Kombinasi Algoritma Tanda Tangan Digital dengan Steganografi untuk Autentikasi File Media

Arief Widhiyasa (13505126)

Departemen Teknik Informatika
Institut Teknologi Bandung
Jl. Ganesha 10 Bandung 40132

E-mail : arief@clawford.net

Abstraksi

Sampai saat ini penggunaan teknik tanda tangan digital seringkali hanya pada file-file teks atau dokumen-dokumen yang masih berwujud teks (misalnya file document office, halaman web, dsb), hal ini disebabkan karena algoritma tanda tangan digital saat ini akan menyebabkan bertambahnya ukuran dokumen oleh karena disisipkan data tanda tangan yang bersangkutan. Pada file-file teks, penyisipan ini bisa tidak memiliki pengaruh sama sekali dengan isi pesan, misalnya pada file halaman web, bila data tanda tangan disisipkan dengan tag tertentu (misalnya tag komentar HTML) maka halaman web yang ditampilkan akan tidak mengalami perubahan sama sekali dari file asli. Namun berbeda halnya dengan file media (sebagian besar format file gambar/citra, suara dan video), segala penyisipan yang terjadi bisa mengakibatkan berubahnya file media tersebut, sehingga cukup sulit untuk melakukan tanda tangan digital. Dalam makalah ini saya akan mencoba mengkombinasikan antara algoritma tanda tangan digital yang ada saat ini (RSA+MD5), dengan teknik steganografi, dimana data tanda tangan digital akan disisipkan di file media dengan mengubah bit sedikit demi sedikit, namun tidak memberikan perubahan yang signifikan terhadap file media yang dirubah tersebut. Segala penerapan kombinasi akan diuraikan dalam makalah ini dengan harapan terdapat suatu kombinasi yang dapat memberikan suatu kualitas file media (setelah ditandatangani) yang hampir tidak memiliki perbedaan dengan aslinya bila dilihat oleh manusia.

Kata kunci
digital signature, RSA, MD5, steganografi, watermark

1. PENDAHULUAN

Seperti telah dijelaskan sebelumnya, penggunaan teknik tanda tangan digital saat ini masih sangat terbatas pada file teks saja. Namun sebenarnya kebutuhan akan autentikasi file-file digital sebenarnya tidak terbatas pada file-file teks saja, misalkan saja contohnya seseorang, sebut saja si A, meminta kepada si B untuk dibuatkan sebuah gambar logo untuk produk baru perusahaannya. Dalam proses pengerjaan, B seringkali mengirimkan salah satu *prototype* logo ke si A untuk dimintai pendapat. Pada saat inilah diperlukan suatu proses autentikasi untuk meyakinkan si A bahwa *prototype* logo yang dikirimkan oleh si B, benar-benar gambar yang dibuat dan dikirimkan oleh si B, serta tidak mengalami perubahan isi data dalam perjalanan.

Apabila ditelaah lebih jauh, dalam bidang steganografi, telah terdapat suatu teknik bernama watermarking, untuk penyimpanan data tambahan dalam suatu file media (gambar/citra, audio, dan video). Teknik ini sampai saat ini banyak diaplikasikan untuk penyembunyian data dan untuk legalisasi suatu file media (pencegahan pembajakan data media). Namun penggunaan teknik *watermarking* untuk tanda tangan digital saya rasa masih kurang efisien. Hal ini dikarenakan dalam teknik *watermarking*, perubahan data yang dilakukan cukup banyak. Walaupun *watermarking* sebenarnya dapat menyimpan data rahasia sangat banyak, namun pada algoritma tanda tangan digital, hanya dibutuhkan beberapa bit data untuk menyimpanan data tanda tangan (misalnya RSA + MD5 yang hanya membutuhkan 128 bit data untuk penyimpanan data tanda tangan).

Pada makalah ini, saya akan mengajukan kombinasi salah satu algoritma tanda tangan digital, yaitu RSA + MD5, dengan teknik steganografi untuk menyisipkan data tanda tangan dalam suatu file media digital (gambar/citra, audio, dan video) sebagai salah satu alternatif. Sebagai pendahuluan saya akan memberikan sedikit penjelasan mengenai digital signature, RSA, MD5 dan *watermarking* agar memperjelas perbedaannya. Serta akan dibahas pula beberapa definisi-definisi format-format file media yang umum digunakan dalam penyimpanan file-file media untuk mempermudah penjelasan analisis selanjutnya dalam proses penyisipan data tanda tangan digital ke dalam file media.

2. DESKRIPSI & PENJELASAN DASAR

2.1. Digital Signature

Tanda-tangan digital merupakan suatu nilai kriptorafis yan terkandung pada pesan dan pengirim pesan. Dengan tanda-tangan digital , maka integritas data dapat terjamin dan juga dapat digunakan untuk membuktikan asal pesan serta pihak yang mengirimkan pesan tersebut. Panandatanganan pesan dapat dilakukan dengan dua cara yaitu :

1. Enkripsi Pesan

Dengan meng-enkripsi pesan, secara otomatis telah menyediakan ukuran otentikasi yang menyatakan bahwa pesan tersebut telah ditandatangani. Penandatanganan dengan cara enkripsi pesan dapat dilakukan dengan menggunakan algoritma kriptografi kunci simetri atau kriptografi kunci public.

2. Dengan Fungsi Hash

Penandatanganan dengan menggunakan fungsi hash hanya menyediakan fitur untuk membuktikan otentikasi pasan saja namun tidak menyembunyikan pesan tersebut kedalam bentuk cipherteks, karena dalam beberapa persoalan, terdapat beberapa dokumen yang memang hanya membutuhkan otentikasi saja tapi tidak untuk kerahasiaannya.

Tahap-tahap yang dilakukan saat penandatanganan adalah sebagai berikut :

- Mula-mula pengirim pesan menghitung *message digest* (MD) dari pesan dengan menggunakan fungsi hash
- Selanjutnya MD tersebut dienkripsi dengan kunci *private* si pengirim.
- MD terenkripsi tersebutlah yang kemudian kita sebut sebagai tanda-tangan digital yang kelak akan dilekatkan pada akhir atau awal dokument

Tahap-tahap yang dilakukan saat melakukan verifikasi adalah sebagai berikut :

- Mula-mula penerima pesan akan menghitung MD' dari pesan yang dikirim kepadanya.
- Kemudian penerima pesan akan mendekripsi pesan dengan kunci *public* sipengirim untuk memperoleh MD yang sesungguhnnya dari dokumen tersebut.
- Kemudian penerima akan membandikan kedua MD tersebut. Jika kedua MD tersebut berbeda, maka telah terjadi perubahan pada pesan tersebut dan jika MD tersebut sama, berati pesan yang dikirim terbukti keotentikannya.

2.2. RSA

RSA merupakan salah satu algoritma kunci public yang cukup popular dan banyak digunakan karena tingkat keamanannnya yang cukup tinggi. RSA pertama kali ditemukan oleh Ron (R)ivest, Adi (S)hamir, dan Leonard (A)dleman pada tahun 1976. Algoritma ini didasarkan pada teorema euler dan tingkat keamanannnya terletak pada sulitnya memfaktorkan bilangan yang besar menjadi faktor-faktor prima.

Algoritma RSA memiliki besaran-besaran sebagai berikut :

- P dan q --> (prima) rahasia
- n=p.q --> tidak rahasia
- Φ(n)=(p-1)(q-1) --> rahasia
- e (kunci enkripsi) --> tidak rahasia
- d (kunci dekripsi) --> rahasia
- m (plainteks) --> rahasia
- c (cipherteks) --> tidak rahasia

Algoritma Membangkitkan pasangan kunci

i. pilih dua buah bilangan prima sembarang , p dan q
ii. hitung $n = p.q$
iii. hitung $\Phi(n)=(p-1)(q-1)$
iv. pilih kunci *public* (kunci enkripsi) *e* yang relatif prima terhadap $\Phi(n)$
v. bangkitkan kunci *private* (kunci dekripsi) *d* dengan menggunakan persamaan :

$$e \cdot d \equiv 1 (\mathrm{mod}\ \Phi(n))$$

sehingga

$$d = \frac{1 + k\,\Phi(n)}{e}$$

Algortima Enkripsi

i. ambil kunci *public* penerima pesan *e* & modulus *n*
ii. ubah plainteks kedalam bentuk heksadecimal
iii. nyatakan plainteks dalam blok-blok *m1, m2, ...,* sedemikian sehingga setiap blok merepresentasikan nilai dalam selang [0, n-1]
iv. setiap blok mi dienkripsi menjadi blok *ci* dengan rumus ***ci = mi ^e mod n***

Algortima Dekripsi

i. setiap blok ci didekripsi kembali menjadi blok m dengan menggunakan rumus ***mi = ci ^d mod n***

2.3. MD5

MD5 adalah fungsi hash satu arah yang dibuat olah Ronald Rivest pada tahun 1991. Algoritma MD5 menerima masukan pesan yang berukuran sembarang dan menghasilkan message digest yang panjangnnya 128 bit. Langkah-langkah pembuatan *message digest* secara garis besar adalah sebagai berikut :

- *Penambahan bit-bit pengganjal (padding bits)*
  Pesan ditambah dengan sejumlah bit pengganjal sehingga panjang pesan kongruen dengan 448 modulo 512. Bit pengganjal diisi dengan sebuah bit 1 kemudia diikuti dengan bit 0 untuk sisanya.
- *Penambahan nilai panjang pesan semula*
  Pesan yang telah diberi bit-bit pangganjal selanjutnya ditambahkan lagi dengan 64 bit yang menyatakan panjang pesan.
- *Inisialisasi penyangga (buffer) MD*
  MD5 membutuhkan 4 buah penyangga yang masing-masingnya berukuran 32 bit. Keempat bit

penyangga ini menampung hasil antara dan hasil akhir. Keempat bit penyangga dan inisialisasinya adalah sebagai berikut :

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

- *Pengolahan pesan dalam blok berukuran 512*

  Pesan dibagi menjadi L buah blok yang masing-masing panjangnya 512 bit. Setiap blok di proses berasama penyangga MD yang menghasilkan keluaran 128 bit, proses ini di sebut HMD5. Proses HMD5 terdiri dari 4 putaran dimana setiap putaran melakukan 16 operasi dasar MD5 dan tiap operasinya memakai sebuah elemen T yang nilainya tertera pada tabel T yang di peroleh dari perihitungan berdasarkan rumus yang telah ditentukan.

Operasi MD5 dapat ditulis dengan persamaan sebagai berikut :

A <-- b + CLSs(a + g(b,c,d) + X[k] + T[i])

yang dalam hal ini ,

a,b,c,d --> empat buah penyangga 32 bit
g --> salah satu fungsi F, G, H, I
CLSs **-->** circular left bit sebanyak s bit
X[k] --> kelompok 32 bit ke-k dari blok 512 bit message ke-q.
T[i] --> elemen tabel ke i (32 bit)
+ --> operasi penjumlahan modulo 2^32

Karena ada 16 kali operasi dasar yang dilakukan, maka setiap kali satu operasi dasar dilakukan, penyangga-penyangga itu di geser ke kanan secara sirkuler dengan cara pertukaran sebagai berikut :

Temp --> d
d --> c
c --> b
b --> temp

Berikut merupakan tabel fungsi-fungsi dasar MD5

Nama	Notasi	g(b,c,d)
Ff	F(b,c,d)	(b ^ c) v (~b ^ d)
fG	G(b,c,d)	(b ^ d) v (c ^ ~d)
fH	H(b,c,d)	b Xor c Xor d
fI	I(b,c,d)	c Xor (b V ~d)

2.4. Digital Signature menggunakan RSA+MD5

Tahap-tahan yang dilakukan saat penandatanganan adalah sebagai berikut :

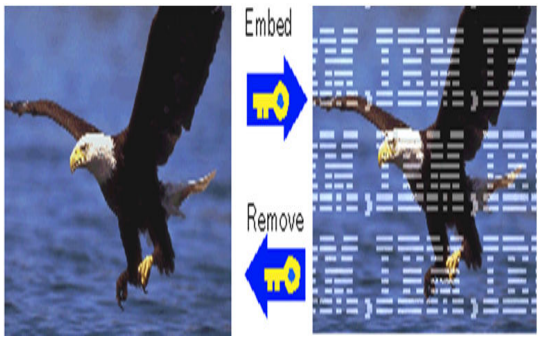
- Mula-mula pengirim pesan menghitung *message digest* (MD) dari pesan dengan menggunakan algoritma MD5 dan menghasilkan MD yaitu ***h***
- Selanjutnya MD tersebut dienkripsi dengan kunci *private* sipengirim dengan rumus ***S = h ^ sk mod n***, dimana sk adalah kunci *private*.
- Pengirim kemudian mentransmisikan (M + S) tersebut ke penerima.

Tahap-tahap yang dilakukan saat melakukan verifikasi adalah sebagai berikut :

- Mula-mula penerima pesan akan menghitung MD dari pesan yang dikirim kepadanya dengan menggunakan algoritma MD5 dan menghasilkan MD yaitu ***h'***
- Kemudian penerima pesan akan mendekripsi pesan dengan kunci *public* sipengirim untuk memperoleh MD yang sesungguhnnya dari dokumen tersebut dengan rumus ***h = S ^ pk mod n,*** dimana pk adalah kunci public pengirim pesan.
- Kemudian penerima akan membandikan h dan h' tersebut. Jika kedua MD tersebut berbeda, maka telah terjadi perubahan pada pesan tersebut dan jika kedua MD tersebut sama, berati pesan yang dikirim terbukti keotentikannya.

2.5. Watermarking

Yang dimaksud dengan watermarking disini adalah *digital watermarking*, yaitu watermarking yang dilakukan pada file-file digital. *Digital watermarking*, secara umum berarti penyisipan informasi (yang biasanya disebut *watermark*, dan dapat berupa apa saja, seperti gambar/citra, audio, video, teks) ke dalam dokumen digital untuk berbagai tujuan, misalnya perlindungan *copyright*/kepemilikan, *fingerprinting*, otentikasi (integritas content), dsb.



Gambar 1. Contoh aplikasi watermark pada citra

Beberapa aplikasi watermarking yang umum sekarang ini, antara lain:

- Memberi label kepemilikan (*ownership*) pada karya digital
- Melindungi isi karya digital (*copyright*).
- Memeriksa integritas isi karya digital (*tamper proofing*) ➔ *Data authentication*
- *User authentication/fingerprinting*: mengotentikasi pengguna spesifik. Contoh: distribusi DVD
- Aplikasi medis: foto sinar-X diberi *watermark* berupa ID pasien (memudahkan identifikasi pasien).
- *Covert communication*: untuk sistem komunikasi di negara2 di mana kriptografi tidak dibolehkan.
- *Piracy protection*: mencegah penggandaan yang tidak berizin.

Terdapat banyak sekali teknik/algoritma dalam melakukan proses *watermarking* ini. Seperti misalnya pada *watermarking* citra, terdapat dua jenis watermarking, yaitu *visible* dan *invisible*, dimana pada proses *visible watermarking*, hasil akhir akan berupa gambar dengan *watermark* yang mudah dilihat dengan mata telanjang (seperti yang dapat dilihat pada gambar 1). Sedangkan untuk *invisible watermark*, dibutuhkan beberapa proses terlebih dahulu untuk dapat melihat/mengetahui watermark yang disisipkan.

Pada makalah ini, saya akan lebih fokus dalam membahas salah satu aplikasi *watermarking*, yaitu pada bagian otentikasi file digital (khususnya file media citra, audio dan video). Seperti yang dibicarakan sebelumnya pada bagian pendahuluan, penggunaan teknik *watermarking* untuk autentikasi file-file digital masih sangat tidak efisien sehingga sangat jarang ada penggunaan *watermark* untuk autentikasi. Pada umumnya sebagian besar orang lebih suka untuk mengkompresi file-file digital tersebut menjadi suatu file kompresi, sehingga proses autentikasi hanya perlu dilakukan pada file kompresi tersebut.

2.6. Format File Gambar (Citra)

Secara mendasar, sebuah file gambar akan berisikan suatu header file (hampir semua file media akan memiliki header file semacam ini). Header ini akan menyimpan data-data gambar seperti nama file gambar, ukuran, format, dsb. Kemudian disambung dengan body file, pada bagian body, akan berisi sederetan bit/karakter yang akan merepresentasikan suatu warna atau kombinasi warna RGB dan terkadang *alpha factor*. Biasanya deretan ini akan ditampilkan dalam bentuk matriks terurut.

Pada saat ini, hampir seluruh format file gambar (citra) merupakan format file gambar yang telah terkonversi atau menggunakan konvensi/aturan-aturan tertentu sehingga memungkinkan gambar disimpan dalam ukuran lebih kecil. Contohnya format gambar GIF, JPEG, PNG, TIFF, dsb. Belum lagi format-format gambar tingkat tinggi yang biasanya untuk pengolahannya harus menggunakan kakas pengolah gambar tingkat tinggi juga, seperti PSD, CDR dsb. Namun gambar-gambar yang sudah dikonversi ini, akan sangat sulit dilakukan modifikasi tanpa menggunakan kakas pengolah gambar yang tersedia. Oleh karena itu untuk selanjutnya pada makalah ini, kita akan mengasumsikan penggunaan file gambar super sederhana dimana baris-baris bitnya dituliskan satu persatu. Sedangkan untuk praktiknya, saya menggunakan format file BMP 24 bit yang sangat sederhana sehingga mudah dilakukan modifikasi.

2.7. Format File Audio

Pendekatan paling umum dalam penyimpanan data audio adalah menyimpan urutan voltase audio (bila dimisalkan dalam suatu pemutaran data, akan mengkorespondensikan suatu posisi dalam membrane speaker) dalam suatu channel individu dengan resolusi (jumlah bit per sampel) tertentu dalam interval yang tetap. Dalam pembahasan format file audio (video juga untuk kedepannya) akan dikenal istilah codec. Codec ini merupakan sesuatu yang akan melakukan proses encoding dan decoding suatu format file audio dan mengambil data audio sebenarnya. Suatu format file audio dapat mendukung berbagai codec untuk proses encode dan decodenya.

Terdapat 3 jenis format file audio, yaitu:

- *Uncompressed audio format*
  Format file audio yang benar-benar murni, belum dikompresi sama sekali sehingga tidak memerlukan codec untuk proses playbacknya. Contohnya format WAV pada Windows, serta format AIFF pada MAC OS.
- *Lossless compression audio format*
  Format file yang mengalami *lossless compression*, artinya file hasil kompresi dapat dikembalikan menjadi file raw dengan kualitas dan isi yang benar-benar sama dengan file raw sebelum dikompresi. Contohnya lossless WMA, TTA, FLAC.
- *Lossy compression audio format*
  Format file yang mengalami *lossy compression*, dimana file hasil kompresi tidak dapat dikembalikan menjadi file raw yang sama persis dengan file raw asli, namun dari segi kualitas tidak terdapat perbedaan yang cukup signifikan antara keduanya. Contohnya MP3, lossy WMA, AAC.

Pada makalah ini diasumsikan bahwa format file audio akan cukup sederhana yaitu berupa rentetan sekelompok bits, dimana setiap kelompok akan menampilkan besarnya voltase yang dibutuhkan per suatu satuan interval waktu format file tersebut.

2.8. Format File Video

Biasa disebut juga format container. Dalam format file ini terkandung gabungan dari file citra, file audio, dan bahkan file teks. Sehingga pengelompokkan file container ini biasanya dilakukan berdasarkan jenis kompresi yang digunakan. Pemutaran file-file container ini juga menggunakan codec (untuk file-file yang sudah dikompresi). Berikut adalah beberapa contoh file container yang sering digunakan:

- 3gp (digunakan pada telepon genggam)
- ASF (container standar untuk WMA dan WMV)
- AVI (container standar Microsoft)
- MP4 (standar untuk MPEG-4)
- MOV (standar quicktime video dari Apple Inc.)
- RealMedia (standar container untuk Real)
- MPEG (standar container video yang sering digunakan saat ini)

Selanjutnya di makalah ini, pembahasan format file video akan diasumsikan sebagai file container yang mengandung file-file citra dan audio sesuai dengan asumsi sebelumnya.

3. KOMBINASI RSA+MD5 PADA FILE MEDIA

3.1. Proses Umum

Secara umum proses yang berjalan sama persis dengan proses pembubuhan tanda tangan digital menggunakan RSA+MD5 pada dokumen teks. Jadi pada awal proses, orang yang akan melakukan pembubuhan tanda tangan, sebut saja si A harus telah memiliki kunci public dan kunci private yang telah digenerasi oleh generator kunci untuk algoritma RSA. Kemudian si A akan membubuhkan tanda tangannya ke file media bersangkutan menggunakan kunci private yang ada di tangannya (teknisnya akan dibahas belakangan). Kemudian file media bersangkutan sudah siap dikirimkan ke (misalkan) si B. Ketika si B menerima file, untuk melakukan proses autentikasi, B hanya memerlukan kunci public dari A.

3.2. Analisis Visibilitas

Terdapat beberapa hal yang dapat dianalisis sebelum dilakukan suatu aplikasi penggunaan teknik kombinasi RSA+MD5 untuk proses tanda tangan digital pada file media. Ada beberapa fakta yang dapat diperhatikan untuk mempermudah proses aplikasi kedepannya. Pertama, fakta bahwa dalam proses tanda tangan digital yang menggunakan kombinasi dari RSA dan MD5, akan dibutuhkan tambahan space pada file yang ditandatangani sebesar data tanda tangan yang dibubuhkan. Karena menggunakan MD5 yang selalu menghasilkan 128 bit, maka ukuran data tanda tangan yang dibubuhkan akan selalu 128 bit. Hal ini akan memudahkan proses pembubuhan tanda tangan karena ukurannya selalu fix.

Fakta kedua adalah bahwa ukuran rata-rata dari file-file media (gambar/citra, audio dan video) sangat besar. Apalagi pada jaman sekarang kualitas selalu menjadi nomor satu sehingga ukuran pun berkembang menjadi sangat besar. Untuk ukuran sebuah gambar logo kecil berukuran 32x32 pixel, bisa mencapai 10kB (untuk yang high quality), sedangkan untuk kualitas sedang bisa mencapai 2-4 kB. Ini adalah ukuran minimum untuk file-file media seperti ini, file-file media yang beredar pada umumnya berukuran jauh lebih besar. Sehingga dapat dilihat disini bahwa penyisipan data tanda tangan sebesar 128 bit, pada file berukuran sebesar itu tidak akan berpengaruh apa-apa. Jadi aplikasi tanda tangan digital menggunakan kombinasi RSA+MD5 untuk file-file media sangat visible untuk dilakukan mengingat ukuran tanda tangannya yang sangat kecil dan ukuran file-file media yang relatif besar, sehingga penyisipan mudah dilakukan.

3.3. Teknis Aplikasi Steganografi

Untuk menyisipkan data 128 bit tanda tangan pada file media, diperlukan aplikasi dari teknik steganografi, dalam menyembunyikan data tersebut dan sebisa mungkin tidak mengubah kualitas dari file media yang disisipkan tersebut. Pada file gambar/citra misalnya, pendekatan paling sederhana adalah dengan menyisipkan bit-bit tanda tangan tersebut pada 128 pixel yang direpresentasikan pertama kali. Namun mau tidak mau kita akan mengorbankan satu bit dari salah satu representasi warna atau alpha transparency. Misalkan saja terdapat sebuah file gambar yang berupa hamparan pixel 3x3 berwarna putih (R=255, G=255, B=255), maka pada file gambar akan muncul representasi gambar sbb (misalkan representasi dituliskan dalam bentuk HEXA, dengan format file gambar yang cuma merepresentasikan 24 bit warna):

```
FFFFFF FFFFFF FFFFFF
FFFFFF FFFFFF FFFFFF
FFFFFF FFFFFF FFFFFF
```

Dimana setiap kombinasi 6 HEXA (2 digit pertama untuk nilai R, 2 digit kedua untuk G, 2 digit terakhir untuk B) akan membentuk suatu *pixel*, dengan urutan representasi *pixel* dari kiri atas ke kanan bawah (sebagian besar format file gambar tidak menggunakan urutan seperti ini). Sehingga apabila diputuskan format penyisipannya adalah pada warna B (*Blue*) dan akan disisipkan bit-bit 001010111, maka file tersebut akan berubah menjadi:

```
FFFFFF FFFFFF FFFFFE
FFFFFF FFFFFE FFFFFF
FFFFFE FFFFFE FFFFFE
```

Dapat dilihat dimana akan terjadi perubahan hanya pada bit terakhir representator untuk salah satu warna yang diputuskan sebagai tempat penyimpanan nilai tanda tangan digital. Sehingga untuk file gambar akan dibutuhkan 128 *pixel* awal untuk tempat penyimpanan data tanda tangan. Ide dasar ini juga berlaku untuk file audio, dimana bit terakhir diubah-ubah untuk merepresentasikan 128 bit tanda tangan. Demikian juga halnya dengan file container yang mengandung kedua jenis file ini.

Ide dasar penyisipan ini sebenarnya dapat dikembangkan lebih jauh, misalnya dengan penerapan pola-pola tertentu dalam penyimpanan data tanda tangan sehingga lebih menyulitkan kriptanalis untuk melakukan penyerangan, bisa juga dengan menggunakan perhitungan-perhitungan tambahan yang melibatkan bit-bit lainnya, dan berbagai metoda lainnya.

3.4. Masalah Implementasi

Dalam aplikasi metoda ini pada file media terdapat satu masalah besar. Coba kita tinjau ulang implementasi pada dokumen teks, data tanda tangan dibubuhkan dengan menambahkan data pada dokumen, sedangkan dengan file media, kita melakukan modifikasi data untuk membubuhkan data tanda tangan. Hal ini sangatlah berpengaruh besar, karena seperti yang kita ketahui, proses pembuatan tanda tangan dan proses verifikasi, akan melibatkan seluruh isi dokumen. Jadi misalkan terdapat suatu fiile media asli X, dimana setelah dibubuhkan tanda tangan akan menjadi file media Y. Ketika dilakukan proses

autentikasi file, maka dibutuhkan file media X untuk melakukan proses hash sebelum dilakukan pencocokan dengan hasil dekripsi kunci public terhadap data tanda tangan digital. Sedangkan file media yang diterima adalah Y, dan tidak ada cara untuk mendapatkan X kembali, karena tanda tangan yang disisipkan pada X sehingga menjadi Y juga dibutuhkan.

Satu-satunya solusi dalam memecahkan masalah ini tanpa melakukan penambahan ukuran dari file media (karena penambahan ukuran pada file media akan mengakibatkan perubahan yang cukup signifikan pada file media tersebut), adalah dengan melakukan pengabaian pada bit-bit yang akan atau telah dibubuhi data-data bit dari tanda tangan digital yang dipergunakan. Dengan demikian maka proses autentikasi masih dapat berjalan dengan baik.

3.5. Keunggulan Metoda

Keunggulan dari metoda ini dalam penggunaannya sebagai metoda alternatif untuk autentikasi file-file media (gambar/citra, audio dan video) antara lain:

- Hanya merubah sebagian kecil (sangat sedikit – 128 bits) dari file-file media yang dibubuhi tanda tangan sehingga tidak memberikan penurunan kualitas yang signifikan.
- Memiliki tingkat keamanan yang sangat tinggi, karena apabila sebuah bit berubah, baik itu bit dokumen maupun bit yang berupa penyisipan tanda tangan, maka file akan dianggap tidak valid.
- Metoda ini dapat dikembangkan jauh lebih dalam untuk menjamin keamanan dan memperkecil tingkat robustness dari file media yang dibubuhi tanda tangan.

3.6. Masalah Aplikatif

Saat melakukan aplikasi dari metoda ini, secara umum proses autentikasi dan penanda tanganan dapat berjalan dengan baik. Namun masih terdapat beberapa masalah aplikatif yang dapat ditemukan dalam beberapa kasus pemakaian, misalnya:

- Pada saat pembubuhan tanda tangan dilakukan berbagai proses konversi dan kompresi file media (karena sebagian besar proses konversi dan kompresi file-file media bersifat *lossy*, maka ada kemungkinan file berubah sehingga dianggap tidak otentik walaupun sebenarnya otentik)

4. KESIMPULAN

Metoda kombinasi algoritma tanda tangan digital menggunakan RSA+MD5 dengan steganografi untuk file-file media dapat dijadikan salah satu metoda alternatif dalam pembubuhan tanda tangan digital karena tingkat keamanan yang cukup tinggi serta metoda ini tidak mengubah file media terlalu banyak (hanya 128 bits) dan juga metoda ini masih sangat dasar dan dapat dikembangkan lebih jauh.

5. DAFTAR REFERENSI

[1] Munir, Ir. Rinaldi, M.T. *Diktat Kuliah IF5054 Kriptografi*. Teknik Informatika ITB, 2006

[2] http://crpit.com/confpapers/CRPITV32Thomborson.pdf

[3] http://www.muppetlabs.com/~breadbox/txt/rsa.html

[4] http://www.mip.informatik.uni-kiel.de/~wwwadmin/Lehre/WS0506/seminar/DigitalWatermarking.pdf

[5] http://www.infosecwriters.com/text_resources/pdf/Steganography_AMangarae.pdf