Penerapan digital signature pada image file untuk owner authentication

Aqil Abdul Aziz Syafiq - 13518002
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): 13518002@std.stei.itb.ac.id

***Abstract*—Dilakukan digital signing pada file image untuk menandakan pemilik sebuah file agar pembuat asli dari suatu image akan tertanda pada file tersebut selama tidak ada tampering atau editing pada image**

Keywords—digital signature, image, pemilik

I. Pendahuluan

Pada zaman ini, sangat mudah bagi seorang seniman untuk membagikan karya digital mereka pada sosial media, ini mendorong generasi muda untuk berkarya dan memproduksi karya karya yang lebih baik untuk dibagikan kepada masyarakat, ini juga baik untuk masyarakat karena menjadi salah satu sumber hiburan dan hobi yang tidak jarang untuk mencari dan menikmati karya digital para seniman.

Namun dengan mudahnya mendapatkan ketenaran dengan membagikan karya, banyak yang menginginkan ketenaran tersebut tanpa kemampuan yang memadai, mereka mencuri karya orang lain dan mengaku karya-karya tersebut adalah milik mereka.

Penelitian ini dilakukan untuk mengexplore kemungkinan menerapkan digital signing untuk menyisipkan tanda kepemilikan seniman akan suatu karya seni digital.

II. Landasan Teori

1. Kriptografi

Kriptografi merupakan aplikasi dan studi tentang teknik pengamanan komunikasi maupun data terhadap pihak ketiga. Alur umum kriptografi adalah dilakukannya enkripsi menggunakan suatu kunci terhadap sebuah plainteks, untuk menghasilkan sebuah cipherteks, dimana untuk mendekripsi cipherteks tersebut kembali menjadi plainteks, diperlukan kunci yang bisa sama atau tidak dengan kunci yang digunakan untuk enkripsi.

2. Kriptografi Asimetrik

Kriptografi asimetrik merupakan sistem kriptografi yang menggunakan pasangan kunci yang terdiri dari kunci publik dan kunci privat. Pembangkitan dan pemakaian kunci bervariasi tergantung pada algoritma enkripsinya, namun umumnya, kunci publik digunakan untuk enkripsi, sedangkan kunci privat digunakan untuk dekripsi.

Cara kerja kriptografi asimetri adalah, kunci publik dari pasangan kunci yang sudah dibangkitkan digunakan untuk mengenkripsi suatu message, dan hanya kunci privat yang merupakan pasangan kunci publik tersebut yang dapat mendekripsi kembali cipherteks yang terbuat. Ini adalah maksud dari asimetri pada nama, dimana kunci yang digunakan untuk enkripsi dan dekripsi berbeda, sehingga terjadi asimetri.

Seperti namanya, kunci publik adalah kunci yang diketahui oleh publik dan dapat digunakan oleh siapapun yang ingin mengirim pesan kepada sang pemilik kunci publik tersebut. Namun hanyalah sang pemilik kunci yang mengetahui kunci privatnya, dengan ini, siapapun yang ingin mengirim pesan cukup menggunakan kunci publik untuk mengenkripsi pesan mereka sebelum mengirimkannya ke pemilik kunci, dimana pemilik kunci akan mendekripsi cipherteks dengan kunci privat yang dipegang.

Kelebihannya dibanding kriptografi klasik adalah, dengan adanya kunci publik dan privat, tidak ada resiko bocornya key saat informasi key digerakkan dari satu pihak ke pihak lain, karena dengan kriptografi asimetrik hanyalah kunci publik yang perlu diberikan ke pengirim pesan, dan kunci publik tidak dapat digunakan untuk mendekripsi pesan.

3. RSA

RSA merupakan kependekan dari Rivest-Shamir-Adleman, RSA adalah salah satu sistem atau algoritma kriptografi asimetrik pertama yang pernah ada, yang dibuat oleh nama nama yang ada pada kepanjangan nama algoritma itu sendiri.

RSA memiliki 4 langkah dalam operasinya, yaitu key generation, key distribution, enkripsi, dan dekripsi.

Key generation pada RSA adalah sebagai berikut :

1. Pilih 2 bilangan prima berbeda p dan q secara random
2. Hitung n = pq
3. Hitung $\lambda(n) = KPK(\lambda(p),\lambda(q))$, dimana $\lambda(n)$ adalah n-1 jika n bilangan prima.
4. Cari e sehingga $1 < e < \lambda(n)$ and $fpb(e, \lambda(n)) = 1$
5. Hitung d dengan d as $d \equiv e-1 \pmod{\lambda(n)}$
6. n,e merupakan public key, n,d merupakan private key

Key Distribution merupakan aksi membagikan kunci publik kepada orang yang ingin mengirimkan pesan, dimana kunci publik akan digunakan untuk mengenkripsi pesan yang ingin dienkripsi. Kunci publik hanya dapat digunakan untuk mengenkripsi, sehingga kebocoran kunci publik tidak akan menjadi masalah. Kunci privat disimpan oleh pemilik untuk mendekripsi cipherteks hasil enkripsi pesan orang yang menggunakan kunci publiknya.

Untuk enkripsi, pengirim pesan harus mengubah pesannya ke dalam bentuk integer, lalu mengaplikasikan rumus ini untuk mendapatkan cipherteksnya.

$$c \equiv m^e \pmod{n}$$

Untuk dekripsi, diaplikasikan rumus ini kepada cipherteks.

$$c^d \equiv (m^e)^d \equiv m \pmod{n}$$

4. SHA256

SHA-2 merupakan sebuah kumpulan fungsi hash kriptografik yang didesain oleh NSA (National Security Agency) pada tahun 2001. SHA2 dibuat menggunakan konstruksi merkle-damgard. 256 pada SHA256 menandakan hash value (digests) pada fungsi tersebut.

5. Steganografi

Steganografi merupakan teknik menyembunyikan suatu pesan ke dalam suatu objek, baik digital, fisik, maupun berupa pesan lain. Dalam konteks digital dan komputer, steganografi merupakan pesan tersembunyi pada file yang memiliki isi yang bisa diubah tanpa disadari indra manusia.

Steganografi memiliki beberapa kelebihan dibandingkan kriptografi biasa, salah satu kelebihan utamanya adalah sifatnya yang tersembunyi dan rahasia, tidak seperti kriptografi yang ibarat menaruh pesan ke dalam sebuah brankas, steganografi ibarat sebuah pesan yang ditulis dengan tinta khusus yang tidak bisa dilihat orang yang tidak tahu. Ini menjamin kerahasiaan pesan, dimana orang yang tidak berkepentingan, termasuk kurir atau pengirim pesan pun tidak akan mengetahui keberadaan pesan tersebut.

6. Digital Signature

Digital Signature merupakan sebuah teknik matematis yang digunakan untuk memverifikasi integritas suatu pesan, atau berbagai macam file lainnya.

Digital Signature bekerja dengan cara menyisipkan suatu Digital Sign pada objek target, Digital Sign ini dapat dibuat tergantung keinginan, bisa menggunakan isi dari file, atau bisa juga menggunakan nama pengarang pesan.

7. Invisible Watermark

Watermark merupakan sebuah pola khusus yang ditambahkan pada suatu gambar untuk menandakan kepemilikan atau sumber dari suatu gambar. Ini digunakan oleh kebanyakan seniman untuk menandai karya mereka, dimana mereka akan menambahkan watermark pada karya mereka sebelum dibagikan untuk konsumsi umum. Ini dilakukan untuk menghindari pencurian karya.

Invisible watermark merupakan watermark yang tidak terlihat secara kasat mata, ini dikarenakan watermarking dilakukan pada pixel level sehingga tidak bisa ditangkap oleh mata manusia.

III. Pembahasan

1. Penjelasan Implementasi

Pada penelitian ini, akan digunakan teknik digital signature dengan nama seniman sebagai objek yang dienkripsi dan dijadikan signature, dimana selanjutnya digital sign akan disisipkan ke dalam file image.

Algoritma kriptografi yang digunakan untuk mengubah nama seniman karya menjadi digital sign adalah algoritma RSA, dimana kemudian hasil enkripsi akan dihash menggunakan hash function SHA256.

Hasil dari RSA+SHA256 akan disisipkan ke dalam file image. Teknik penyisipan pesan adalah sebagai berikut.

1. Diambil suatu byte secara terurut dari pesan yang ingin disisipkan pada image.
2. Byte tersebut diubah ke dalam bentuk binary.
3. Sebuah byte akan disisipkan kepada 3 buah pixel RGB dari suatu gambar. Dimana 8 value pertama dari 3 pixel tersebut akan diubah sesuai dengan bit yang akan disisipkan, bit 1 akan mengubah value menjadi ganjil, sedangkan bit 0 akan mengubah value menjadi genap, value ke 9 digunakan untuk menandakan posisi pada pesan, genap berarti pesan belum berakhir, dan ganjil berarti pesan sudah selesai.
4. Setelah semua byte pesan disisipkan, ditambahkan byte 11111111 untuk menandakan akhir pesan.

Untuk verifikasi dan pengambilan pesan rahasia dari suatu file image, berikut adalah langkah-langkahnya.

1. Diambil setiap 3 pixel pada gambar sampai ditemukan kelompok 3 pixel yang value ke-9nya ganjil.
2. Diekstrak byte dari tiap 3 pixel, dimana 8 value pertama pixel diambil dan ditransformasi kembali menjadi byte, genap menjadi 0 dan ganjil menjadi 1.
3. Hasil ekstraksi ditranslasi kembali menjadi digital signature yang bisa dibandingkan dengan aplikasi RSA+SHA256 pada suatu nama input.
4. Jika sama, maka validasi berhasil, jika beda, berarti validasi tidak berhasil dikarenakan nama yang diinput tidak sesuai.

2. Kode Program

Kode program untuk penelitian diimplementasikan dalam bahasa python. Ini dilakukan karena pemahaman penulis yang baik terhadap bahasa tersebut dibandingkan dengan bahasa pemrograman lain.

Berikut merupakan beberapa bagian dari source code untuk algoritma RSA.

```
def generate_keys():
    done = False
    primes_file = open('primes.txt','r')
    primes = primes_file.readlines()

    while not done:
        r1 = random.randint(30,75)
        r2 = random.randint(30,75)

        p = int(primes[r1])
        q = int(primes[r2])

        if len(str(p*q)) % 2 == 0:
            done = True

    n = p * q
    toitent = (p-1) * (q-1)

    searching = True
    while (searching):
        e = random.randrange(2, toitent)
        if(gcd(e, toitent) == 1):
            searching = False

    # e = 79
    # toitent e
    found = False
    k = 1
    while not found:
        d = (1 + (k*toitent)) / e
        if (d % 1 != 0):
            k += 1
        else:
            found = True
            d = int(d)

        with open('key.pub', 'w') as f:
            f.write(f"{str(n)},{str(e)}")
            f.close()

        with open('key.pri', 'w') as f:
            f.write(f"{str(n)},{str(d)}")
            f.close()

    return n,e,d
```

Ini merupakan fungsi yang digunakan untuk generate key untuk RSA. File ‘primes.txt’ berisi semua bilangan prima dibawah 10000000. Pemilihan bilangan prima dilakukan secara acak.

```
def encrypt_number(text, keys=None):
    if not keys:
        n, e, d = generate_keys()
    else:
        with open('key.pub','r') as f:
            n,e = f.read().strip().split(',')
            n,e = int(n), int(e)
        with open('key.pri','r') as f:
            n,d = f.read().strip().split(',')
            n,d = int(n), int(d)

    res = int(text) ** d
    res = res % n
    res = hex(res)

    return res
```

Ini merupakan fungsi enkripsi RSA yang dipakai. Hasil merupakan integer dalam bentuk hexadecimal.

```
def sha256(message):
    k = initializer(K)
    h0, h1, h2, h3, h4, h5, h6, h7 = initializer(h_hex)
    chunks = preprocessMessage(message)
    for chunk in chunks:
        w = chunker(chunk, 32)
        for _ in range(48):
            w.append(32 * [0])
        for i in range(16, 64):
            s0 = XORXOR(rotr(w[i-15], 7), rotr(w[i-15], 18),
shr(w[i-15], 3) )
            s1 = XORXOR(rotr(w[i-2], 17), rotr(w[i-2], 19),
```

```
shr(w[i-2], 10))
            w[i] = add(add(add(w[i-16], s0), w[i-7]), s1)
        a = h0
        b = h1
        c = h2
        d = h3
        e = h4
        f = h5
        g = h6
        h = h7
        for j in range(64):
            S1 = XORXOR(rotr(e, 6), rotr(e, 11), rotr(e, 25) )
            ch = XOR(AND(e, f), AND(NOT(e), g))
            temp1 = add(add(add(add(h, S1), ch), k[j]), w[j])
            S0 = XORXOR(rotr(a, 2), rotr(a, 13), rotr(a, 22))
            m = XORXOR(AND(a, b), AND(a, c), AND(b, c))
            temp2 = add(S0, m)
            h = g
            g = f
            f = e
            e = add(d, temp1)
            d = c
            c = b
            b = a
            a = add(temp1, temp2)
        h0 = add(h0, a)
        h1 = add(h1, b)
        h2 = add(h2, c)
        h3 = add(h3, d)
        h4 = add(h4, e)
        h5 = add(h5, f)
        h6 = add(h6, g)
        h7 = add(h7, h)
    digest = ''
    for val in [h0, h1, h2, h3, h4, h5, h6, h7]:
        digest += b2Tob16(val)
    return digest
```

Berikut merupakan implementasi fungsi SHA256 dalam program python. Hasil return adalah sama dengan rsa, dalam bentuk hexadecimal.

```
def stegano_encode(message, imarr):
    cx, cy = 0,0
    dim = imarr.shape
    for i in range(len(message)):
        ch = message[i]
        # print(ch)
        if i == len(message)-1:
            end=True
        else:
            end=False
        imarr = placebit(imarr,ord(ch),(cx,cy),end)
        cy += 3
        if cy+3 > dim[1]:
            cx += 1
            cy =0
    return imarr
```

Ini merupakan algoritma yang digunakan untuk menyisipkan pesan pada suatu file gambar. Placebit merupakan fungsi untuk menaruh suatu byte pada posisi (cx,cy) dalam array image imarr.

```
def stegano_decode(imarr):
    arr = imarr.flatten()
    bits = []
    for i in range(0,len(arr),9):
        bits.append([0 if x % 2 == 0 else 1 for x in arr[i:i+8]])
#         print(arr[:9])
        if arr[i+8] % 2 == 0:
            # print(arr[i:i+8])
            # print("Tetot")
#             print(len(bits))
            # print(bits)
            break
    ret = [chr(numly(bit)) for bit in bits]
    # print(ret)
    return "".join(ret)
```

Ini merupakan algoritma yang digunakan untuk mengekstrak kembali pesan yang tersembunyi dalam suatu image array.

Main dari program yang menggunakan kedua tugas tersebut adalah seperti berikut :

```
filename = input("Masukkan Nama File : ")
author = input("Masukkan Nama Author : ")
print("Signing file")
digital_signed_file = digital_sign(author, filename)
print("Test part")
print(f"encrypted authorname = {str(encrypt(author, key='fromfile'))}")
print(f"encrypted non-author name = {str(encrypt("random non author name", key='fromfile'))}")
print(f"extracted sign from file = {stegano_decode(digital_signed_file)}")
```

Gambar 3.1 Source code untuk main program testing

Pada kode ini, digital_sign hanyalah kombinasi fungsi rsa, sha, dan stegano_encode, serta tambahan untuk melakukan save file hasil signing.

Pada kode tersebut dapat 2 bagian utama, signing dan verifying, dapat dilihat bahwa setelah dilakukan digital signing, dilakukan sebuah bentuk test, dimana nama author diencrypt, dan diberikan random string non-author encrypted, serta diberikan sign yang diekstrak dari file yang baru saja

disign, dari situ dapat dilihat apakah hasil ekstraksi sama dengan nama author dan nama non-author untuk dibandingkan sebagai bentuk verifikasi. Dilakukan pencetakan untuk dokumentasi.

3. *Pengujian*

Dilakukan pengujian pada sebuah file image berikut.

Gambar 3.2 20056.jpg

Berikut adalah hasil run program dengan input nama author ‘aqil’ dan input file image ‘20056.jpg’.

```
Masukkan Nama File : 20056.jpg
Masukkan Nama Author : aqil
Signing file
Signed file saved!
Test part
encrypted authorname = 0xe24f
encrypted non-author name = 0x343d
extracted sign from file = 0xe24f
```

Gambar 3.3 Hasil Running program untuk pengujian

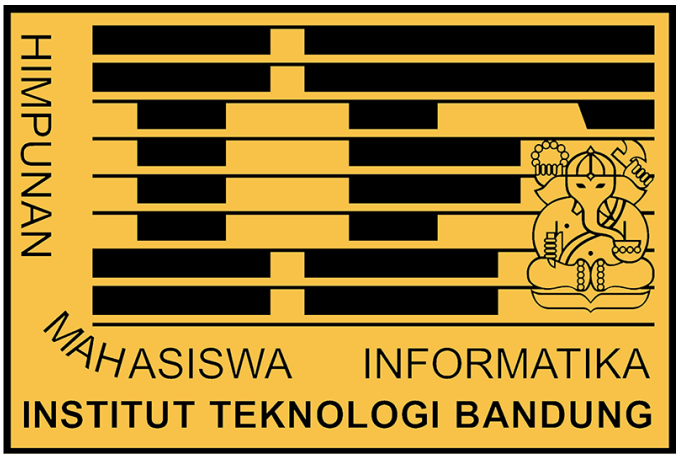
Bisa dilihat bahwa encrypted author name sama dengan extracted sign from file, dengan ini, hanya hasil enkripsi nama author yang akan menghasilkan hasil hash yang sama dengan sign yang diekstrak dari dalam file.

Berikut merupakan file 20056.jpg yang diberi sign.

Gambar 3.4 20056-signed.jpg

Dapat dilihat bahwa tidak ada perbedaan dalam image yang dapat dilihat oleh mata manusia, ini menandakan keberhasilan invisible watermarking.

Selanjutnya dilakukan testing pada sebuah image file berikut.



Gambar 3.5 HMIF.png

Selanjutnya, berikut adalah hasil program dengan input nama author ‘aqil’, dan input file image ‘HMIF.png’.

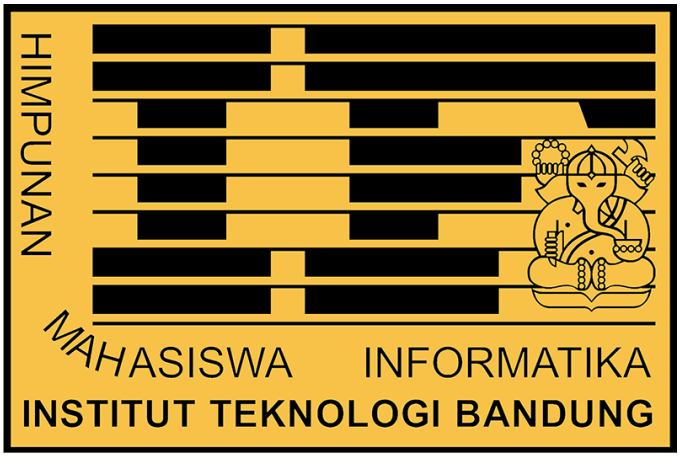
```
Masukkan Nama File : HMIF.png
Masukkan Nama Author : aqil
Signing file
Signed file saved!
Test part
encrypted authorname = 0xfdae
encrypted non-author name = 0x14408
extracted sign from file = 0xfdae
```

Gambar 3.6 Hasil running program

Bisa dilihat bahwa hasil kurang lebih sama dengan test pertama, ini berarti, selama bisa dilakukan konversi dari image menjadi array of RGB, tipe file tidak masalah, jpg dan png bisa, bahkan sudah dicoba menggunakan bmp, dan tidak ada masalah juga. Encrypted authorname pada file tersebut berbeda dengan test pertama dikarenakan pembangkitan ulang key yang dipakai pada algoritma RSA.

Terakhir, ini merupakan file HMIF-signed.png yang, sama seperti tes pertama, tidak mengalami perubahan yang kasat

mata.



Gambar 3.7 HMIF-signed.png

IV. Kesimpulan

Kombinasi digital signature dan steganografi merupakan cara yang efektif untuk melakukan invisible watermarking. Nama seniman yang sudah dienkripsi dan dihash disembunyikan dalam image, sehingga jika diperlukan bukti kepemilikan, dapat dilakukan digital verification dimana hanya nama author yang akan menghasilkan hasil yang sama dengan sign pada image.

Namun masih banyak kekurangan pada implementasi yang saya lakukan, ini karena penelitian ini berfokus pada ide dan kemungkinannya, dimana detail dan fungsionalitas lain dapat ditambahkan sesuai keperluan saja.

References

[1] Munir, Rinaldi. Slide kuliah terkait Steganografi

[2] M.Sreerama Murty, D.Veeraiah, A.Srinivas Rao, "Digital Signature and Watermarking for Image Authentication using Cryptography Analysis"

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, 18 Desember 2021

Aqil Abdul Aziz S