pada blok data berukuran berapa saja

Keenam sifat diatas penting sebab sebuah fungsi *hash* seharusnya berlaku seperti fungsi acak. Namun, ada beberapa kasus dimana dapat terjadi suatu kolisi (*coll ision*), yaitu ketika terdapat dua pesan yang berbeda yang mempunyai pesan ringkas yang sama. Fungsi *hash* memiliki algoritma yang *iterative* dan searah, yang dapat memproses pesan yang diberikan untuk menghasilkan representasi yang lebih pendek yang disebut *message digest*. Untuk menghasilkan nilai *message digest* dari pesan besar, fungsi *hash* menerima masukan berupa isi blok pesan saat ini serta nilai hash dari blok pesan sebelumnya,

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

Skema fungsi hash ditunjukkan pada Gambar 5.2. Fungsi *hash* adalah publik (tidak dirahasiakan), dan keamanannya terletak pada sifat satu arahnya itu. Ada beberapa fungsi hash satu arah yang telah diciptakan, antara lain MD4, MD5, Secure Hash Algorithm (SHA), RIPEMD, WHIRLPOOL, dan lain lain.



Gambar 5. 2 Fungsi Hash Satu Arah

Tabel 5.1 membandingkan antara beberapa algoritma fungsi hash

Tabel 5. 1 Beberapa Fungsi Hash

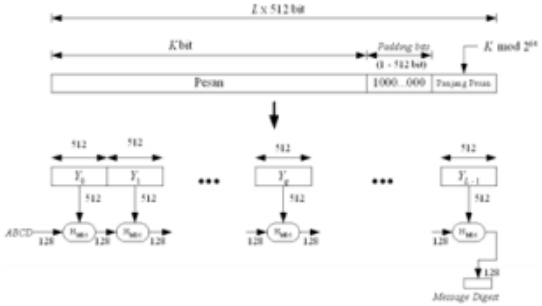
Algoritma	Ukuran MD (bit)	Ukuran Blok Pesan	Kolisi
MD2	128	128	Ya
MD4	128	512	Hampir
MD5	128	512	Ya
RIPEMD	128	512	Ya
RIPEMD-128/256	128/256	512	Tidak
RIPEMD-160/320	160/320	512	Tidak
SHA-0	160	512	Ya
SHA-1	160	512	Ada Cacat
SHA-256/224	256/224	512	Tidak
SHA-512/384	512/384	1024	Tidak
WHIRLPOOL	512	512	Tidak

Contoh penggunaan fungsi *hash* adalah pengiriman dan pencocokan kata kunci (*password*) antara *client* dan *server*. Masukan kata kunci diberikan fungsi *hash* sehingga menghasilkan nilai *hash* yang khas untuk kemudian dikirimkan kepada *server* dan dibandingkan dengan nilai *hash* yang tersimpan pada basisdata server. Kata kunci dinyatakan benar jika nilai hashnya sesuai dengan nilai hash yang tersimpan pada server. Dan MD5 adalah algoritma yang secara luas dipakai dalam melakukan otentikasi kata kunci tersebut.

5.2. Message Digest 5 (MD5)

MD5 yang didesain oleh Profesor Ronald Rivest pada tahun 1991, merupakan salah satu fungsi *hash* satu arah yang paling umum dan luas dipakai. MD5 merupakan perbaikan dari MD4, karena MD4 dianggap tidak aman lagi setelah adanya serangan analitik yang melemahkan algoritma tersebut.

MD5 menerima masukan pesan dengan ukuran sembarang dan mengonversi pesan tersebut dengan algoritma *hash*nya menjadi *message digest* berukuran 128 bit, yang biasanya merupakan rangkaian 32 digit karakter heksadesimal. Lebih spesifik lagi, MD5 bekerja pada satuan blok-blok masukan berukuran 512 bit yang diproses secara berulang.



Gambar 5. 3 Pembuatan Message Digest dengan algoritma MD5

Gambar 5.3 adalah gambaran umum pemrosesan pesan menjadi message digest menggunakan algoritma MD5.

Langkah pemrosesannya secara umum ada 4, yaitu:

1. Menambahkan bit-bit pengganjal pada pesan masukan agar memiliki panjang kelipatan 512 bit dikurang 64 bit.

   Pesan ditambah dengan sejumlah bit pengganjal sedemikian sehingga panjang pesan kongruen dengan 448 bit modulo 512, artinya panjang pesan setelah ditambahi bit-bit pengganjal adalah 64 bit kurang dari kelipatan 512, seperti yang sudah disebutkan sebelumnya. Hal ini berlaku juga untuk pesan dengan panjang 448 bit, dimana pesan tersebut harus ditambahi bit-bit pengganjal sebanyak 512 bit menjadi 960 bit. Dari peristiwa khusus tersebut, dapat dilihat bahwa panjang bit-bit pengganjal yang diperbolehkan adalah 1 sampai 512 bit. Penambahan bit-bit pengganjal tersebut dilakukan dengan prosedur tersendiri, yaitu diawali dengan satu buah bit 1 pada akhir pesan dan selanjutnya menambahkan bit-bit 0 sampai memenuhi syarat jumlah 64 bit kurang dari kelipatan 512.

2. Mengisi sisa ruang 64 bit tersebut dengan informasi panjang pesan semula.

   Pesan yang sudah diberi bit-bit pengganjal, seharusnya berukuran 64 bit kurang dari kelipatan 512, selanjutnya ditambah dengan 64 bit yang menyatakan informasi panjang pesan semula sehingga panjang pesan menjadi tepat kelipatan 512 bit. Jika panjang pesan lebih besar dari 264 maka dilakukan proses modulo 264 terhadap panjang tersebut, baru kemudian ditambahkan pada pesan. Dengan kata lain, jika panjang pesan semula adalah n bit maka 64 bit yang ditambahkan menyatakan n modulo 264

3. Menginisialisasi penyangga untuk memroses pesan.

   Untuk melakukan proses hashing, MD5 membutuhkan empat buah penyangga (buffer) yang masing-masing panjangnya 32 bit sehingga totalnya 128 bit. Penyangga-penyangga tersebut berfungsi sebagai penampung hasil antara dan hasil akhir selama menjalankan algoritma hash. Keempat penyangga ini diberi nama A, B, C, dan D. Setiap penyangga diinisialisasi dengan pengisian nilai-nilai dalam notasi heksadesimal sebagai berikut:

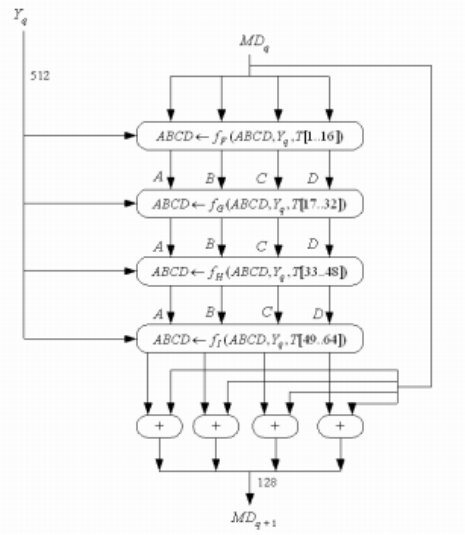
***A* = 01234567**	***C* = FEDCBA98**
***B* = 89ABCDEF**	***D* = 76543210**

   Beberapa versi MD5 memiliki nilai inisialisasi yang berbeda, yaitu:

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

4. Memecah pesan dalam satuan blok-blok berukuran 512 bit dan mengolahnya masing-masing.

   Setiap blok berukuran 512 bit diproses bersama dengan penyangga MD menjadi keluaran 128 bit. Proses ini terdiri dari 4 buah putaran dan masing-masing putaran melakukan operasi dasar MD5 sebanyak 16 kali. Setiap operasi dasar memakai sebuah elemen T secara spesifik, yang akan ditampilkan nanti. Jadi setiap putaran memakai 16 elemen tabel T, dan jumlah total putaran proses yang dilakukan adalah 64 buah. Proses tersebut diperlihatkan dalam gambar 5.4



Gambar 5. 4 Pengolahan blok 512 bit

6. Analisis Masalah

6.1. Serangan Pada MD5

Algoritma kriptografis manapun sampai saat ini belum ada yang benar-benar sempurna. Selalu ada kemungkinan serangan berhasil yang ditujukan pada suatu algoritma. Hal ini juga berlaku pada algoritma MD5. MD5 telah menjadi sorotan publik kriptografi sejak pertama kali dipublikasikan. Pada tahun 2004, hampir semua riset serangan terhadap MD5 hanya dapat menunjukkan kelemahan – kelemahan kecil pada desain algoritmanya. Meskipun ada 3 serangan yang dapat menunjukkan adanya permasalahan serius pada desain. Tiga serangan tersebut dengan properti L adalah panjang nilai hash, yaitu:

1. Serangan terhadap properti *Collision Resistant*, yaitu *Collision Attack*, yang artinya usaha menemukan dua pesan M1 dan M2 yang memiliki nilai hash yang sama dengan percobaan sebanyak kurang dari 2L/2
2. Serangan terhadap properti satu arah, yaitu *First Preimage Attack*, yang berarti usaha untuk menemukan pesan masukan jika diketahui nilai hashnya dalam percobaan sejumlah kurang dari 2L
3. Serangan kedua terhadap properti satu arah, yaitu *Second Preimage Attack*, yang berarti usaha untuk menemukan pesan masukan M2 jika diketahui pesan M1 yang memiliki nilai hash yang sama dalam percobaan sejumlah kurang dari 2L

Pada tanggal 16 Agustus 2004, Xiaoyun Wang, Dengguo Feng,, Xuejia Lai, dan Hongbo Yu menerbitkan suatu tulisan mengenai kolisi berbagai macam algoritma fungsi hash, salah satunya adalah algoritma MD5. Dengan rumus sebagai berikut: [5]

$$IV_0 : A_0 = 0x67452301, B_0 = 0xefcdab89, C_0 = 0x98badcfe, D_0 = 0x10325476$$

$$M' = M + \Delta C_1, \Delta C_1 = (0,0,0,0,2^{31},...,2^{15},...,2^{31},0)$$

$$N_i' = N_i + \Delta C_2, \Delta C_2 = (0,0,0,0,2^{31},...,-2^{15},...,2^{31},0)$$

sehingga:

$$MD5(M, N_i) = MD5(M', N_i').$$

Dibutuhkan waktu hanya sekitar 1 jam untuk mendapatkan M dan M' dengan menggunakan komputer bermesinkan IBM P690, untuk kemudian mendapatkan N dan N' dalam waktu sekitar 15 detik s.d. 5 menit.

Karena tidak dibutuhkan waktu yang lama untuk mendapatkan kolisi suatu pesan yang dienkripsi dengan MD5, maka hal ini menjadi sangat berbahaya untuk sistem-sistem yang menggunakan MD5 untuk autentikasi penggunanya. Seseorang dapat masuk ke dalam sistem karena dia terautentikasi dengan memasukkan kolisi dari kata kunci yang sebenarnya.

Sistem-sistem yang biasanya menggunakan algoritma MD5 untuk autentikasi penggunanya adalah sistem yang berbasiskan web, baik dalam bahasa PHP, JAVA, ASP, dan C#. Karena di dalam library bahasa pemrograman tersebut sudah terkandung fungsi algoritma MD5. Bahkan dalam bahasa pemrograman PHP, pengguna cukup memanggil fungsi MD5 saja tanpa perlu modifikasi apapun. Hal inilah yang menyebabkan sistem berbasiskan web gemar memakai algoritma MD5 untuk autentikasi penggunanya, tak terkecuali web situs jejaring sosial.

Web situs jejaring sosial seperti Facebook, Friendster, dll sedang menanjak pesat. Bahkan peningkatan Facebook terbilang cukup tajam. Berbagai orang penting pun menggunakan Facebook sebagai sarana untuk mempromosikan diri. Sebut saja Obama, SBY, David Beckham, dan berbagai orang penting lainnya telah lama bergabung dalam situs yang dibuat pertama kali untuk kalangan mahasiswa Universitas Harvard ini. Akan sangat berbahaya jika web situs jejaring sosial tidak memiliki sistem pengamanan yang kuat. Akun-akun orang penting seperti yang telah disebutkan di atas akan mudah untuk 'diganggu' dengan memanfaatkan lubang dalam algoritma MD5, yaitu kolisi.

6.2. Pemecahan Masalah

Dihasilkannya suatu kolisi dalam algoritma MD5 dengan sangat cepat seperti yang telah disebutkan sebelumnya adalah masalah besar bagi administrator

sistem. Untuk menangani permasalahan tersebut yang dianggap cukup sulit untuk dipecahkan, administrator harus memodifikasi algoritma MD5 yang digunakan dengan fungsi *hash* yang jauh lebih rumit, dengan tingkat pemecahannya yang jauh lebih tinggi.

Salah satu caranya adalah dengan menambahkan beberapa bit untuk kemudian disambungkan dengan nilai hash pesan tersebut. Hal ini digunakan sebagai penanda jumlah panjang karakter pesan asli yang akan dihitung nilai hashnya.

Situs http://www.x-ways.net/md5collision.html [6] memberikan contoh dua file berekstensi html yang jauh berbeda satu sama lainnya namun memiliki nilai *hash* yang sama. Contoh lainnya adalah dengan menambahkan beberapa karakter aneh pada pesan palsu dengan tujuan untuk mendapatkan nilai *hash* yang sama.

Dengan adanya panjang karakter pesan asli yang diletakkan di akhir nilai hash pesan asli (dalam makalah ini disarankan 10 karakter sehingga mampu menampung pesan dengan panjang 16^{10}), penyerangan seperti pada contoh di atas tidak akan bisa terjadi karena panjang pesan asli dan palsu pasti berbeda.

Ditambahkannya panjang karakter pesan asli di akhir nilai hash pesan asli juga bisa menambah jumlah digit nilai hash yang dihasilkan, yang artinya untuk menemukan kolisi dari pesan tersebut akan semakin sulit dan semakin lama.

Cara lainnya untuk meminimalkan terjadinya kolisi adalah dengan menambah jumlah penyangga. [AND08] Penambahan jumlah penyangga ini dimaksudkan untuk meningkatkan kompleksitas yang dapat dihasilkan oleh fungsi MD5. Dalam makalah ini penyangga ditambahkan 4 variabel penampung lagi sehingga output dari MD5 menjadi 256 bit, sehingga menjadi

***A* = 01234567**	***E* = abcd1111**
***B* = 89abcdef**	***F* = 0000abcd**
***C* = fedcba98**	***G* = efab1111**
***D* = 76543210**	***H* = 1111efab**

Dengan ditambahkannya penyangga menjadi dua kali lipatnya akan meningkatkan kompleksitas nilai hash yang dihasikan karena jumlah operasi dasar yang digunakan pun akan berlipat menjadi 64 dari sebelumnya 16 operasi dasar. Dengan banyaknya operasi dasar yang digunakan, nilai hash yang dihasilkan pun akan semakin sulit dan dibutuhkan waktu yang sangat lama untuk di dapatkan kolisinya.

Dua bentuk modifikasi dari algoritma MD5 ini dirasakan sudah cukup efektif untuk meminimalkan terjadinya (atau sengaja dicarinya) kolisi dari suatu pesan, setidaknya sampai belum ditemukannya algoritma baru yang semakin canggih untuk menemukan kolisi dari modifikasi algoritma MD5 ini. Jika kasus kolisi seperti yang telah disebutkan pada contoh diterapkan pada modifikasi algoritma MD5 ini, maka dapat dipastikan akan dihasilkan nilai hash yang unik.

7. Kesimpulan

Kesimpulan yang dapat diambil dari studi dan modifikasi MD5 untuk mengatasi kolisi dan implementasinya dalam situs jejaring sosial adalah:

1. Sistem dengan jumlah pengguna yang sangat besar seperti situs jejaring sosial harus benar-benar menggunakan sistem pengamanan yang kuat, yang sulit untuk ditaklukan, atau setidaknya perlu waktu yang sangat lama untuk memecahkannya.
2. Pengamanan suatu sistem dengan menggunakan algoritma MD5 sudah tidak aman lagi. Dengan menggunakan algoritma tertentu, kolisi suatu pesan dapat dihasilkan dalam kurun waktu kurang dari 2 jam.
3. Modifikasi dengan menambahkan jumlah karakter di akhir nilai hash yang dihasilkan dan menambahkan jumlah penyangga akan cukup efektif untuk meminimalkan terjadinya kolisi tanpa harus mengorbankan performansi yang sangat banyak.

8. Referensi

1 [MAE07] Maestri, Nicole, *Wal-Mart using Facebook to win back-to-school sales*, 2007.

2 [AND08] Andzarrahim, *Studi mengenai Collision pada MD5 dan Modifkasi Program Lama dalam menjawab solusi tersebut*, 2008.

3 [MAS07] Masykur Marhendra S.N, *Serangan Kolisi MD5 pada Duplikasi Ekstraksi File Paket*, 2007.

4. [RIN06] Rinaldi Munir, *Kriptografi*, Informatika: Bandung, 2006

5 [ANON] http://eprint.iacr.org/2004/199, waktu akses: 16 Mei 2009

6. [ANON] http://www.x-ways.net/md5collision.html, waktu akses: 16 Mei 2009