Aplikasi *digital signature* untuk memverifikasi data user pada *HTML Forms*

Alfan Farizki Wicaksono – 135 06 067
Teknik Informatika, Institut Teknologi Bandung
Jalan Ganesha nomor 10, Kota Bandung, Jawa Barat, Indonesia
E-mail: farizki@comlabs.itb.ac.id, Website : http://students.itb.ac.id/~alfan_fw

ABSTRAK

Salah satu kelemahan yang dimiliki oleh HTML forms adalah tidak mempunyai suatu mekanisme secara langsung untuk memverifikasi identitas user yang melakukan pengiriman form. Oleh karena itu, hal ini menyebabkan HTML forms sama saja dengan sebuah form yang diisi oleh orang yang tidak diketahui dan dikirimkan lewat email. Oleh karena itu, kita butuh suatu mekanisme yang dapat memvalidasi user ketika sedang mengisi sebuah forms. Salah satu tekniknya adalah dengan menggunakan tanda tangan digital.

Prinsip tanda tangan pada dokumen kertas juga diterapkan untuk otentikasi pada data digital seperti pesan yang dikirimkan melalui saluran komunikasi. Tanda tangan digital bukanlah tanda tangan yang didigitisasi dengan alat scanner, tetapi suatu nilai kriptografis yang bergantung pada pesan dan pengirim pesan. Dengan tanda tangan digital, maka integritas data dapat dijamin, disamping itu ia juga digunakan untuk membuktikan asal pesan dan nirpenyangkalan.

Kata kunci: Tanda tangan digital, HTML, Forms.

1. PENDAHULUAN

forms merupakan salah satu fitur yang dimiliki oleh HTML sebagai interaksi input user ke server. Masalah muncul ketika HTML forms tidak mempunyai mekanisme secara langsung untuk mengautentikasi pengisi forms tersebut.

Jika sistem web yang dibuat menggunakan sistem username dan password, masalah ini sudah dapat ditangani dengan baik. Tetapi, bagaimana jika user baru pertama kali menggunakan sistem dan belum memiliki account untuk sistem tersebut. Masalah lain muncul ketika terdapat sebuah sistem yang menghendaki user mengisi HTML forms tanpa login terlebih dahulu. Dalam kasus bisnis penjualan barang secara online, hal ini menjadi hal yang sangat penting. Penjual yang berada di server tidak dapat memverifikasi identitas user yang memesan barang. Penjual mungkin hanya dapat memverifikasi alamat email yang diketahui dari identifikasi kartu kredit. Resiko dapat ditekan lebih jauh jika penjual juga dapat memverifikasi identitas user.

Mekanisme yang dilakukan oleh HTML forms pada umumnya adalah dengan cara menampung dahulu data yang ingin dikirim pada memori client. Setelah user menekan tombol submit barulah data yang dimasukkan oleh user tersebut dikirimkan ke server dengan 2 method. Method pertama adalah method GET yang melewatkan data bersama URL (Uniform Resources Locator). Method yang kedua adalah method POST yang melewatkan data melalui Body pada protokol HTTP. Kita perhatikan disini bahwa data baru dikirim ke server ketika user menekan tombol submit. Oleh karena itu, dalam makalah kali ini saya mencoba untuk menambahkan fitur tanda tangan digital setelah user menekan tombol submit dan sebelum data dikirimkan ke server.

Tanda tangan digital sendiri mempunyai beberapa cara implementasi antara lain tanda tangan digital dengan menggunakan enkripsi pesan menggunakan algoritma simettri atau kunci publik, tanda tangan digital dengan menggunakan fungsi hashing, dan tanda tangan digital yang menggabungkan kedua mekanisme sebelumnya.

Pada makalah saya kali ini, saya akan menggunakan metode tanda tangan digital yang ketiga yaitu yang menggunakan penggabungan antara enkripsi dan fungsi hash. Alasannya adalah keamanan dan tingkat overhead di jaringan tidak menjadi masalah. Keamanan dapat diatasi dengan adanya algoritma enkripsi dan overhead jaringan dapat diatasi karena tanda tangan yang dikirimkan ke server adalah hasil enkripsi dari message digest yang pendek bukan berasal dari hasil enkripsi data sesungguhnya.

Server juga mempunyai mekanisme untuk memverifikasi data yang dikirimkan melalui client. Server akan melakukan komputasi terhadap data yang dikirimkan kemudian hasil komputasi dengan algoritma tersebut dibandingkan dengan tanda tangan yang diterima. Jika hasilnya sama, user berhasil diverifikasi.

2. METODE

Bagian ini akan menjelaskan mengenai metode yang digunakan untuk membuat aplikasi tanda tangan digital ini, mulai dari struktur pesan yang akan dikirimakan ke server, teknik menandatangani input dari pengguna, dan teknik verifikasi dari sisi server.

2.1 Struktur Pesan

Jika kita perhatikan sebagian besar web yang menggunakan forms pada salah satu halamannya. Kita dapat melihat bahwa web tersebut terdiri dari jenis input yang beraneka ragam seperti inputbox, checkbox, combobox, selectionbox, tombol submit dan yang lainnya.

Setiap jenis input mempunyai data yang dikandung. Jika kita melakukan tanda tangan digital untuk masing – masing data pada masing – masing jenis input, hal ini tentu saja sangat tidak efisien. Karena proses komputasi untuk menghitung tanda tangan digital sangat kompleks dan cukup lama. Jadi salah satu tekniknya adalah dengan cara menggabungkan semua data yang ada pada masing – masing jenis input dengan cara konkatenasi string pesan.

Dengan teknik seperti ini, proses komputasi tanda tangan digital akan dilakukan sekali saja. Jadi, pada akhirnya struktur pesan yang akan dikirimkan adalah semua data pada jenis input yang dikonkat dengan hasil enkripsi dari message digestnya.

Berikut (gambar satu) adalah contoh dari sebuah strukur pesan yang dikirimkan ke server jika struktur form adalah seperti pada gambar dua.

```
[Alfan,13506067,Informatika]#
8949ef0189cf5f56fa148568b839b0
f5468a7b689ea65369f13341a1581b
94fb
442f2979189ec8f4772fe9bd550992
05cf4e30e112b452ea953c47d546c8df
```

Gambar 1. Struktur pesan yang dikirimkan

Gambar 2. Contoh forms

2.2 Teknik Menandatangani Masukan Pengguna

2.2.1 Peletakkan Fungsi Tanda Tangan Digital

Sekarang, masalah yang harus kita hadapi adalah bagaimana caranya kita meletakkan fungsi penghitung tanda tangan digital pada sebuah form. HTML form akan mengirimkan data – data yang dimasukan jika pengguna menggunakan tombol submit. Jika pengguna sudah menekan tombol submit, data yang dimasukan oleh pengguna tersebut akan dikirimkan ke server.

Oleh karena itu, kita akan mencoba untuk meletetakkan fungsi tanda tangan digital sebelum data dikirimkan ke server dan setelah pengguna menekan tombol submit. Salah satu event handler pada HTML yang sudah didefinisikan oleh W3C dapat kita gunakan untuk aplikasi ini. Event tersebut adalah event *onSubmit*, event ini bekerja jika pengguna menekan tombol submit. Jika ia berisi false, submit data tidak akan diteruskan ke server. Akan tetapi, jika ia berisi true submit data akan diteruskan ke server. Oleh karena itu, kita dapat menggunakan event ini dengan cara mengisinya dengan sebuah fungsi javascript yang menghitung tanda tangan digital. Berikut adalah contoh deklarasi dari tag forms.

```
<form action="verify.php" method="post"
onSubmit="return signForm(…);">
```

Kita lihat diatas bahwa pada event onSubmit telah diletakkan sebuah fungsi bernama *signForm()* yang bertugas untuk melakukan tanda tangan digital pada data yang akan dikirimkan. Fungsi tersebut akan mengembalikan false jika terjadi kegagalan pada proses komputasi atau pengguna tidak mau menandatanganinya. Jika ia mengembalikan false, sistem secara otomatis tidak akan melanjutkan pengiriman data ke server.

2.2.2 Metode Tanda Tangan Digital

Pada bagian pendahuluan sudah disebutkan bahwa saya akan menggunakan metode campuran antara metode enkripsi dan metode fungsi hash untuk melakukan tanda tangan digital. Adapun metode enkrispi yang digunakan adalah dengan menggunakan algoritma kunci publik RSA dan fungsi hash yang digunakan adalah fungsi MD5 (message digest 5).

Proses pemberian tandatangan adalah sebagai berikut. Fungsi pertama-tama akan menghitung message digest

dari data masukan user yang sudah diformat seperti pada bagian 2.1. Hasilnya adalah berupa hasil transformasi data masukan menjadi message digest 32 karakter (MD5).

M = MD5(data user) **(1)**

Selanjutnya, M (message digest) dienkripsi dengan menggnakan algoritma kriptograsi kunci publik RSA dengan menggunakan kunci privat dari pengirim. Jadi, pasangan kunci privat dan kunci publik akan dihasilkan otomatis oleh sistem dan ditampilkan di layar. Pengguna akan diminta untuk memasukan nilai kunci privat yang tampil di layar ke dalam form sebelum menekan tombol submit. Kunci publik tidak perlu dikirimkan ke server, karena sesungguhnya generator kunci publik dan kunci privat dilakukan dari sisi server yang akan meng-output kode HTML berisi kunci private dan publik pengguna. Jadi, data menguneni kunci publik sudah ada di server.

S = Esk (M) **(2)**

S adalah tanda tangan digital yang akan dilekatkan dengan ke data user yang akan dikirimkan ke server. Setelah itu data yang dikirimkan adalah data user asli dikonkatenasi dengan S. Jika proses ini berhasil fungsi yang melakukan proses ini akan mengembalikan nilai true dan jika ada kegagalan fungsi yang melakukan prosedur ini akan mengembalikan nilai false dan data tidak akan dikirimkan ke server.

2.3 Teknik Verifikasi dari Sisi Server

Di sisi server, tanda tangan diverifikasi dengan menggunakan kunci publik pengisi form. Tanda tangan S didekripsi dengan menggunakan kunci publik menghasilkan message digest semula.

M = Dpk(S) **(3)**

Server kemudian mengubah data masukan user yang dikirimkan menjadi message digest dengan menggunakan MD5.

M' = MD5(data user) **(4)**

Jika M = M', kita dapat pastikan bahwa data yang dikirimkan oleh pengguna adalah otentik berasal dari pengisi form yang benar.

Yang perlu kita perhatikan di sini adalah bahwa algoritma dan parameter mengenai RSA dan MD5 haruslah sama dengan yang digunakan pada sisi client. Maksudnya adalah metode yang digunakan pada javascript di client juga harus sama dengan bahasa pemrograman di sisi server.

3. IMPLEMENTASI

3.1 Pembuatan HTML Forms

kita dapat membuat HTML Forms untuk kebutuhan kita. Artinya, kita dapat membuat forms untuk pendaftaran, kuesioner, dan yang lainnya dan tidak terikat komponen apa saja yang dimasukan. Pengguna dapat dengan bebas menentukan komponen yang digunakan dalam sebuah forms, apakah itu komponen inputbox, combobox, textbox, atau yang lainnya.

Tetapi, untuk argumen yang diletakan pada tag form, kita harus menggunakan event handler onSubmit sebagai event yang menangani persetujuan user. Fungsi yang diletakkan pada event handler onSubmit kita berinama fungsi *signForm()*.

Misalkan kita ingin membuat form yang berisikan masukan nama, NIM, dan jurusan seperti pada gambar nomor 2. Pertama-tama, kita harus ingat bahwa kode HTML akan di-generate dari sisi server sekaligus men-generate kunci privat (secara random) dan menyimpan kunci publik di sisi server. Jadi, kita akan meletakkan kode HTML forms bersamaan dengan bahasa pemrograman sisi server. Untuk kali ini, kita akan mencoba mengimplementasikannya dengan PHP.

```
<?php
Session_start();
$pri = gerPriRSA(); // asumsi method //sudah
ada
$_SESSION[pub]    =    genPubRSA();    //asumsi
//method sudah ada

Echo "
<form      action='ver.php'      method='post'
onSubmit='return signFom()'>
<input type="hidden" name="signature">

Nama:<input
type="input"name="data_input"><br/>

NIM:<input                          type="input"
name="data_input"><br/>

PrivateKey:<input  type="input"  name="private"
value=$pri><br/>

Jurusan :
        <select>
                <option>Informatika</option>
        </select><br/>

<input type="submit" value="Sign/Send">
</form>
";
?>
```

3.2 Pembuatan Fungsi Tandatangan Digital dengan Javascript

Secara umum, ada 2 fungsi utama yang harus kita implementasikan pada bagian client. Fungsi pertama adalah fungsi yang dapat melakukan ekstraksi pesan yang dikandung pada komponen form sebelum dikirim ke server. Pada kali ini kita akan memempergunakan DOM (document object model) untuk mengakses perkomponen pada form. Kedua fungsi diatas akan dibungkus oleh sebuah fungsi yang bernama *signForm()*.

Pertama kita akan mencoba untuk membuat fungsi yang bertugas untuk mengambil seluruh data yang ada pada komponen form. Disini, kita menggunakan atribut *element* pada komponen window form. Atribut element merupakan array yang berisi tiap komponen pada form.

```
Function getData()
{
Var sign="";
var formSize = theForm.elements.length;
for(var i = 0; i < formSize; i++)
{
  elem = theForm.elements[i];
  switch (elem.type)
  {
  case "hidden":
  case "button":
  case "submit":
  case "reset":
  case "image":
  case "radio":
  case "checkbox":
  case "password":
  case "select-one":
  case "file": break;
  case "select-multiple":
  for(var op = 0; op < elem.length; op++)
  {
    if(elem.options[op].selected)
    {
      sign += sign + elem.value;
    }
  }
  default: //input text
    sign += sign + elem.value;
  }
}
Return sign;
}
```

Kita perhatikan pada contoh kode javascript diatas. Kode diatas menggunakan DOM untuk mengakses komponen – komponen yang ada pada forms. Tetapi semua data komponen tersebut masih dibungkus dalam satu objek, jadi kita harus melakukan ekstraksi terhadap object itu untuk mendapatkan elemennya satu-persatu. Hal yang kita lakukan adalah melakukan perulangan sekaligus menjebak identitas tiap komponen. Kode diatas hanya melakukan penanganan terhadap komponen inputbox dan komponen combobox. Seluruh data dari tiap komponen yang diperoleh kita lakukan konkatenasi string. String hasil konkatenasi barulah dikembalikan dilakukan proses enkripsi dan hashing.

Berikut adalah pseudocode untuk fungsi *signForm()* yang sudah menggabungkan fungsi *getData()* dan juga fungsi enkripsi dan hashing.

```
Function signForm(theForm, theWindow)
{
//tempel hasi enrkipsi di variabel hidden form
Var text = getData();
Var        enc        =        encryptRSA(text,
theForm.private.value);

theForm.signature.value = enc;

//tambahkan kode Anda disini
}
```

3.3 Pembuatan Fungsi Validasi di Sisi Server

Server melakukan validasi dengan cara mengambil terlebih dahulu isi variabel dari masing-masing komponen yang dikiri. Server juga mengambil data tanda tangan digital yang dikirimkan bersama dengan atribut hidden form. Setelah itu, data tandatangan digital yang diperoleh didekripsi menggunakan kunci publik RSA. Lalu, data asli yang dikirimkan diperlakukan seperti sebelumnya yaitu dilakukan konkatenasi barulah dimasukkan ke dalam fungsi hashing.

```
<?php
//file : ver.php

Session_start();
$pub = $_SESSION[pub];

//ambil data input
$data1 = $_POST[nama];
$data2 = $_POST[NIM];
$data3 = $_POST[jurusan];
$total = $data1.$data2.$data3;

//ambil tanda tangan digital
$ttd = $_POST[signature];

//cek validasi
$total_h = md5($total);
$de_ttd = decryptRSA($ttd, $pub);

if ($total_h == $de_ttd)
{
   echo "validasi berhasil";

   // kode selanjutnya setelah sukses
   // ...
   // ...
   // ...
   // ...
   // ...
   // ...
}
```

```
Else
{
   Echo “validasi gagal”;
}
?>
```

Kita dapat lihat bahwa kode diatas merupakan implementasi dari prinsip yang dijelaskan pada bagian 2.3. Jika server mendeteksi adanya kesalahan pada data yang dikirim atau tidak otentik, server akan mengirimkan pesan kesalahan kembali ke user.

4. UJI PERFORMANSI

Dengan adanya penambahan fitur tanda tangan digital ketika kita mengirimkan data melalui HTML forms, tentunya performansi pengiriman data akan menjadi lebih lama. Hal yang membuat lama suatu proses disini dibandingkan sebelum adanya penambahan fitur adalah adanya enkripsi dan dekripsi RSA yang cukup memakan waktu lama.

Katakanlah waktu proses yang dilakukan sebelum adanya penambahan fitur adalah **Ta**. Delay jaringan mempunyai kontribusi paling besar pada nilai Ta. Kemudian, waktu proses yang dilakukan setelah penambahan fitur adalah **Tp**. Kita definisikan **DT** adalah selisih antara **Tp** dan **Ta**.

Saya asumsikan yang mempunyai andil lebih besar terhadap DT adalah enkripsi RSA, dekripsi RSA, dan dua kali hashing MD5. jadi,

DT = Te + Td + 2Th **(5)**

Dimana Te adalah waktu yang dibutuhkan untuk mengenkripsi data, Td adalah waktu yang dibutuhkan untuk melakukan dekripsi data dan Th adalah waktu yang dibutuhkan untuk melakukan hashing MD5.

Saya mencoba untuk melakukan percobaan sebanyak 5 kali terhadap dengan data yang berbeda-beda dan menghitung waktu **DT**. kemudian, dari 5 percobaan itu saya akan menyimpulkan rata-rata perbedaan waktu antara forms yang menggunakan aplikasi tanda tangan digital dan forms yang tidak menggunakan aplikasi tanda tangan digital.

Adapun metode penghitungan waktu yang saya gunakan adalah dengan menggunakan **selisih timestamp** antara state sebelum dilakukan sebuah proses dan state setelah melakukan proses tersebut. Proses disini adalah proses enkripsi RSA, dekripsi RSA, dan hashing MD5. kemudian dilakukan penjumlahan dari ketiga nilai tersebut sesuai persamaan nomor 5.

No	Enkripsi	Dekripsi	MD5	Total (ms)
1	60ms	1127ms	0.22ms	1187.44
2	59ms	1127ms	0.09ms	1186.18
3	60ms	1127ms	0.08ms	1187.16
4	60ms	1126ms	0.07ms	1186.14
5	60ms	1126ms	0.06ms	1186.12

Tabel 1. Hasil Uji Performansi

Kita dapat lihat dari tabel diatas bahwa hasil uji menunjukan bahwa rata-rata pemrosesan forms yang menggunakan aplikasi tanda tangan digital akan **lebih lama 1,1 detik** dari pemrosesan dengan forms tanpa aplikasi tanda tangan digital.

5. KESIMPULAN

Secara umum tujuan dari tandatangan digital pada HTML form ini ada dua. Pertama adalah untuk mengautentikasi user pengirim data dan yang kedua adalah sebagai tanda setuju terhadap data yang dikirimkan. Di beberapa negara maju tandatangan digital mempunyai nilai legal yang sama dengan tandatangan di atas kertas.

Waktu yang dibutuhkan dalam pemrosesan forms pasti akan lebih lama. Oleh karena itu, aplikasi tanda tangan digital ini sebaiknya digunakan sesuai dengan kebutuhan. Jika kita mementingkan tingkat otentikasi dibandingkan dengan waktu response, kita diutamakan menggunakan aplikasi ini. Tetapi jika kita lebih mementingkan waktu response dibandingkan dengan tingkat otentikasi pesan, kita sebaiknya tidak perlu menggunakan aplikasi ini.

REFERENSI

[1] Munir, Rinaldi. *Diktat Kuliah Kriptografi*. Bandung 2006.

[2] W3C. *HTML 4.01 Specification* http://www.w3.org/TR/html4/

[3] W3C. Document Object Model http://www.w3schools.com/htmldom/dom_reference.asp