Seminar Nasional Aplikasi Teknologi Informasi 2005 (SNATI 2005)
Yogyakarta, 18 Juni 2005
ISBN: 979-756-061-6

PENGGUNAAN TANDA-TANGAN DIGITAL UNTUK MENJAGA INTEGRITAS BERKAS PERANGKAT LUNAK

Rinaldi Munir
Laboratorium Ilmu dan Rekayasa Komputasi
Departemen Teknik Informatika, Fakultas Teknologi Industri, Institut Teknologi Bandung
Jl. Ganesha 10 Bandung
E-mail: rinaldi@informatika.org

Abstrak

Pendistribusian perangkat lunak melalui web di internet memiliki sejumlah masalah. Salah satunya adalah masalah integritas berkas yang telah di-download. Integritas berkas perangkat lunak berkaitan dengan keaslian berkas program, keutuhan, dan keabsahan pengembang perangkat lunak. Berkas program dapat dimodifikasi oleh pihak ketiga (menjadi tidak asli) atau mengalami kerusakan (corrupt) oleh virus atau gangguan selama transmisi dari komputer server ke komputer client (menjadi tiadk utuh). Selain itu, pengguna perangkat lunak perlu memastikan bahwa program yang ia download dibuat oleh pengembang program yang sah, dan bukan pengembang lain yang menyamar sebagi pengembang program yang asli. Masalah integritas berkas perangkat lunak ini dapat diselesaikan dengan menggunakan tanda-tangan digital. Tanda tangan digital dibangkitkan dengan algoritma kriptografi kunci-publik. Tanda-tangan digital bergantung pada isi berkas program dan kunci pengembang perangkat lunak. Melalui proses verifikasi, pengguna dapat membuktikan integriats berkas perangkat lunak yang ia downlaod dari situs web pengembang.

***Kata kunci**: perangkat lunak, internet, integritas, kriptografi, tanda-tangan digital, kunci, verifikasi*

1. Pendahuluan

Perkembangan internet yang pesat telah memudahkan orang dalam mendistribusikan berkas digital (*file*), termasuk dokumen program atau perangkat lunak. Salah satu cara pendistribusian yang sering digunakan adalah menaruh berkas program di dalam situs *web server*. Pengguna (*user*) cukup mengakses situs *web* tersebut kemudian men-*download* berkas program ke dalam *disk*.

Program yang di-*download* dari situs *web* ada yang gratis (*free*) dan ada pula yang harus dibeli. Untuk yang terakhir ini, pembeli harus melakukan pembayaran secara *online* (biasanya dengan menggunakan kartu kredit) terlebih dahulu sebelum dapat melakukan proses *download* program.

Salah satu masalah yang timbul dari pendistrubisian program melalui internet adalah mengenai integritas program tersebut. Pengguna (atau pembeli) yang telah men-*download* berkas program perlu meyakini bahwa program yang ia *download* masih asli dan utuh. Berkas program dikatakan asli apabila ia tidak diubah oleh pihak lain seperti *hacker* atau oleh virus. Berkas program dikatakan utuh apabila ia tidak rusak (*corrupt*) karena gangguan fisik selama proses transmisi dari komputer *server* ke komputer pengguna atau rusak karena serangan virus. Selain itu, pengguna juga perlu memastikan bahwa program tersebut memang benar dibuat oleh pengembang (*developer*) yang sah, bukan pengembang lain yang menyamar sebagai pengembang yang asli. Penyamaran mungkin dimaksudkan untuk mencuri data rahasia pembeli perangkat lunak (misalnya kode PIN kartu kredit).

Masalah integritas berkas perangkat lunak dapat diselesaikan dengan menggunakan tanda-tangan digital (*digital signature*). Dalam kehidupan sehari-hari, tanda tangan sudah lama digunakan sebagai bukti otentik dokumen cetak (seperti surat, ijazah, karya seni, dan lain-lain). Dokumen yang sudah ditandatangani tidak diragukan lagi integritasnya (baik keotentikan isinya maupun keabsahan orang yang menulis dokumen tersebut). Fungsi tanda-tangan pada dokumen cetak juga diterapkan pada dokumen digital dengan cara yang berbeda.

Tanda-tangan digital adalah suatu nilai kriptografis yang bergantung pada isi berkas digital dan kunci pemilik berkas digital. Tanda-tangan ini dapat ditaruh (*embed*) di dalam berkas digital atau disimpan di dalam berkas terpisah. Proses verifikasi dilakukan untuk membuktikan otentikasi tanda tangan digital tersebut. Jika tanda-tangan digital otentik, berarti berkas digital masih asli dan dan pemiliknya adalah orang yang sah.

Tanda tangan digital dapat diberikan pada sembarang jenis berkas digital, termasuk berkas program. Hal ini membuka peluang kemungkinan penggunaan tanda tangan digital untuk menjaga integritas perangkat lunak.

2. Dasar Teori

Sebelum membahas penggunan tanda tangan pada berkas perangkat lunak, terlebih dahulu dijelaskan beberapa konsep yang melatarbelakangi makalah ini.

 ISBN: 979-756-061-6

2.1 Kriptorafi

Kriptografi adalah ilmu dan seni untuk menjaga keamanan pesan [1]. Dalam kriptografi, pesan yang mempunyai makna disebut plainteks (*plaintext*), dan pesan yang tidak bermakna lagi disebut cipherteks (*ciphertext*). Dua proses utama dalam kriptografi adalah **enkripsi** (*encryption*) dan **dekripsi** (*decryption*). Enkripsi adalah transformasi plainteks menjadi cipherteks, sedangkan transformasi sebaliknya dari cipherteks menjadi plainteks semula disebut dekripsi. Baik enkripsi maupun dekripsi, proses keduanya melibatkan penggunaan **kunci** (*key*). Enkripsi dan dekripsi dinyatakan secara matematis sebagai fungsi:

$$C = E_K(P) \quad (1)$$

dan

$$P = D_K(C) \quad (2)$$

yang dalam hal ini, *C* adalah cipherteks, *P* adalah plainteks, E adalah fungsi enkripsi, *D* adalah fungsi dekripsi, dan K adalah kunci yang digunakan selama transformasi.

Sistem kritpografi (yang terdiri atas algoritma kriptografi, kunci, plainteks, dan cipherteks) dapat digolongkan menjadi dua kelompok: sistem kriptografi simetri dan sistem kriptografi kunci-publik. Pada sistem kriptografi simetri, kunci untuk enkripsi sama dengan kunci untuk dekripsi. Kunci ini bersifat rahasia sehingga ia disebut juga *secret key*. Contoh algoritma kriptorafi simetri adalah *DES* (*Data Encryption Standard*) dan yang terbaru adalah *AES* (*Advanced Encryption Standard*).

Sedangkan pada sistem kriptografi kunci-publik, kunci untuk enkripsi tidak sama dengan kunci untuk dekripsi. Kunci untuk enkripsi tidak rahasia, sehingga dinamakan juga kunci publik (*public key*), sedangkan kunci kunci untuk dekripsi rahasia, sehingga dinamakan kunci privat (*privat key*). Pengirim pesan mengenkripsi pesan dengan menggunakan kunci publik si penerima pesan; hanya penerima pesan yang dapat mendekripsi pesan menjadi plainteks semula dengan menggunakan kunci privatnya. Contoh algoritma kriptografi kunci-publik adalah RSA (*Rivest-Shamir-Adleman*) dan ElGamal.

2.2 Fungsi *Hash*

Fungsi *hash* adalah fungsi yang menerima masukan *string* yang panjangnya sembarang dan mengkonversinya menjadi *string* keluaran yang panjangnya tetap (*fixed*) (umumnya berukuran jauh lebih kecil daripada ukuran *string* semula).

Jika *string* menyatakan pesan (*message*), maka sembarang pesan *M* berukuran sembarang dikompresi oleh fungsi *hash H* melalui persamaan

$$h = H(M) \quad (3)$$

Keluaran fungsi *hash* disebut juga **nilai *hash*** (*hash-value*) atau **pesan-ringkas** (*message digest*). Pada persamaan (3), *h* adalah nilai *hash* atau *message digest* dari fungsi *H* untuk pesan *M*.

Fungsi *hash* satu-arah adalah fungsi *hash* yang bekerja dalam satu arah: pesan yang sudah diubah menjadi pesan-ringkas tidak dapat dikembalikan lagi menjadi pesan semula. Contoh fungsi *hash* satu-arah adalah *MD5* dan *SHA*. *MD5* menghasilkan mesan-ringkas yang berukuran 128 bit, sedangkan *SHA* menghasilkan pesan-ringkas yang berukuran 160 bit.

2.3 Tanda-tangan Digital

Tanda tangan pada data digital disebut **tanda-tangan digital** (*digital signature*). Yang dimaksud dengan tanda-tangan digital di sini bukanlah tanda tangan yang di-dijitasi dengan alat *scanner*, tetapi suatu nilai kriptografis yang bergantung pada pesan dan pengirim pesan (Hal ini kontras dengan tanda tangan pada dokumen kertas yang bergantung hanya pada pengirim dan selalu sama untuk semua dokumen).

Teknik yang umum digunakan untuk membentuk tanda-tangan digital adalah dengan fungsi *hash* dan melibatkan algoritma kriptografi kunci-publik [3]. Mula-mula pesan *M* ditransformasi oleh fungsi *hash H* menjadi pesan ringkas *h*. Pesan ringkas tersebut dienkripsi dengan kunci privat (*PK*) pengirim pesan:

$$S = E_{SK}(h) \quad (4)$$

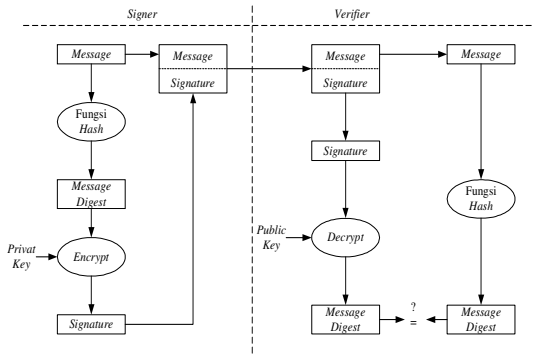
Hasil enkripsi (*S*) inilah yang disebut tanda-tangan digital. Tanda-tangan digital dapat ditambahkan (*append*) pada pesan atau terpisah dari pesan dan dikirim secara bersamaan.

Di tempat penerima, tanda-tangan diverifikasi untuk dibuktikan keotentikannya dengan cara berikut:

a. Tanda-tangan digital *S* didekripsi dengan menggunakan kunci publik (*PK*) pengirim pesan, menghasilkan pesan-ringkas semula, *h*, sebagai berikut:
$$h = D_{PK}(S) \quad (5)$$
b. Pengirim kemudian mengubah pesan *M* menjadi pesan ringkas *h'* dengan menggunakan fungsi *hash* satu-arah yang sama dengan fungsi *hash* yang digunakan oleh pengirim.
c. Jika *h'* = *h*, berarti tanda-tangan yang diterima otentik dan berasal dari pengirim yang benar.

Gambar 1 memperlihatkan proses pembangkitan tanda tangan digital (*signing*) dan verifikasi tanda tangan digital (*verivying*) [2].

ISBN: 979-756-061-6



Gambar 1. Proses penandatanganan dan verifikasi

Keotentikan dapat dijelaskan sebagai berikut:

a. Apabila pesan *M* yang diterima sudah berubah, maka *h'* yang dihasilkan dari fungsi *hash* berbeda dengan *h* semula. Ini berarti pesan tidak asli lagi.
b. Apabila pesan *M* tidak berasal dari orang yang sebenarnya, maka *h* yang dihasilkan dari persamaan 3 berbeda dengan *h'* yang dihasilkan pada proses verifikasi (hal ini karena kunci publik yang digunakan oleh penerima pesan tidak berkoresponden dengan kunci privat pengirim).
c. Bila *h* = *h'*, ini berarti pesan yang diterima adalah pesan yang asli dan orang yang mengirim adalah orang yang sebenarnya.

3. Aplikasi Tanda-tangan Digital untuk Berkas Perangkat Lunak

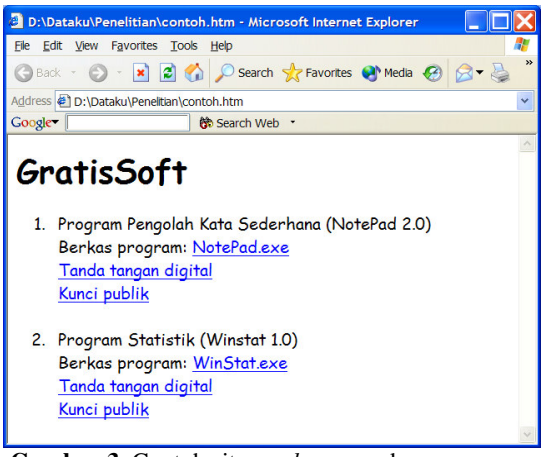
Tanda tangan digital dapat diberikan ke sembarang data digital, tidak hanya pesan, termasuk di dalamnya berkas program. Ada dua cara yang dapat dilakukan dalam penggunaan tanda tangan digital untuk berkas program.

Cara pertama, tanda tangan diletakkan pada berkas terpisah dari berkas *executable*. Menambahkan (*append*) tanda tangan digital langsung ke dalam berkas program *executable* dapat menyebabkan program menjadi rusak sehingga ia tidak bisa dieksekusi (*run*). Oleh karena itu, tanda tangan harus diletakkan pada berkas terpisah. Tanda tangan digital dibangkitkan mula-mula dengan menghitung nilai *hash* (atau pesan-ringkas) dari berkas program, lalu nilai *hash* ini dienkripsi dengan kunci privat pengembang program. Gambar 2 memperlihatkan tanda tangan digital dari berkas program notepad.exe yang dibangkitkan dengan perangkat lunak PGP (*Pretty Good Privacy*).

```
-----BEGIN PGP SIGNATURE-----
iQA/AwUAQnibsbPbxejK4Bb3EQJXvQCg8zN6UL0xnw
BTPR5FfWNt4uxh3AEAn2NC
/G2VTUrLpcSyo2l/S/D/+rUl=pZeh
-----END PGP SIGNATURE-----
```

Gambar 2. Contoh tanda-tangan digital program NotePad

Di dalam situs *web*, berkas program ditaruh bersama-sama dengan berkas tanda tangan digitalnya beserta kunci publik pengembang perangkat lunak (Gambar 3).



Gambar 3. Contoh situs *web* pengembang program

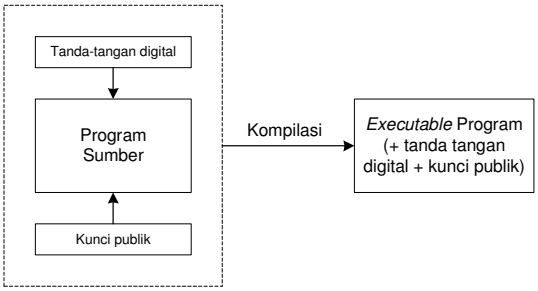
Pengguna (pembeli) program men-*download* ketiga macam berkas ini (berkas *executable*, berkas tanda tangan digital, dan berkas kunci publik pengembang perangkat lunak). Pengguna menguji integritas perangkat lunak yang ia *download* dengan melakukan verifikasi terhadap tanda tangan digital. Verifikasi tanda tangan digital menggunakan kunci publik pengembang program (asumsikan pengguna memiliki program verifikasi tanda tangan digital seperti *PGP*, atau program verifikasi disediakan oleh pengembang program di dalam situs *web* tersebut). Jika verifikasi OK, berarti program yang di-*download* asli, utuh, dan dibuat oleh pengembang program yang absah. Tetapi jika verifikasi gagal, berarti program yang di-*download* mengalami kerusakan, perubahan, atau tidak dibuat oleh pengembang program yang benar.

Namun, pemberian tanda tangan dengan cara pertama ini memiliki kelemahan [4]. Pihak ketiga yang memodifikasi program dapat membangkitkan kunci privat dan kunci publiknya sendiri. Lalu, ia menghitung tanda tangan digital dengan menggunakan kunci privatnya, selanjutnya mengganti kunci publik pengembang yang sah dengan kunci publiknya. Pengguna yang men-*download* ketiga berkas ini tidak mengetahui bahwa ia telah men-*download* program yang sudah berubah.

Cara kedua yang lebih aman adalah menambahkan tanda tangan digital (mungkin juga termasuk kunci publiknya) pada saat pengembangan program. Tanda tangan ini dibangkitkan dari hasil enkripsi terhadap nilai *hash* dari berkas *executable* program tersebut. Dengan teknik tertentu, program sumber dikompilasi kembali bersama-sama dengan tanda tangan digital tersebut (dan kunci publik), sehingga tanda-tangan digital (dan kunci publik) menyatu dengan berkas *executable*. Berkas

ISBN: 979-756-061-6

executable ini masih dapat dieksekusi. Gambar 4 memperlihatkan proses penyisipan tanda-tangan pada saat pengembangan program.



Gambar 4. Penyisipan tanda-tanagn digital pada saat pengembangan program.

Pengguna men-*download* berkas program tersebut dari situs *web*. Integritas perangkat lunak diuji di awal eksekusi program. Jika verifikasi OK, berarti program yang di-*download* asli, utuh, dan dibuat oleh pengembang program yang benar. Tetapi jika verifikasi gagal, berarti program yang di-*download* mengalami kerusakan, perubahan, atau tidak dibuat oleh pengembang program yang benar.

Meskipun cara kedua ini relatif aman daripada cara pertama, namun kelemahannya tetap masih ada. Jika program mengalami kerusakan selama transmisi, proses verifikasi tidak dapat dilakukan karena program tidak dapat dieksekusi (yang berarti ia tidak dapat memerika integritas dirinya sendiri).

4. Kesimpulan

Tanda-tangan digital dapat digunakan untuk menjaga integritas berkas program atau perangkat lunak yang diletakkan di dalam situs web. Integritas perangkat lunak menyangkut tiga hal: keaslian berkas, keutuhan berkas, dan keabsahan pengembang program.

Tanda tangan digital dibangkitkan dengan mengenkripsi nilai *hash* dari berkas program dengan menggunakan kunci publik pengembang program. Selanjutnya, tanda tanagn digital dapat disimpan di dalam berkas terpisah atau dikompialsi dengan berkas program sumber sehingga menyatu di dalam program.

Integritas perangkat lunak dilakukan dengan memverifikasi tanda tangan digital. Proses Verifikasi membutuhkan kunci publik pengembang program.

5. Saran Pengembangan

Dua cara pemberian tanda tangan yang sudah dikemukakan di atas masih mengandung sejumlah kelemahan. Perlu dipikirkan strategi pemberian tanda tangan lain yang cukup aman dari intervensi pihak ketiga.

Daftar Pustaka

[1] Bruce Schneier, Bruce, *Aplied Cryptography 2nd*, John Wiley & Sons, 1996

[2] Fred Piper & Sean Murphy, *Cryptography, A Very Short Introduction*, Oxford University Peress, 2002.

[3] Rinaldi Munir, *Bahan Kuliah IF5054 Kriptografi*, Departemen Teknik Informatiak ITB, 2004.

[4] Anugerah Redja Kusuma, Rizki Yulianto, dan Widhiyo Sudiyono, *Proteksi Perangkat Lunak dengan Algoritma Kriptogarfi Kunci Publik*, Tugas mata kuliah IF5054 Kriptografi, Departemen Teknik Informatika ITB, 2004.