Penerapan Tanda Tangan Digital berbasis Kurva Eleptikal untuk Autentikasi *Website*

Willsen Sentosa 13517036
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13517036@std.stei.itb.ac.id

***Abstract*—Tanda tangan digital digunakan untuk menjamin keabsahan dan integritas dari sebuah dokumen. *Digital Signature Algorithm* (DSA) merupakan sebuah algoritma kriptografi yang menggunakan kunci public untuk mengimplementasikan tanda tangan digital. *Elliptic Curve Digital Signature Algorithm* (ECDSA) adalah sebuah modifikasi dari DSA yang menggunakan perhitungan kurva eleptikal untuk membangkitkan tenda tangan digital. ECDSA yang sudah terbukti dapat menjamin keabsahan suatu dokumen akan digunakan untuk autentikasi keaslian *website*.**

***Keywords—Elliptic Curve Digital Signature Algorithm, Hash, Phising, Website*.**

I. Latar Belakang

Tidak dapat dipungkiri bahwa teknologi telah berkembang dengan pesat. Tiap aspek dalam kehidupan manusia telah dipengaruhi oleh teknologi, termasuk dalam bidang komunikasi. Pada zaman dahulu, manusia hanya dapat berkomunikasi secara tatap muka. Namun seiring berkembangnya teknologi, manusia pada belahan bumi lain dapat secara langsung berkomunikasi dengan mudah.

Sejak Sir Timothy John "Tim" Berners-Lee menemukan *World Wide Web*, manusia dapat berkomunikasi dan menukar informasi dengan mudah melalu internet. Karena penemuan ini juga, internet dan *World Wide Web* berkembang dengan pesat. Saat ini, *World Wide Web* bukan hanya digunakan untuk pertukaran informasi, namun *World Wide Web* dapat digunakan untuk pemesanan barang, manajemen finansial, pembayaran, hiburan, media sosial, dan lain-lain.

Dikarenakan *World Wide Web* ini banyak digunakan oleh publik dan terus berkembang dengan pesat, maka ada peluang tindak criminal dapat dilakukan melalui *World Wide Web*, salah satunya dengan metode *Phising*.

Phising adalah sebuah tindak kejahatan yang berupaya untuk mendapatkan informasi pribadi seseorang dengan cara pengelabuan. *Phising* biasanya mengincar data-data seperti data pribadi (nama, usia, nomor telepon, nomor Kartu Tanda Penduduk), data finansial (nomor rekening, nomor kartu kredit), dan data akun (username, alamat email, password). Salah satu cara *phising* ini dilakukan adalah dengan membuat halaman *website* palsu yang sangat mirip dengan aslinya sehingga pengguna tersebut memasukkan data pribadi, data akun, dan data finansial mereka tanpa menyadari bahwa data tersebut akan jatuh ke tangan orang yang tidak bertanggung jawab. Jika orang tersebut sudah memiliki data tersebut, maka mungkin saja orang tersebut menggunakan data tersebut untuk penipuan, atau bahkan pengambilan uang dalam rekening bank.

Karena maraknya *website phising* yang beredar, maka diperlukan sebuah solusi untuk autentikasi keaslian *website*. Dengan menggunakan tanda tangan digital, maka keaslian dan keabsahan suatu *website* dapat dibuktikan sehingga dapat mengurangi penipuan berbasis *website phising*.

II. Dasar Teori

A. Tanda Tangan Digital

Tanda tangan digital adalah sebuah *string* yang merepresentasikan keabsahan sebuah dokumen. *String* ini didapatkan dengan sebuah skema matematis. Dengan skema ini, maka dapat dijamin bahwa data dan informasi yang ditampilkan adalah benar-benar berasal dari sumber asalnya.

Tanda tangan digital dibuat dengan sebuah algoritma, yaitu *Digital Signature Algorithm* (DSA) yang merupakan sebuah algoritma kunci publik. Untuk membuat tanda tangan digital, maka diperlukan sebuah *Message Digest* dari pesan yang akan dikirim. *Message Digest* sendiri didapatkan dengan mengubah pesan yang akan dikirim dengan fungsi *hash* satu arah. Setelah didapatkan *Message Digest*, *Message Digest* akan dienkripsi dengan sebuah algoritma kriptografi kunci privat, misalnya menggunakan algoritma *Rivest-Shamir*-Andleman (RSA). Hasil enkripsi dari tanda tangan digital tersebut akan disisipkan pada pesan yang akan dikirim. Kemudian pesan dapat dikirim sehingga pesan tersebut sudah memiliki tanda tangan digital.

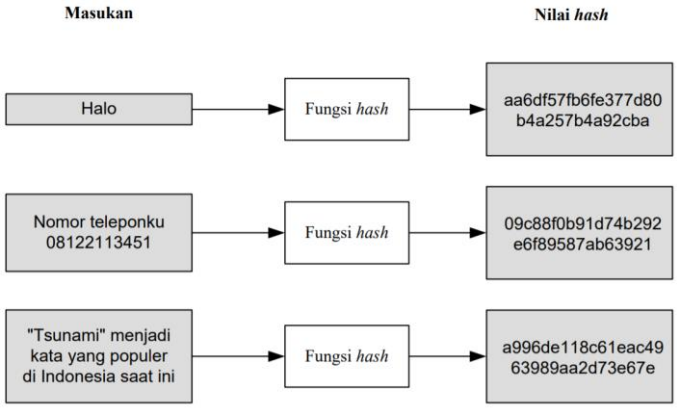
Untuk melakukan verifikasi keaslian pesan yang telah diterima menggunakan tanda tangan digital, maka kebalikan dari proses pembangkitan tanda tangan digital akan dilakukan, yaitu dengan mendekripsikan tanda tangan digital dengan kunci publik yang sudah diberikan. Hasil dekripsi ini akan menghasilkan sebuah *Message Digest*. Lalu penerima pesan ini akan mengubah pesan yang telah diterima menjadi *Message Digest* lain menggunakan fungsi *hash* yang sama. Jika *Message Digest* yang dihasilkan oleh tanda tangan digital sama dengan *Message Digest* yang dihasilkan dari pesan menggunakan fungsi *hash*, maka pesan tersebut dapat dijamin autentikasinya. Namun jika *Message Digest* berbeda, maka pesan yang dikirim sudah dilakukan modifikasi, baik modifikasi pesan, maupun modifikasi dari tanda tangan digital.

B. Fungsi *Hash*

Fungsi *hash* adalah sebuah fungsi yang digunakan untuk mengkompresi sebuah data berukuran sembarang menjadi sebuah *string* yang berukuran tetap[1]. *String* tersebut dinamakan sebagai *Message Digest* atau *Hash Value. Message Digest* yang dihasilkan dapat digunakan untuk merepresentasikan data yang sudah dikompresi tersebut. Namun, *Message Digest* yang sudah dihasilkan tidak dapat diubah kembali menjadi data sebelumnya sehingga membuat fungsi *hash* adalah sebuah fungsi yang *irreversible*

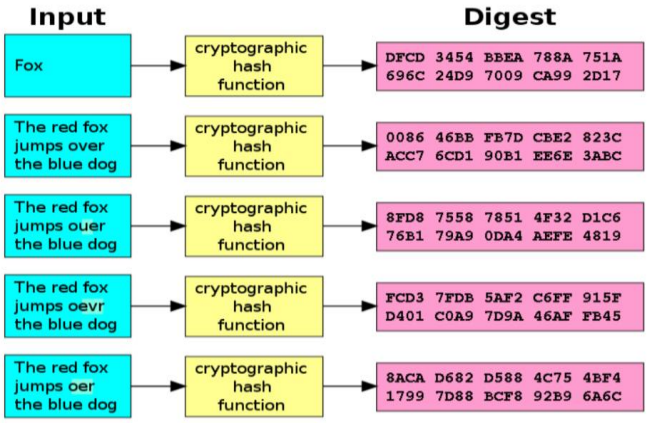
Fungsi *hash* memiliki 3 jenis sifat, yaitu *collision resistance*, *preimage resistance*, dan *second preimage resistance*. *Collision resistance* pada fungsi *hash* berarti bahwa sangat sukar untuk menemukan 2 buah input *a* dan *b* sehingga hasil dari *hash a* sama dengan hasil dari *hash b*. Sedangkan *preimage resistance* menyatakn bahwa untuk sembarang output *y*, sulit untuk menemukan input *a* sehingga *hash a* adalah *y*. Sifat terakhir fungsi *hash*, yaitu *second preimage resistance*, menyatakan bahwa untuk input *a* dan output *y = hash(a)*, sulit untuk menemukan input *b* dimana *hash(b) = y*.

Fungsi *hash* tidak dapat sebagai enkripsi, walaupun hasil nilai *hash* tidak memiliki makna, dikarenakan fungsi *hash* adalah *irreversible* dan fungsi *hash* tidak menggunakan kunci. Contoh dari hasil fungsi *hash* dapat dilihat pada gambar dibawah.



Gambar 1. Hasil Fungsi *Hash*

Hasil fungsi *hash* juga cukup aman dikarenakan jika sebuah bit atau huruf diganti pada *message*, maka hasil fungsi *hash* juga akan berubah secara sepenuhnya. Hasil fungsi *hash* tersebut dapat dilihat pada gambar dibawah.



Gambar 2. Hasil *Hash* pada perubahan sedikit huruf

C. Kriptografi Kunci Publik

Kriptografi kunci publik, atau disebut juga kriptografi kunci-nirsimetri, adalah sebuah sistem kriptografi yang menggunakan sebuah pasangan kunci, yaitu kunci publik yang dapat disebarluaskan, dan kunci privat yang hanya diketahui oelh pemiliknya saja. Kunci publik akan dihasilkan dari kunci privat menggunakan fungsi satu arah sehingga kerahasiaan kunci privat dapat dijaga.

Dengan sistem kriptografi kunci publik ini, maka siapapun dapat mengenkripsi pesan dengan menggunakan kunci publik penerima dan mengirimkan pesan tersebut pada saluran yang tidak aman. Hanya penerima yang mempunyai kunci privat yang digunakan untuk menghasilkan kunci publik tersebut yang dapat mendekripsi pesan tersebut.

D. *Elliptic Curve Digital Signature Algorithm*

Elliptic Curve Digital Signature Algorithm (ECDSA) adalah sebuah modifikasi dari *Digital Signature Algorithm* (DSA) dengan cara menggunakan *Elltiptic Curve Criptography*. Sama seperti DSA, ECDSA menggunakan pasangan kunci public dan kunci privat untuk membuat sebuah tanda tangan digital. ECDSA merupakan perkembangan dari DSA dikarenakan keamanan ECDSA lebih ketat dibandingkan DSA, yaitu pada kesuliatan dalam pemecahan persamaan kurva eliptik

ECDSA dibagi menjadi 2 proses, yaitu proses pembuatan tanda tangan, dan proses verifikasi tanda tangan.

1) Tahapan Pembuatan Tanda Tangan

Sebelum terjadinya pertukaran pesan antara kedua belah pihak (anggap saja Alice dan Bob), mereka harus pertama-tama menyetujui parameter kurva yang akan digunakan. Parameter tersebut dapat dilihat pada tabel dibawah ini.

Parameter	Deskripsi
CURVE	Persamaan kurva yang ditentukan
G	Titik dasar kurva eleptikal
n	*Integer order* dari G
d_A	Kunci Privat
Q_A	Kunci Publik
m	Pesan yang akan dikirim

Tabel 1. Parameter Kurva Eliptik

d_A didapatkan dengan membangkitkan sebuah bilangan prima besar, dan Q_A yang merupakan sebuah titik didapatkan dengan melakukan perkalian titik G dengan *dA*.

Tahapan penandatanganan yang harus dilakukan oleh Alice adalah sebagai berikut:

1. Membangkitkan sebuah bilangan bulat *k*
2. Dengan menggunakan perkalian titik, hitung titik $P = k * G$.
3. Jika $Px \bmod n = 0$, maka kembali ke langkah pertama
4. Jika $Px \neq 0$, maka $r = Px$.
5. Ubah pesan *m* menjadi *Message Digest* (MD) dengan menggunakan fungsi *hash*
6. Hitung $s = k^{-1}(MD + d_A \times r) \bmod n$, dengan k^{-1} adalah *modular multiplicative inverse* dari *k*
7. *Digital signature* adalah pasangan (r,s).

Setelah Alice mendapatkan tanda tangan digital, Alice akan menyisipkan tanda tangan digital tersebut kedalam pesan yang akan dikirimkan. Untuk memisahkan antara pesan dan tanda tangan digital, dapat digunakan pemisah atau penanda.

2) Tahapan Verifikasi Tanda Tangan

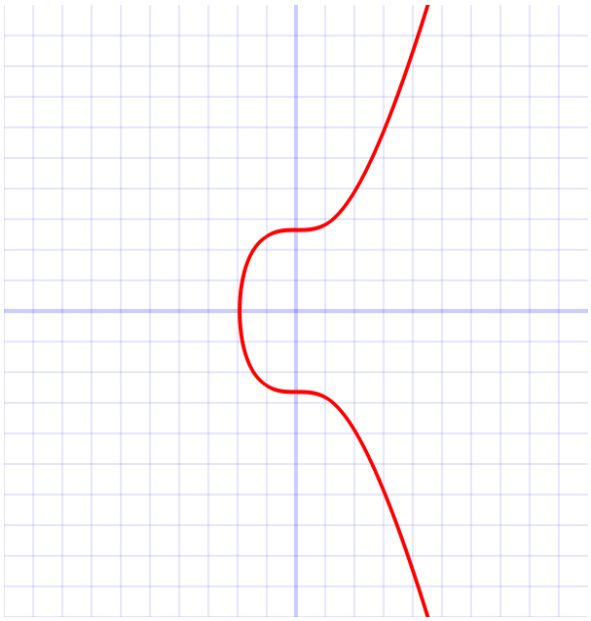
Untuk melakukan verifikasi tanda tangan, maka Bob sebagai penerima harus terlebih dahulu mendapatkan kunci public QA dari Alice. Setelah mendapatkan kunci public, maka tahapan yang akan dilakukan untuk menverifikasi adalah sebagai berikut:

1. Mengecek apakah *r* dan *s* adalah sebuah bilangan bulat diantara 1 sampai *n*-1. Jika tidak, maka tanda tangan digital tidak valid
2. Menghitung *Message Digest* (MD) dengan cara mengubah *message m* menggunakan fungsi *hash* yang digunakan oleh Alice.
3. Menghitung $P = s^{-1} \times MD \times G + s^{-1} \times r \times Q_A$
   Dimana s^{-1} adalah *modular multiplicative inverse* dari *s*
4. Jika $P_x = r$, maka pesan terbukti autentikasinya.

III. RANCANGAN SISTEM

Pada bagian ini, akan dibahas mengenai perancangan sistem yang diajukan untuk mengurangi maraknya *website phising*. Dengan menggunakan tanda tangan digital yang dibuat dengan metode ECDSA, maka pengguna *website* dapat mengecek keaslian *website* sebelum memasukkan data pribadi.

Seperti yang dibahas pada dasar teori, pengirim (pemilik *website*) dan penerima (pengguna *website*) harus menyetujui kurva eliptik beserta parameter-parameter yang akan digunakan. Penulis mengusulkan untuk menggunakan kurva secp256k1. Kurva ini digunakan juga sebagai parameter dari algoritma kriptografi kunci publik bitcoin, dan juga merupakan *Standard for Efficient Cryptography*[2]. Berikut adalah gambar kurva secp256k1.



Gambar 3. Visualisasi Kurva secp256k1

Kurva secp256k1 mengikuti persamaan kurva sebagai berikut:

$$y^2 = x^3 + 7$$

Dengan parameter kurva secp256k1 adalah sebagai berikut:

$p = 2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1$

$a = 0$

$b = 7$

$G\ (compressed) =$
02 79BE667E F9DCBBAC 55A06295 CE870B07
029BFCDB 2DCE28D9 59F2815B 16F81798

$n =$ FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE
BAAEDCE6 AF48A03B BFD25E8C D0364141

Setelah pengirim dan penerima menyepakati kurva yang akan digunakan, pengirim akan membangkitkan sebuah kunci privat dari bilangan acak, setelah itu pengirim akan membangkitkan kunci publik yang dihitung dari kurva dan kunci privat pengirim.

Kunci publik ini dapat digunakan oleh siapa saja dan disebarluaskan ke publik. Kunci publik ini dapat disimpan di sebuah *database* independent yang berisi semua kunci publik *website* yang asli. Pengguna dapat mengunduh kunci publik atau mengambil kunci public yang sudah disediakan oleh *database* tersebut sebelum memasukkan data.

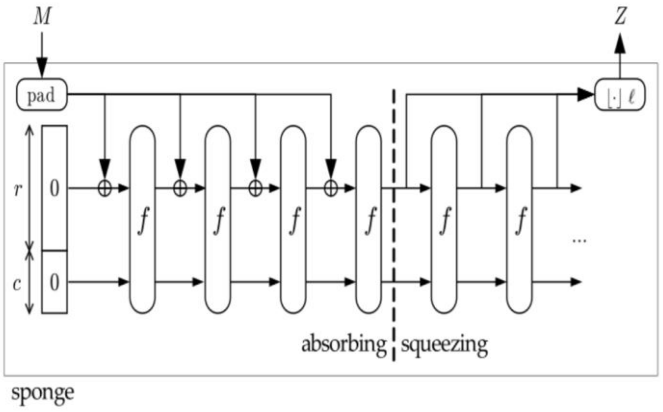
Setelah pemilik website dan pengguna website memiliki pasangan kunci publik. Maka setiap kali pengguna masuk ke dalam *website* yang autentik, maka server *website* akan mengirimkan sebuah *cookie* yang berisi tanda tangan digital kepada komputer *client.*

Untuk memastikan bahwa tanda tangan digital sesuai dengan *website* yang dimasukkan, *public IP Address* dari pengunjung *website* akan diubah menjadi tanda tangan digital. Tujuan mengapa *public IP Address* yang diubah menjadi tanda tangan digital adalah mencegah sebuah proses *automation* dimana pihak penindak *phising* akan mengambil *cookie* yang diberikan oleh *website* asli, menuju ke pengguna pada *website phising*.

Sebagai contoh, seorang pengguna masuk ke sebuah *website phising*. Pengguna tersebut memiliki *IP Address* 130.126.157.2. Pihak *phising* yang melakukan *automation* tidak dapat membangkitkan sebuah tanda tangan digital menggunakan kunci privat pemilik *website* asli dengan *IP Address* tersebut dikarenakan sebuah *IP Address* hanya dapat digunakan oleh sebuah perangkat saja dan *server* dari *website* asli akan mengirimkan tanda tangan digital berdasarkan pengunjung dari *website* tersebut.

Sebagai kontras, jika dibangkitkan sebuah tanda tangan digital berdasarkan sebuah *keyword* statis (sebagai contoh: url), maka pihak website phising dapat masuk ke website asli, mengambil *cookie* yang berupa tanda tangan digital, lalu meneruskan *cookie* tersebut kepada pengguna. Untuk penggunaan *session id* pun juga tidak dapat digunakan dikarenakan *session id* memerlukan pengguna untuk melakukan *sign in* terlebih dahulu, dimana hal tersebut menghapus maksud dari perlindungan data pribadi dari *website phising*.

Penggunaan fungsi *hash* untuk membangkitkan tanda tangan digital dengan menggunakan fungsi *hash* SHA-3. Penggunaan SHA-3 dikarenakan menggunakan prinsip konstruksi spons, dimana hasil *hash* akan sulit untuk ditelusuri asalnya. Berikut gambar konstruksi spons yang digunakan SHA-3



Gambar 4. Konstruksi Spons pada SHA-3 [3]

Setelah *IP Address* pengguna diubah menjadi *Message Digest* dengan menggunakan SHA-3, akan dihasilkan sebuah tanda tangan digital dengan menggunakan kunci privat dari pemilik *website*, dan akan dikirimkan sebagai *cookie* setiap kali pengguna mengunjungi halaman *website* tersebut.

Setelah pengguna *website* mendapatkan *cookie* yang berisi tanda tangan digital, maka pengguna dapat menverifikasi *website* tersebut berdasarkan *cookie* yang sudah didapat. Untuk program yang digunakan untuk menvalidasi tanda tangan digital tersebut, dapat berupa sebuah program *add-on* pada *browser* pengguna.

IV. Implementasi Sistem

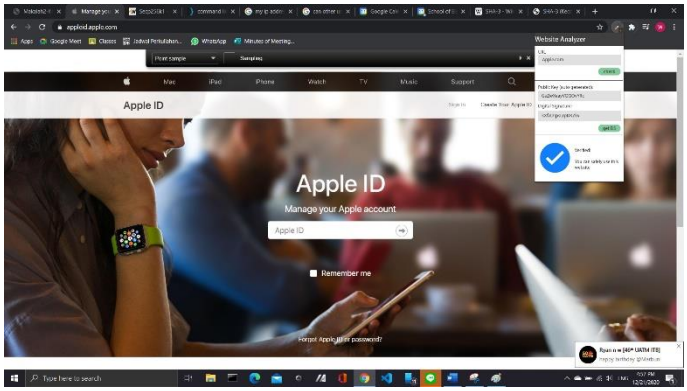
Tahapan yang dilakukan oleh program *add-on* tersebut adalah sebagai berikut.

1. Pengguna memasukkan website yang ingin dicek autentikasinya kepada program *add-on* pada *browser*
2. Program *add-on* akan mengakses *database* tempat penyimpanan kunci publik dan mengambil kunci publik *website* tersebut
3. Program akan mengambil *cookie* yang berisi tanda tangan digital yang diberikan oleh *server website*
4. Program mengambil *public IP Address* pengguna, dan mengubahnya menjadi *Message Digest* menggunakan fungsi *hash* yang sama.
5. Dari *Message Digest* tersebut, akan dilakukan proses verifikasi menggunakan kunci publik yang sebelumnya didapatkan, dan perhitungan kurva eliptik
6. Program *add-on* memberikan hasil dari verifikasi autentikasi *website* tersebut

Kasus yang mungkin terjadi jika program menghasilkan bahwa website tersebut tidak valid adalah:

1. *Website* yang dikunjungi adalah *website phising* (karena kunci privat website dapat diperoleh oleh pihak *phising*)
2. *Website* tidak mengirimkan *cookie* yang berisi tanda tangan digital
3. Pengguna mengganti *public IP Address* yang digunakan saat mengakses *website*
4. Terjadinya modifikasi *cookie* yang dikirimkan oleh *server* terhadap *client*

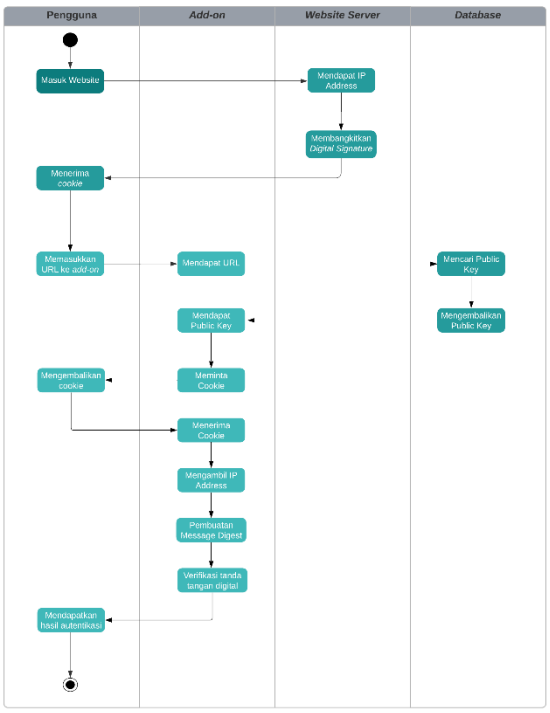
Untuk tampilan muka dari program *add-on* yang diusulkan dapat dilihat pada beberapa gambar dibawah ini.



Gambar 5. Tampilan antarmuka *add-on* jika berhasil diverifikasi

Gambar 6. Tampilan antarmuka *add-on* jika *website* tidak berhasil diverifikasi

Berikut adalah activity diagram dengan swimlane pada sistem yang diusulkan.



Gambar 7. Activity Diagram with Swimlane

Seperti yang dapat dilihat pada tampilan antarmuka tersebut, pengguna dapat memasukkan secara manual URL yang akan diperiksa. Secara *default*, program akan mengambil URL *website* saat program *add-on* dinyalakan. Setelah pengguna menekan tombol "check", maka program akan menelusuri *database* untuk menampilkan *public key* website tersebut.

Setelah program mendapatkan *public key*, public key akan ditampilkan di textbox yang tersedia. Untuk mendapatkan digital signature pada *cookie*, pengguna perlu menekan tombol "get DS" yang dimana program akan mengambil *digital signature* dari *cookie*, setelah itu melakukan proses verifikasi dengan *public IP address* pengguna tersebut. Jika terverifikasi, akan muncul hasil bahwa website tersebut terautentikasi keasliannya dan pengguna dapat dengan aman memasukkan data kedalam website tersebut.

V. Analisis Sistem

Sistem yang diusulkan pada tulisan ini memerlukan seluruh *website* memiliki sebuah *private key* dan juga *server* yang mempunyai fungsi untuk mengirimkan *digital signature* sebagai *cookie* kepada pengguna. Untuk sementara, sistem ini tidak dapat digunakan dikarenakan tidak ada *server* yang menerapkan hal ini. Tulisan ini dibuat sebagai sebuah proposal untuk pengembangan kedepannya.

ECDSA digunakan untuk membangkitkan tanda tangan digital dengan tujuan keamanan. ECDSA dibandingkan dengan DSA yang menggunakan RSA memiliki keunggulan, yaitu memiliki sistem keamanan yang sama dengan menggunakan kunci yang lebih pendek.

SHA-3 digunakan untuk melakukan *hashing* dikarenakan sejauh ini SHA-3 belum dapat dipecahkan tanpa menggunakan *effort* yang sangat besar. SHA-3 juga merupakan varian SHA yang paling terbaru jika dibandingkan dengan SHA-1 dan SHA-2.

VI. Kesimpulan dan Saran

Kesimpulan yang dapat diambil dari tulisan ini adalah tanda tangan digital dapat dimanfaatkan untuk melakukan validasi *website* untuk membedakan *website phising* dengan *website* asli. Namun dalam tulisan ini, belum ada pengujian mengenai performa dan *workload server* untuk secara terus menerus membangkitkan tanda tangan digital para penggunanya. Tulisan ini hanya ditujukan sebagai proposal dan gagasan dasar penerapan *digital signature* berbasis ECDSA untuk autentikasi *website*.

Saran untuk pengembangan selanjutnya adalah untuk lebih matang memikirkan *vulnerability* yang dapat terjadi dengan menggunakan sistem ini. Selain itu, perlu juga diuji mengenai performa dan *workload server* dalam membangkitkan *digital signature*.

VI. Acknowledgement

Penulis pertama-tama ingin berterima kasih kepada Tuhan Yang Maha Esa atas berkat dan anugrahNya yang telah mengizinkan penulis untuk dapat menuliskan tulisan ini dan memiliki kesempatan untuk menimba ilmu didalam bidang kriptografi. Selain itu, penulis juga ingin berterima kasih kepada keluarga saya yang telah mendukung saya dan merawat saya dikala saya sedang menulis tulisan ini. Saya juga berterima kasih sebesar-besarnya kepada dosen pengajar penulis, yaitu Pak Rinaldi Munir selaku dosen pengampu mata kuliah IF 4020 Kriptografi yang telah mengajarkan penulis mengenai kriptografi. Terakhir, saya berterima kasih kepada teman-teman saya, yang saya tidak bisa sebutkan satu per satu, yang telah mendukung saya, memberikan saran, serta menjadi teman berdiskusi dalam pembuatan tulisan ini.

VII. Referensi

[1] R. Munir, "Fungsi hash," Teknik Informatika STEI - ITB, Bandung.

[2] En.bitcoin.it. 2020. *Secp256k1 - Bitcoin Wiki*. [online] Available at: <https://en.bitcoin.it/wiki/Secp256k1> [Accessed 21 December 2020].

[3] R. Munir, "Fungsi hash SHA-3 (Keccak)," Teknik Informatika STEI - ITB, Bandung.

[4] Instructables.com. 2020. [online] Available at: <https://www.instructables.com/Understanding-how-ECDSA-protects-your-data/> [Accessed 21 December 2020].

[5] En.wikipedia.org. 2020. *Elliptic Curve Digital Signature Algorithm*. [online] Available at: <https://en.wikipedia.org/wiki/Elliptic_Curve_Digital_Signature_Algorithm> [Accessed 21 December 2020].

[6] Twinarko, A., n.d. Elliptic Curve Digital Signature Algorithm (ECDSA).

[7] Tools.ietf.org. 2020. *RFC 6979 - Deterministic Usage Of The Digital Signature Algorithm (DSA) And Elliptic Curve Digital Signature Algorithm (ECDSA)*. [online] Available at: <https://tools.ietf.org/html/rfc6979> [Accessed 21 December 2020].

[8] Johnson, D., Menezes, A. and Vanstone, S., 2001. The Elliptic Curve Digital Signature Algorithm (ECDSA). *International Journal of Information Security*, 1(1), pp.36-63.

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

Willsen Sentosa
13517036