Implementasi Algoritma DSA untuk Otentikasi dan Otorisasi POST *Request*

Ariel Ansa Razumardi 13517040[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]*13517040@std.stei.itb.ac.id*

***Abstract*—Pada zaman informasi ini, salah satu teknologi yang paling berkembang dengan cepat adalah teknologi *web* dengan salah satu perhatian utamanya pada aspek keamanan. Salah satu cara untuk meningkatkan keamanan pada sebuah aplikasi *web* yang terdiri dari *client* dan *web service* adalah dengan mengganti sistem otentikasi dan otorisasi dari menggunakan *password* menjadi menggunakan tanda tangan digital pada saat mengirim POST *request*. Penggunaan tanda tangan digital ini dapat meningkatkan keamanan dalam penggunaan *web*, tetapi memiliki sisi negatif data menyusahkan pengguna jika dibandingkan dengan *password*.**

***Keywords*—POST *request*, Otentikasi, Otorisasi, *Digital Signature Algorithm*.**

I. PENDAHULUAN

Pada zaman informasi ini, salah satu teknologi yang paling berkembang dengan sangat cepat adalah teknologi *web*. Segala macam proses bisnis sekarang menggunakan *web* sebagai perantaranya. Semakin hari semakin banyak perusahaan atau pihak yang mulai menggunakan teknologi *web* ini untuk melaksanakan kegiatannya. Topik yang penting mengenai teknologi *web* ini salah satunya adalah soal keamanan. Dengan banyaknya data orang-orang yang disimpan di web, maka sangat penting bagi pemilik aplikasi *web* untuk menjaga agar aplikasi yang dimilikinya seaman mungkin dari akses yang tidak diizinkan.

Cara yang populer untuk membuat aplikasi web saat ini adalah dengan memisahkan aplikasi menjadi *web client* dan *web service*. Implementasi dari logika proses bisnis dan masalah pengaksesan data akan ditaruh pada *web service*. *Web client* akan fokus kepada halaman yang dilihat oleh pengguna aplikasi. Pada kasus seperti itu, proses yang perlu dilakukan terkait dengan aspek keamanan adalah memastikan pengguna yang mengakses suatu *web service* sesuai dengan informasi yang dikirimkan bukan pengguna lain yang berpura-pura sebagai pengguna tersebut. Proses ini biasa disebut sebagai proses otentikasi. Selain itu perlu dibatasi juga pengguna mana saja yang dapat mengakses *service* tertentu. Misalnya pada suatu situs media sosial, pengguna biasa seharusnya tidak dapat mengubah data pengguna lain sedangkan admin dapat mengubah data semua pengguna. Proses ini biasa disebut sebagai proses otorisasi.

Tanda tangan digital merupakan sebuah metode yang dapat digunakan untuk otentikasi sebuah dokumen atau *file*. Tanda tangan digital ini dilakukan dengan menggunakan kombinasi sebuah fungsi *hash* dan algoritma kunci public. Dengan mengaplikasikan tanda tangan digital untuk proses otentikasi dan otorisasi pada *web service*, keamanan data yang dapat diakses pada *web service* tersebut dapat lebih terjaga.

Pada makalah ini, akan dibahas mengenai otentikasi dan otorisasi POST *request* pada sebuah web *service* dengan menggunakan algoritma DSA.

II. DASAR TEORI

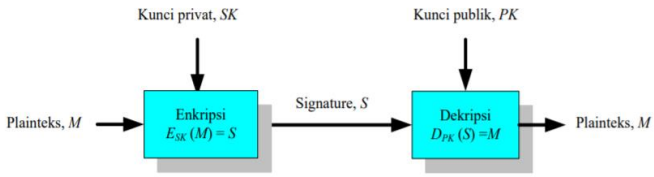
A. Tanda Tangan Digital

Pada kriptografi, terdapat 4 aspek keamananan, yaitu kerahasiaan pesan (*confidentiality*), otentikasi (*authentication*), keaslian pesan (*data integrity*), dan anti-penyangkalan (*non-repudiation*). Dari keempat aspek tersebut, tanda tangan digital dapat memenuhi aspek otentikasi, keaslian pesan, dan anti-penyangkalan. Tanda tangan digital merupakan tanda tangan untuk data yang bersifat digital dan merupakan hal yang berbeda dari tanda tangan yang didigitasi seperti tanda tangan pada kertas yang difoto.

Terdapat 2 cara untuk memberikan sebuah tanda tangan digital pada data, yaitu:

1. Mengenkripsi pesan
2. Menggunakan kombinasi fungsi hash dan kriptografi kunci publik.

Dengan cara pertama, enkripsi bisa menggunakan enkripsi simetris atau kriptografi kunci publik. Dengan cara ini kerahasiaan pesan akan terjaga. Jika menggunakan enkripsi simetris, maka diperlukan seorang pihak ketiga terpercaya yang disebut penengah agar dapat menjaga aspek anti-penyangkalan. Enkripsi juga dapat dilakukan dengan kriptografi kunci publik. Pesan akan di enkripsi dengan kunci privat pengirim. Jika dengan kunci publik pengirim pesan tersebut dapat didekripsi, maka pengirim akan terotentikasi sebagai pengirim dari pesan tersebut.



Gambar 1. Ilustrasi tanda tangan digital dengan enkripsi kunci publik
(http://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2020-2021/Tanda-tangan-digital-2020.pdf)

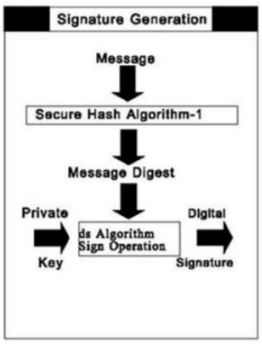
Pada beberapa kasus, tidak diperlukan penjagaan kerahasiaan dari pesan yang ditandatangani, tetapi hanya perlu aspek otentikasinya. Pada kasus seperti itu, bisa digunakan gabungan dari algoritma kunci publik dan fungsi hash.

B. Digital Signature Algorithm

Digital Signature Algorithm (DSA) adalah sebuah algoritma pengembangan dari algoritma ElGamal yang dapat digunakan untuk proses tanda tangan digital. Tidak seperti algoritma lainnya seperti RSA atau ElGamal, DSA tidak dapat digunakan untuk enkripsi, melainkan dispesifikasikan untuk tanda tangan digital.

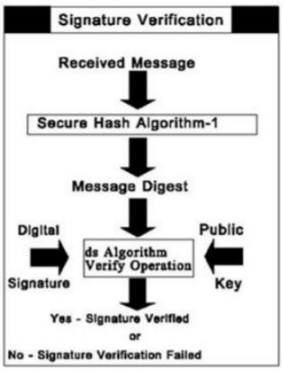
DSA merupakan komponen dari *Digital Signature Standard* (DSS). DSS merupakan sebuah *standard* untuk tanda tangan digital dengan menggunakan DSA sebagai algoritmanya dan SHA-1 sebagai fungsi hashnya. Walaupun seperti itu, fungsi hash pada DSA dapat diganti dengan fungsi hash lainnya. Berikut adalah spesifikasi dari algoritma DSA:

1. Parameter DSA
   Algoritma DSA membutuhkan parameter-parameter sebagai berikut:
   a. p, sebuah bilangan prima dengan panjang antara 512 sampai 1024 bit dan merupakan kelipatan dari 64. Parameter ini bersifat publik.
   b. q, sebuah bilangan prima dengan panjang 160 bit dan merupakan faktor dari (p-1). Parameter ini juga bersifat publik.
   c. g, parameter ini bisa didapatkan dengan rumus sebagai berikut:
   $$g = h^{(p-1)/q} \bmod p, h < p - 1 \quad (1)$$
   h yang disebutkan pada rumus tersebut harus memenuhi kondisi berikut:
   $$h^{(p-1)/q} \bmod p > 1 \quad (2)$$
   Parameter ini juga bersifat publik. p, q, dan g biasa disebut sebagai parameter domain.
   d. x, parameter ini merupakan kunci privat dan merupakan bilangan bulat yang kurang dari 1.
   e. y, parameter ini merupakan kunci publik dan didapatkan melalui rumus berikut:
   $$y = g^x \bmod p \quad (3)$$
   f. m, pesan yang akan ditandatangani oleh algoritma DSA.
2. Pembangkitan Kunci
   Berikut ini adalah langkah-langkah pembangkitan kunci pada algoritma DSA:
   a. Pilih bilangan prima p dan q dengan q merupakan faktor dari (p-1).
   b. Hitung nilai g dengan rumus yang telah diberikan pada bagian Parameter DSA.
   c. Tentukan kunci privat x, dengan syarat x < q
   d. Hitung kunci publik y dengan rumus yang telah diberikan pada bagian Parameter DSA.
3. Proses Pemberian Tanda Tangan (*Signing*)
   Berikut ini adalah proses pemberian tanda tangan dengan algoritma DSA dengan suatu fungsi hash H. Tanda tangan digital berupa dua bilangan, yaitu r dan s.
   a. Hitung H(m) (*message digest*) dari pesan m dengan fungsi hash H.
   b. Tentukan sebuah bilangan acak k yang lebih kecil dari q.
   c. Hitung r dengan rumus sebagai berikut:
   $$r = (g^k \bmod p) \bmod q \quad (4)$$
   d. Hitung s dengan rumus sebagai berikut:
   $$s = (k^{-1}(H(m) + x \cdot r)) \bmod q \quad (5)$$



Gambar 2. Ilustrasi proses *signing* DSA dengan fungsi hash SHA-1
(http://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2020-2021/DSS-2020.pdf)

4. Proses Verifikasi Tanda Tangan (*Verification*)
   Berikut ini adalah langkah-langkah dari proses verifikasi tanda tangan pada algoritma DSA:
   a. Hitung H(m) (*message digest*) dari pesan m dengan fungsi hash H.
   b. Hitung w dengan rumus sebagai berikut:
   $$w = s^{-1} \bmod q \quad (6)$$
   c. Hitung u_1 dengan rumus sebagai berikut:
   $$u_1 = (H(m) \cdot w) \bmod q \quad (7)$$
   d. Hitung u_2 dengan rumus sebagai berikut:
   $$u_2 = (r \cdot w) \bmod q \quad (8)$$
   e. Hitung v dengan rumus sebagai berikut:
   $$v = ((g^{u_1} \cdot y^{u_2}) \bmod p) \bmod q) \quad (9)$$
   f. Jika v sama dengan r, maka tanda tangan digital terverifikasi secara sah.



Gambar 3. Ilustrasi proses *verification* DSA dengan fungsi hash SHA-1 (http://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2020-2021/DSS-2020.pdf)

C. HTTP Request

Sebuah aplikasi web biasa dibagi menjadi *client* dan *service*. Hal ini dilakukan agar *client* bisa fokus kepada interaksi dengan pengguna dan *service* dapat fokus kepada implementasi proses bisnis. Hal ini juga membuat agar *client* dan *service* dapat diimplementasikan dengan teknologi yang berbeda, seperti misalnya *client* diimplementasikan dengan *javascript* dan *service* diimplementasikan dengan *java*. Karena penggunaan teknologi yang bisa berbeda tersebut, *client* dan *service* membutuhkan cara agar dapat bisa tetap berkomunikasi tidak peduli teknologi apa yang digunakan.

Untuk mengatasi permasalahan tersebut, *client* dan *service* menggunakan HTTP (*Hypertext Transfer Protocol*). Dengan HTTP, *client* dapat mengirimkan sebuah *request* HTTP dan *service* dapat membalas dengan sebuah *response* HTTP. Dengan mengikuti *standard* HTTP ini, masalah komunikasi antara *client* dan *service* berhasil terselesaikan.Sebuah HTTP *request* terdiri dari baris yang menyatakan *request*-nya (*request line*), *header*, dan *body*. Pada *request line* dimasukkan sebuah penanda *method* yang digunakan pada *request* tersebut. Salah satu dari *method* tersebut adalah POST.

POST request biasa digunakan untuk memasukkan data dari *client* ke *service*. Contohnya adalah pada saat proses registrasi seorang pengguna. Parameter dari pengguna biasa ditaruh di dalam *body* dari *request* jika menggunakan POST. Ini berbeda dari *method* lain bernama GET yang biasa menaruh parameter requestnya pada *link request*-nya. Perbedaan lain dari GET dan POST adalah GET biasa digunakan untuk mengambil data dari *service* tanpa mengubah data yang berada pada *service* tersebut.

III. IMPLEMENTASI

A. Pembangkitan Kunci

Karena pengiriman *request* akan berasal dari pengguna *service* pada *web server*, maka pasangan kunci akan menjadi milik pengguna. Oleh karena itu, pembangkitan kunci akan dilakukan pada sisi pengguna. Setelah itu, parameter domain dan kunci public harus disimpan di *web server* agar dapat mengenali pengguna.

Oleh karena itu, pada saat registrasi pengguna, setelah dibangkitkan pasangan kunci, kunci publik dan parameter domain perlu dikirimkan ke *web server* bersama dengan data registrasinya. Lalu kunci public dan parameter domain akan disimpan di dalam *web server* dan direlasikan kepada pengguna tersebut untuk memungkinkan otentikasi dan otorisasi terhadap pengguna tersebut nantinya.

B. Tanda Tangan Digital pada POST Request

Seperti yang telah dijelaskan sebelumnya, Sebuah POST *request* biasnya menyimpan parameter requestnya di dalam *body*. Oleh karena itu, tanda tangan digital akan menggunakan *body* tersebut sebagai pesan yang akan ditandatanganinya. Setelah itu tanda tangan digital perlu dimasukkan ke dalam *request* juga agar *web service* dapat melakukan verifikasi terhadap tanda tangan tersebut. Lalu identitas yang dapat membedakan pengguna yang mengirim *request*, seperti id pengguna tersebut pada basis data, juga perlu dikirimkan ke *web service* agar *web service* mengetahui harus melakukan verifikasi tanda tangan tersebut dengan parameter domain dan kunci publik milik pengguna mana.

Berikut adalah contoh sebuah *body* POST *request* yang nantinya akan ditandatangani dengan algoritma DSA:

```
{
        "id": "2",
        "nama_panggilan": "ariel"
}
```

Contoh *body* tersebut digunakan untuk mengubah data nama panggilan dan umur dari seorang pengguna dengan nama ariel. Data id pada *body* tersebut digunakan sebagai pengenal pengguna yang mengirimkan *request* tersebut. Namun, dengan sistem yang saya jelaskan sebelumnya, *request* seperti ini tidak akan diterima oleh *web service* karena tidak memiliki tanda tangan digital padanya. Oleh karena itu, agar *request* dapat diterima oleh *web service* dan dijalankan, perlu dimasukkan tanda tangan digital ke dalamnya. Akan tetapi, sebelum dimasukkan tanda tangan digital ke dalamnya, ada suatu pemrosesan awal yang harus dilakukan terlebih dahulu. *Body* tersebut harus diubah menjadi seperti berikut:

```
{"id": "2", "nama_panggilan": "ariel"}
```

Alasan dari pemrosesan ini adalah bisa jadi terdapat perbedaan pemrosesan *body* menjadi *string* antara *client* dan *service*. Oleh karena itu, diperlukan format yang sama untuk *client* dan *service* saat mengubah *body* menjadi *string* untuk dijadikan pesan pada saat proses penandatanganan. Persamaan format yang dilakukan disini adalah dengan cara menghilanhkan semua *whitespace* seperti spasi yang berlebih, *newline*, dan tabulasi dari *body* saat diubah menjadi *string* yang akan ditandatangani. Setelah pemrosesan tersebut, barulah dapat dilakukan proses penambahan tanda tangan pada *body* POST *request*. Tanda tangan digital akan dihasilkan dengan menggunakan algoritma DSA dengan fungsi hash SHA-256. Berikut ini adalah parameter domain, kunci privat yang

digunakan, dan tanda tangan yang dihasilkan dari parameter dan kunci tersebut:

Tabel 1. Parameter Domain p

Parameter Domain p
25092090249929350779849287833748958068613015073302515286690061472803631755623584771589497898775469523385337425061297461139175357713220592819184126758736935376063844462752696739924401550878690970031452142705989766600459307709751095886798012465074287686603813335413950048395392807369806770348677712071134153714822861742342306044146195183862721406657854197441439920932791385149409072273015314397649471787446383620886918278975193209934386278429760482865813720445934774635122283912076246478409879816482363164147627355333664268885045792734799764253979822648760929170218066424940267682953921037065926022781652167949012627649

Tabel 2. Parameter Domain q

Parameter Domain q
17226399291152022739952146283132915465134070583045614268781843580863

Tabel 3. Parameter Domain g

Parameter Domain g
16453829383264082010520601164257876618574891472600547282630884418267633662229370942246407012676206938812501264639276406904403720466604952111560621750480448129633186825489877565108992697777088830904731462237416205080010777365384207392360378706693030297195023491667067969474559030620201177946743894853547710594130133221752595199285908679427584651861393384477025289929248362069832741909874750250557562262629645944159980756102064549249368567483018005231155897257043471632986468760829787790137753232022390221763635945898670746588888079006608500408043924456003562018505189258086393296227423338648086280363422795795191082945

Tabel 4. Kunci privat dalam format pem

Kunci Privat
-----BEGIN PRIVATE KEY----- MIICXQIBADCCAjYGByqGSM44BAEwggIpAoIBAQD GxIQ1MXL5IMOJN5hdTra2zwuw 1vgpScToelE2W2QyFuZZZXTmhP4xG8wX/wPQP+IwW2 hrj4EqCNeSfP1NYFMDLIqB /SzFlkyWFjEJ74mECniQXgH3bLjhlTap79v6mfQuc4GP0 K57KXh0iUTwlzs5jTsO QbQh8qAd/aizSvoznJD9E5S5CO++ZtTnjCCWdWZ040nF y1XkidRRqYTfznegXd1R np/3hqzz7UoxGzpxBcRptRibiD6PwSjwD+n6Etv9ZSf6Hkd xRGVshOwnuLlKWMvo yo9RCcoe2hloz0BpDL815j2so3wslhDciCkJnfCi9+IzFkk6 Vw9H5MqqnvDBAh0A o5MOmN4BdKNF1AT+5JUdq7OZSdPwG409+1oHvwKC AQEAglblO3JsxhN4X/gvTNXM K069jA0nzfh/NV3se463pYr0GUTouktdg05atUYmiX3HFir aKzRvhFrL8psVYckC YtD9sEQgKzeImRBMbfVwNkaBqIAbPeJC+ctZdps/oS33z iZMZ4fiCJUQOlPS2o44 j/kN8kmC5CiBYBs5ROd6pG3f9IxLDXtsfm44Af/mIHs5Q gHxNvmiKihzBam/0LgU camJrben/wtQ4luPyxJNGgoT3u5nfZbkyphj41u51Eswf0iK 37MaqIobqISW+tBj lV5P3t9BhkUCbfOmgJqmuc0N3ozundwZG8ZWOHkN8rt 8CGnrnD8piV+m9bPE4atE xgQeAhxLaT/YfE5m2b0cO0Vxt95NRurfbx85jp7LGclk -----END PRIVATE KEY-----

Tabel 5. Tanda tangan digital yang dihasilkan

Tanda Tangan Digital yang Dihasilkan
WOJFsxFGlz/90FNk339VMtqr5brq0czaxGcRTIAGt41nN4apK4YAw0inEHaoyPiwe4qz5Pza0k0=

Berikut adalah *body* setelah ditambahkan dengan tanda tangan digital:

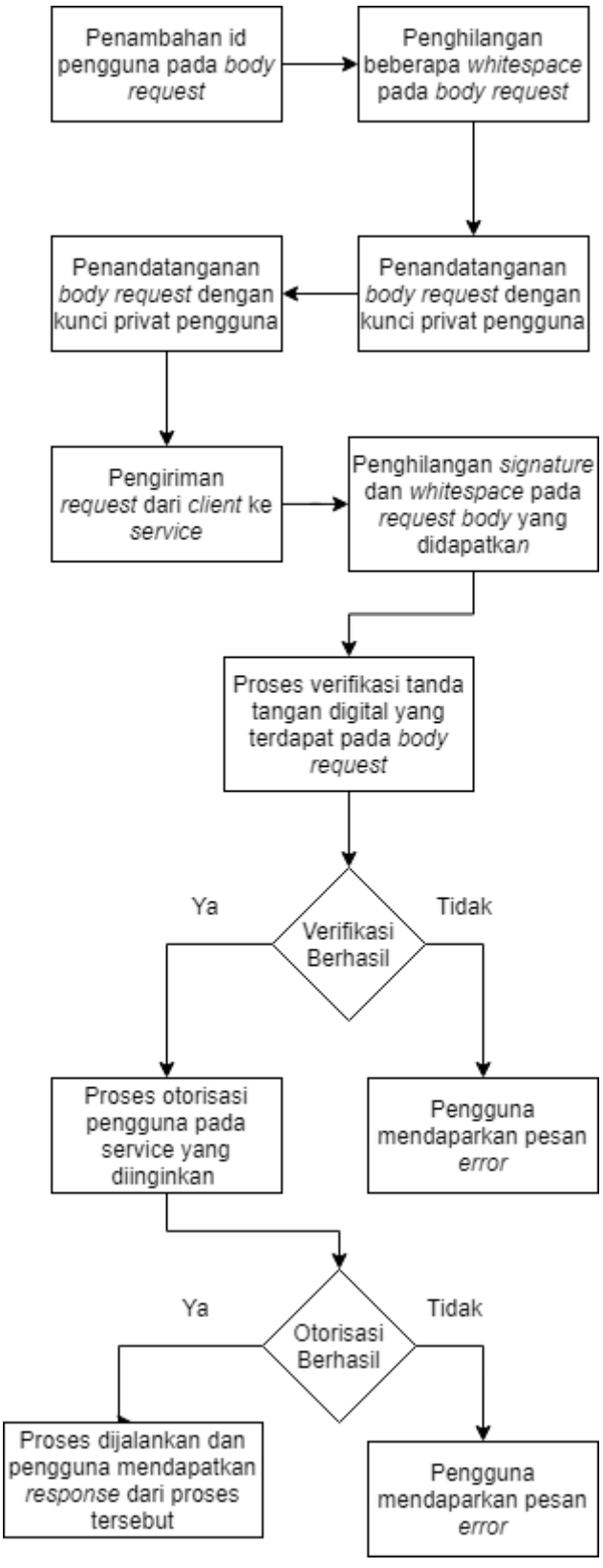
```
{
	"id": "2",
	"nama_panggilan": "ariel",
	"signature":
"WOJFsxFGlz/90FNk339VMtqr5brq0czaxGcRTIAGt41nN
4apK4YAw0inEHaoyPiwe4qz5Pza0k0="
}
```

Seperti yang terlihat di atas, tanda tangan digital telah ditambahkan pad *body* dengan nama *signature*. *Encoding* yang digunakan pada *signature* adalah base64 agar dapat dikirim ke *backend* tanpa ada masalah. Hal ini penting karena dapat muncul berbagai masalah apabila menggumakan *encoding* lain.

Untuk proses verifikasi pada *service*, akan dihilangkan terlebih dahulu kolom tanda tangan digital pada *body* baru dihilangkan *whitespace* agar menjadi sama seperti pesan yang ditandatangani pada *client*.

Apabila proses verifikasi tidak berhasil pada *service*, maka akan dikembalikan *response* berupa pesan *error* kepada *client*. Apabila proses verifikasi berhasil, maka proses otentikasi berhasil dan akan dilanjutkan ke tahap selanjutnya, yaitu proses otorisasi, yaitu melihat apakah pengguna tersebut berhak menjalankan *request* yang dimintanya atau tidak. Jika pengguna terotorisasi, maka proses akan dijalankan. Sebaliknya, jika pengguna tidak terotorisasi, maka proses tidak akan dijalankan dan pengguna akan mendapatkan pesan *error*.

Berikut ini bisa dilihat gambaran dari keseluruhan proses penandatanganan POST *request*, verifikasi tanda tangan pada POST *request*, dan otorisasi *web service* yang telah disebutkan sebelumnya:



Gambar 4. Proses otentikasi dan otorisasi pada POST *request*
(Data Pribadi)

IV. PENGUJIAN

Pada bagian ini, akan diimplementasikan sebuah *web service* sederhana dengan fitur mengubah data pribadi yang berada pada basis data *server*. Terdapat 2 orang pengguna biasa yang hanya dapat mengubah data mereka sendiri. Terdapat pula satu orang admin yang dapat mengubah data semua orang selain datanya sendiri. Berikut adalah id pengguna berikut nama panggilan yang ditetapkan mereka:

Tabel 6. Data pengguna

id	Nama Panggilan
0	admin
1	alice
2	bob

Setiap pengguna memiliki pasangan kunci masing-masing dengan kunci publik dan parameter domainnya disimpan pada basis data *server*. Kunci privat tidak disimpan di dalam *server* dan harus disimpan dan dijaga kerahasiaannya oleh masing-masing pengguna. Pasangan kunci yang digunakan tidak akan diberikan pada makalah ini karena terlalu panjang. Parameter Domain yang digunakan sama seperti parameter domain yang telah diberikan pada Tabel 1, Tabel 2, dan Tabel 3.

Akan dilakukan dua jenis pengujian, yaitu pengujian otentikasi dan pengujian otorisasi. Pengujian dilakukan ke sebuah *endpoint* yang telah dibuat untuk pengujian ini, yaitu localhost:3001/user/detail/<id> dengan id yang dimaksud adalah id pengguna yang nama panggilannya ingin diganti. *Request* yang akan dikirim ke *endpoint* tersebut memiliki body yang berisi id pengirim, nama panggilan baru, dan tanda tangan digital. *Response* yang akan diterima adalah pesan yang menyatakan keberhasilan dari *request* atau pesan *error* terkait.

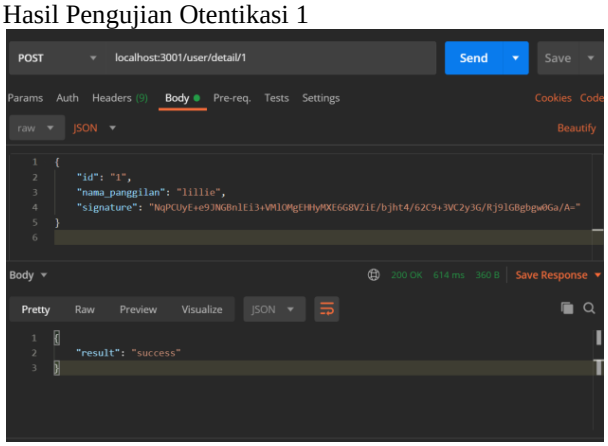
A. Pengujian Otentikasi

Pada pengujian ini, ada 3 kasus uji yang akan dicoba, yaitu:

1. Pengguna dengan id 1 akan mencoba mengubah nama panggilannya sendiri dengan tanda tangan digital yang valid.
2. Pengguna dengan id 1 akan mencoba mengubah nama panggilannya sendiri dengan tanda tangan digital yang tidak valid.
3. Pengguna dengan id 1 akan mencoba mengubah nama panggilannya sendiri tanpa tanda tangan digital.

Berikut adalah hasil yang didapatkan dari masing-masing kasus uji tersebut:

1. Hasil Pengujian Otentikasi 1

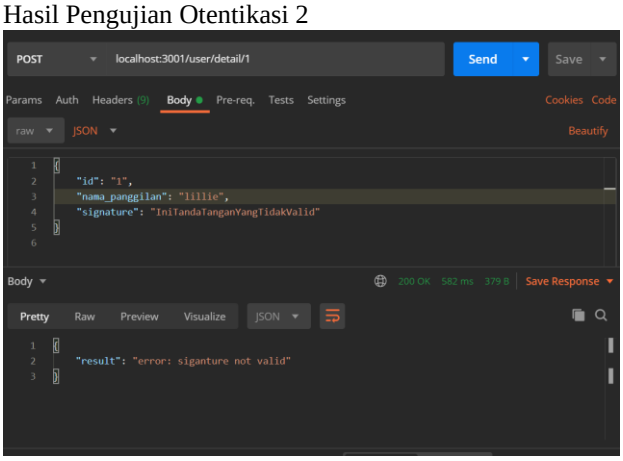


Gambar 5. Pengujian Otentikasi 1

(Data Pribadi)

Pada pengujian ini, karena tanda tangan digital sudah benar, maka hasilnya adalah nama panggilan pengguna dengan id 1 akan berhasil diganti menjadi lillie sesuai *body request* tersebut pada basis data.

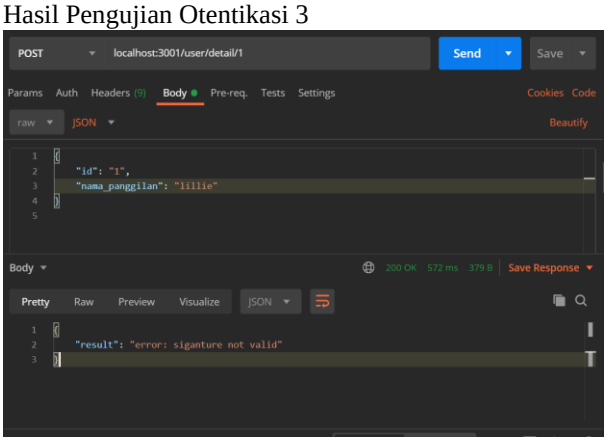
2. Hasil Pengujian Otentikasi 2



Gambar 6. Pengujian Otentikasi 2
(Data Pribadi)

Pada pengujian ini, dapat dilihat bahwa kalau tanda tangan digital yang diberikan tidak benar, akan muncul pesan *error*. Oleh karena itu, diperlukan kunci privat dari seorang pengguna untuk dapat mengaku sebagai pengguna tersebut.

3. Hasil Pengujian Otentikasi 3



Gambar 7. Pengujian Otentikasi 3
(Data Pribadi)

Pada pengujian ini, dapat dilihat bahwa kalau tanda tangan digital tidak diberikan sama sekali, akan muncul pesan *error*. Seperti kasus sebelumnya, dapat disimpulkan bahwa diperlukan kunci privat dari seorang pengguna untuk dapat mengaku sebagai pengguna tersebut.

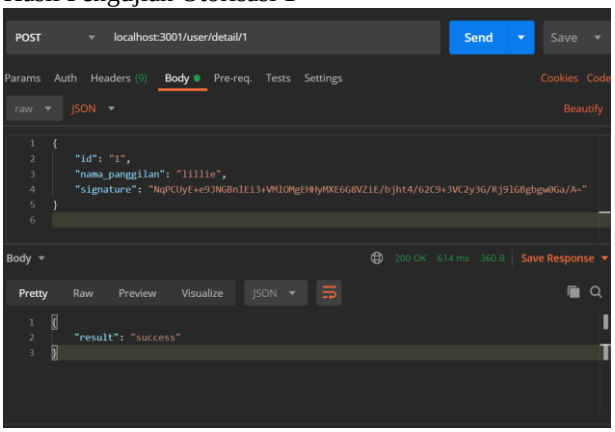
B. Pengujian Otorisasi

Pada pengujian ini, ada tiga kasus yang akan dicoba dengan tanda tangan digital yang valid, yaitu:

1. Pengguna dengan id 1 akan mencoba mengubah nama panggilannya sendiri.
2. Pengguna dengan id 1 akan mencoba mengubah nama panggilan pengguna dengan id 2.
3. Pengguna dengan id 0 atau admin akan mencoba mengubah nama panggilan pengguna dengan id 2.

Berikut adalah hasil yang didapatkan dari masing-masing kasus uji tersebut:

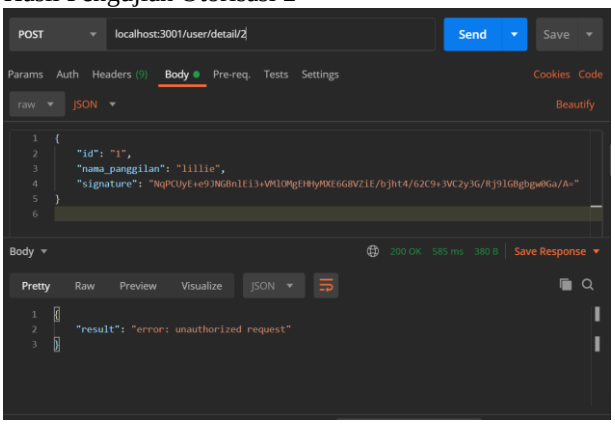
1. Hasil Pengujian Otorisasi 1



Gambar 8. Pengujian Otorisasi 1
(Data Pribadi)

Seperti pada pengujian otentikasi 1, pengguna dengan id 1 berhasil mengubah nama panggilannya sendiri karena pengguna biasa dapat diotorisasi untuk mengubah nama panggilannya masing-masing.

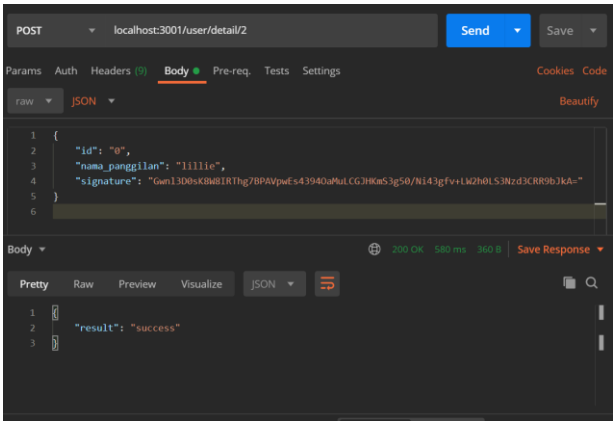
2. Hasil Pengujian Otorisasi 2



Gambar 9. Pengujian Otorisasi 2
(Data Pribadi)

Pada pengujian ini, dapat dilihat bahwa saat pengguna biasa mencoba mengubah nama panggilan dari pengguna lain, akan muncul pesan *error*. Hal ini terjadi karena pengguna biasa tidak dapat diotorisasi untuk mengubah nama panggilan dari pengguna lain.

3. Hasil Pengujian Otorisasi 3



Gambar 10. Pengujian Otorisasi 3
(Data Pribadi)

Pada pengujian ini, dapat dilihat bahwa saat pengguna yang merupakan admin mencoba mengubah nama panggilan dari pengguna lain, tidak muncul pesan *error*. Hal ini terjadi karena tidak seperti pengguna biasa, pengguna yang merupakan admin dapat diotorisasi untuk mengubah nama panggilan dari pengguna lainnya.

V. KESIMPULAN

Implementasi tanda tangan digital dalam pengiriman *request* ke sebuah *service* dapat membantu dalam aspek keamanan data yang terdapat pada aplikasi web. Selama pengguna menjaga kunci privat mereka dengan baik, akan sangat sulit bagi pihak ketiga dengan niat tidak baik untuk mengirimkan *request* atas nama pengguna tersebut. Hal ini berbeda dengan otentikasi menggunakan *password* yang sangat bergantung kepada kekuatan *password* yang dibuat oleh pengguna. Pada sistem usulan ini, pengguna tidak mempunyai pilihan untuk membuat pasangan kunci sendiri, melainkan akan diberikan oleh aplikasi web *client*.

Akan tetapi, sistem ini juga memiliki kelemahan, yaitu pengguna harus menyimpan kunci privat mereka dengan baik. Ini akan bermasalah saat pengguna memutuskan untuk mengganti *device* yang digunakannya dan terlupa untuk memindahkan *file* kunci privat yang dimilikinya. Hal ini berbeda dari sistem *password* di mana pengguna hanya perlu mengingatnya dengan baik.

Dengan itu dapat disimpulkan bahwa implementasi tanda tangan digital untuk otentikasi dan otorisasi dalam pengiriman sebuah POST *request* kepada sebuah *web service* dapat meningkatkan keamanan pengguna dari retasan (*hacking*) oleh pihak ketiga. Namun, sistem ini juga merepotkan pengguna karena mereka harus menyimpan kunci privat mereka dengan baik, jika dibandingkan dengan *password* yang hanya harus mereka ingat.

VII. UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih ke semua pihak yang telah membantu dalam kelancaran dan selesainya pembuatan makalah ini. Pertama-tama penulis berterima kasih kepada Tuhan Yang Maha Esa atas berkat rahmat-Nya penulis dapat membuat makalah yang berjudul "Implementasi Algoritma DSA untuk Otentikasi dan Otorisasi POST Request" ini. Lalu, penulis juga ingin berterima kasih kepada Dr. Ir. Rinaldi Munir, MT. selaku dosen pengajar mata kuliah Kriptografi penulis selama satu semester ini. Lalu, penulis juga berterima kasih kepada Orang Tua yang selalu mendukung kegiatan kuliah penulis. Terakhir, penulis juga berterima kasih kepada teman-teman seperjuangan di informatika yang selalu membantu penulis.

DAFTAR PUSTAKA

[1] Munir, Rinaldi. 2020. "Digital Signature Standard (DSS)". http://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2020-2021/DSS-2020.pdf diakses pada 20 Desember 2020.

[2] Munir, Rinaldi. 2020. "Tanda-tangan Digital". http://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2020-2021/Tanda-tangan-digital-2020.pdf diakses pada 20 Desember 2020.

[3] "HTTP - Requests". https://www.tutorialspoint.com/http/http_requests.htm diakses pada 20 Desember 2020.

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, 21 Desember 2020

Ariel Ansa Razumardi 13517040