Pemanfaatan Tanda Tangan Digital Menggunakan ECDSA dan Keccak pada Teks Editor

Dandy Arif Rahman
Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Bandung, Indonesia
13516086@std.stei.itb.ac.id

***Abstrak*—Tanda tangan digital sangat dibutuhkan untuk menjamin *authenticity, integrity,* dan *non-repudiation* dokumen yang ditandatangani. *Elliptic Curve Digital Signature Algorithm (ECDSA)* adalah algoritma kunci publik yang digunakan untuk implementasi sebuah skema tanda tangan digital. Fungsi hash Keccak merupakan fungsi yang memetakan sebuah string menjadi ukuran tertentu, algoritma ini juga digunakan untuk skema tanda tangan digital. ECDSA dan fungsi hash Keccak terbukti bekerja dengan baik dari algoritma kunci publik dan fungsi hash lainnya. Skema tanda tangan digital ini akan diimplementasikan pada sebuah teks editor, sehingga dokumen tersebut bisa diverifikasi keasliannya.**

***Kata Kunci—Elliptic Curve Digital Signature Algorithm, Keccak,* Tanda Tangan Digital.**

I. Latar Belakang

Di era ini, teknologi informasi seperti sudah tidak bisa dipisahkan dari kehidupan kita. Teknologi informasi berperan banyak dalam kehidupan kita dari mulai perkara besar, seperti sistem pesawat terbang yang sangat kompleks, hingga ke hal-hal kecil seperti pemesanan makanan yang biasa kita lakukan sehari-hari.

Semakin berkembangnya teknologi informasi, semakin banyak pula data digital yang berpindah dari satu tempat ke tempat yang lain. Muncul isu tentang keamanan data digital yang pertukarkan tersebut, lalu muncul peran ilmu kriptografi. Kriptografi berperan penting dalam salah satu metode pengamanan data digital, kriptografi berperan dalam mengenkripsi data sehingga orang yang tidak berwenang tidak bisa menyalahgunakan data digital tersebut.

Aplikasi lain yang tidak kalah penting dari ilmu kriptografi yaitu tanda tangan digital.Tanda tangan digital memastikan identitas dari pembuat suatu dokumen, email, atau data digital lainnya dengan menerapkan suatu algoritma kriptografi. Sebenarnya apa yang dilakukan oleh tanda tangan digital adalah menjamin tiga hal, yaitu *authenticy, integrity,* dan *non-repudiation*. *Authenticity* menjelaskan siapa yang membuat dokumen tersebut. *Integrity* yaitu tidak ada perubahan dari dokumen tersebut setelah penandatanganan digital dilakukan. *Non-repudiation* maknanya bahwa sang pembuat dokumen tidak bisa menyangkal di kemudian hari bahwa dia tidak pernah membuat dokumen tersebut.

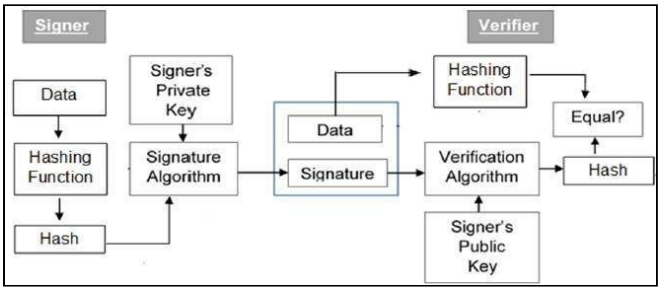
Di zaman sekarang hampir seluruh pembuatan dokumen dilakukan secara digital dengan menggunakan bantuan teks editor, dari mulai penulisan tugas hingga dokumen kenegaraan yang sangat penting. Maka dari aplikasi tanda tangan digital menjadi sangat penting untuk menjamin aspek *authenticity, integrity,* dan *non-repudiation* dari dokumen tersebut, terlebih lagi pada dokumen-dokumen yang bersifat penting. Sayangnya saat ini masih sedikit teks editor yang menyediakan layanan tanda tangan digital sebagai salah satu fiturnya. Maka dari itu penulis merasa hal ini sangat penting dan berharap makalah ini bisa menjadikan suatu konsep dasar penerapan tanda tangan digital pada teks editor.

II. Dasar Teori

A. Tanda Tangan Digital

Tanda tangan digital adalah skema matematika untuk memverifikasi keaslian pesan atau dokumen digital. Tanda tangan digital yang valid, dengan syarat-syarat yang terpenuhi, memberikan penerima pesan alasan yang sangat kuat untuk percaya bahwa pesan itu dibuat oleh pengirim yang dikenal (otentikasi), dan bahwa pesan itu tidak diubah dalam transit (integritas).[1] Selain itu tanda tangan digital juga dapat memastikan aspek *non-repudiation* sehingga penulis pesan tidak bisa menyangkal di kemudian hari bahwa pesan tersebut bukan ditulis oleh dirinya.

Sistem tanda tangan digital menggunakan algoritma kunci publik dan juga fungsi hash untuk menjamin *authenticity, integrity,* dan *non-repudiation* pesan yang ditandatangani. Pertama, data/pesan yang akan ditandatangani di-hash kemudian hasil hash tersebut di enkripsi dengan kunci privat penandatangan yang disebut tanda tangan. Kemudian data/pesan di-*append* dengan tanda tangan digital. Di sisi *verifier* data dan tanda tangan digital dipisah. Kemudian hasil hash dari data dibandingkan dengan hasil dekripsi tanda tangan digital menggunakan kunci publik penandatangan, jika sama maka data tersebut adalah data yang benar atau dalam kata lain *authenticity, integrity,* dan *non-repudiation* nya terjamin. Berikut skema tanda tangan digital.



Gambar 1. Skema Tanda Tangan Digital
https://www.tutorialspoint.com/cryptography/cryptography_digital_signatures.htm

B. Kriptografi Kunci Publik

Kriptografi kunci publik, atau kriptografi asimetris, adalah sistem kriptografi yang menggunakan pasangan kunci: kunci publik yang dapat disebarluaskan secara luas, dan kunci privat yang hanya diketahui oleh pemiliknya. Generasi kunci tersebut tergantung pada algoritma kriptografi berdasarkan masalah matematika untuk menghasilkan fungsi satu arah. Keamanan yang efektif hanya membutuhkan menjaga kunci privat privat; kunci publik dapat didistribusikan secara terbuka tanpa mengurangi keamanan.[2] Beberapa contoh algoritma kriptografi kunci publik adalah algoritma pertukaran kunci Diffie–Hellman, ElGamal, beberapa teknik algoritma kurva eliptik, RSA, dll.

C. Fungsi Hash

Fungsi hash adalah fungsi apa pun yang dapat digunakan untuk memetakan data dengan ukuran acak ke data dengan ukuran tetap. Nilai yang dikembalikan oleh fungsi hash disebut nilai hash, kode hash, *digest*, atau hanya hash. Fungsi hash kriptografi memungkinkan seseorang untuk dengan mudah memverifikasi apakah beberapa input data memetakan ke nilai hash yang diberikan, tetapi jika data input tidak diketahui, sulit untuk merekonstruksi ulang dengan mengetahui nilai hash yang tersimpan. Ini digunakan untuk memastikan integritas data yang dikirim, dan otentikasi pesan. Beberapa contoh fungsi hash adalah SHA-1, SHA-2, MD-5, Keccak, dll.

D. *Elliptic Curve Digital Signature Algorithm* (ECDSA)

Elliptic Curve Digital Signature Algorithm (ECDSA) merupakan sebuah algoritma tanda tangan digital yang menggunakan pasangan kunci berdasarkan algoritma kriptografi kunci publik Elliptic Curve Cryptography (ECC). Letak keamanannya adalah kesulitan pemecahan persamaan kurva eliptik. Algoritma ini terdiri dari 2 tahap yaitu tahap penandatanganan dan tahap verifikasi.

1) Tahap Penandatanganan

Misalkan Alice ingin mengirim pesan yang ditandatangani kepada Bob. Awalnya, mereka harus menyetujui parameter kurva (CURVE, G, n). CURVE adalah bidang kurva elips dan persamaan yang digunakan, G adalah titik dasar kurva elips, dan n adalah bilangan prima urutan integer G, berarti n x G = O, dimana O adalah elemen identitas.

Alice menciptakan pasangan kunci, yang terdiri dari integer kunci privat d_A, dipilih secara acak dalam interval [1, n-1] dan titik kurva kunci publik $Q_A = d_A$ x G. X adalah simbol untuk menunjukkan titik kurva eliptik perkalian dengan skalar.

Alice menandatangani pesan m, ia mengikuti langkah-langkah ini:

1. Hitung e = HASH(m), HASH merupakan fungsi hash, kemudian output nya dikonversi ke integer.
2. z merupakan L_n bit paling kiri dari e, di mana L_n adalah panjang bit dari *group order* n. (Perhatikan bahwa z dapat lebih besar dari n tetapi tidak lebih panjang.
3. Pilih integer acak k antara [1, n-1].
4. Hitung titik kurva $(x_1, y_1) = k$ x G
5. Hitung r = x_1 mod n, jika r = 0, lakukan kembali dari langkah 3.
6. Hitung s = $k^{-1}(z + rd_A)$ mod n, jika s = 0, lakukan kembali dari langkah 3.
7. Tanda tangan digital merupakan pasangan (r, s).

Perlu diperhatikan, k bukan hanya harus rahasia tetapi juga harus berbeda untuk tiap tanda tangan digital yang berbeda.

2) Tahap Verifikasi

Di sisi Bob untuk mengautentikasi tanda tangan Alice, ia harus memiliki salinan titik kurva kunci publik Q_A. Bob dapat memverifikasi Q_A adalah titik kurva yang valid sebagai berikut:

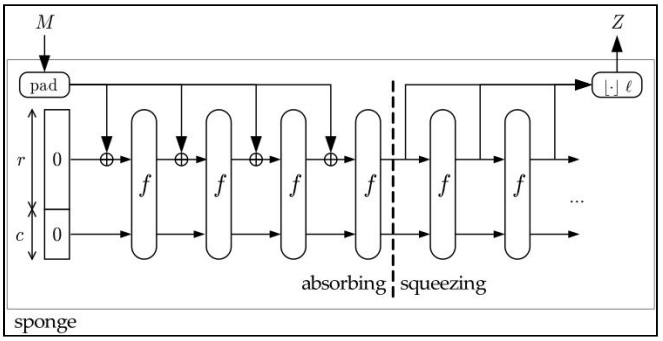
1. Periksa apakah Q_A tidak sama dengan elemen identitas O, dan koordinatnya dinyatakan valid.
2. Periksa apakah Q_A terletak pada kurva.
3. Periksa apakah n x Q_A = O

Setelah itu, Bob mengikuti langkah-langkah ini:

1. Pastikan bahwa r dan s adalah bilangan bulat antara [1, n-1]. Jika tidak, tanda tangan tidak valid.
2. Hitung e = HASH(m), dengan menggunakan fungsi hash yang sama pada tahap penandatanganan.
3. z merupakan L_n bit paling kiri dari e.
4. Hitung $u_1 = zs^{-1}$ mod n dan $u_2 = rs^{-1}$ mod n.
5. Hitung titik kurva $(x_1, y_1) = u_1$ x G + u_2 x Q_A. Jika (x_1, y_1) = O, maka tanda tangan tidak valid
6. Tanda tangan valid jika r ekivalen x_1, jika tidak invalid.

E. Keccak Hash Function

Fungsi Hash Keccak merupakan pemenang dari kompetisi terbuka untuk SHA-3 yang diadakan oleh NIST. Keccak berbeda dari finalis SHA-3 lainnya dalam hal menggunakan konstruksi 'spons' (sponge construction). Jika desain lainnya bergantung pada 'fungsi kompresi, Keccak menggunakan fungsi non-kompresi untuk menyerap dan kemudian 'memeras' digest.



Gambar 2. Skema fungsi hash *Keccak*

https://en.bitcoinwiki.org/wiki/SHA-3

III. Rancangan Sistem

Pada bagian isi akan dijelaskan mengenai rancangan teks editor yang akan dibuat dengan fitur tanda tangan digital untuk memastikan aspek *authenticity, integrity,* dan *non-repudiation* dokumen yang ditandatangani.

Setiap user akan memiliki pasangan kunci publik dan kunci privat, asumsi nya kunci publik seluruh user tersimpan di sebuah tempat yang dapat diakses oleh user lainnya. Kemudian setelah penulis dokumen telah selesai membuat dokumen tersebut terdapat pilihan '*save with digital signature*' kemudian muncul dialog box yang meminta pembuat dokumen untuk memasukan kunci privatnya.

Pada skema ini yang menjadi m atau pesan adalah keseluruhan file dokumen tersebut, termasuk metadata nya. Lalu file tersebut akan di hash menggunakan algoritma fungsi hash keccak, digest dari hash tersebut kemudian dikonversi menjadi bilangan integer. Lalu *digest* tersebut dienkripsi menggunakan algoritma kriptografi kunci publik ECDSA dengan menggunakan kunci privat penandatangan, hasil dari enkripsi ini disebut dengan tanda tangan digital. Lalu file yang disimpan di-*append* dengan tanda tangan digital.

Di sisi pembaca/pengguna dokumen, ia ingin memastikan keaslian dari dokumen yang telah ditandatangani oleh ini sang pembuat dokumen. Pembaca dokumen akan membuka file tersebut dan memilih fitur '*check with digital signature*' untuk memverifikasi dokumen tersebut. Pada skema verifikasi dari sistem tanda tangan digital ini, pesan akan di hash dengan fungsi hash yang sama saat tahap penandatanganan, yaitu fungsi hash keccak. Lalu tanda tangan digital yang di-*append* ke dokumen tersebut akan di dekripsi menggunakan algoritma kriptografi kunci publik ECDSA dengan menggunakan kunci publik penandatangan. Lalu hasil nya akan dibandingkan dengan hasil hash, jika sama ini artinya dokumen tersebut adalah dokumen yang asli dan akan muncul pesan 'Dokumen Anda adalah asli' jika tidak maka akan muncul pesan 'Maaf, ada kesalahan pada dokumen Anda'.

IV. Implementasi Sistem

Untuk implementasi dari skema penandatanganan digital ini sendiri penulis bermaksud untuk mengimplementasikannya pada teks editor *google docs* tapi karena rumitnya perizinan untuk membuat *adds on* pada *google docs*, akhirnya penulis memutuskan untuk membuat teks editor sendiri menggunakan Java.

Teks editor tersebut diimplementasikan menggunakan library java swing. Untuk saat ini implementasi dari teks editor tersebut masih sangat sederhana, yaitu untuk membuat teks tanpa *formatting* apapun. Lalu di menu bar teks editor tersebut terdapat opsi file dan edit. Opsi file memiliki sub opsi *new, open, save, save with digital signature,* dan *exit*. Sedangkan opsi edit memiliki sub opsi *cut, copy, paste.*

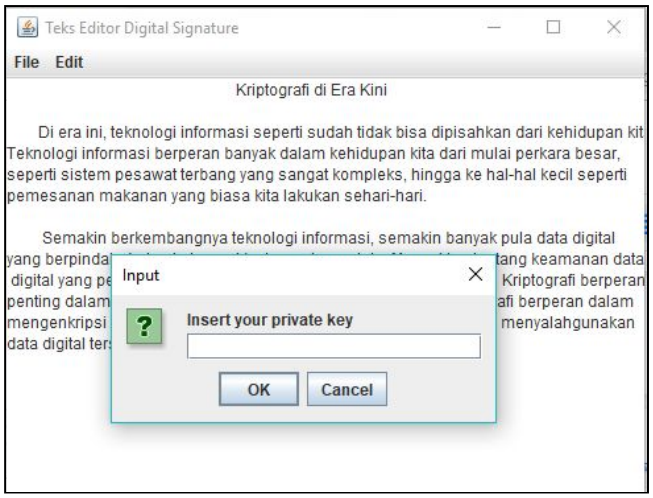
Saat pembuat dokumen sudah selesai membuat dokumen tersebut dan memilih opsi save with digital signature, akan muncul dialog box yang meminta private key d_A dari user tersebut, sementara untuk parameter CURVE kurva eliptik, G, dan n sudah diimplementasikan di dalam program tanpa ditunjukkan ke user, lalu perhitungan public key Q_A juga dilakukan di dalam program, yang asumsi nya disimpan disebuah tempat yang diketahui oleh user yang akan memverifikasi pesan tersebut.

Kemudian di sisi user yang memverifikasi saat membuka dokumen dia harus memasukkan kunci publik penandatangan, lalu jika dokumen tersebut *verified* akan diberikan pesan sukses, ini menandakan *authenticity, integrity,* dan *non-repudiation* dokumen yang ditandatangani benar. Tetapi jika terjadi perubahan pada pesan sehingga tidak sama dengan tanda tangan digital maka ini menunjukan dokumen tidak *verified* dan akan muncul pesan error.

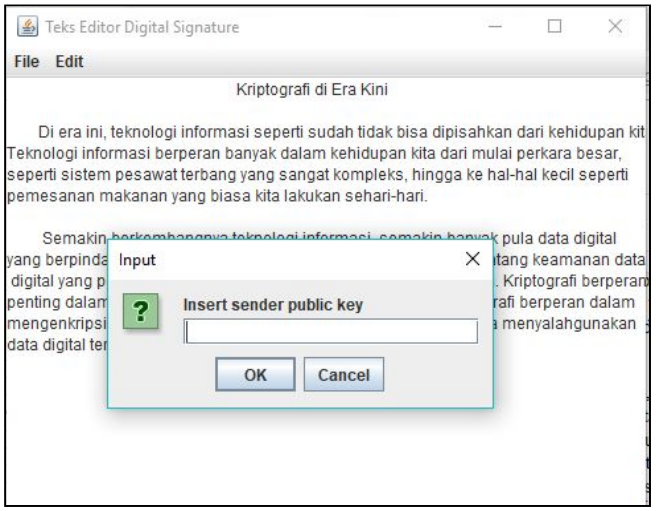
Adapun backend dari program ini adalah implementasi *Elliptic Curve Digital Signature Algorithm* (ECDSA) dan juga algoritma fungsi hash *Keccak*, keduanya diimplementasikan menggunakan bahasa pemrograman java, menyesuaikan dengan teks editor yang telah dibuat dengan menggunakan java juga.

Gambar 3. Teks Editor

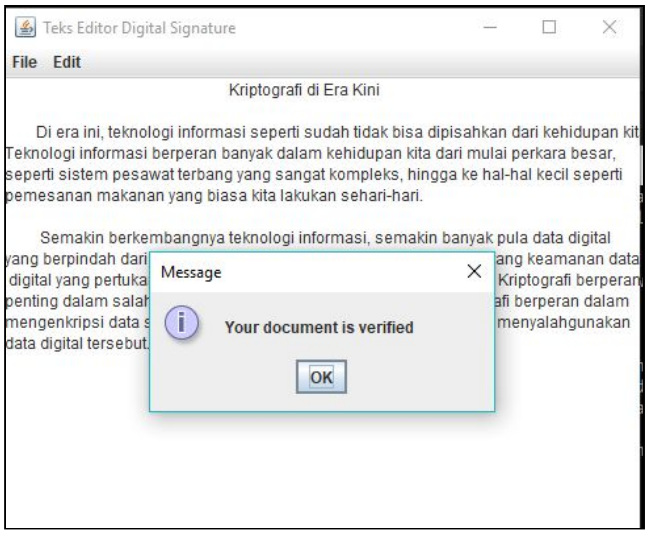
Gambar 4. Option File



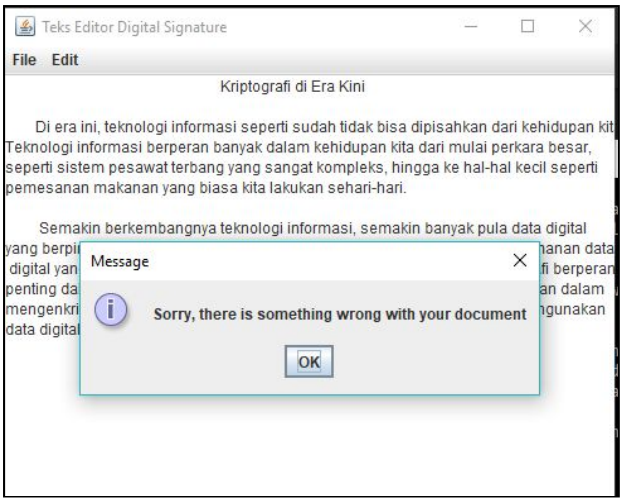
Gambar 5. Sub Option Save with Digital Signature



Gambar 6. Sub Option Check with Digital Signature



Gambar 7. Dialog Box jika dokumen verified



Gambar 8. Dialog Box jika dokumen tidak verified

V. Analisis Sistem

Pada dasarnya banyak algoritma penandatanganan digital yang tersedia, begitu pula dengan fungsi *hash* yang tersedia. Tetapi pada sistem penandatanganan digital yang dibuat kali ini, penulis memilih untuk memilih implementasi dengan algoritma kriptografi kunci publik *Elliptic Curve Digital Signature Algorithm* (ECDSA) dan juga algoritma fungsi hash *Keccak*. Hal ini didasarkan pada keamanan disediakan pada kedua algoritma tersebut.

Jika dibandingkan dengan algoritma kunci publik yang popular lainnya untuk penandatanganan digital, misal RSA. ECDSA memiliki keunggulan yaitu dengan level keamanan yang sama ECDSA memiliki kunci yang lebih pendek, hal ini berhubungan erat dengan persamaan kurva eliptik yang digunakan oleh ECDSA. Karena sampai saat ini komunitas matematika belum bisa membuat algoritma yang cukup cepat untuk memecahkan persamaan tersebut. Untuk algoritma fungsi hash keccak, jika dibandingkan dengan pendahulunya, yaitu SHA-1 dan SHA-2, lebih kuat, karena SHA-1 sudah bisa dipecahkan, walaupun SHA-2 masih banyak digunakan pada saat ini.

VI. Kesimpulan dan Saran

Kesimpulannya kriptografi berperan penting dalam melakukan pengamanan dokumen digital di era ini, salah satu nya adalah algoritma kriptografi kunci publik dan fungsi hash yang menjadi dasar dari implementasi tanda tangan digital. Tanda tangan digital diperlukan untuk memastikan *authenticity, integrity,* dan *non-repudiation* dokumen digital yang ditandatangani. Salah satu penerapannya adalah tanda tangan digital menggunakan ECDSA dan fungsi hash Keccak pada teks editor, algoritma ini terbukti lebih baik daripada algoritma-algoritma lainnya.

Saran dari penulis adalah, mungkin penerapan dokumen digital ini bisa dikembangkan lagi untuk, teks editor kolaboratif, dimana tidak hanya satu orang yang membuat dokumen tersebut, mungkin bisa dikembangkan untuk penandatanganan oleh banyak penulis.

Referensi

[1] Paul, Eliza (12 September 2017). "What is Digital Signature- How it works, Benefits, Objectives, Concept". EMP Trust HR.

[2] Stallings, William (3 Mei 1990). Cryptography and Network Security: Principles and Practice. Prentice Hall. p. 165. ISBN 9780138690175.

Pernyataan

Dengan ini kami menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 10 Mei 2019

Dandy Arif Rahman