Penerapan Tanda Tangan Digital dan Teknik Steganografi pada Surat Resmi

Hansel Valentino Tanoto - 13520046 (*Author*)
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): 13520046@std.stei.itb.ac.id

***Abstract*—Pada era digital saat ini, seringkali kita mengirim atau menerima surat resmi/formal secara daring melalui *email* atau platform elektronik lainnya. Namun, dibalik kemudahannya, terdapat ancaman keamanan dan integritas isi dari surat tersebut karena mudahnya modifikasi isi oleh pihak ketiga saat dalam proses pengiriman. Oleh karena itu, solusi untuk menjaga integritas data tersebut adalah dengan menggunakan tanda tangan digital sehingga penerima surat nantinya dapat melakukan verifikasi untuk memastikan keabsahan surat tersebut. Makalah ini akan membahas penerapan tanda tangan digital dan steganografi pada surat resmi. Dalam implementasinya, akan digunakan teknik *hashing* menggunakan SHA-3 dan kriptografi kunci public ECCEG (*Eliptic Curve Cryptography ElGamal*). Dari hasil pengujian, diperoleh hasil yang memuaskan, yaitu perubahan sedikit saja pada konten surat dapat terdeteksi melalui verifikasi tanda tangan digitalnya. Selain itu, algoritma ini juga tidak membutuhkan waktu yang cukup lama dalam proses *signing* dan *verification* untuk ukuran *file* surat standar yang umumnya tidak terlalu besar.**

Kata Kunci—surat resmi; tanda tangan digital; steganografi; hash; enkripsi; dekripsi

I. PENDAHULUAN

Dalam era digital saat ini, surat resmi yang dikeluarkan oleh suatu organisasi, instansi pemerintah, perusahaan, atau lembaga lainnya juga mengalami transformasi menuju bentuk yang lebih modern. Surat-surat resmi tersebut tak jarang dikirimkan secara elektronik melalui *email* atau platform komunikasi digital lainnya. Hal ini tentu memberikan efisiensi dan kemudahan dalam pendistribusian informasi karena informasi dalam dokumen tersebut dapat sampai lebih cepat ke tujuan dan juga menghemat biaya dalam hal pencetakan dan peggunaan kertas. Meskipun demikian, penggunaan surat resmi elektronik juga membawa tantangan baru dalam hal keamanan dan integritas konten dokumen tersebut.

Keamanan dan integritas surat resmi merupakan hal yang sangat penting, terutama dalam konteks hukum, bisnis, dan administrasi publik. Manipulasi dokumen, seperti perubahan konten, pencurian data/identitas, atau penambahan informasi palsu, dapat mengakibatkan kerugian yang cukup serius dan dapat merusak kepercayaan pihak penerima terhadap organisasi/lembaga yang mengeluarkan surat tersebut.

Gambar 1. Contoh kasus pemalsuan terhadap surat edaran dari WRAM ITB (sumber: dokumentasi penulis)

Untuk mengatasi persoalan ini, penerapan tanda tangan digital (*digital signature*) telah menjadi solusi yang populer dan efektif. Tanda tangan digital ini sebenarnya sudah sering diterapkan pada dokumen-dokumen penting lainnya, terutama sejak masa pandemic COVID-19 yang lalu, misalnya ijazah digital, sertifikat, dan/atau akta-akta penting lainnya. Penggunaan tanda tangan digital memungkinkan pengguna untuk mengautentikasi dan mengamankan surat resmi elektronik dengan menerapkan beberapa algoritma kriptografi. Dengan menggunakan kunci-kunci kriptografi yang unik, tanda tangan digital memberikan keabsahan dan keutuhan pada dokumen elektronik tersebut.

Selain tanda tangan digital, akan diterapkan juga teknik steganografi yang digunakan untuk menyembunyikan informasi penting secara rahasia ke dalam dokumen surat resmi tersebut sehingga menjadi tidak terlihat (tersembunyi) tanpa menimbulkan kecurigaan dari pihak yang tidak berwenang. Dalam konteks makalah ini, informasi yang disembunyikan pada surat resmi ini adalah tanda tangan digital yang dihasilkan sebelumnya.

Makalah ini akan membahas penerapan tanda tangan digital dan teknik steganografi tersebut pada surat resmi, dengan fokus pada penggunaannya dalam konteks keamanan (*security*) dan integritas (*integrity*) dokumen. Di dalamnya, akan dijelaskan konsep dasar tanda tangan digital, steganografi, algoritma kriptografi yang terlibat untuk *hashing*, enkripsi, dan dekripsi, serta bagaimana implementasi/penerapannya pada surat resmi elektronik.

II. DASAR TEORI

A. *Surat Resmi*

Surat resmi adalah surat yang digunakan untuk keperluan formal oleh pihak tertentu, baik perseorangan, organisasi, lembaga, maupun instansi tertentu untuk melakukan komunikasi secara resmi. Dalam penulisannya, surat resmi dibuat dengan mengikuti kaidah (struktur dan format) yang jelas dan juga menggunakan bahasa yang baku. Surat resmi biasanya digunakan untuk berbagai keperluan yang bersifat formal, seperti mengajukan permohonan, menyampaikan informasi, mengundang, memberikan instruksi, memberikan pemberitahuan resmi, atau menjalankan kegiatan administratif.

Umumnya surat resmi memiliki struktur yang terdiri atas kop surat, alamat dan tanggal, alamat penerima, pengantar, isi, penutup, salam penutup, tanda tangan, dan lampiran.

Seiring perkembangan zaman, surat resmi yang awalnya menggunakan media cetak, sekarang mulai beralih pada penggunaan *email* dan media komunikasi elektronik lainnya sehingga ada aspek tambahan yang perlu dipertimbangkan, yaitu masalah keamanan dan integritas data, karena surat digital dapat dipalsukan oleh pihak yang tak bertanggung jawab. Sehingga, apabila integritas konten surat tidak terjaga, tentu akan timbul rasa ketidakpercayaan penerima terhadap pengirim surat. Oleh karena itu, makalah ini menawarkan solusi berupa penerapan tanda tangan digital dan steganografi untuk menjaga integritas dan keaslian surat tersebut.

B. *Kriptografi*

Kriptografi merupakan sebuah cabang ilmu/seni yang bertujuan untuk menjaga dan menjamin keamanan informasi. Kata *Cryptography* sendiri berasal dari bahasa Yunani, *Cryptos* dan *Graphien* yang berarti *secret*/*hidden writing*. Kriptografi menyediakan 4 layanan utama dalam keamanan, yaitu kerahasiaan pesan (*confidentiality*), keaslian pesan (*data integrity*), keaslian pengirim dan penerima pesan (*authentication*), dan anti penyangkalan (*non-repudiation*).

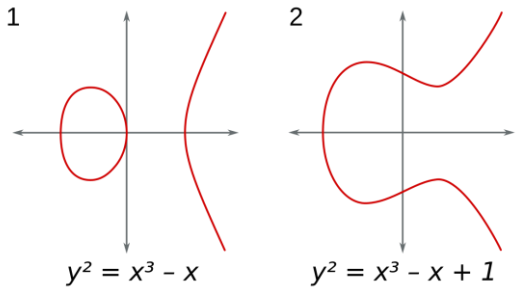
Secara umum, proses dalam kriptografi dapat dibagi menjadi 2 proses utama, yaitu enkripsi dan dekripsi. Enkripsi adalah proses mentransformasi suatu pesan asli (plainteks) menggunakan suatu kunci enkripsi untuk menghasilkan pesan yang terenkripsi (cipherteks). Sebaliknya, dekripsi adalah proses mentransformasi kembali pesan rahasia yang terenkripsi tersebut (cipherteks) kembali ke wujud aslinya (plainteks) menggunakan kunci dekripsi.

Berdasarkan jenis kuncinya, kriptografi dapat dibagi menjadi 3 jenis, yaitu kriptografi simetri, kriptografi asimetri (kunci publik), dan fungsi *hash*. Algoritma kriptografi simetri menggunakan satu kunci yang sama untuk melakukan enkripsi dan dekripsi. Sedangkan, kriptografi asimetri menggunakan 2 kunci yang berbeda untuk proses enkripsi dan dekripsinya sehingga disebut kriptografi asimetri. Umumnya, kunci untuk enkripsi disebut kunci publik karena dapat disebarkan dan diketahui oleh masyarakat umum, sedangkan kunci untuk dekripsi disebut sebagai kunci privat karena bersifat rahasia dan hanya diketahui oleh pemiliknya. Terakhir, fungsi *hash* adalah jenis kriptografi yang tidak memungkinkan cipherteks untuk dikembalikan menjadi pesan semula (bersifat *irreversible*).

Tanda tangan digital yang digunakan pada makalah ini akan menggunakan algoritma kunci publik El Gamal dan fungsi *hash* SHA-3 yang akan dijelaskan lebih lanjut pada subbab selanjutnya. Alasan penggunaan kriptografi asimetri dibanding kriptografi simetri adalah faktor keamanannya yang lebih baik karena pemilik kunci tidak perlu mengirimkan kunci privatnya ke pengirim pesan, melainkan pengirim pesan tersebut dapat menggunakan kunci publik penerima (pemilik kunci) yang diketahui secara umum.

C. *Eliptic Curve Cryptography (ECC)*

Eliptic curve cryptography merupakan teknik kriptografi yang memanfaatkan konsep kurva eliptik untuk mengoptimasi proses enkripsi dan dekripsi pada kriptografi kunci publik. Kurva eliptik adalah kurva dengan bentuk umum persamaan $y^2 = x^3 + ax + b$, dengan syarat $4a^3 + 27b^2 \neq 0$ untuk mencegah adanya perpotongan kurva dengan dirinya sendiri dan menjaga kurva agar tetap *smooth* (tidak patah).



Gambar 2. Contoh kurva eliptik
(sumber: https://upload.wikimedia.org/wikipedia/commons/thumb/d/d0/ECClines-3.svg/1200px-ECClines-3.svg.png)

Algoritma ECC ini masih tergolong algoritma yang baru karena baru pertama kali dikembangan oleh Neal Koblitz dan Victor S. Miller pada tahun 1985. Umumnya, kriptografi kunci publik memiliki permasalahan dalam ruang penyimpanan kunci dan pesan yang besar serta waktu komputasi yang lama karena algoritma tersebut biasanya bergantung pada kompleksitas komputasi bilangan menggunakan bilangan-bilangan besar, misalnya persoalan logaritma diskrit. Namun, ECC ini menawarkan alternatif solusi dengan melakukan komputasi matematis berbasis kurva eliptik sehingga memungkinkan untuk dihasilkannya tingkat keamanan yang setara dengan algoritma kunci publik pada umumnya, tetapi dengan menggunakan ukuran kunci yang lebih kecil.

Konsep dasar dari ECC adalah terminologi pada aljabar abstrak, yaitu grup dan medan (*field*). Grup adalah sistem aljabar yang terdiri dari sebuah himpunan G dan sebuah operasi biner * sedemikian sehingga elemen-elemen pada G memenuhi aksioma *closure* dan asosiatif pada operator *, serta memiliki elemen netral dan elemen invers. Sementara itu, medan terdiri atas sebuah himpunan F dan 2 buah operasi biner, yaitu penjumlahan (+) dan perkalian (•) yang memenuhi syarat grup <F,+> dan <F-{0},•> adalah grup abelian, serta

operasi perkalian bersifat menyebar terhadap operasi penjumlahan (distributif).

Keamanan pada ECC didasarkan pada *Eliptic Curve Discrete Logarithm Problem* (ECDLP), yaitu mudahnya menghitung kP = Q, tetapi sulit untuk menghitung k dari P dan Q, dengan k merupakan sebuah konstanta (bilangan besar), sedangkan P dan Q merupakan titik pada kurva eliptik. Implementasinya pada kriptografi adalah dengan menggunakan Q sebagai kunci publik dan k sebagai kunci privat, sedangkan P adalah sembarang titik pada kurva eliptik. Berikut adalah langkah-langkah umum dalam algoritma ECC.

1. Kedua pihak menyepakati nilai a, b, dan sebuah bilangan prima p yang menyatakan sebuah kurva eliptik $y^2 = x^3 + ax + b \bmod p$.
2. Kedua pihak menghitung grup eliptik dari persamaan kurva eliptik tersebut.
3. Kedua pihak menentukan titik basis B dari grup eliptik.
4. Masing-masing pihak membangkitkan pasangan kunci publik dan kunci privat. Kunci privatnya adalah sebuah bilangan x dalam selang [1, p-1], sedangkan kunci publiknya adalah titik Q, dengan Q adalah x.B.

D. Algoritma Kriptografi Kunci Publik ElGamal

Algoritma ElGamal merupakan salah satu algoritma kriptografi kunci publik (asimetri) yang dikemukakan oleh Taher Elgamal pada tahun 1985. Faktor keamanan dari algortima ini terdapat pada sulitnya menyelesaikan persoalan logaritma diskrit, yaitu bagaimana mencari nilai x dalam persamaan $g^x \equiv y \bmod p$, dengan g dan y merupakan sembarang bilangan bulat dan p merupakan bilangan prima. Pada makalah ini, tanda tangan digital yang digunakan akan memanfaatkan versi *Eliptic Curve ElGamal* (ECEG), dengan mekanisme sebagai berikut.

1. Pengirim pesan (pihak 1) dan penerima pesan (pihak 2) menyepakati sebuah kurva eliptik dan sebuah titik basis B.
2. Kedua pihak akan membangkitkan pasangan kunci masing-masing, yaitu pihak 1 dengan kunci privat a dan kunci publik P_1 = a.B. Sedangkan pihak 2 memiliki kunci privat b dan kunci publik P_2 = b.B.
3. Pengirim akan mengkodekan/memetakan (*encoding*) pesan plainteks terlebih dahulu menjadi titik-titik (P_M) pada kurva eliptik.
4. Pengirim akan memilih bilangan acak k yang berada pada selang [1, p-1].
5. Pengirim akan mengkomputasi cipherteks dari P_M, yaitu P_C = [(k.B), (P_M + k.P_2)] dan mengirimkannya kepada penerima.
6. Penerima akan melakukan dekripsi dengan cara mengurangi titik kedua dari P_C, yaitu P_M + k.P_2 dengan hasil kali kunci privatnya, yaitu b terhadap titik pertama dari P_C, yaitu k.B sehingga diperoleh:

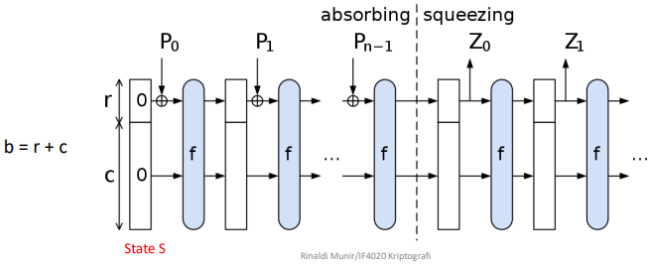
   (P_M + k.P_2) – (b.k.B) = P_M + k.b.B – b.k.B = P_M
7. Terakhir penerima dapat men-*decode* P_M kembali menjadi pesan asalnya.

Salah satu metode yang bisa digunakan untuk melakukan *encoding* dan *decoding* adalah dengan menggunakan metode Kolbitz yang langkah-langkahnya sebagai berikut.

1. Mengkodekan setiap karakter penyusun pesan menjadi angka, misalnya jika karakter penyusunnya adalah 0..9A..Z, maka bisa dikodekan menjadi 0-35 secara berturut-turut.
2. Memilih parameter basis berupa sebuah bilangan bulat k yang disepakati kedua belah pihak.
3. Untuk setiap nilai m, sulihkan x = m.k + 1 ke dalam persamaan kurva eliptik yang telah disepakati tadi, lalu tentukan nilai y yang memenuhi.
4. Apabila tidak ada nilai y yang memenuhi, maka coba sulihkan x = m.k + 2, lalu x = m.k + 3, dan seterusnya sampai diperoleh nilai y yang memenuhi.
5. Proses *decoding* dilakukan dengan menghitung m = ⌊(x-1) / k⌋, lalu memetakan kembali m ke karakter aslinya.

E. Fungsi Hash SHA-3 (Keccak)

Fungsi *hash* adalah fungsi yang digunakan untuk melakukan kompresi pesan berukuran sembarang (*message*) menjadi pesan yang berukuran tetap (*message digest*). Sifat dari *hash function* adalah *irreversible* sehingga hasil *hashing* tidak akan bisa dikembalikan menjadi pesan semula. Salah satu keluarga fungsi *hash* yang cukup populer adalah SHA (*Secure Hash Algorithm*) yang dibuat oleh NIST (*National Institute of Standards and Technology*), dengan salah satu variannya yang terbaru adalah SHA-3 (Keccak).



Gambar 3. Ilustrasi tahapan pada algoritma SHA-3
(sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2022-2023/27-SHA-3-2023.pdf)

Secara garis besar, algoritma SHA-3 menggunakan konstruksi spons menggunakan fungsi non-kompresi untuk menyerap kemudian memeras digest. Berikut adalah penjelasan dari tahapan algortima tersebut tersebut.

1. Praproses
   - Menentukan panjang *message digest* (d) dan panjang setiap blok (r).
   - Pesan M ditambah dengan bit *padding* menjadi *string* P sehingga habis dibagi r.

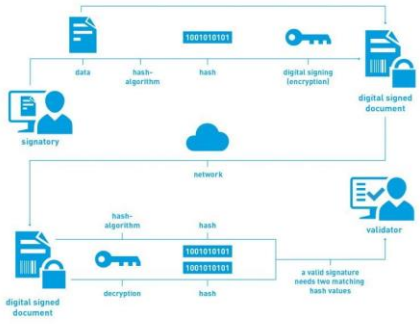
- *String* P dibagi menjadi blok-blok P_i berukuran r bit.
- Inisialisasi sebuah peubah status (*state*) S berukuran b-bit dengan 0. Ukuran b di sini adalah penjumlahan r dengan suatu konstanta c.

2. Fase Penyerapan (*Absorbing*)
   - Lakukan XOR antara r-bit P_i dengan r-bit dari *state* S.
   - Hasilnya dimasukkan ke dalam fungsi permutasi f untuk menghasilkan *state* S baru.
   - Lakukan langkah tersebut hingga semua blok P_i telah diproses.
3. Fase Pemerasan (*Squeezing*)
   - Inisialisasi *message digest* (Z) sepanjang d-bit dengan 0.
   - Sambungkan r-bit pertama dari *state* S ke Z. Hal ini dilakukan hingga panjang Z sudah sebesar d-bit.
   - Jika panjang Z belum sama dengan panjang d-bit, lakukan permutasi kembali pada S dengan menggunakan fungsi permutasi f sehingga diperoleh *state* S baru.

F. Tanda Tangan Digital

Tanda tangan digital (*digital signature*) merupakan salah satu mekanisme dalam kriptografi yang menyediakan layanan terhadap otentikasi (*authentication*) dan anti penyangkalan (*nonrepudiation*). Berbeda dengan tanda tangan pada dokumen cetak yang selalu sama, tanda tangan digital akan selalu berbeda-beda sesuai isi pesan/dokumen dan kunci yang digunakan. Secara umum, proses pada tanda tangan digital dapat dibagi menjadi proses menandatangani pesan (*signing*) dan memverifikasi keabsahan pesan (*verification*).

Cara umum yang digunakan untuk menandatangani pesan adalah dengan menggunakan kombinasi *hashing* dan kriptografi kunci public. Pada makalah ini akan digunakan *hashing* menggunakan fungsi *hash* SHA-3 dan kriptografi kunci publik ElGamal.



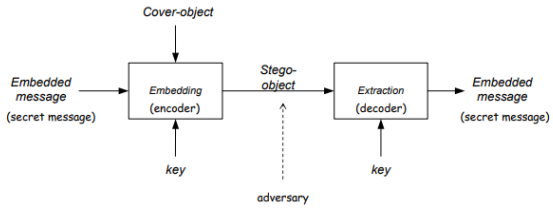
Gambar 4. Ilustrasi proses *signing* (atas) dan *verification* (bawah)
(sumber: https://easy-software.com/wp-content/uploads/2019/11/digital_signature_process-en-kr-1024x644.jpg)

Proses *signing* dilakukan dengan melakukan *hashing* terlebih dahulu terhadap pesan yang ingin dikirim sehingga dihasilkan sebuah *message digest*. Lalu *message digest* tersebut akan dienkripsi menggunakan kunci privat pengirim. Selanjutnya hasil enkripsi tersebut disebut sebagai *signature* dan akan disambungkan pada bagian akhir dari pesan.

Sementara itu, proses *verification* dilakukan dengan mempartisi kembali antara pesan dengan *signature*-nya. Kemudian, dilakukan dekripsi dari *signature* menggunakan kunci publik dari pengirim sehingga diperoleh kembali *message digest* yang sebelumnya. Lalu dilakukan perbandingan apakah *message digest* tersebut sama dengan hasil *hashing* dari pesan yang diterima. Apabila sama, artinya tanda tangan tersebut sah dan integritas pesan terjaga.

G. Steganografi

Steganografi adalah teknik pada kriptografi yang digunakan untuk mnyembunyikan pesan rahasia sehingga sulit terdeteksi dan tidak menimbulkan kecurigaan. Pada makalah ini, teknik steganografi digunakan untuk menyembunyikan *signature* pada *file* surat resmi. Berikut adalah diagram yang menggambarkan langkah umum dalam teknik steganografi.



Gambar 5. Ilustrasi proses pada steganografi
(sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2022-2023/08-Steganografi-Bagian1-2023.pdf)

Salah satu teknik umum yang digunakan untuk menyisipkan pesan adalah metode LSB, yaitu dengan mengubah *Least Significant Bits* (LSB) pada *cover object* dengan bit-bit pada pesan yang ingin disisipkan. Sedangkan cara lainnya adalah dengan menyisipkan tanda tangan digital tersebut ke dalam metadata dari *file* surat sehingga tidak akan mengubah konten surat sama sekali.

III. Rancangan Solusi dan Implementasi

A. Rancangan Solusi

Untuk mengatasi permasalahan integrtas konten surat resmi, penulis menawarkan solusi berupa penerapan tanda tangan digital dan steganografi untuk menyembunyikan tanda tangan digital tersebut agar tidak mempengaruhi konten surat. Berikut adalah penjelasan alur/proses yang dilakukan untuk penandatanganan surat.

1. Mengekstrak konten pada *file* pdf menjadi struktur data *string*. *String* tersebut selanjutnya akan disebut sebagai *message* M.

2. Melakukan penandatanganan surat
    - Melakukan *hashing* terhadap M mengunakan fungsi *hash* SHA-3 sehingga dihasilkan sebuah *message digest* MD.
    - Melakukan enkripsi menggunakan kriptografi kunci publik *Eliptic Curve ElGamal* (ECEG) dengan kunci privat terhadap *message digest* MD. Hasil yang diperoleh disebut sebagai *signature* S.
    - Menyambungkan *message* M dengan *signature* S.
3. Menyelipkan *signature* ke dalam surat dengan menambahkannya sebagai salah satu atribut pada metadata.

Sementara itu, alur untuk proses verifikasi tanda tangan digital adalah sebagai berikut.

1. Mengekstrak konten pada *file* pdf menjadi struktur data *string*. *String* tersebut selanjutnya akan disebut sebagai SM (*signed message*).
2. Mengekstrak metadata dari *file* pdf untuk memperoleh *digital signature* S.
3. Melakukan verifikasi tanda tangan digital
    - Melakukan *hashing* pada *signed message* menggunakan fungsi *hash* SHA-3 sehingga dihasilkan sebuah *message digest* MD1.
    - Melakukan dekripsi terhadap *signature* S menggunakan algoritma kriptografi kunci publik *Eliptic Curve ElGamal* dengan kunci publik. Hasil dari dekripsi ini adalah *message digest* MD2.
    - Melakukan perbandingan antara *message digest* MD1 dengan *message digest* MD2. Apabila hasilnya sama, maka tanda tangan digital dinyatakan valid dan sah. Sedangkan, jika tidak, berarti tanda tangan digital tidak sah dan sudah terjadi perubahan pada konten surat yang dikirim.

B. Implementasi

Dalam implementasinya, akan dibuat 4 buah kelas, yaitu ElipticCurve, ECCElGamal, DigitalSignature, dan Steganography. Berikut adalah penjelasan dari masing-masing kelas.

1. Kelas Eliptic Curve

   Kelas ini memiliki *methods* untuk melakukan operasi-operasi dasar pada *eliptic curve*, seperti penjumlahan titik dan perkalian titik dengan konstanta. Berikut adalah *pseudocode* untuk fungsi perkalian dan penjumlahan titik.

```
function addPoint(P1: tuple, P2: tuple, a:
integer, modulo: integer) → tuple:
    x1, y1 ← P1[0], P1[1]
    x2, y2 ← P2[0], P2[1]
    if (x1 = x2) and (y1 = y2) then
        m ← ((3*x1*x2+a)*pow(2*y1,-1,modulo)) %
modulo
    else
        m ← ((y2-y1)*pow(x2-x1,-1,modulo)) %
modulo
    x3 ← (pow(m,2)-x1-x2) % modulo
    y3 ← (m*(x1-x3)-y1) % modulo
    → x3, y3
```

```
function applyDoubleAndAdd(P: tuple, k: integer)
→ tuple:
    k ← bin(k)[2:]
    Q ← P
    i traversal [1..length(k)]
        bit ← k[i]
        Q ← addPoint(Q, Q)
        if (bit = '1') then
            Q ← addPoint(Q, P)
    → Q
```

2. Kelas ECC ElGamal

   Kelas ini memiliki *methods* untuk membangkitkan pasangan kunci publik dan privat, serta melakukan enkripsi dan dekripsi dengan algoritma El Gamal. Berikut adalah *pseudocode* untuk fungsi pembangkitan pasangan kunci privat dan publik.

```
function generateKeys(curve: ElipticCurve,
base_point: tuple) → tuple:
    private_key ← random.getrandbits(256)
    public_key ← curve.applyDoubleAndAdd(curve,
base_point, private_key)
    → private_key, public_key
```

3. Kelas Digital Signature

   Kelas ini memiliki *methods* untuk melakukan penandatanganan dan verifikasi tanda tangan digital. Berikut adalah *pseudocode* untuk fungsi *hash*, *sign* dan *verify*.

```
function hash(message: string) → integer:
    message ← str(message)
    hashObj ← hashlib.sha3_256()
    hashObj.update(message.encode('utf-8'))
    hashInt ← int.from_bytes(hashObj.digest(),
byteorder='big')
    → hashInt
```

```
function sign(message: string, privateKey:
integer, ecElGamal: ECCElGamal) → tuple:
```

```
    publicKey ← getPublicKey(privateKey)

    r ← hash(hash(strToInt(message)) +
strToInt(message)) % ecElGamal.curve.modulo

    R ← ecElGamal.curve.applyDoubleAndAdd(
curve.base_point, r)

    h ← hash(R[0] + publicKey[0] +
strToInt(message)) % ecElGamal.curve.modulo

    s ← (r + h * privateKey)

    → R, s
```

```
function verify(message: string, publicKey:
tuple, signature: tuple, ecElGamal: ECCElGamal) →
boolean:

    R ← signature[0]

    s ← signature[1]

    h ← hash(R[0] + publicKey[0] +
strToInt(message)) % ecElGamal.curve.modulo

    P1 ← ecElGamal.curve.applyDoubleAndAdd(
ecElGamal.curve.base_point, s)

    P2 ← ecElGamal.curve.addPoint(R,
ecElGamal.curve.applyDoubleAndAdd(publicKey, h))

    if (P1[0] = P2[0]) and (P1[1] = P2[1]) then

        → True

    else

        → False
```

4. Kelas Steganography

Kelas ini memiliki *methods* untuk melakukan ekstraksi dan penyisipan isi surat serta metadata, terutama untuk *signature*-nya. Berikut adalah *pseudocode* untuk fungsi *read*, *write*, dan *extract signature*.

```
procedure read(reader: PdfReader):

    i traversal [0..length(reader.pages)]

        page ← reader.pages[i]

        content_in_str ← content_in_str +
page.extract_text()

        j traversal [0..length(content_in_str)]

            content_in_bits ← content_in_bits +
format(ord(x), ‘b’)

        output("File length:", length(
content_in_bits), "bits")
```

```
procedure write(reader: PdfReader, writer:
PdfWriter, output_file_path: string, sign:
string, pub_key: string):

    signature = sign

    public_key = pub_key

    i traversal [0..length(reader.pages)]

        page ← reader.pages[i]

        writer.add_page(page)

        writer.add_metadata(reader.metadata)

        writer.add_metadata({

            '/Signature': signature,

        })

        writer.add_metadata({

            '/PublicKey': public_key,

        })

        with open(output_file_path, 'wb') as
signed_file

            writer.write(signed_file)
```

```
function extractSignature(file_path: string) →
tuple:

    reader ← PyPDF2.PdfReader(file_path)

    meta ← reader.metadata

    signStr ← meta['/Signature']

    signTuple ← ast.literal_eval(signStr)

    → signTuple
```

Jadi, secara umum, alur eksekusinya adalah akan dilakukan pembacaan *file* terlebih dahulu menggunakan fungsi *read*. Kemudian dilakukan penandatanganan dengan fungsi *sign* yang didalamnya akan memanggil fungsi lainnya seperti *hash* serta perkalian dan penjumlahan titik kurva eliptik. Dan terakhir akan dilakukan penyisipan *signature* tersebut menggunakan fungsi *write*. Hal yang sama berlaku juga untuk proses verifikasi, tetapi tanpa pemanggilan *write*, karena operasi yang dilakukan hanya operasi pembacaan *file*.

IV. Pengujian dan Pembahasan

Berikut ini adalah hasil pengujian program tersebut berdasarkan kinerja keamanannya dan waktu eksekusinya.

A. Analisis keamanan dan integritas data

Berikut adalah contoh penerapan tanda tangan digital pada sebuah surat resmi, terkait penawaran *internship*.

Isi Surat:

Hasil *Signing*:

```
================================================
==== Digital Signature for Formal Letter (pdf) ====
================================================

 1. Sign a file
 2. Verify a file
 3. Exit
 Input your choice     : 1
 Input file name (pdf) : OfferLetter.pdf
 File length           : 12212 bits
 Input private key     : Internship
 Your private key      : 346770282333291906886000
 Your public key       : 979141854448866907533625304639576304313852998674607026677304505383816928503 84 ,
                         101415811239912452721586732310816807321552466995300330630189404717266460228420
 Your signature        :
 ( ( 896059248820430382456984439705254255503872832215847679067579345107749703208 56 ,
     183092946230218316453175074267222715763152715572691170955049022099272400187 84 ) ,
     173753284665491070261722021750488754588999483025109900496640492891995342145316090079761149018949708 62 )
 Time elapsed          :  0.024744749069213867 s
 The file has been signed successfully!
 Your signed file path :  signed-OfferLetter.pdf

 Metadata              :
{'/Producer': 'Microsoft® Word for Microsoft 365', '/MSIP_Label_38b525e5-f3da-4501-8f1e-526b6769fc56_Enabled': 'true', '/MSIP_Label_38b525e5-f3da-4501-8f1e-526b6769fc56_SetDate': '2023-04-28T02:18:19Z', '/MSIP_Label_38b525e5-f3da-4501-8f1e-526b6769fc56_Method': 'Standard', '/MSIP_Label_38b525e5-f3da-4501-8f1e-526b6769fc56_Name': 'defa4170-0d19-0005-0004-bc88714345d2', '/MSIP_Label_38b525e5-f3da-4501-8f1e-526b6769fc56_SiteId': 'db6e1183-4c65-405c-82ce-7cd53fa6e9dc', '/MSIP_Label_38b525e5-f3da-4501-8f1e-526b6769fc56_ActionId': '7bc4164c-c29a-48e4-b6f3-785e0cfed637', '/MSIP_Label_38b525e5-f3da-4501-8f1e-526b6769fc56_ContentBits': '0', '/Author': 'Amalina', '/Creator': 'Microsoft® Word for Microsoft 365', '/CreationDate': "D:20230522144635+07'00'", '/ModDate': "D:20230522144635+07'00'", '/Signature': '((896059248820430382456984439705254255503872832215847679067579345107749703208 56 , 183092946230218316453175074267222715763152715572691170955049022099272400187 84 ) , 173753284665491070261722021750488754588999483025109900496640492891995342145316090079761149018949708 62 )', '/PublicKey': '(979141854448866907533625304639576304313852998674607026677304505383816928503 84, 101415811239912452721586732310816807321552466995300330630189404717266460228420 84)'}

 The signature is valid. The file is verified!

 Time elapsed          :  0.034657955169677734 s
```

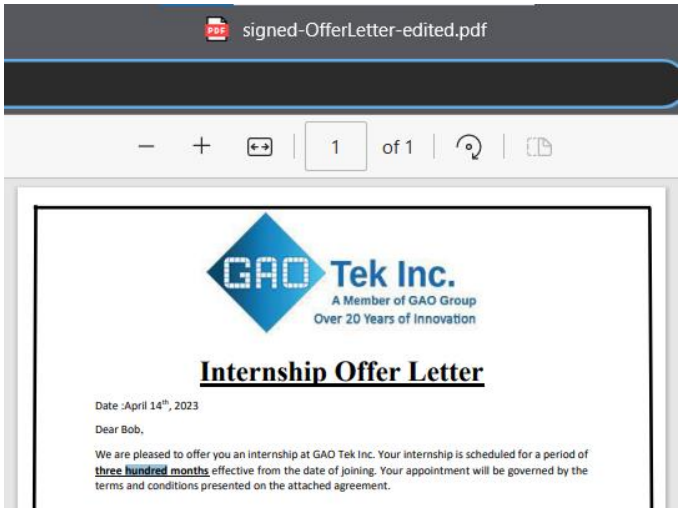
Berdasarkan hasil pengujian tersebut, dapat dilihat bahwa surat tersebut sudah berhasil ditandatangani dan tanda tangannya juga berhasil disimpan ke dalam metadata-nya. Selanjutnya, apabila dilakukan verifikasi, hasilnya adalah sebagai berikut.

Hasil Verifikasi Berhasil:

```
================================================
==== Digital Signature for Formal Letter (pdf) ====
================================================

 1. Sign a file
 2. Verify a file
 3. Exit
 Input your choice     : 2
 Input file name (pdf) : signed-OfferLetter.pdf
 File length           : 12212 bits
 Metadata              :
{'/Producer': 'Microsoft® Word for Microsoft 365', '/MSIP_Label_38b525e5-f3da-4501-8f1e-526b6769fc56_Enabled': 'true', '/MSIP_Label_38b525e5-f3da-4501-8f1e-526b6769fc56_SetDate': '2023-04-28T02:18:19Z', '/MSIP_Label_38b525e5-f3da-4501-8f1e-526b6769fc56_Method': 'Standard', '/MSIP_Label_38b525e5-f3da-4501-8f1e-526b6769fc56_Name': 'defa4170-0d19-0005-0004-bc88714345d2', '/MSIP_Label_38b525e5-f3da-4501-8f1e-526b6769fc56_SiteId': 'db6e1183-4c65-405c-82ce-7cd53fa6e9dc', '/MSIP_Label_38b525e5-f3da-4501-8f1e-526b6769fc56_ActionId': '7bc4164c-c29a-48e4-b6f3-785e0cfed637', '/MSIP_Label_38b525e5-f3da-4501-8f1e-526b6769fc56_ContentBits': '0', '/Author': 'Amalina', '/Creator': 'Microsoft® Word for Microsoft 365', '/CreationDate': "D:20230522144635+07'00'", '/ModDate': "D:20230522144635+07'00'", '/Signature': '((896059248820430382456984439705254255503872832215847679067579345107749703208 56 , 183092946230218316453175074267222715763152715572691170955049022099272400187 84 ) , 173753284665491070261722021750488754588999483025109900496640492891995342145316090079761149018949708 62 )', '/PublicKey': '(979141854448866907533625304639576304313852998674607026677304505383816928503 84, 101415811239912452721586732310816807321552466995300330630189404717266460228420 84)'}

 The signature is valid. The file is verified!

 Time elapsed          :  0.03465795516967773 4 s
```

Karena memang tidak ada perubahan konten surat yang terjadi, maka verifikasi memberikan hasil berhasil, ditunjukkan pada pesan "*The signature is valid. The file is verified!*". Namun, apabila dilakukan penambahan 1 kata saja, seperti contoh di bawah ini, proses verifikasi akan langsung memberikan hasil gagal, ditunjukkan oleh pesan "*The signature is invalid!*".

Isi Surat:



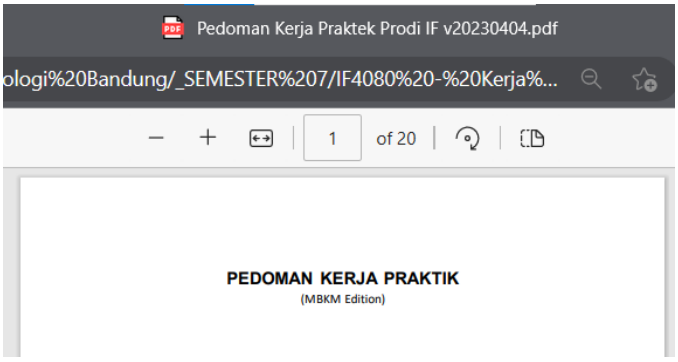
Hasil Verifikasi Gagal:

```
==================================================
==== Digital Signature for Formal Letter (pdf) ====
==================================================

 1. Sign a file
 2. Verify a file
 3. Exit
 Input your choice     : 2
 Input file name (pdf) : signed-OfferLetter-edited.pdf
 File length           : 12279 bits
 Metadata              :
{'/Producer': 'Microsoft® Word for Microsoft 365', '/MSIP_Label_38b525e5-f3da-4501-8f1e-526b6769fc56_Enabled': 'true', '/MSIP_Label_38b525e5-f3da-4501-8f1e-526b6769fc56_SetDate': '2023-04-28T02:18:19Z', '/MSIP_Label_38b525e5-f3da-4501-8f1e-526b6769fc56_Method': 'Standard', '/MSIP_Label_38b525e5-f3da-4501-8f1e-526b6769fc56_Name': 'defa4170-0d19-0005-0004-bc88714345d2', '/MSIP_Label_38b525e5-f3da-4501-8f1e-526b6769fc56_SiteId': 'db6e1183-4c65-405c-82ce-7cd53fa6e9dc', '/MSIP_Label_38b525e5-f3da-4501-8f1e-526b6769fc56_ActionId': '7bc4164c-c29a-48e4-b6f3-785e0cfed637', '/MSIP_Label_38b525e5-f3da-4501-8f1e-526b6769fc56_ContentBits': '0', '/Author': 'Amalina', '/Creator': 'Microsoft® Word for Microsoft 365', '/CreationDate': 'D:20230522145446+07'00'", '/ModDate': 'D:20230522145446+07'00'", '/Signature': '((8960592488204303824569844397052542555038728322158476790675793451077497032085 6,1830929462302183164531750742672227157631527155726911709550490220992724001878 4),1737532846654910702617220217504887545889994830251099004966404928919953421453160900797611490189497086 2)', '/PublicKey': '(979141854448866907533625304639576304313852998674607026677304505383816928503 84,1014158112399124527215867323108168073215524669953003306301894047172666402284 20)'}

 The signature is invalid!

 Time elapsed          :  0.03632092475891113 s
```

B. Analisis waktu eksekusi

Untuk waktu eksekusi sendiri, proses *signing* dan verifikasi hanya memerlukan waktu yang sedikit, yaitu rata-rata di bawah 0.05 detik sehingga penggunaan *digital signature* pada surat resmi masih bisa dikatakan efisien dan layak dicoba. Selain itu, umumnya *file* surat resmi hanya terdiri atas beberapa halaman sehingga tidak memiliki ukuran yang terlalu besar. Berikut adalah contoh waktu eksekusi untuk *file* dengan 20 halaman.

Isi *File*:



Hasil *Signing*:

```
==================================================
==== Digital Signature for Formal Letter (pdf) ====
==================================================

 1. Sign a file
 2. Verify a file
 3. Exit
 Input your choice     : 1
 Input file name (pdf) : PedomanKP.pdf
 File length           : 250624 bits
 Input private key     :
 Your private key      : 92618910946905925582707694024007303118208526796683366829632979771068606662251
 Your public key       : 53557325583278004955528264062289473913460393251454166615210961384666430200068 ,
                         77735972062250409996230924039173411393008044025132764977485473941332155771103

 Your signature        :
 ( ( 643387783662107147896859366175076572907100722679192976353189273482650585060 10 ,
     62417189648304000697245583891321232442905756422024454726044235535739021577106 ) ,
     66663938494356962145771721163425074063731996387889094678094372399590590109120457962267795227421817191919895324291179368967064924340925123283768096210337 55 )
 Time elapsed          :  0.5850098133087158 s
 The file has been signed successfully!
 Your signed file path :  signed-PedomanKP.pdf

 Metadata              :
{'/Producer': 'Adobe PDF Library 23.1.125', '/Author': 'Tricya Widagdo', '/Company': 'Hewlett-Packard', '/ContentTypeId': '0x010100224164C7377F934BB8A348905512758C', '/CreationDate': 'D:20230405091829+07'00'", '/Creator': 'Acrobat PDFMaker 23 for Word', '/Keywords': '', '/MSIP_Label_38b525e5-f3da-4501-8f1e-526b6769fc56_ActionId': 'b70f534f-c66b-4664-8c2a-cca3395cb121', '/MSIP_Label_38b525e5-f3da-4501-8f1e-526b6769fc56_ContentBits': '0', '/MSIP_Label_38b525e5-f3da-4501-8f1e-526b6769fc56_Enabled': 'true', '/MSIP_Label_38b525e5-f3da-4501-8f1e-526b6769fc56_Method': 'Standard', '/MSIP_Label_38b525e5-f3da-4501-8f1e-526b6769fc56_Name': 'defa4170-0d19-0005-0004-bc88714345d2', '/MSIP_Label_38b525e5-f3da-4501-8f1e-526b6769fc56_SetDate': '2023-04-03T03:27:55Z', '/MSIP_Label_38b525e5-f3da-4501-8f1e-526b6769fc56_SiteId': 'db6e1183-4c65-405c-82ce-7cd53fa6e9dc', '/MediaServiceImageTags': '', '/Mendeley Citation Style_1': 'http://www.zotero.org/styles/chicago-author-date', '/Mendeley Document_1': 'True', '/Mendeley Unique User Id_1': '326f2146-bfd8-34bd-9edb-d52cc148f65a', '/ModDate': 'D:20230405091836+07'00'", '/SourceModified': 'D:20230405021818', '/Subject': '', '/TaxCatchAll': '', '/Title': 'Pedoman Kerja Praktek', '/1cf76f155ced4ddcb4097134ff3c332f': '', '/Signature': '((643387783662107147896859366175076572907100722679192976353189273482650585060 10,62417189648304000697245583891321232442905756422024454726044235535739021577106), 66663938494356962145771721163425074063731996387889094678094372399590590109120457962267795227421817191919895324291179368967064924340925123283768096210337 55)', '/PublicKey': '(535573255832780049555282640622894739134603932514541666152109613846664302000 68, 77735972062250409996230924039173411393008044025132764977485473941332155771103)'}

 The signature is valid. The file is verified!

 Time elapsed          :  0.20508599281311035 s
```

Hasil Verifikasi:

Dapat dilihat, hasil eksekusi masih berada pada batas waktu yang wajar, yaitu 0.5 detik untuk *signing* dan 0.2 detik untuk verifikasi. Jadi, secara umum, waktu eksekusi bukan menjadi permasalahan yang harus terlalu diperhatikan.

V. KESIMPULAN DAN SARAN

Berdasarkan hasil pengujian dan pembahasan pada subbab sebelumnya, dapat disimpulkan penerapan tanda tangan digital pada *file* surat dapat menjaga intergritas data dengan baik dan dapat dilaksanakan dalam waktu eksekusi yang cepat dan efisien. Saran untuk pengembangan selanjutnya adalah dengan menerapkan algoritma-algoritma kriptografi yang mungkin dapat meningkatkan efisiensi dan lebih mempercepat waktu eksekusi. Selain itu, mungkin penerapan tanda tangan digital ini tidak hanya diterapkan pada surat saja, melainkan bisa diterapkan pada hal lain yang mementingkan integritas pada datanya.

UCAPAN TERIMA KASIH

Pertama-tama, penulis ingin memanjatkan puji dan syukur kepada Tuhan Yang Maha Esa sehingga penulis bisa menyelesaikan pembuatan makalah ini. Penulis juga ingin mengucapkan terima kasih yang sebesar-besarnya kepada Bapak Dr. Ir. Rinaldi Munir, M.T. selaku dosen mata kuliah IF4020 Kriptografi atas ilmu dan bimbingan yang telah diberikan selama pembelajaran di kelas. Penulis juga ingin meminta maaf apabila terdapat kekurangan dalam penulisan makalah ini. Semoga isi dari makalah ini dapat bermanfaat dan bisa menambah wawasan pembaca.

REFERENSI

[1] Munir, R. (2023). 01-Pengantar Kriptografi [Presentasi PowerPoint]. Diakses dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2022-2023/01-Pengantar-Kriptografi-(2023).pdf

[2] Munir, R. (2023). 08-Steganografi (Bagian 1) [Presentasi PowerPoint]. Diakses dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2022-2023/08-Steganografi-Bagian1-2023.pdf

[3] Munir, R. (2023). 20-Algoritma ElGamal [Presentasi PowerPoint]. Diakses dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2022-2023/20-Algoritma-Elgamal-2023.pdf

[4] Munir, R. (2023). Elliptic Curve Cryptography (ECC) - Bagian 1 [Presentasi PowerPoint]. Diakses dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2022-2023/23-ECC-Bagian1-2023.pdf

[5] Munir, R. (2023). Elliptic Curve Cryptography (ECC) - Bagian 2 [Presentasi PowerPoint]. Diakses dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2022-2023/24-ECC-Bagian2-2023

[6] Munir, R. (2023). Fungsi Hash [Presentasi PowerPoint]. Diakses dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2022-2023/25-Fungsi-hash-2023.pdf

[7] Munir, R. (2023). Fungsi Hash SHA-3 (Keccak) [Presentasi PowerPoint]. Diakses dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2022-2023/27-SHA-3-2023.pdf

[8] Munir, R. (2023). Tanda-tangan Digital (Digital Signature) [Presentasi PowerPoint]. Diakses dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2022-2023/28-Tanda-tangan-digital-2023.pdf

[9] Umam. (2022, 24 Mei). Pengertian Surat Resmi: Tujuan, Fungsi, Ciri, Struktur, Dan Contohnya. Gramedia Literasi. Diakses dari https://www.gramedia.com/literasi/pengertian-surat-resmi/