Lisensi Perangkat Lunak dengan Menggunakan Tanda-Tangan Digital dengan Enkripsi

Sibghatullah Mujaddid – NIM : 13507124
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
sibgha07@students.itb.ac.id

***Abstract*—Pembajakan perangkat lunak merupakan salah satu permasalahan serius bagi kebanyakan pembuat perangkat lunak terutama bagi produk perangkat lunak komersil seperti sistem operasi, aplikasi perkantoran, dan lain-lain. Pembajak perangkat lunak dapat saja melakukan modifikasi, duplikasi, maupun menyebarkan perangkat lunak tanpa seizin pembuatnya. Dengan semakin berkembangnya penggunaan internet membuat semakin mudahnya pendistribusian perangkat lunak sehingga setiap orang dapat menggunakan perangkat lunak tanpa perlu seizin pembuatnya. Hal ini membuat sebagian besar pembuat perangkat lunak komersil membatasi penggunaan perangkat lunaknya sehingga hanya pihak tertentu yang dapat menggunakannya. Makalah ini membahas tentang salah satu cara yang dapat digunakan untuk mengatasi permasalahan pembajakan perangkat lunak dengan menerapkan tanda-tangan digital dengan enkripsi.**

***Index Terms*—Pembajakan, perangkat lunak, tanda-tangan digital, enkripsi.**

I. PENDAHULUAN

Pembajakan perangkat lunak merupakan salah satu permasalahan besar dalam perkembangan perangkat lunak terutama yang bersifat komersil. Hal ini sangat merugikan bagi pembuat perangkat lunak yang ingin membatasi agar perangkat lunaknya hanya dapat digunakan oleh pihak tertentu saja.

Dengan perkembangan internet yang semakin pesat, penyebaran dan pendistribusian perangkat lunak pun dapat dilakukan dengan cepat dan mudah. Mudahnya penyebaran file maupun perangkat lunak, tentunya semakin membantu penyebaran perangkat lunak bajakan. File dengan ukuran hingga gigabyte pun tidak menjadi masalah untuk didistribusikan, dengan bantuan bandwith Internet yang besar, setiap orang dapat memindahkan sebuah file berukuran beberapa gigabyte dari server di benua lain ke dalam harddisk komputer dalam hitungan jam bahkan menit.

Pembajak perangkat lunak dapat melakukan modifikasi, duplikasi, dan menyebarkan perangkat lunak tanpa seizin pembuatnya. Untuk sebuah perangkat lunak tanpa proteksi, duplikasi sangat mudah dilakukan. Cukup melakukan copy dan paste file-file perangkat lunak tersebut, maka seseorang dapat memiliki dua perangkat lunak yang identik. Kekhawatiran mengenai pembajakan perangkat lunak jauh-jauh hari telah disampaikan oleh Bill Gates pada tahun 1976 melalui sebuah sebuah surat terbuka.

Terdapat berbagai cara untuk proteksi perangkat lunak, namun, terdapat pula berbagai cara untuk melakukan pembajakan perangkat lunak. Berikut realita pembajakan perangkat lunak saat ini:

1. Serial Number

   Dengan maraknya pembajakan, maka developer berusaha mengamankan perangkat lunak-nya dengan berbagai cara proteksi.

   Berbagai cara proteksi mungkin akan semakin berkembang, namun, dalam sekejap bisa menjadi ketinggalan zaman, karena para pembajak juga akan terus berusaha melewati proteksi yang diciptakan, terutama jika perangkat lunak yang diproteksi tersebut memiliki kualitas dan nilai komersial yang tinggi.

   Proteksi serial number adalah proteksi dengan memberikan serial number pada perangkat lunak yang harus diisi dengan benar (umumnya pada saat instalasi) oleh pengguna perangkat lunak. Proteksi ini cukup sederhana tetapi juga lemah, developer menyediakan satu atau kumpulan serial number yang valid, dan setiap orang yang telah memperoleh salinan perangkat lunak dan serial number tersebut, akan dapat menggunakan perangkat lunak tersebut di komputer lain.

   Walaupun demikian, proteksi seperti ini masih cukup banyak digunakan. Pembajak perangkat lunak tidak identik dengan bajak laut yang seram dan menggunakan otot dan kekerasan, terkadang pembajak perangkat lunak disetarakan atau identik dengan hacker/cracker yang dapat memanipulasi perangkat lunak agar proteksi terlewati, ataupun mengorek informasi serial number dari rutin yang dijalankan executable file itu sendiri.

   Pembajak perangkat lunak umumnya menggunakan teknik debugger dan disassembler berikut tools yang siap digunakan, dan yang terutama adalah menggunakan kreativitas dan motivasi.

2. Activation Code

   Mirip dengan penggunaan serial number, yaitu pengguna harus memiliki serial number untuk dapat menggunakan perangkat lunak. Perbedaannya adalah pada cara mendapatkan serial number tersebut.

   Cara kerja activation code adalah seperti ilustrasi berikut. Saat diinstal, perangkat lunak akan memerika spesifikasi komputer pengguna dan menghasilkan kode aktivasi, kode aktivasi tersebut harus diregistrasikan oleh pengguna (umumnya via website atau telepon) ke vendor perangkat lunak untuk mendapatkan serial number yang sesuai.

   Proteksi ini lebih kuat pengamanannya dibandingkan dengan hanya memberikan serial number yang langsung dapat digunakan pada komputer mana saja.

   Permasalahannya adalah jika spesifikasi komputer pengguna berubah dan memerlukan instalasi ulang perangkat lunak. Serial number yang baru akan dibutuhkan, sehingga support dari vendor akan tetap diperlukan.

   Sementara dengan perkembangan hardware yang pesat, kegiatan seperti upgrade komputer atau instal ulang mungkin akan cukup sering dilakukan.

3. Dongle

   Dongle merupakan hardware yang dipasangkan secara fisik pada komputer melalui port yang terpasang (saat ini umumnya menggunakan port USB). Cara kerja dongle cukup sederhana. Dongle menyimpan informasi lisensi dalam bentuk hardware yang dibaca dan oleh perangkat lunak, perangkat lunak melakukan otentifikasi dan tidak akan bekerja jika dongle tidak terpasang atau tidak memiliki lisensi yang benar.

   Bagi para pembajak, menyalin dongle tidak semudah menyalin perangkat lunak. Tetapi, keamanan sering kali memiliki konsekuensi kenyamanan, paling tidak ketergantungan perangkat lunak terhadap dongle sangatlah tinggi, sementara alat kecil tersebut dapat dengan mudah dicabut dan dikantungi. Ini berarti pengguna perangkat lunak harus mempersiapkan proteksi pada dongle itu sendiri. Permasalahan lain adalah jika perangkat lunak tersebut harus terpasang pada sejumlah unit komputer, maka sebanyak jumlah unit komputer itu pula dongle yang diperlukan, kecuali pengguna perangkat lunak menggunakan konsep terminal services di mana perangkat lunak hanya diinstal pada server dan client mengakses perangkat lunak tersebut langsung dari server.

   Hingga saat ini kebanyakan perangkat lunak yang menggunakan proteksi dongle merupakan paket perangkat lunak high-end yang cukup mahal, dongle tidak umum digunakan oleh perangkat lunak seperti game atau aplikasi ringan.

4. Demo Version

   Developer perangkat lunak dapat membuat dua versi perangkat lunak yang berbeda. Versi pertama adalah demo version dengan keterbatasannya dan versi kedua adalah full version yang memiliki seluruh fitur yang dibutuhkan, full version hanya akan diberikan jika customer telah membayar setelah mencoba menggunakan demo version dan tertarik untuk membeli full version.

   Demo version dapat dibuat dengan berbagai keterbatasan, misalnya hanya dapat membaca jumlah record yang terbatas atau beberapa fitur penting ditiadakan, atau justru selalu menambahkan pesan yang mengganggu, seperti menampilkan footer pada data/laporan, atau menampilkan layar peringatan setiap waktu tertentu. Tetapi, intinya adalah mengusahakan agar pengguna terbantu membayangkan apa yang mereka dapatkan dari membeli full version. Tentunya tidak ada jaminan bahwa full version yang diberikan kemudian tidak dibajak. Tetapi, paling tidak pengguna dapat dengan leluasa melakukan publikasi demo version di Internet atau media massa untuk menarik minat calon pembeli.

5. Hard Code

   Alternatif lain adalah menggunakan hard code, artinya informasi atau format tertentu ditanamkan pada source code, sehingga tidak dapat diubah tanpa mengubah source code.

   Contohnya, menyimpan informasi nama perusahaan X pada source code, sehingga laporan-laporan yang dihasilkan memiliki format dan logo perusahaan X yang tidak dapat diganti-ganti.

   Perangkat lunak tersebut bisa saja diduplikasikan dan diinstal pada perusahaan lain, misalkan perusahaan Z, tetapi format laporan yang dihasilkan tetap menampilkan perusahaan X.

   Dengan demikian, perusahaan Z harus menghubungi vendor perangkat lunak yang bersangkutan untuk memesan perangkat lunak versi khusus untuk perusahaannya, di mana vendor perangkat lunak akan melakukan hard code ulang dengan nama perusahaan Z.

6. Obfuscated Code

   Tidak hanya file biner/executable file saja yang perlu diproteksi, bisa jadi source code perangkat lunak tidak cukup aman sehingga memerlukan proteksi.

   Hal ini mungkin ditemui pada pembuatan web, di mana source code tidak dicompile menjadi executable file, tetapi hanya diupload dan dijalankan dari sisi server, seperti PHP, ASP, Perl, dan lain-lain. Untuk memproteksi source code,

dikenal istilah obfuscated code, yang bisa diartikan kode yang membingungkan, karena memang source code tersebut sengaja dibuat sedemikian kompleks agar tidak mudah dimodifikasi.

Source kode tanpa indent/tab pada setiap barisnya (rata kiri semua) dan tanpa baris kosong, bisa dikategorikan sebagai obfuscated code yang sederhana. Dengan catatan, source code harus tetap dapat dijalankan, tentunya hal ini tidak dapat berlaku pada bahasa pemrograman atau script yang memiliki aturan penulisan tertentu, misalnya COBOL.

Pada bahasa pemrograman seperti C, C++, dan Perl, menghasilkan obfuscated code yang benar-benar memusingkan dan membuat frustasi bagi yang membacanya, adalah sangat dimungkinkan. Tentunya obfuscated code yang kompleks akan sulit dimodifi kasi dan di-debug bahkan oleh developer-nya sendiri.

Karena itu, developer yang menggunakan obfuscated code biasanya memiliki dua versi source code. Versi pertama adalah source code orisinal yang tentunya diusahakan semudah mungkin untuk dipahami. Source code kedua adalah source code yang telah dibuat obfuscated code (dapat dihasilkan dengan menggunakan tools obfuscator).

Permasalah utama yang umum dalam pembuatan perangkat lunak adalah bagaimana membatasi perangkat lunak agar hanya dapat digunakan oleh pihak tertentu.

Makalah ini membahas tentang salah satu cara yang dapat digunakan untuk mengatasi permasalahan pembajakan perangkat lunak yaitu membatasi perangkat lunak agar hanya dapat digunakan oleh pihak tertentu. Cara yang digunakan adalah menerapkan protokol tanda-tangan digital dengan enkripsi dalam pemberian lisensi kepada pengguna.

II. TANDA-TANGAN DIGITAL

A. Tanda-Tangan Digital

Sebagai sebuah alat pemeriksa keabsahan sebuah tanda tangan secara garis besar memiliki karakteristik sebagai berikut:

- Merupakan bukti yang otentik.
- Tidak dapat dilupakan.
- Tidak dapat dipindahtangankan.
- Tidak dapat disangkal.
- Dokumen yang telah ditandatangani tidak dapat berubah.

Fungsi tanda tangan pada dokumen kertas diterapkan untuk otentikasi pada data digital seperti pesan yang dikirim melalui saluran komunikasi dan dokumen elektronis yang disimpan di dalam memori komputer.

Tanda tangan pada data digital ini dinamakan tanda tangan digital (digital signature). Perlu ditekankan bahwa tanda tangan digital bukanlah tanda tangan manual (pada dokumen cetak) yang didigitasi dengan alat pemindai. Tanda tangan digital sebenarnya adalah suatu nilai kriptografis yang bergantung pada pesan dan pengirim pesan. Oleh karena itu tanda tangan digital seorang akan berbeda pada dua buah dokumen atau pesan yang berbeda.

Pembubuhan tanda tangan digital pada sebuah dokumen, umumnya menggunakan hash. Dokumen yang hendak dikirim diubah terlebih dahulu menjadi bentuk yang ringkas yang disebut message digest MD.

Selanjutnya, message digest MD dienkripsi dengan algoritma kriptografi kunci-publik menggunakan kunci rahasia (SK) milik penandatangan atau pengirim dokumen. Hasil dari enkripsi inilah yang kita sebut sebagai tanda tangan digital (digital signature) S.

Dokumen M dan tanda tangan digital S kemudian dikirim melalui saluran komunikasi. Dalam hal ini, dokumen M telah ditandatangani oleh pengirim dengan tanda tangan S. Sebuah tanda tangan digital bisa dilekatkan (embedded) pada sebuah dokumen atau bisa juga disimpan di dokumen yang terpisah.

Setelah sampai di penerima, dokumen diverifikasi untuk dibuktikan keabsahannya. Adapun caranya adalah dengan mendekripsi tanda tangan digital S dengan kunci publik PK milik pengirim menjadi message digest MD kembali. Jika message digest hasil dekripsi cocok berarti dokumen yang diterima valid atau terbukti keabsahannya.

Dalam penandatanganan secara digital ini ada dua standar yang umum digunakan ketiga standar tersebut adalah :

- RSA (Rivest-Shamir-Adleman)
- DSA (Digital Signature Algorithm)

B. Tanda-Tangan Digital Menggunakan RSA

RSA adalah salah satu algoritma kriptografi dengan kunci publik. Saat ini RSA bisa dikatakan sebagai algoritma kunci publik paling populer. Nama RSA sendiri diambil dari singkatan nama-nama penemu algoritmanya, yaitu Ron Rivest, Adi Shamir dan Leonard Adleman.

Dalam melakukan enkripsi dan dekripsi secara umum algoritma RSA memiliki besaran-besaran sebagai berikut:

- p dan q, bilangan prima rahasia
- $n = p \cdot q$, tidak rahasia
- $\phi(n) = (p-1) \cdot (q-1)$, rahasia
- SK, kunci privat rahasia
- PK, kunci publik tidak rahasia.
- M, pesan yang dienkripsi.

Kunci publik PK dalam RSA merupakan hasil pembangkitan bilangan secara acak dari pencarian bilangan yang relatif prima terhadap $\phi(n)$. Sedangkan kunci privat SK dibangkitkan dengan menggunakan persamaan

$$PK \cdot SK = 1 \left(mod\ \phi(n)\right)$$

Dalam enkripsi biasa pesan dienkripsi dengan kunci publik baru didekripsi dengan kunci privat. Namun pada

praktek tanda tangan digital digunakan sebaliknya.

Secara ringkas, pembubuhan tangan digital dengan RSA adalah sebagai berikut:

1. Pengirim menghitung nilai hash dari pesan M yang akan dikirim, misalkan nilai hash dari M adalah MD.
2. Pengirim mengenkripsi MD dengan kunci privatnya menggunakan persamaan enkripsi RSA.

Sedangkan untuk verifikasi tanda tangan digital :

1. Penerima menghitung nilai hash dari pesan M yang akan dikirim, misalkan nilai hash dari M adalah MD'.
2. Penerima melakukan dekripsi terhadap tanda-tangan S dengan kunci publik si pengirim.
3. Penerima membandingkan MD dengan MD'. Jika MD = MD' maka dokumen beserta tanda-tangan digitalnya adalah otentik. Jika sebaliknya, berarti dokumen atau pengirimnya dinyatakan tidak valid.

III. Protokol Tanda-Tangan Digital dengan Enkripsi

Protokol ini dapat dianalogikan seperti pengiriman surat yang menggunakan amplop terututup. Tanda tangan pada surat memberikan bukti kepemilikan, hal ini sama dengan fungsi tanda-tangan digital pada dokumen elektrinis. Sedangkan amplop memberikan perlindungan kemananan (privacy), hal ini sama dengan fungsi enkripsi pada dokumen.

Tanda-tangan digital diberikan dengan menggunakan kunci privat pengirim dan dokumen dienkripsi dengan kunci publik penerima.

Protokol untuk tanda-tangan digital dengan enkripsi adalah sebagai berikut:

(1) Alice menandatangani dokumen atau pesan (M) dengan menggunakan kunci privat (A).

$$S_A(M)$$

(2) Alice mengenkripsi dokumen yang sudah ditandatangani dengan kunci publik Bob (B) dan mengirimkannya kepada Bob.

$$E_B\big(S_A(M)\big)$$

(3) Bob mendekripsi cipherteks yang diterima dengan kunci privatnya.

$$D_B\left(E_B\big(S_A(M)\big)\right) = S_A(M)$$

(4) Bob melalukan verifikasi dengan mendekripsi hasil pada langkah 3 dengan menggunakan kunci publik Alice dan sekaligus mendapatkan kembali dokumen yang belum dienkripsi.

$$V_A\big(S_A(M)\big) = M$$

Alice tidak harus menggunakan kunci publik/privat yang sama untuk enkripsi dan tanda-tangan. Alice dapat menggunakan dua pasang kunci: sepasang untuk enkripsi dan sepasang untuk pemberian tanda-tangan.

IV. Lisensi Perangkat Lunak

Protokol tanda-tangan digital dengan enkripsi diadaptasi untuk penggunaan lisensi pada perangkat lunak. Algoritma tanda-tangan dan enkripsi/dekripsi menggunakan sistem algoritma kriptografi kunci-publik yaitu algoritma RSA. Ada dua alternatif protokol adaptasi ini, yaitu sebagai berikut:

a. Protokol Alternatif 1

- Pemberi lisensi bertindak sebagai Alice dan mempunyai sepasang kunci privat dan kunci publik.
- Penerima lisensi bertindak sebagai Bob dan mempunyai sepasang kunci privat dan kunci publik.
- Lisensi bertindak sebagai dokumen atau pesan (M) yang berisi informasi berupa minimal nama pemberi lisensi, kunci publik pemberi lisensi, nama penerima lisensi dan masa berlaku lisensi.
- Informasi nama diperoleh dari pihak penerima lisensi, sedangkan informasi masa berlaku lisensi ditentukan oleh pihak pemberi lisensi.
- Langkah-langkah:
  (1) Pemberi lisensi menandatangani lisensi dengan menggunakan kunci privat (A).
  $$S_A(M)$$
  (2) Pemberi lisensi mengenkripsi dokumen yang sudah ditandatangani dengan kunci publik penerima lisensi dan mengirimkannya kepada penerima lisensi.
  $$E_B\big(S_A(M)\big)$$
  (3) Perangkat lunak melakukan validasi terhadap lisensi penerima lisensi dengan cara melakukan dekripsi lisensi yang berupa cipherteks dengan menggunakan kunci privat penerima lisensi dan kemudian melakukan verifikasi terhadap tanda-tangan digital dengan kunci publik pemberi lisensi sekaligus mendapatkan informasi penerima dan masa berlaku lisensi.
  $$D_B\left(E_B\big(S_A(M)\big)\right) = S_A(M)$$
  $$V_A\big(S_A(M)\big) = M$$
- Pada protokol ini, perangkat lunak hanya bertindak sebagai pihak yang melakukan validasi terhadap lisensi dengan masukan kunci privat penerima lisensi dan kunci publik pemberi lisensi.

b. Protokol Alternatif 2

- Pemberi lisensi bertindak sebagai Alice dan mempunyai sepasang kunci privat dan kunci publik.
- Perangkat lunak bertindak sebagai Bob dan mempunyai sepasang kunci privat dan kunci publik.

- Lisensi bertindak sebagai dokumen atau pesan (M) yang berisi informasi berupa minimal nama pemberi lisensi, kunci publik pemberi lisensi, nama penerima lisensi dan masa berlaku lisensi.
- Informasi nama diperoleh dari pihak penerima lisensi, sedangkan informasi masa berlaku lisensi ditentukan oleh pihak pemberi lisensi.
- Langkah-langkah:
  (1) Pemberi lisensi menandatangani lisensi dengan menggunakan kunci privat (A).
  $$S_A(M)$$
  (2) Pemberi lisensi mengenkripsi dokumen yang sudah ditandatangani dengan kunci publik perangkat lunak dan mengirimkannya kepada penerima lisensi.
  $$E_B\left(S_A(M)\right)$$
  (3) Perangkat lunak melakukan validasi terhadap lisensi dengan cara melakukan dekripsi lisensi yang berupa cipherteks dengan menggunakan kunci privat perangkat lunak itu sendiri dan kemudian melakukan verifikasi terhadap tanda-tangan digital dengan kunci publik pemberi lisensi sekaligus mendapatkan informasi penerima dan masa berlaku lisensi.
  $$D_B\left(E_B\left(S_A(M)\right)\right) = S_A(M)$$
  $$V_A\left(S_A(M)\right) = M$$

- Pada protokol ini, perangkat lunak bertindak sebagai pihak yang melakukan validasi terhadap lisensi dengan masukan kunci privat perangkat lunak itu sendiri dan kunci publik pemberi lisensi.
- Pasangan kunci privat dan kunci publik untuk setiap salinan perangkat lunak boleh sama dan boleh berbeda satu dengan yang lain.
- Dalam hal ini, kunci publik perangkat lunak dapat dikatakan sebagai nomor serial unik perangkat lunak.
- Jika setiap salinan perangkat lunak memiliki pasangan kunci yang berbeda, maka sebelum pemberi lisensi mengeluarkan lisensi, penerima lisensi harus memberikan informasi nomor serial unik perangkat lunak yang dimilikinya kepada pembuat lisensi.

Untuk memperkuat kedua protokol ini, pada saat validasi lisensi perlu dilakukan pemeriksaan terhadap informasi penerima lisensi (misalnya nama penerima lisensi) dengan masukan dari pengguna perangkat lunak. Hal ini untuk mencegah pembajakan lisensi oleh pihak ketiga.

V. ANALISIS

1. Kelebihan kedua protokol alternatif ini, yaitu:
   - Aman terhadap perubahan isi informasi lisensi karena isi informasi lisensi dienkripsi.
   - Jika kunci privat penerima lisensi dibajak atau diketahui oleh pihak ketiga, informasi lisensi masih tetap aman dari perubahan selama kunci privat pihak pemberi lisensi juga aman. Perubahan terhadap informasi lisensi (misalnya kunci privat penerima lisensi diketahui) akan menyebabkan ketidaksesuaian tanda-tangan digital pada lisensi pada saat verifikasi tanda-tangan digital.
2. Kelebihan protokol alternatif 1, yaitu:
   - Aman dari manipulasi perangkat lunak untuk memperoleh informasi kunci privat penerima lisensi.
3. Kelebihan protokol alternatif 2, yaitu:
   - Penerima lisensi tidak perlu membuat dan menjaga pasangan kunci rahasia dan kunci publik. Cukup perangkat lunak yang memilikinya.
4. Kelemahan protokol alternatif 1, yaitu:
   - Informasi kunci privat penerima lisensi harus dapat dijaga dengan baik agar tidak diketahui oleh pihak ketiga. Hal ini tentu membutuhkan standar pengamanan lain yang lebih baik.
   - Panjang kunci privat yang terlalu besar akan merepotkan penerima lisensi untuk mengamankan kunci privatnya, dan dibutuhkan suatu media penyimpanan yang mungkin tidak aman.
5. Kelemahan protokol alternatif 2, yaitu:
   - Perangkat lunak dapat saja dimanipulasi oleh pembajak untuk memaksakan perangkat lunak agar dapat menerima lisensi tanpa perlu melakukan validasi ataupun memperoleh informasi kunci privat perangkat lunak.

.

VI. KESIMPULAN

1. Pembajakan perangkat lunak dapat diatasi dengan cara menerapkan konsep protokol tanda-tangan digital dengan enkripsi untuk pemberian lisensi dan validasi lisensi.
2. Distributor perlu memiliki informasi tentang komputer pengguna perangkat lunak untuk membuat lisensi. Informasi yang digunakan boleh unik dan boleh tidak.
3. Protokol adaptasi ini aman dari perubahan atau manipulasi informasi lisensi.

VII. SARAN

Untuk memperkuat validasi lisensi, perlu dilakukan pemeriksaan data informasi penerima lisensi pada lisensi dengan masukan dari pengguna perangkat lunak pada saat validasi lisensi.

Selain itu, perlu ada mekanisme untuk pembuatan/penentuan pasangan kunci privat dan publik bagi penerima lisensi, agar tidak terjadi pemalsuan maupun pembajakan informasi pasangan kunci dan lisensi.

REFERENCES

Munir, Rinaldi. 2005. *Diktat Kuliah IF5054 Kriptografi*. Departemen Teknik Informatika, Institut Teknologi Bandung.

http://if.web.id/informatics-knowledge-and-news/proteksi-software/ diakses pada Ahad, 16 Mei 2010.

http://en.wikipedia.org/wiki/Public-key_cryptography, diakses pada Ahad, 16 Mei 2010.

Rizaldy, Mohamad Ray. 2009. "Perbandingan Tanda Tangan Digital RSA dan DSA Serta Implementasinya untuk Antisipasi Pembajakan Perangkat Lunak". Departemen Teknik Informatika, Institut Teknologi Bandung.

Kusuma, Anugrah Redja. Yulianto, Rizky. Sudiyono, Widhiyo. 2004. "Proteksi Perangkat Lunak dengan Algoritma Kriptografi Kunci Publik". Departemen Teknik Informatika, Institut Teknologi Bandung.

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, 17 Mei 2010

ttd

Sibghatullah Mujaddid
13507124