Implementasi Algoritma ElGamal untuk Pengiriman Pesan Rahasia Melalui *Messenger*

Lukas Kurnia Jonathan 13517006
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13517006@std.stei.itb.ac.id

***Abstract*—Dengan pesatnya kemajuan teknologi yang terjadi, pertukaran informasi semakin mudah dilakukan dengan berbagai media. Salah satu media yang sering digunakan untuk berkomunikasi adalah *messenger*. Dengan adanya kemudahan ini tidak jarang membuat seseorang mengirimkan pesan yang bersifat rahasia ke orang lain melalui *messenger*. Keamanan pesan rahasia selama proses pengiriman pada umumnya sudah difasilitasi oleh aplikasi *messenger*. Akan tetapi, keamanan pesan rahasia seringkali belum terjamin ketika pesan telah selesai dikirimkan dan diterima oleh penerima. Salah satu cara untuk menjaga keamanan pesan ini adalah menerapkan kriptografi kunci publik yang asimetrik untuk mengenkripsi dan dekripsi pesan. Salah satu algoritma yang dapat digunakan untuk menerapkan hal ini adalah algoritma ElGamal.**

Keywords*—asimetrik, dekripsi, ElGamal, enkripsi, kriptografi kunci publik, *messenger

I. Pendahuluan

Pada era digital seperti ini perkembangan teknologi terjadi dengan begitu cepat dan pesat. Penggunaan teknologi dan media komunikasi secara *online* semakin meningkat. Hal ini juga didorong dengan munculnya penyakit pernapasan menular akut pada akhir tahun 2019 yang mengakibatkan adanya situasi darurat internasional pada tahun 2020 menurut *World Health Organization* (WHO). Berbagai sektor dan pekerjaan membatasi pertemuan secara tatap muka dan mendorong kegiatan untuk dilakukan secara *online*. Dengan banyaknya kegiatan yang dilakukan secara *online*, tidak jarang dilakukan pertukaran informasi dimulai dari informasi yang biasa hingga informasi yang bersifat rahasia menggunakan media komunikasi. Salah satu media komunikasi yang biasa dan mudah dipergunakan adalah *messenger*.

Salah satu aspek yang menjadi perhatian dalam pertukaran informasi adalah aspek keamanan dan kerahasiaan suatu pesan. Sayangnya, dengan kemajuan ilmu pengetahuan serta teknologi, penyadap dapat dengan mudah mengambil informasi pada pesan yang sedang ditransmisikan [1]. Oleh karena itu, diperlukan suatu cara yang aman untuk menjaga kerahasiaan informasi pada pesan yang ditransmisikan. Salah satu cara untuk menjaga aspek keamanan pesan yang dikirimkan adalah menggunakan kriptografi.

Kriptografi merupakan salah satu ilmu yang digunakan untuk menjaga keamanan pesan dengan menerapkan teknik-teknik matematika pada pesan yang dikirimkan [2]. Dengan kriptografi, pesan yang dikirimkan akan dilakukan enkripsi menjadi sebuah *cipher text* dan dilakukan dekripsi untuk mendapatkan kembali pesan utuh yang sebenarnya. Salah satu kategori dari enkripsi yang dilakukan oleh komputer adalah enkripsi asimetrik (*asymmetric encryption*) atau *public-key encryption*.

Salah satu referensi dari platform *messenger*, yaitu *WhatsApp Messenger* menerapkan *asymmetric encryption* pada saat pengiriman pesan dari seorang pengguna ke pengguna lainnya. Kunci publik disimpan pada server *messenger* sedangkan kunci privat disimpan pada *device* milik pengguna [3]. *Messenger* memastikan pesan yang dikirimkan dalam keadaan aman dengan *end to end encryption*. Akan tetapi, ada masalah keamanan lain selain proses pengiriman pesan dari pengguna satu ke pengguna lainnya yang dirasakan cukup penting, yaitu adalah keamanan pada saat menampilkan pesan dan membaca pesan.

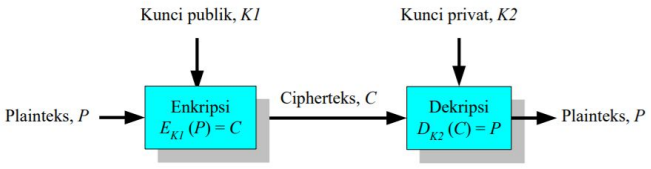
Ketika sebuah pesan rahasia dan penting diterima oleh pengguna lainnya, kebanyakan *messenger* akan langsung menampilkan *plaintext* pesan yang dikirimkan tanpa ada penjagaan kerahasiaan apapun. Sebagai contoh, ketika seorang karyawan mendapatkan pesan yang berisi informasi rahasia dan *confidential* bagi perusahaan dari manajernya, pesan tersebut akan terjamin keamanannya saat pengiriman melalui *messenger*. Namun, keamanan menjadi tidak terjamin ketika aplikasi *messenger* milik karyawan tersebut dibuka oleh orang lain. Hal ini dapat menimbulkan masalah keamanan jika *device* pengguna hilang, dilihat oleh orang lain, tidak terkunci, dan kelalaian lainnya. Meskipun seharusnya pesan rahasia dikirimkan tidak melalui *messenger*, namun hal ini masih saja terjadi karena metode komunikasi ini dianggap jauh lebih mudah dibandingkan metode lainnya.

Dengan demikian, penulis terpikirkan untuk berkontribusi dengan melakukan implementasi menggunakan algoritma kunci publik dan privat pada pesan yang ditampilkan pada sebuah *messenger*. Tentunya tanpa menghilangkan aspek keamanan yang sudah dimiliki *messenger* sebelumnya saat melakukan pengiriman pesan dari seorang pengguna ke pengguna lainnya. Salah satu algoritma yang dapat digunakan untuk membangkitkan kunci publik dan privat untuk melakukan enkripsi dan dekripsi pada pesan adalah algoritma ElGamal.

II. Dasar teori

A. Kriptografi Kunci Publik

Ide dari kunci publik muncul pada tahun 1976. Algoritma kunci publik dibuat dikarenakan terdapat permasalahan keamanan dengan kriptografi kunci simetri yang menggunakan kunci yang sama ketika proses enkripsi dan juga proses dekripsi. Ide dari algoritma kunci publik ini adalah memiliki 2 (dua) buah kunci berbeda untuk melakukan enkripsi dan dekripsi. Pada saat melakukan enkripsi digunakan kunci publik yang bersifat tidak rahasia dan dapat diketahui oleh orang banyak. Sedangkan kunci privat digunakan untuk melakukan dekripsi dan bersifat rahasia karena hanya diketahui oleh pengguna seorang diri saja.



Gambar 1. Proses kriptografi kunci publik [2].

Cara termudah untuk menggambarkan proses kriptografi kunci publik adalah dengan ilustrasi sebuah gembok. Misalkan terdapat dua orang yang akan berkirim pesan dimana A akan mengirimkan pesan kepada B. Mula-mula B akan mengirimkan kotak surat dengan gembok terbuka kepada A untuk diisi oleh pesan yang akan dikirimkan. A akan menuliskan pesannya, dimasukkan ke dalam kotak surat dan mengunci gembok pada kotak surat. Kotak surat dikirimkan kepada B dan hanya bisa dibuka oleh B karena kunci gembok hanya dimiliki oleh B. Kunci publik digambarkan dengan gembok terbuka pada kotak surat dan ketika dilakukan enkripsi gembok tersebut ditutup dengan kotak surat berisi pesan yang terenkripsi. Sedangkan kunci gembok menggambarkan kunci privat yang hanya dimiliki oleh pemilik gembok tersebut, dengan kata lain hanya pemegang kunci privat yang bisa membuka pesan yang dikirimkan.

Gambar 2. Ilustrasi kriptografi kunci publik [2].

Salah satu keuntungan yang diberikan oleh kriptografi kunci publik ini adalah tidak diperlukan pengiriman kunci privat seperti yang dilakukan pada kriptografi kunci simetri. Selain itu, kunci yang digunakan tidak perlu sering diubah dan dapat digunakan untuk jangka waktu yang panjang. Pembuatannya pun dilakukan dengan menggunakan operasi perpangkatan dan bilangan yang besar.

Pengaplikasian kriptografi kunci publik diantaranya adalah melakukan enkripsi dan dekripsi pesan, *digital signature*, dan pertukaran kunci (*key exchange*). Pengaplikasian yang akan lebih banyak dibahas pada penelitian ini adalah mengenai enkripsi dan dekripsi pada pesan yang dikirimkan serta diterima oleh pengguna.

B. Algoritma ElGamal

Algoritma ElGamal merupakan salah satu algoritma kriptografi kunci publik yang dapat digunakan untuk melakukan enkripsi, dekripsi, dan *digital signature*. Algoritma ini dibuat oleh Taher Elgamal (1985) pada makalah berjudul *"A public key cryptosystem and a signature scheme based on discrete algorithm"*. Seperti yang disampaikan pada judul makalah yang dibuat oleh Taher Elgamal, algoritma ElGamal dibuat dengan perhitungan logaritma diskrit yang rumit untuk meningkatkan aspek keamanan.

Properti yang terdapat pada algoritma ElGamal antara lain:

1. Bilangan prima, p (tidak rahasia)
2. Bilangan acak, g ($g < p$) (tidak rahasia)
3. Bilangan acak, x ($x < p$) (rahasia, kunci privat)
4. $y = g^x \bmod p$ (tidak rahasia, kunci publik)
5. m (plainteks) (rahasia)
6. a dan b (cipherteks) (tidak rahasia)

Terdapat 3 (tiga) buah prosedur pembangkitan kunci publik dan privat untuk algoritma ElGamal, antara lain sebagai berikut.

1. Melakukan pemilihan sembarang bilangan prima p
2. Melakukan pemilihan 2 (dua) bilangan acak, g dan x dengan syarat $g < p$ dan $1 \leq x \leq p - 2$
3. Melakukan perhitungan $y = g^x \bmod p$

Kunci publik didapatkan dari pasangan tripel y, g, dan p. Sedangkan kunci privat merupakan pasangan x dan p. Kunci publik dan privat ini yang akan digunakan untuk melakukan enkripsi serta dekripsi.

Prosedur tahapan melakukan enkripsi adalah sebagai berikut.

1. Pertama-tama dilakukan penyusunan terhadap plainteks pesan menjadi blok-blok m1, m2, …., mn yang berada di dalam selang [0, p-1]
2. Selanjutnya lakukan pemilihan bilangan acak k, dengan ketentuan $1 \leq k \leq p - 2$
3. Lakukan proses enkripsi untuk setiap blok m dengan rumus

$$a = g^k \bmod p \quad (1)$$

$$b = y^k m \bmod p \quad (2)$$

Pasangan a dan b yang dihasilkan merupakan pasangan cipherteks yang dihasilkan untuk satu buah blok m. Dari prosedur ini dapat dilihat bahwa ukuran cipherteks akan 2 (dua) kali lebih besar dari ukuran plainteks awal.

Selanjutnya untuk prosedur tahapan melakukan dekripsi pada sebuah cipherteks adalah sebagai berikut.

1. Pertama-tama ambil x dalam pasangan kunci privat untuk menghitung:

$$(a^x)^{-1} = a^{p-1-x} \bmod p \quad (3)$$

2. Untuk masing-masing blok, hitung plainteks m dengan persamaan berikut.

$$m = \frac{b}{a^x} \bmod p = b(a^x)^{-1} \bmod p \quad (4)$$

Hasil perhitungan dari masing-masing blok teks akan disatukan kembali hingga menjadi sebuah *plaintext* utuh yang memiliki informasi awal.

C. Messenger

Messenger adalah sebuah aplikasi *instant messaging* yang digunakan untuk mengirimkan pesan dari seseorang ke orang lainnya [4]. Pesan yang dikirimkan dapat berupa pesan teks, audio, video, dan dokumen. Beberapa contoh *messenger* adalah *Facebook Messenger* dan *WhatsApp Messenger* yang juga merupakan produk dari *facebook*.

Pengiriman pesan yang dilakukan oleh berbagai *messaging service* memiliki tipe enkripsi yang berbeda-beda. Tujuan dilakukan enkripsi terhadap pesan yang dikirimkan melalui *messaging service* adalah memastikan pesan tidak dapat dibaca oleh pihak yang tidak diinginkan jika terdapat intersepsi saat pengiriman pesan. Salah satu tipe enkripsi yang umum dilakukan adalah *end to end encryption*.

End to end encryption memastikan pesan yang dikirimkan hanya dapat dibaca oleh pengirim pesan dan penerima pesan yang ditujukan oleh pengirim pesan. Dengan *end to end encryption* bahkan *messaging service* sendiri tidak dapat membaca pesan yang dikirimkan [5]. Salah satu proses yang digunakan untuk *end to end encryption* adalah kriptografi kunci publik.

Sebagai contoh pada aplikasi WhatsApp, ketika pengguna melakukan instalasi terhadap aplikasi maka kunci publik dari pengguna tersebut akan disimpan dalam server aplikasi. Kunci privat yang dimiliki pengguna hanya akan disimpan secara rahasia pada perangkat yang digunakan oleh pengguna. Ketika seorang pengguna akan mengirimkan pesan kepada pengguna lainnya, maka dilakukan pengambilan kunci publik milik penerima pesan. Kunci publik yang diambil digunakan untuk melakukan enkripsi pesan dan mengirimkannya kepada penerima pesan. Penerima pesan akan menggunakan kunci privat miliknya untuk melakukan dekripsi pada pesan. Sesi yang digunakan pada kunci publik dan privat akan dibuat kembali ketika aplikasi di *install* ulang atau terjadi penggantian *device* [6].

III. Rancangan Solusi dan Implementasi

Dalam menyelesaikan permasalahan ini terdapat beberapa langkah yang akan dilakukan untuk dapat menghasilkan solusi, yaitu Deskripsi Umum Solusi, Rancangan Solusi, dan Implementasi Solusi.

A. Deskripsi Umum Solusi

Seperti yang sudah dijelaskan pada bagian pendahuluan, terdapat permasalahan keamanan pada sebuah aplikasi *messenger* ketika pesan telah sampai kepada penerima pesan. Terdapat kemungkinan suatu informasi baik pesan yang mengandung informasi rahasia maupun tidak rahasia untuk terlihat atau terbaca oleh pihak lain yang tidak berwenang. Salah satu penyebab hal ini terjadi adalah karena tidak adanya mekanisme penyembunyian pesan yang bersifat rahasia tersebut ketika sudah sampai kepada penerima pesan. Kasus yang mungkin terjadi adalah ketika *device* yang digunakan pengguna hilang atau terdapat beberapa orang yang menggunakan sebuah *device* yang sama dimana di dalamnya terdapat informasi dan pesan rahasia.

Salah satu solusi yang dapat ditawarkan untuk permasalahan ini adalah tetap menggunakan kemampuan *end to end encryption* yang sudah dimiliki oleh *messenger* untuk menjamin keamanan pesan ketika dikirimkan. Akan tetapi, ditambahkan *preprocessing* terhadap pesan yang dikirimkan terlebih dahulu dengan melakukan enkripsi pada pesan yang dikirimkan. Sehingga pesan yang dikirimkan adalah pesan yang sudah terenkripsi dan tidak diketahui makna sebenarnya kecuali dilakukan dekripsi terhadap pesan tersebut. Penerima pesan tidak akan merasa khawatir jika terdapat musibah atau hal yang tidak diinginkan terjadi kepada *device* yang digunakannya dikarenakan pesan yang ditampilkan adalah pesan yang sudah terenkripsi dan tidak diketahui makna sebenarnya.

B. Rancangan Solusi

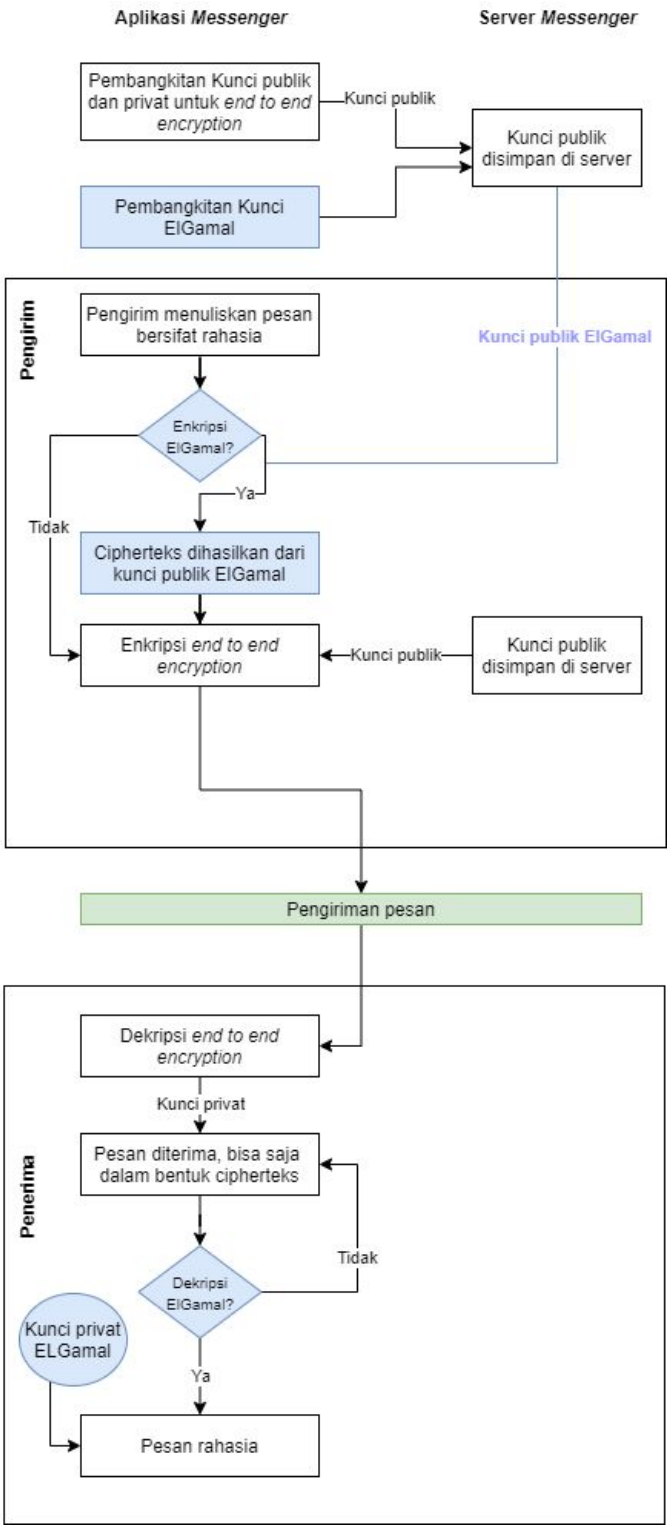
Solusi dirancang dengan menggunakan algoritma kunci publik ElGamal untuk membangkitkan kunci publik dan privat. Proses pengiriman pesan tetap sama seperti halnya aplikasi *messenger* mengirimkan pesan, namun terdapat tambahan *preprocessing* yang merupakan enkripsi pada pesan. Arsitektur solusi yang dirancang dapat dilihat pada Gambar 3.

Rancangan solusi yang dibuat pada Gambar 3 dilakukan dengan penambahan beberapa proses yang belum dilakukan sebelumnya pada aplikasi *messenger* biasa. Proses yang ditambahkan adalah proses berwarna biru pada Gambar 3 yang terdapat pada saat awal aplikasi, saat pengirim akan mengirimkan pesan, dan saat penerima menerima pesan. Proses yang bukan berwarna biru yang sudah ada pada aplikasi *messenger* tidak akan banyak dibahas pada bagian ini.

Alur pengerjaan solusi dimulai dengan membangkitkan kunci publik ElGamal yang disimpan pada server aplikasi bersamaan dengan mekanisme awal yang sudah dimiliki oleh *messenger* dalam penyimpanan kunci publik untuk *end to end encryption*. Kunci publik ELGamal ini akan digunakan untuk melakukan enkripsi pesan terlebih dahulu sebelum pesan dikirimkan kepada penerima. Hasil dari enkripsi menggunakan ELGamal adalah cipherteks yang menyembunyikan makna sebenarnya dari pesan rahasia yang dikirimkan oleh pengirim pesan.

Proses pengiriman pesan tetap dilakukan seperti halnya *messenger* bekerja. Namun, ketika pesan diterima oleh penerima pesan, yang ditampilkan pada aplikasi *messenger* adalah cipherteks yang tidak dapat dimengerti oleh orang lain.

Hal ini dapat menjadi solusi dan mempertahankan keamanan pesan meskipun *device* dari penerima pesan rahasia mengalami kehilangan atau digunakan oleh pihak yang tidak berwenang. Jika penerima ingin mengetahui isi dari pesan rahasia yang disampaikan oleh pengirim pesan, maka penerima pesan dapat melakukan dekripsi dengan kunci privat yang dimilikinya saat melakukan pembangkitan kunci ElGamal.



Gambar 3. Arsitektur rancangan solusi
Sumber: Dokumen penulis

C. Implementasi Solusi

Solusi yang diimplementasikan memiliki beberapa buah batasan implementasi agar penelitian lebih terfokus dengan topik permasalahan yang ada. Beberapa batasan implementasi yang dilakukan adalah sebagai berikut.

1. Sistem *messenger* dianggap sudah ada, sehingga implementasi hanya berfokus kepada proses enkripsi dan juga dekripsi menggunakan algoritma kunci publik ElGamal.
2. Implementasi untuk *messenger* hanya dilakukan pada level *frontend* aplikasi menggunakan *framework* ReactJS tanpa menggunakan *backend* aplikasi, yaitu server dari *messenger*.
3. Implementasi pembangkitan kunci ElGamal serta proses enkripsi dan dekripsi menggunakan bahasa pemrograman Python3.

Oleh karena itu, implementasi hanya dilakukan pada proses yang belum terdapat pada sistem aplikasi *messenger* secara umum, atau bagian yang berwarna biru pada Gambar 3 mengenai arsitektur rancangan solusi.

Proses pembangkitan kunci dilakukan sebagaimana dijelaskan pada dasar teori, yaitu kunci publik dengan properti (y, g, dan p) dan kunci privat dengan properti (x dan p)

```
def generate(self):
      # random int from 8 bit to  16 bit
      # BITS is setted to 16
      seed = random.randint(8,const.BITS)
      p = number.getPrime(seed)
      while p <= 256 or p > 2**const.BITS:
          p = number.getPrime(seed)

      g = random.randint(1,p-1)
      x = random.randint(1,p-2)
      y = pow(g,x,p)
      # Separate each properties with '|'
      pub = str(hex(y)) + SEPARATOR + str(hex(g))
+ SEPARATOR + str(hex(p))
      pri = str(hex(x)) + SEPARATOR + str(hex(p))

      # pub is public key string triple (y, g,
and p), saved as y|g|p
      # pri is private key string pair (x and p),
saved as x|p
      return pub, pri
```

Untuk memudahkan implementasi, bilangan prima p dipastikan lebih besar dari 256 (dua ratus lima puluh enam) dikarenakan pada saat melakukan enkripsi terdapat persyaratan blok pesan dalam selang [0, p-1]. Maka blok pesan yang digunakan hanya sebesar 1 character sehingga mengikuti nilai *extended ASCII code* yang memiliki nilai hingga 256. Dengan demikian syarat enkripsi pasti akan terpenuhi.

Proses enkripsi dilakukan dengan memasukkan teks serta kunci publik yang dimiliki oleh penerima pesan. Proses

enkripsi menggunakan kunci publik serta persamaan (1) dan (2) yang sudah dijelaskan pada bagian teori dasar. Hasil enkripsi adalah cipherteks dengan ukuran pesan yang 2 (dua) kali lebih besar dari pesan semula.

```
def encrypt(self, text, key):
    # Remove '|' separator to obtain y,g, and p
value
    y, g, p = process_keys(key)
    # Create a block for each character
    # max size of block = 256 (1 char with ASCII
256)
    m = [ord(c) for c in text]

    k = random.randint(1,p-2)
    arr_integer = []
    for block in m:
        # For each block create a and b tuple
        a = pow(g,k,p)
        b = pow(y,k)*block % p

        arr_integer.append(a)
        arr_integer.append(b)

    # return cipher with 2x size
    return arr_integer
```

Proses dekripsi dilakukan dengan memasukkan cipherteks yang berukuran 2 (dua) kali lebih besar dan kunci privat yang dimiliki oleh penerima pesan. Proses dekripsi menggunakan persamaan (3) dan (4) yang sudah dijelaskan pada bagian teori dasar.

```
def decrypt(self, text, key):
    # Remove '|' separator to obtain x and p value
    x, p = process_keys(key,False)

    decipher = ''
    for i in range(0,len(text), 2):
        a_invers = pow(text[i],p-1-x, p)
        m = a_invers*text[i+1] % p
        decipher += chr(m)

    # return a deciphered text, plainteks
    return decipher
```

Pada bagian implementasi *messenger* mekanisme pengiriman pesan dengan *end to end encryption* diasumsikan sudah ada, sehingga fokus dari penelitian ini hanya terdapat pada isi pesan yang dienkripsi ataupun dilakukan dekripsi. Pembangunan antarmuka aplikasi dibuat semirip mungkin dengan *messenger* yang ada. Berikut adalah hasil implementasi antarmuka untuk *messenger* yang dibangun.

Gambar 4. Antarmuka *messenger* yang diimplementasikan

IV. Analisis dan Pengujian

Mula-mula dilakukan pembangkitan kunci publik dan privat ElGamal sebagai berikut.

Bilangan prima random (p) = 263 Bilangan acak, g = 41 Bilangan acak, x = 178 Hasil perhitungan, y = 66
Kunci publik ElGamal yang dihasilkan: (dalam hexadecimal) 0x42|0x29|0x107 Kunci privat ElGamal yang dihasilkan: (dalam hexadecimal) 0xb2|0x107

Jika bilangan prima (p) diganti dengan bilangan yang cukup dekat dengan 263 yaitu 257, seluruh kunci publik dan privat berubah. Hal ini menunjukkan bahwa perhitungan logaritma diskrit sulit untuk dipecahkan.

Bilangan prima random (p) = **257** Bilangan acak, g = 217 Bilangan acak, x = 50 Hasil perhitungan, y = 228
Kunci publik ElGamal yang dihasilkan: (dalam hexadecimal) 0xe4|0xd9|0x101

Kunci privat ElGamal yang dihasilkan: (dalam hexadecimal) 0x32|0x101

Selanjutnya akan dilakukan tahapan enkripsi dan dekripsi menggunakan kunci publik dan privat yang sudah dibentuk dengan bilangan prima (p) 257. Misalkan seorang manajer perusahaan mengirimkan pesan rahasia yaitu pin perusahaan kepada karyawannya dengan menggunakan *messenger*. Hasil yang didapat adalah sebagai berikut.

Pesan: Pin account perusahaan 123456
Public key (karyawan): 0xe4|0xd9|0x101
Waktu eksekusi (second): 0.0015521049499511719
Ukuran pesan (byte): 116
Ciphertext: (pasangan tuple a dan b) [17, 10, 17, 238, 17, 78, 17, 4, 17, 237, 17, 173, 17, 173, 17, 46, 17, 111, 17, 78, 17, 143, 17, 4, 17, 14, 17, 109, 17, 207, 17, 111, 17, 175, 17, 237, 17, 13, 17, 237, 17, 237, 17, 78, 17, 4, 17, 231, 17, 199, 17, 167, 17, 135, 17, 103, 17, 71]
Ciphertext: (sudah dinormalisasi menjadi ASCII 256) ¨¨3¨í¨o¨Ø¨ï¨ï¨x¨½¨í¨ 1¨o¨¨¨¨¨½¨¦¨Ø¨ ¨¨Ø¨Ø¨í¨o¨²¨=¨É¨T¨à¨

Tampilan yang diterima oleh karyawan adalah seperti Gambar 5 berikut.

Gambar 5. Pesan pada *messenger* penerima masih dalam bentuk ciphertext (sumber: Dokumen penulis)

Dapat dilihat bahwa pesan dengan informasi rahasia yang dikirimkan sudah tidak dapat dipahami lagi oleh penerima pesan, kecuali dilakukan dekripsi dengan menggunakan kunci privat yang dimiliki oleh penerima pesan (karyawan) tersebut. Tampilan yang dilihat oleh karyawan jika pesan didekripsi dengan kunci privat adalah seperti Gambar 6 berikut.

Pesan: ¨¨3¨í¨o¨Ø¨ï¨ï¨x¨½¨í¨ 1¨o¨¨¨¨¨½¨¦¨Ø¨ ¨¨Ø¨Ø¨í¨o¨²¨=¨É¨T¨à¨
Private key (karyawan): 0x32|0x101
Waktu eksekusi (second): 0.0029578208923339844
Ukuran pesan (byte): 29
Decipher text: **Pin account perusahaan 123456**

Gambar 6. Pesan pada *messenger* penerima yang sudah didekripsi (sumber: Dokumen penulis)

Hasil pesan rahasia yang dikirimkan masih terjaga keamanannya dikarenakan pengguna dapat memilih kapan pesan ingin didekripsikan dengan kunci privat yang ia miliki.

V. Kesimpulan dan Saran

Algoritma kriptografi kunci publik dapat digunakan untuk mengirimkan pesan rahasia dengan aman dikarenakan pesan yang dikirimkan hanya dapat dibuka oleh orang yang memiliki pasangan kunci privatnya. Pada aplikasi *messenger* secara umum keamanan sudah dijaga dengan menerapkan *end to end encryption* dalam melakukan pengiriman pesan. Akan tetapi, *messenger* akan jauh lebih aman jika setelah pesan diterima oleh penerima tetap dijaga keamanannya. Salah satu cara mencapainya dengan menerapkan algoritma ElGamal dalam pengiriman pesan rahasia melalui *messenger*.

Saran yang dapat dilakukan untuk kedepannya adalah melakukan eksplorasi lebih jauh untuk aplikasi algoritma ElGamal selain proses enkripsi dan dekripsi seperti pada penelitian ini. Misalnya pada media komunikasi lain seperti surat elektronik yang memanfaatkan *digital signature* dalam pengiriman pesannya sehingga diketahui jika terdapat perubahan pesan dan intersepsi pada sebuah pesan.

VI. Ucapan Terima Kasih

Pertama-tama penulis mengucapkan ucapan syukur dan terima kasih kepada Tuhan Yang Maha Esa atas kasih dan berkat-Nya sehingga penulis masih dapat menempuh pembelajaran di Teknik Informatika, Institut Teknologi Bandung.

Selanjutnya, penulis juga mengucapkan terima kasih kepada kedua orang tua dan keluarga yang selalu mendoakan dan mendukung apa yang dikerjakan penulis. Penulis juga ingin berterima kasih kepada Bapak Rinaldi Munir selaku dosen mata kuliah Kriptografi yang membantu penulis dalam memahami materi yang disampaikan serta penyusunan makalah ini.

Akhir kata, penulis juga mengucapkan banyak terimakasih kepada teman-teman satu angkatan UNIX 2017, terkhususnya teman-teman grup Jiwa yang senantiasa saling memberikan semangat kepada satu dengan lainnya, termasuk dalam penyusunan makalah ini.

Referensi

[1] York, D. 2010. *Seven Deadliest Unified Communications Attacks*. Syngress.

[2] Munir, Rinaldi. 2020. Slide Kuliah Kriptografi (Bandung: Institut Teknologi Bandung).

[3] WhatsApp. 2020. *WhatsApp Encryption Overview White Paper*. https://scontent.whatsapp.net/v/t39.8562-34/122249142_4698577206422 75_2152527586907531259_n.pdf/WA_Security_WhitePaper.pdf?ccb=2 &_nc_sid=2fbf2a&_nc_ohc=9v2E5walbJgAX-fvt2G&_nc_ht=scontent. whatsapp.net&oh=ac4aebab397cfb1cc8522cf04872c83a&oe=60035199.

[4] Webwise. 2018. *Explainer: What is Messenger?*. https://www.webwise.ie/parents/explained-what-is-messenger/.

[5] BBC News. 2018. *Encryption on Facebook Messenger and other chat apps*. https://www.bbc.com/news/newsbeat-43485511#:~:text=Facebook%20M essenger%20encrypts%20messages%20by,have%20the%20stop%20in% 20between.

[6] Arampatzis, A. 2019. *What Are the Best Use Cases for Symmetric vs Asymmetric Encryption?*. Verafi, https://www.venafi.com/blog/what-are-best-use-cases-symmetric-vs-asy mmetric-encryption

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, 20 Desember 2020

Lukas Kurnia Jonathan 13517006