Implementasi Kriptografi Kunci Publik dengan Algoritma RSA dalam QR Code untuk Verifikasi Pemilik Tiket

Michel Vito Adinugroho - 18220066 (*Author*)
Program Studi Sistem dan Teknologi Informasi
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: vitoadinugroho2207@gmail.com

***Abstract*—Fenomena keberadaan calo tiket online yang melakukan kecurangan pembelian tiket secara masif menggunakan bot pada e-commerce menimbulkan kontroversi sosial di masyarakat. Tiket yang dijual kembali oleh calo dengan harga berkali-kali lipat dianggap merugikan dan menghilangkan hak calon pembeli tiket yang telah mengantre untuk membeli tiket secara legal. Untuk menangani permasalahan ini, diperlukan sebuah sistem keamanan tiket yang mengaitkan data diri pembeli dengan tiket fisik sehingga tiket tidak dapat diperjualbelikan kembali. Melalui makalah ini ini, penulis akan menjelaskan penggunaan algortima RSA pada QR Code tiket sebagai pengait data diri pembeli tiket dengan tiket fisik sekaligus mengenkripsi data pembeli tiket demi keamanan privasi dan menjaga tiket agar tidak dapat dipalsukan.**

Kata Kunci—tiket; QR Code, algoritma RSA; enkripsi; kriptografi

I. PENDAHULUAN

Dalam era digital, penggunaan teknologi internet dapat memberikan kemudahan jual beli dan perdagangan. Penggunaan internet untuk aktivitas transaksi jual beli dan bisnis dikenal dengan istilah *Electronic Commerce* (*e-commerce*) [1]. Layanan *e-commerce* merupakan salah satu bukti perkembangan era digital yang telah merubah cara manusia dalam melakukan jual beli, termasuk jual beli tiket. Saat pertama kali sistem tiket digunakan sebagai sebagai bukti pembayaran dan bukti untuk mendapatkan fasilitas dan jasa, sistem pembeliannya masih bersifat *on spot*, dimana calon pembeli harus melakukan pembelian langsung di lokasi acara atau penggunaan fasilitas berlangsung. Dengan keberadaan internet dan layanan *e-commerce*, kini pembelian tiket telah beralih menjadi *e-ticket* atau tiket *online*.

Perkembangan teknologi jual beli menggunakan internet bukanlah tanpa dampak negatif. Aspek keamanan dan keadilan jual beli seringkali dikorbankan dalam jual beli *online*. Pada konteks pembelian tiket secara *online*, terutama tiket *event* konser, fenomena calo tiket yang merebut tiket pembeli dan melakukan *reselling* tiket dengan harga tinggi menjadi salah satu permasalahan. Para calo tiket melakukan aksi perebutan tiket menggunakan banyak akun dan banyak perangkat sekaligus, dan seringkali menggunakan bot yang diprogram otomatis untuk melakukan *loading* atau akses berulang. Hal ini menimbulkan kecurangan antrean dimana para calo mendapatkan ‘prioritas tiket’ akibat penggunaan bot dan *multiple devices.*

Untuk mengatasi masalah perebutan tiket oleh calo, diperlukan sebuah mekanisme yang mengatur bahwa hanya pembeli tiketlah yang berhak menggunakan tiket. Mekanisme ini dapat dilakukan dengan mengaitkan data diri pembeli tiket di dalam tiket yang kemudian dapat diverifikasi oleh promotor sebagai pemilik tiket yang asli. Salah satu cara konservatif dalam menerapkan mekanisme ini adalah dengan menuliskan nama serta identitas pembeli pada tiket. Cara ini terlihat solutif, tetapi masih memiliki celah keamanan dimana tiket dapat dipalsukan. Para pemalsu tiket dapat dengan mudah meniru desain tiket dan memalsukan identitas sesuai permintaan calon pemilik tiket.

Oleh karena itu, diperlukan pengamanan tambahan dengan perahasiaan identitas tiket yang hanya bisa diketahui oleh promotor. Pengamanan tiket ini dapat dilakukan dengan menerapkan dan memodifikasi salah satu algoritma kriptografi kunci publik yakni algoritma RSA. Dengan modifikasi penggunaan algoritma RSA, identitas pemilik di dalam tiket akan dienkripsi (disamarkan) dengan kunci privat yang hanya dimiliki pembuat kunci dan menjadi sebuah *ciphertext* yang kemudian disembunyikan di dalam QR Code. *Ciphertext* ini dapat didekripsi (dikembalikan menjadi identitas semula) dengan sebuah kunci publik yang dibagikan kepada petugas pengecek tiket. Dengan cara ini, petugas pengecek tiket dapat dengan mudah memverifikasi pemilik tiket serta pemalsuan tiket menjadi sulit untuk dilakukan. Hal ini disebabkan pihak yang mengetahui kunci privat yang digunakan untuk membuat *ciphertext* identitas tiket hanyalah sang pembuat kunci. Pembuatan *ciphertext* asal-asalan oleh pemalsu tiket membuat *ciphertext* identitas tiket tidak dapat didekripsi sehingga tiket terdeteksi tidak sah.

II. METODE PENELITIAN

A. *Studi Literatur*

Untuk membuat makalah ini, penulis terlebih dahulu melakukan studi literatur dengan mencari informasi dari sumber digital berupa jurnal, paper, maupun materi perkuliahan mengenai tiket, sistem *e-ticketing*, QR Code, kriptografi kunci publik, serta algortima RSA".

B. *Eksperimen*

Usai melakukan studi literatur, penulis menerapkan pengetahuan yang didapat dan melakukan eksperimen dengan membuat program desktop app "Safe Ticket" untuk membuat sistem e-ticketing dengan modifikasi algoritma RSA dan penggunaan QR Code untuk 'menyembunyikan' hasil enkripsi. Program ini ditulis menggunakan bahasa Python dengan bantuan library Tkinter untuk membuat GUI program.

Terdapat tiga halaman yang dimiliki oleh desktop app "Safe Ticket" yakni *Key Generator*, *QR Generator* dan *Decryptor*. Pada halaman *Key Generator*, pengguna dapat membangkitkan sepasang kunci RSA (kunci publik dan privat). Pada halaman *QR Generator*, pengguna dapat membuat QR Code tiket dengan mengisi identitas pemilik tiket dan mengenkripsinya dengan kunci privat RSA. Pada halaman *Decryptor*, pengguna dapat mendekripsi chipertext pada QR Code mengunakan kunci publik RSA menjadi identitas asli pemilik tiket.

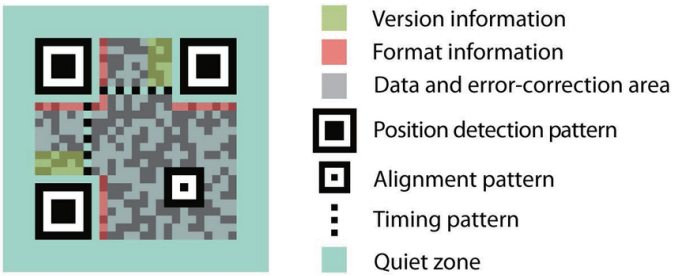
III. LANDASAN TEORI

A. *QR Code*

Quick Response Code atau yang biasa dikenal dengan sebutan QR Code adalah kode batang dua dimensi penyimpan data yang diciptakan oleh Denso Wave pada tahun 1994. Data yang terkandung dalam QR Code dapat dibaca dengan cara dipindai oleh perangkat digital seperti *smartphone*. QR Code merupakan pengembangan dari teknologi kode batang satu dimensi pada Barcode, dengan kemampuan untuk menyimpan informasi secara horizontal dan vertikal berkat bentuk dua dimensinya, dan secara otomatis memiliki kemampuan penyimpanan informasi yang jauh lebih besar dibandingkan Barcode. QR code menyimpan informasi sebagai rangkaian piksel berbentuk persegi dalam kisi.

QR Code standar yang beredar mampu menyimpan 7089 digit angka dan 4296 karakter alphanumeric. Karakter alphanumeric termasuk huruf, angka, dan simbol-simbol lain seperti simbol matematika juga dapat ditampung dalam QR Code. Beberapa jenis QR Code yang baru juga diperkenalkan untuk memenuhi kebutuhan yang lebih, seperti QR Mikro untuk memenuhi kebutuhan kode yang lebih kecil dan SQRC untuk kebutuhan kriptografi dengan *publik-private code*. QR Code telah disetujui sebagai standar kode internasional oleh International Organization for Standardization (ISO) pada tahun 2000 [2].

Berikut adalah stuktur penyusun bagian QR Code[3].



Gambar 1. Struktur QR Code [3]

1. *Position Detection Pattern*

   merupakan bagian yang berfungsi mengidentifikasi orientasi pemindaian QR Code sehingga QR Code mampu dipindai dari segala arah

2. *Allignment Pattern*

   merupakan kumpulan kotak yang berfungsi agar QR code dapat dipindai pada permukaan yang bergelombang

3. *Timing Pattern*

   merupakan pola titik yang membantu pemindai mengkonfigurasi jaringan data secara akurat dan menentukan besar matriks data yang dimuat pada QR Code

4. *Version Information*

   adalah pola yang menandakan salah satu versi yang digunakan pada QR Code dari total 40 jenis versi yang beredar,

5. *Format Information*

   adalah pola yang berisi informasi mengenai kapasitas koreksi kesalahan data. Saat ini terdapat 4 tingkat level koreksi kesalahan data yang tersedia dengan level tertinggi mampu mengoreksi kesalahan hingga 30%

6. *Data and Error Correction Area*

   adalah inti tempat semua data disimpan dalam tubuh QR Code. Di bagian ini terdapat error correction block yang mengoreksi kesalahan data yang timbul dari kerusakan pada badan QR Code,

7. *Quiet Zone*

   adalah zona putih yang mengeliligi stuktur QR Code, berfungsi memberikan jarak QR code dari objek lain sehingga QR Code dapat dipindai tanpa terpengaruh objek di sekitarnya.

Keamanan penggunaan QR Code dari segi kriptografi terletak pada keterkaitan struktur QR Code yang memampukan QR dideteksi secara akurat serta mampu melakukan *error correction*. Dengan kemampuan ini, sebuah QR Code dapat mempertahankan integritas data pada struktur QR Code yang telah rusak hingga maksimal 30% kerusakan fisik [4].

B. Algoritma RSA

RSA adalah sebuah alogritma kriptografi yang dibuat oleh tiga peneliti dari MIT (Massachussets Institute of Technology), yaitu Ronald Rivest, Adi Shamir, dan Leonard Adleman. Alogritma ini menerapkan kriptografi kunci-publik dimana kunci yang digunakan untuk melakukan enkripsi berdeda dengan kunci untuk melakukan dekripsi.

Pada algoritma kunci-publik, sebuah kunci publik disebarluaskan ke khalayak umum dan digunakan untuk mengenkripsi data. Sebaliknya, sebuah kunci privat akan dirahasiakan dan hanya diketahui oleh penerima data. Kunci privat akan digunakan penerima datauntuk mendekripsi data menjadi data semula.

Algoritma RSA dibuat dengan dasar keamanan sulitnya memfaktorkan bilangan bulat besar n menjadi faktor-faktor prima (p dan q), dalam hal ini $n = p * q$. Dibutuhkan waktu komputasi 4 miliar tahun untuk mencari faktor bilangan bulat 200 digit, dan membutuhkan waktu 10^{25} tahun untuk bilangan bulat 500 digit [5]. Algoritma RSA akan dianggap tetap aman selama komputer kuantum belum diciptakan.

Dalam menggunakan RSA, tedapat dua prosedur yang harus dilakukan. Prosedur pertama adalah pembangkitan sepasang kunci (kunci publik dan kunci privat). Prosedur kedua adalah prosedur enkripsi dan dekripsi data.

Prosedur pembangkitan kunci harus dijalankan terlebih dahulu karena proses enkripsi dan dekripsi data membutuhkan pasangan kunci RSA. Berikut adalah tahapan pembangkitan kunci pada algoritma RSA [6]:

1. Pilihlah dua buah bilangan prima rahasia yang akan disimpan dalam variabel p dan q. Pilihlah bilangan prima yang berdigit besar, misalnya 100-digit p dan 100-digit q sehingga menghasilkan n lebih dari 200-dikit. Makin besar digit n, makin kuat keamanan RSA.
2. Hitung n dengan persamaan berikut:

   $$n = p * q$$

   Nilai n bersifat tidak rahasia.
3. Hitung nilai *Totient Euler* dengan persamaan berikut:

   $$\phi(n) = (p - 1) * (q - 1)$$

   Nilai *totient euler* bersifat rahasia.
4. Pilihlah sebuah bilangan bulat e yang akan menjadi kunci publik untuk melakukan enkripsi. Bilangan e harus bersifat relatif prima dengan nilai *totient euler*, dalam kata lain nilai gdc(*totient euler*, e) $= 1$. Nilai e bersifat tidak rahasia
5. Hitung nilai d dari e dan n yang akan menjadi kunci privat untuk melakukan dekripsi. Nilai d dapat dicari dengan persamaan berikut:

   $$d \equiv e - 1 mod\ \phi(n)$$

   Kunci dekripsi d bersifat rahasia

Hasil dari algoritma pembangkitan kunci RSA di atas adalah kunci publik berupa pasangan (e, n) dan kunci privat berupa pasangan (d, n)

Prosedur yang dilakukan setelah membangkitkan kunci adalah proses enkripsi dan dekripsi data menggunakan pasangan kunci publik dan pasangan kunci privat. Pesan yang akan dikirim pengirim pesan akan dienkripsi dengan menggunakan kunci publik. Berikut adalah tahapan enkripsi pada algoritma RSA [5]:

1. Apabila pesan berukutan besar, pecah data menjadi blok-blok plainteks dengan ukuran yang lebih kecil yakni m_1 , m_2 , m_3 , … , m_i dengan syarat $0 \leq m_i < n - 1$
2. Hitung nilai cipherteks c_i untuk tiap blok plainteks m_i. Nilai c_i dihitung menggunakan kunci publik (e, n) dengan persamaan:

   $$c_i = m_i^e\ mod\ n$$

Selanjutnya ciphertext yang dikirim oleh pengirim akan diterima oleh penerima. Si penerima pesan akan melakukan dekripsi ciphertext tersebut dengan kunci privatnya agar data dapat dibaca. Berikut adalah tahapan dekripsi pada algoritma RSA [5]:

1. Pecah ciphertext data menjadi blok-blok cipherteks dengan ukuran yang lebih kecil yakni c_1 , c_2 , c_3 , … , c_i
2. Hitung nilai plainteks m_i dari tiap blok cipherteks c_i. Nilai m_i dihitung menggunakan kunci privat (d, n) dengan persamaan:

   $$m_i = c_i^e\ mod\ n$$

Algoritma RSA hanya menyediakan opsi enkripsi dan dekripsi untuk bilangan bulat.. Seringkali data yang ingin dikirim oleh pengirim adalah data string dan bukan data angka bulat. Untuk mengatasi hal tersebut, diperlukan konversi string menjadi integer dengan algoritma yang telah disepakati. Misalnya sebelum melakukan enkripsi dan dekripsi, data m dikodekan dengan A = 00, B = 01, …, Z = 25 (spasi diabaikan).

Pada pemrograman dengan bahasa Python, terdapat beberapa cara melakukan konversi data string menjadi data integer atau sebaliknya. Salah satu konversinya adalah menggunakan library binascii dengan konversi int(binascii.hexlify(string.encode("utf-8")), 16) untuk mengubah string menjadi integer, serta binascii.unhexlify(format(number, "x").encode("utf-8")).decode("utf-8") untuk mengubah integer menjadi string.

C. Sistem E-Ticketing

Seiring dengan perkembangan teknologi informasi, penggunaan tiket perlahan beralih menjadi *e-ticket*. *E-ticketing* atau *electronic ticketing* adalah suatu cara untuk mendokumentasikan penjualan (oleh penjual/penyedia) dan memperoleh tiket (oleh pembeli/pengguna) secara daring tanpa memerlukan paper *ticket*. Sistem e-ticketing menghasilkan produk berupa *e-ticket*.

Peberadaan *e-ticket* menyediakan kemudahan sarana pembayaran serta fleksibilitas dimana pengguna tiket tidak perlu khawatir akan kekurangan tiket fisik yang dapat hilang, tertinggal, dicuri, atau rusak. Sistem antre yang digunakan untuk membeli tiket juga beralih ke sistem antre secara daring. Calon pembeli yang terlebih dulu menekan tombol pembelian atau mengakses website memiliki hak untuk memilih tiket lebih dulu.

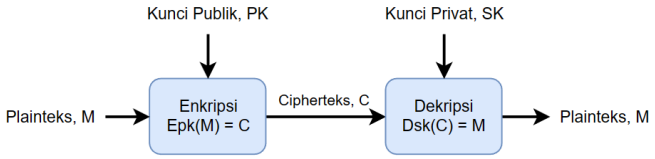
Meski demikian, antre daring pada sistem *e-ticketing* cenderung memiliki kelemahan karena tidak adanya verifikasi fisik atas pemilik antrean yang berakibat antrean tiket dapat disela atau direbut. Hal ini dapat dilakukan dengan melakukan pembelian tiket menggunakan banyak akun dan banyak perangkat sekaligus, serta menggunakan *bot* yang diprogram otomatis untuk melakukan *loading* atau akses berulang.

IV. Desain dan Rancangan

Dalam menerapkan pengetahuan pengamanan data menggunakan kriptografi, penulis melakukan desain sistem yang dapat menyelesaikan permasalahan calo tiket sebagaimana telah dibahas pada bagian pendahuluan. Desain pengamanan sistem *e-ticket* anti calo dan anti pemalsuan didasari ide bahwa tiket hanya dapat sah apabila informasi yang terkandung di dalam tiket sama dengan informasi pada KTP/kartu identitas lainnya yang dibawa pemilik tiket saat proses validasi tiket di lokasi *event*.

Desain sistem *e-ticket* ini dapat diterapkan melalui pengenkripsian identitas pemilik tiket dengan sebuah algoritma kriptografi. Data yang dienkripsi adalah gabungan nama pembeli tiket serta informasi pendukung lainnya seperti NIK atau NIM.

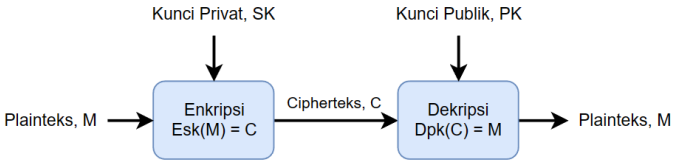
Pengamanan identitas tiket dengan enkripsi pertama-tama dilakukan dengan memodifikasi prosedur kriptografi kunci-publik pada algoritma RSA. Prosedur kriptografi kunci-publik yang umum digunakan adalah pengenkripsian data menggunakan kunci publik dan pendekripsian data menggunakan kunci privat.



Gambar 2. Prosedur umum algoritma kriptografi kunci-publik (sumber: dokumentasi pribadi)

Namun cara ini tidak dapat digunakan untuk mengamankan data *e-tiket* yang terdapat pada cipherteks, karena kunci yang digunakan untuk mengenkripsi informasi pemilik tiket haruslah kunci yang rahasia. Penggunaan kunci publik untuk mengenkripsi tiket menyebabkan tiket dapat dipalsukan dan dapat dibuat oleh siapa saja, karena kunci publik tersebar secara bebas.

Oleh karena itu, agar kriprografi kunci-publik dapat digunakan untuk mengamankan tiket, maka prosesnya dibalik: tiket dienkripsi dengan kunci privat milik promotor, lalu tiket didekripsi dengan kunci privat milik promotor. Promotor pada kasus ini diasumsikan adalah pihak yang melakukan pembuatan serta penjualan tiket. Dengan cara ini, identitas yang tercetak dalam cipherteks hanya bisa dibuat oleh pihak yang memiliki kunci privat, dalam hal ini sang promotor. Pembuatan tiket dengan cipherteks asal-asalan oleh pemalsu tiket menyebkan hasil dekripsi identitas pada ciphertext tidak sesuai dengan KTP/kartu identitas lainnya sehingga tiket dinyatakan palsu.

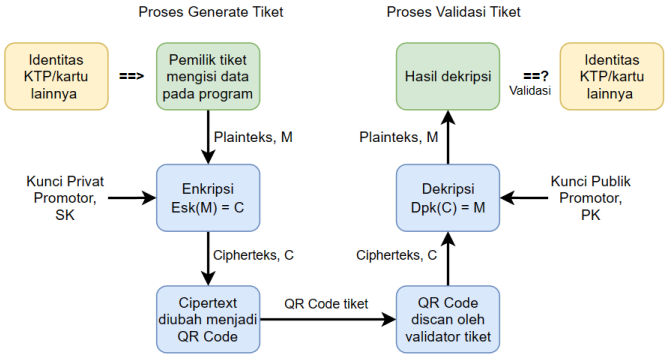


Gambar 3. Prosedur modifikasi algoritma kriptografi kunci-publik untuk sistem e-ticket (sumber: dokumentasi pribadi)

Usai mengenkripsi data pembeli tiket, cipherteks tiket hasil enkripsi akan di-*embed* ke dalam QR Code. Pemanfaatan QR Code pada implementasi sistem "Safe Ticket" adalah untuk 1) melakukan sedikit steganografi informasi, dalam hal ini penyembunyian hasil enkripsi identitas pemilik tiket, sehingga keberadaan ciphertext tidak terlalu mencolok dan 2) membuat proses verifikasi e-ticket menjadi lebih cepat dan efektif karena seorang validator tiket hanya perlu melakukan scanning untuk dapat menyalin seluruh informasi yang terkandung di dalam QR Code.

Hasil QR Code kemudian akan dipindai oleh validator tiket dan didekripsi dengan kunci publik promotor. Hasil dekripsi akan dibandingkan dengan identitas asli pemilik tiket sebagai validasi sah tidaknya tiket.

Keseluruhan proses dari sistem *e-ticket* anti calo dan anti pemalsuan dapat dilihat pada diagram berikut.



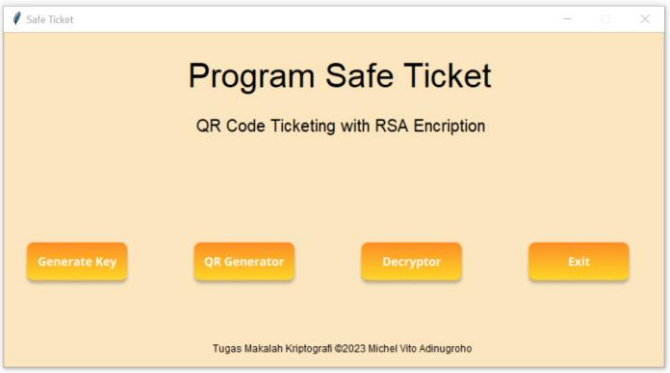
Gambar 4. Desain proses sistem e-ticket anti calo dan anti pemalsuan (sumber: dokumentasi pribadi)

V. Implementasi Prototipe

Pada bagian ini, penulis melakukan implementasi yang dapat menyelsaikan permasalahan calo tiket sebagaimana telah dibahas pada bagian pendahuluan. Hasil implementasi solutif dari masalah yang diangkat adalah program desktop app "Safe Ticket" untuk digunakan dalam sistem *e-ticketing* dengan modifikasi algoritma kriptografi kunci-publik.

Pada implementasi “Safe Ticket”, algoritma kriptografi kunci-publik yang dimodifikasi adalah RSA. Alasan penggunaan RSA adalah keamanannya yang tinggi dan sangat sulit dikriptanalisis. Digunakan pula QR Code untuk ‘menyembunyikan’ hasil enkripsi. Program ini ditulis menggunakan bahasa Python dengan bantuan library Tkinter untuk membuat GUI program serta library qrcode untuk melakukan *generate* QR Code.

Berikut adalah halaman utama dari program “Safe Ticket”.



Gambar 5. Halaman utama program safe ticket (sumber: dokumentasi pribadi)

Terdapat tiga modul yang dimiliki oleh desktop app “Safe Ticket” yakni modul *Key Generator*, modul *QR Generator* dan modul *Decryptor*. Fungsi tiap modul akan dibahas sebagai berikut.

A. Modul Key Generator

Modul pertama pada program adalah modul untuk membangkitkan sepasang kunci RSA (kunci publik dan privat). Berikut merupakan potongan kode program pembangkitan kunci RSA:

```
    def getkey():
        #membuat p dan q
        content_q = entry_q.get()
        content_p = entry_p.get()
        if len(content_q) == 0 or len(content_p) == 0:
            a = generate_key_pair(generate_prime_number(),
generate_prime_number())
        else:
            a      =      generate_key_pair(int(content_q),
int(content_p))


        entry_pub.delete(0, len(entry_pub.get()))
        entry_pub.insert(0, a[0][0])
        pub.set(a[0])


        entry_pri.delete(0, len(entry_pri.get()))
        entry_pri.insert(0, a[1][0])
        pri.set(a[1])

    #save kunci ke file
    def write():
        with open('public_key.pub', 'w') as f:
            f.write(pub.get())
        with open('private_key.pri', 'w') as f:
            f.write(pri.get())
        messagebox.showinfo("Success",   "Key   has   been
saved!")
```

Kode Program 1. Potongan kode prosedurpembangkitan kunci RSA

Pada potongan kode program di atas, prosedur getkey() akan membangkitkan sepasang kunci p dan q dari angka acak. Prosedur write() kemudian akan menyimpan sepasang kunci dengan format private key (d, n) dan kunci publik (e, n) dengan ekstensi .pub untuk kunci publik dan .pri untuk kunci privat.

B. Modul QR Generator

Pada modul *QR Generator*, dilakukan pengenkripsian atas identitas pemilik tiket menggunakan kunci privat RSA. Identitas yang telah dienkripsi kemudian disimpan di dalam QR Code. Berikut merupakan potongan kode program pembangkitan QR Code.

```
    def enkripsi():
        if len(privkey.get()) == 0:
            messagebox.showerror("Error", "Private Key is
empty")
        elif len(plain.get()) == 0:
            messagebox.showerror("Error", "Plain text is
empty")
        else:
            print("plain: ",plain.get())


            print("privkey: ",privkey.get())



            stringkey = str(privkey.get())
            real_key = stringtokey(stringkey)


            chi = encrypt(real_key, str2num(plain.get()))
            entry_cipher.delete(0,
len(entry_cipher.get()))
            entry_cipher.insert(0, chi)
            cipher.set(chi)


            print("cipher: ",cipher.get())
            print("cipher type: ", type(cipher.get()))
            print("chi type: ", type(chi))
```

```
def makeqr():
    myQR = qrcode.make(cipher.get())
    print(type(myQR))
 myQR.save("image_QR.png")
```

Kode Program 2. Potongan kode prosedur pembangkitan QR Code

Pada potongan kode program modul QR Generator, prosedur enkripsi() akan melakukan enkripsi terhadap identitas pemilik tiket menggunakan kunci privat RSA. Setelah itu, prosedur makeqr() akan membangkitkan QR Code dan melakukan penyimpanan QR Code. Pembangkitan kode QR pada prosedur ini memanfaatkan library qrcode. Berikut merupakan contoh kode QR yang berhasil dibangkitkan oleh prosedur makeqr():

Gambar 6. QR Code yang dibangkitkan dengan masukkan "ANTO SUBEJO 18220200" (sumber: dokumentasi pribadi)

C. *Modul Decryptor*

Pada modul *Decryptor*, dilakukan dekripsi atas chipertext pada QR Code mengunakan kunci publik RSA menjadi identitas asli pemilik tiket. Berikut merupakan potongan kode progam modul *decryptor*.

```
def dekripsi():
    if len(pubkey.get()) == 0:
        messagebox.showerror("Error", "Public Key is empty")
    elif len(cipher.get()) == 0:
        messagebox.showerror("Error", "Cipher text is empty")
    else:
        print("cipher: ",cipher.get())
        print("pubkey: ",pubkey.get())

        stringkey = str(pubkey.get())
        real_key = stringtokey(stringkey)

        pla = decrypt(real_key, int(cipher.get()))
        print("pla: ",pla)
        plaintext = num2str(pla)
        entry_plain2.delete(0, len(entry_plain2.get()))
        entry_plain2.insert(0, plaintext)
        plain2.set(plaintext)

        print("plain2: ",plain2.get())
        print("plain2 type: ", type(cipher.get()))
    print("plaintext type: ", type(plaintext))
```

Kode Program 3. Potongan kode prosedur pendekripsi ciphertext

Pada potongan kode program modul QR Generator, prosedur dekripsi() akan melakukan dekripsi terhadap cipherteks identitas pemilik tiket menggunakan kunci publik RSA. Apabila hasil dekripsi kemudian menghasilkan nama serta identitas yang sesuai dengan KTP/kartu identitas lainnya, maka tiket dinyatakan valid.

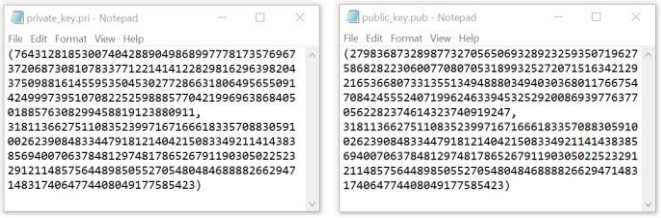
VI. PENGUJIAN

Setelah melakukan implementasi, dekstop app yang telah dibuat akan diuji untuk memastikan sistem "Safe Ticket" dapat digunakan sesuai fungsinya. Pengujian dilakukan menggunakan metode *blackbox* dengan menguji seluruh fungsional yang terdapat pada program. Proses pengujian dilakukan per-modul program.

A. *Pengujian Modul Key Generator*

Modul pertama yang akan diuji adalah modul key generator. Parameter keberhasilan dari pengujian ini adalah program mampu menghasilkan sepasang kunci publik dan kunci privat dan menyimpannya dalam file private_key.pri dan public_key.pub. Pengujian dapat dilihat pada gambar berikut:

Gambar 7. Hasil pengujian modul key generator (sumber: dokumentasi pribadi)

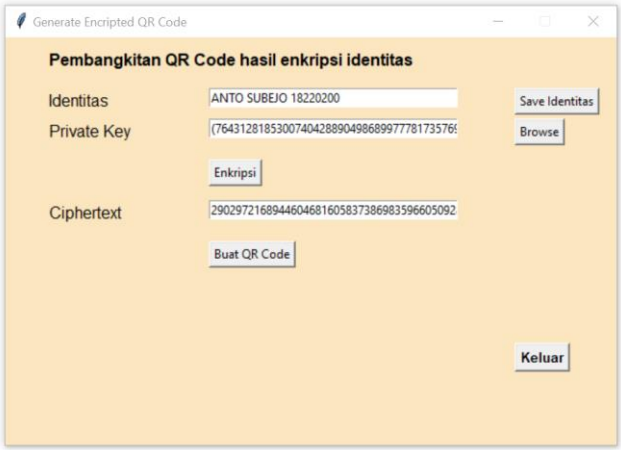


Gambar 8. File kunci hasil pengujian modul key generator (sumber: dokumentasi pribadi)

Berdasarkan hasil pengujian, program berhasil membangkitkan sepasang kunci publik dan kunci privat RSA dan menyimpannya ke dalam file private_key.pri dan public_key.pub

B. Pengujian Modul QR Generator

Modul kedua yang akan diuji adalah modul QR generator. Parameter keberhasilan dari pengujian ini adalah program mampu mengenkripsi data pemilik tiket dan membangkitkan serta menyimpan QR Code dari cipherteks hasil enkripsi. Pengujian akan menggunakan identitas "ANTO SUBEJO 182200200". Hasil pengujian dapat dilihat pada gambar berikut:



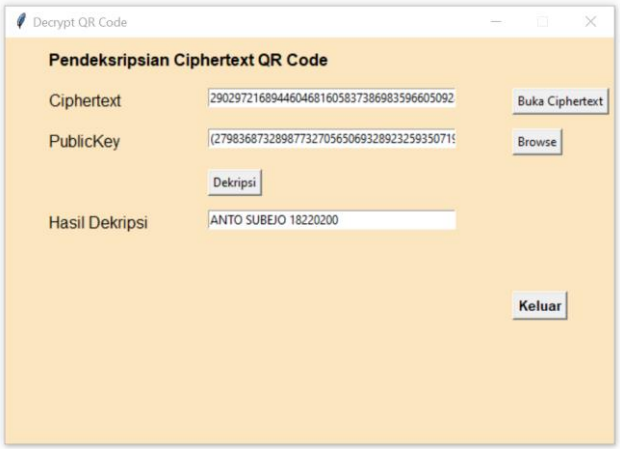
Gambar 9. Hasil pengujian modul QR *generator* (sumber: dokumentasi pribadi)

Gambar 10. Hasil QR Code yang dibangkitkan dengan masukkan "ANTO SUBEJO 18220200" (sumber: dokumentasi pribadi)

Berdasarkan hasil pengujian, program berhasil mengenkripsi data pemilik tiket dan membangkitkan serta menyimpan QR Code dari cipherteks hasil enkripsi.

C. Pengujian Modul Decryptor

Modul terakhir yang akan diuji adalah modul QR *decryptor*. Parameter keberhasilan dari pengujian ini adalah program mampu mendekripsi cipherteks menjadi plainteks awal yang berisi informasi identitas pemilik tiket, serta program menghasilkan error apabila cipherteks yang dimasukan telah diubah (QR Code palsu). Pengujian dapat dilihat pada gambar berikut:



Gambar 11. Hasil pengujian modul *dercyptor* dengan cipherteks asli (sumber: dokumentasi pribadi)

Gambar 11. Hasil pengujian modul *dercyptor* dengan cipherteks palsu (sumber: dokumentasi pribadi)

Berdasarkan hasil pengujian, program berhasil mendekripsi data pemilik tiket di dalam ciphertext asli menjadi plainteks yang merupakan identitas pemilik tiket, yaitu "ANTO SUBEJO 18220200". Selain itu, hasil pengujian dengan ciphertext palsu yang digit keduanya telah diubah (cipherteks awal "2902972...." diubah menjadi "2802972...) menunjukan bahwa program error dan ciphertext tidak dapat didekripsi. Hal ini menunjukan bahwa tiket telah dipalsukan.

VII. KESIMPULAN DAN SARAN

Untuk mengatasi masalah perebutan tiket oleh calo, diperlukan sebuah mekanisme yang mengatur bahwa hanya pembeli tiketlah yang berhak menggunakan tiket. Mekanisme ini dapat dilakukan dengan mengaitkan data diri pembeli tiket di dalam tiket yang kemudian dapat diverifikasi oleh promotor sebagai pemilik tiket yang asli. Mekanisme pengamanan tiket ini dapat dilakukan dengan mengenkripsi informasi pribadi pemilik tiket melalui memodifikasi salah satu algoritma kriptografi kunci publik yakni algoritma RSA. Hasil enkripsi kemudian di-embed ke dalam QR Code untuk menyisipkan sedikit steganografi dan mempermudah proses verifikasi tiket.

Seluruh mekanisme sistem pengamanan tiket tersebut telah berhasil diimplementasikan dalam prototipe program "Safe Ticket". Program "Safe Ticket" ini telah lolos semua pengujian. Saran pengembangan dari program ini adalah penambahan modul pembacaan QR Code menjadi ciperteks, sehingga QR Code dapat dipindai langsung melalui program dan tidak secara manual melalui *smartphone.*

SOURCE CODE & VIDEO LINK AT YOUTUBE

https://github.com/Vittskuy/SafeTicket (link youtube terdapat pada file readme pada github)

ACKNOWLEDGMENT

Terima kasih saya ucapkan kepada Tuhan Yang Maha Esa karena atas berkat-Nya yang berlimpah penulis dapat menyelesaikan makalah yang berjudul "Implementasi Kriptografi Kunci Publik dengan Algoritma RSA dalam QR Code untuk Verifikasi Pemilik Tiket" dengan tepat waktu. Terima kasih saya ucapkan juga kepada Bapak Dr. Ir. Rinaldi Munir, M.T selaku pengampu mata kuliah II4031 Kriptografi dan Koding yang telah membantu saya selama kegiatan belajar mengajar mata kuliah ini. Semoga hasil makalah ini dapat menjadi inspirasi bagi seluruh orang yang membaca untuk dapat mengimplementasikan serta mengembangkan keseluruhan bahasan makalah ini.

REFERENSI

[1] Rahmawati, F. 2018. Faktor-faktor yang Mempengaruhi Penerimaan Sistem Tiket Elektronik PT Transjarakta.

[2] Denso Wave, "History of QR Code." qrcode.com. https://www.qrcode.com/en/history/ (accessed May. 21, 2023)

[3] Gao, Zhongpai & Zhai, Guangtao & Hu, Chunjia. (2015). The Invisible QR Code. 10.1145/2733373.2806398.

[4] E. Nurdiansyah and I. Afrianto, "IMPLEMENTASI QRCODE SEBAGAI TIKET MASUK EVENT DENGAN MEMPERHITUNGKAN TINGKAT KOREKSI KESALAHAN", JATI, vol. 7, no. 2, pp. 25-44, Sep. 2017.

[5] R. Munir. 2023. Algoritma RSA

[6] Milanov, E. (2009). The RSA algorithm. RSA laboratories, 1-11.

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, 22 Mei 2023

Michel Vito Adinugroho (18220066)