Jurnal Mahajana Informasi, Vol.1 No 2, 2016
e-ISSN: 2527-8290

PENERAPAN INTERLOCK PROTOCOL DALAM PENCEGAHAN MAN-IN-THE-MIDDLE-ATTACK

Jepri Paulanda
Program Studi Sistem Informasi Universitas Sari Mutiara Indonesia
jepripaulanda@gmail.com

ABSTRAK

Problema *man-in-the-middle-attack* ini dapat diilustrasikan sebagai berikut, misalkan Alice dan Bob sedang berkomunikasi dan Mallory ingin menyadapnya. Ketika Alice mengirimkan kunci publiknya kepada Bob, Mallory dapat menangkap kunci ini dan mengirimkan kunci publiknya sendiri kepada Bob. Kemudian, ketika Bob mengirimkan kunci publiknya kepada Alice, Mallory juga dapat menangkap kunci tersebut dan mengirimkan kunci publiknya sendiri kepada Alice. Ketika Alice mengirimkan pesan kepada Bob yang dienkripsi dengan menggunakan kunci publik Bob, Mallory dapat menangkapnya. Karena pesan tersebut dienkripsi dengan menggunakan kunci publik Mallory, maka Mallory dapat mendekripsikan pesan tersebut dengan menggunakan kunci privatnya dan kemudian dienkripsi kembali dengan menggunakan kunci publik dari Bob dan mengirimkannya kepada Bob. Hal yang sama juga terjadi ketika Bob mengirimkan pesan kepada Alice. Mallory dapat mengetahui semua pesan yang dikirimkan oleh Bob dan Alice tersebut. Problema *man-in-the-middle-attack* ini dapat dicegah dengan menggunakan *interlock protocol*. *Interlock protocol* ini diciptakan oleh Ron Rivest dan Adi Shamir. Setelah menyelesaikan perangkat lunak simulasi pemanfaatan metode *interlock protocol* untuk mengatasi *man-in-the-middle-attack*, perangkat lunak mensimulasikan proses kerja *man-in-the-middle-attack* sebagai salah satu bentuk penyerangan terhadap metode kriptografi publik dan proses kerja *interlock protocol* untuk mengatasinya, sehingga perangkat lunak dapat digunakan untuk mendukung proses belajar mengajar, terutama dalam mata kuliah Kriptografi.

Kata Kunci : Kriptografi, *man-in-the-middle-attack*, *interlock protocol*

I. PENDAHULUAN

1.1 Latar Belakang

Dalam proses komunikasi data, walaupun data telah dienkripsi, terdapat kemungkinan data tersebut dapat diketahui oleh orang lain. Salah satu kemungkinan tersebut adalah orang tersebut menyadap media komunikasi yang digunakan oleh kedua orang yang sedang berkomunikasi tersebut. Hal inilah yang disebut dengan *man-in-the-middle-attack*. Dalam keadaan ini, orang yang menyadap berada di antara kedua orang yang sedang berkomunikasi. Data-data yang dikirimkan oleh orang yang sedang berkomunikasi satu sama lain selalu melalui orang yang menyadap tersebut, sehingga orang yang menyadap tersebut dapat mengetahui semua informasi yang dikirimkan satu sama lain. Keadaan ini muncul karena kedua orang yang sedang berkomunikasi tersebut tidak dapat mem-verifikasi status dari orang yang berkomunikasi dengannya tersebut, dengan mengambil asumsi bahwa proses penyadapan tersebut tidak menyebabkan gangguan dalam jaringan.

Problema *man-in-the-middle-attack* ini dapat diilustrasikan sebagai berikut, misalkan Alice dan Bob sedang berkomunikasi dan Mallory ingin menyadapnya. Ketika Alice mengirimkan kunci publiknya kepada Bob, Mallory dapat menangkap kunci ini dan mengirimkan kunci publiknya sendiri kepada Bob. Kemudian, ketika Bob mengirimkan kunci publiknya kepada Alice, Mallory juga dapat menangkap kunci tersebut dan mengirimkan kunci publiknya sendiri kepada Alice. Ketika Alice mengirimkan pesan kepada Bob yang dienkripsi dengan menggunakan kunci publik Bob, Mallory dapat menangkapnya. Karena pesan tersebut dienkripsi dengan menggunakan kunci publik Mallory, maka Mallory dapat mendekripsikan pesan tersebut dengan menggunakan kunci privatnya dan kemudian dienkripsi kembali dengan menggunakan kunci publik dari Bob dan mengirimkannya kepada Bob. Hal yang sama juga terjadi ketika Bob mengirimkan pesan kepada Alice. Mallory dapat mengetahui semua pesan yang dikirimkan oleh Bob dan Alice tersebut. Problema *man-in-the-middle-attack* ini dapat dicegah dengan menggunakan *interlock protocol*. *Interlock protocol* ini diciptakan oleh Ron Rivest dan Adi Shamir. Algoritma inti dari protokol ini yaitu protokol ini mengirimkan 2 bagian pesan terenkripsi. Bagian pertama dapat berupa hasil dari fungsi hash satu arah (*one way hash function*) dari pesan tersebut dan bagian kedua berupa pesan terenkripsi itu sendiri. Hal ini menyebabkan orang yang menyadap tersebut tidak dapat mendekripsi pesan pertama dengan menggunakan kunci privatnya. Ia hanya dapat membuat sebuah pesan baru dan mengirimkannya kepada orang yang akan menerima pesan tersebut.

Penulis merasa sangat tertarik untuk mempelajari problema *man-in-the-middle-attack* ini dan solusinya dengan menggunakan metode *interlock protocol*.

1.2 Rumusan Masalah

Berdasarkan latar belakang pemilihan judul, maka yang menjadi permasalahan adalah bagaimana menjelaskan proses kerja *man-in-the-middle-attack* dalam menyadap dan mengubah pesan, menjelaskan proses kerja *interlock protocol* untuk mengatasi problema *Man-In-The-Middle-Attack*, menampilkan algoritma dari sistem kriptografi kunci publik metode RSA dan merancang *interface* dari perangkat lunak simulasi.

1.3 Batasan Masalah

Karena keterbatasan waktu dan pengetahuan penulis, maka ruang lingkup permasalahan dalam merancang perangkat lunak ini adalah sebagai berikut :

1. Proses kerja dari perangkat lunak ditampilkan dalam bentuk biner.
2. Proses enkripsi dan dekripsi pesan menggunakan algoritma RSA.
3. Fungsi *hash* satu arah yang digunakan adalah fungsi SHA-1.

4. Nilai-nilai yang dibutuhkan dalam algoritma RSA dapat di-*input* secara manual atau dihasilkan secara acak oleh komputer.
5. Algoritma-algoritma pendukung lainnya yang digunakan mencakup :
    a. Metode *Rabin-Miller* untuk menghasilkan bilangan prima.
    b. Algoritma GCD untuk mengecek prima relatif antar bilangan.
    c. Algoritma *Fast Exponentiation* untuk menghitung perpangkatan modulo bilangan besar.
6. Perangkat lunak akan menampilkan proses enkripsi dan dekripsi dari pesan tersebut secara singkat.
7. Proses kerja yang akan ditampilkan :
    a. Proses terjadinya *man-in-the-middle-attack*.
    b. Proses solusi mengatasi *man-in-the-middle-attack* dengan menggunakan *interlock protocol*.
8. *Interlock protocol* yang dibahas memiliki 2 alternatif berikut :
    a. Alternatif pertama, yaitu *message* terenkripsi dibagi menjadi 2 bagian yang sama besar.
    b. Alternatif kedua, yaitu pecahan pertama berupa nilai *hash* dari *one way hash function* dari *message* dan pecahan kedua berupa *message* terenkripsi.
9. Proses kerja dari perangkat lunak akan didukung dengan animasi gambar.
10. Panjang pesan (*message*) dibatasi maksimal 50 karakter.

1.4 Tujuan Penelitian

Tujuan dari penelitian ini adalah:

1. Untuk merancang suatu perangkat lunak simulasi yang mampu untuk menjelaskan proses kerja dari *man-in-the-middle-attack* ini, dan
2. Proses solusi dengan menggunakan *interlock protocol*.

1.5 Manfaat Penelitian

Manfaat dari penelitian ini, yaitu :

1. Membantu pemahaman proses terjadinya *man-in-the-middle attack* dan proses pencegahannya dengan menggunakan *interlock protocol*.
2. Perangkat lunak dapat digunakan sebagai fasilitas pendukung dalam proses belajar mengajar, terutama untuk mata kuliah Kriptografi.

II. LANDASAN TEORI

2.1 Pengenalan Kriptografi

Cryptography adalah ilmu yang mempelajari teknik-teknik matematika yang berhubungan dengan aspek-aspek pada keamanan informasi misalnya kerahasiaan, integritas data, otentikasi pengirim / penerima data, dan otentikasi data.

Cryptography (kriptografi) berasal dari bahasa Yunani yaitu dari kata '*crypto*' dan '*graphia*' yang berarti penulisan rahasia. Kriptografi adalah suatu ilmu yang mempelajari penulisan secara rahasia. Kriptografi merupakan bagian dari suatu cabang ilmu matematika yang disebut *cryptology*. Kriptografi bertujuan menjaga kerahasiaan informasi yang terkandung dalam data sehingga informasi tersebut tidak dapat diketahui oleh pihak yang tidak sah.

Menurut Stalling, ada beberapa tuntutan yang terkait dengan isu keamanan data, yaitu:

1. *Confidentiality*. Menjamin bahwa data-data tersebut hanya bisa diakses oleh pihak-pihak tertentu saja.
2. *Authentication*. Baik pada saat mengirim atau menerima informasi, kedua belah pihak perlu mengetahui bahwa pengirim dari pesan tersebut adalah orang yang sebenarnya seperti yang diklaim.
3. *Integrity*. Tuntutan ini berhubungan dengan jaminan setiap pesan yang dikirim pasti sampai pada penerimanya tanpa ada bagian dari pesan tersebut yang diganti, diduplikasi, dirusak, diubah urutannya, dan ditambahkan.
4. *Nonrepudiation*. Mencegah pengirim maupun penerima mengingkari bahwa mereka telah mengirimkan atau menerima suatu pesan/informasi. Jika sebuah pesan dikirim, penerima dapat membuktikan bahwa pesan tersebut memang dikirim oleh pengirim yang tertera. Sebaliknya, jika sebuah pesan diterima, pengirim dapat membuktikan bahwa pesannya telah diterima oleh pihak yang ditujunya.
5. *Access Control*. Membatasi sumber-sumber data hanya kepada orang-orang tertentu.
6. *Availability*. Jika diperlukan setiap saat semua informasi pada sistem komputer harus tersedia bagi semua pihak yang berhak atas informasi tersebut.

Dari keenam aspek keamanan data tersebut, empat diantaranya dapat diatasi dengan menggunakan *cryptography* yaitu *confidentiality*, *integrity*, *authentication*, dan *nonrepudiation.*

2.1.1 *Confidentiality*

Confidentiality & privacy terkait dengan kerahasiaan data atau informasi. Pada sistem *e-government* kerahasiaan data-data pribadi (privacy) sangat penting. Hal ini kurang mendapat perhatian di sistem *e-government* yang sudah ada. Bayangkan jika data pribadi anda, misalnya data KTP atau kartu keluarga, dapat diakses secara *online*. Maka setiap orang dapat melihat tempat dan tanggal lahir anda, alamat anda, dan data lainnya. Data ini dapat digunakan untuk melakukan penipuan dan pembobolan dengan mengaku-aku sebagai anda (atau keluarga anda).

Ancaman atau serangan terhadap kerahasiaan data ini dapat dilakukan dengan menggunakan penerobosan akses, penyadapan data (*sniffer, key logger*), *social engineering* (yaitu dengan menipu), dan melalui kebijakan yang tidak jelas (tidak ada).

Untuk itu kerahasiaan data ini perlu mendapat perhatian yang besar dalam implementasi sistem *e-government* di Indonesia. Proteksi terhadap data ini dapat dilakukan dengan menggunakan *firewall* (untuk membatasi akses), segmentasi jaringan (juga untuk membatasi akses), enkripsi (untuk menyandikan data sehingga tidak mudah disadap), serta kebijakan yang jelas mengenai kerahasiaan data tersebut.

Pengujian terhadap kerahasian data ini biasanya dilakukan secara berkala dengan berbagai metode.

Ada beberapa jenis informasi yang tersedia didalam sebuah jaringan komputer. Setiap data yang berbeda pasti mempunyai grup pengguna yang berbeda pula dan data dapat dikelompokkan sehingga beberapa pembatasan kepada pengunaan data harus ditentukan. Pada umumnya data yang terdapat didalam suatu perusahaan bersifat rahasia dan tidak boleh diketahui oleh pihak ketiga yang bertujuan untuk menjaga rahasia perusahaan dan strategi perusahaan. *Backdoor*, sebagai contoh, melanggar kebijakan perusahaan dikarenakan menyediakan akses yang tidak diinginkan kedalam jaringan komputer perusahaan.

Kerahasiaan dapat ditingkatkan dan didalam beberapa kasus pengengkripsian data atau menggunakan VPN. Topik ini tidak akan, tetapi bagaimanapun juga, akan disertakan dalam tulisan ini. Kontrol akses adalah cara yang lazim digunakan untuk membatasi akses kedalam sebuah jaringan komputer. Sebuah cara yang mudah tetapi mampu untuk membatasi akses adalah dengan menggunakan kombinasi dari *username*-dan-*password* untuk proses otentifikasi pengguna dan memberikan akses kepada pengguna (*user*) yang telah dikenali. Didalam beberapa lingkungan kerja keamanan jaringan komputer, ini dibahas dan dipisahkan dalam konteks otentifikasi.

Dalam menjaga kerahasiaan data, kriptografi mentransformasikan data jelas (*plaintext*) ke dalam bentuk data sandi (*ciphertext*) yang tidak dapat dikenali. *Ciphertext* inilah yang kemudian dikirimkan oleh pengirim (*sender*) kepada penerima (*receiver*). Setelah sampai di penerima, *ciphertext* tersebut ditranformasikan kembali ke dalam bentuk *plaintext* agar dapat dikenali.

2.1.2 *Authentication*

Otentikasi merupakan identifikasi yang dilakukan oleh masing – masing pihak yang saling berkomunikasi, maksudnya beberapa pihak yang berkomunikasi harus mengidentifikasi satu sama lainnya. Informasi yang didapat oleh suatu pihak dari pihak lain harus diidentifikasi untuk memastikan keaslian dari informasi yang diterima. Identifikasi terhadap suatu informasi dapat berupa tanggal pembuatan informasi, isi informasi, waktu kirim dan hal-hal lainnya yang berhubungan dengan informasi tersebut.

Aspek ini berhubungan dengan metode untuk menyatakan bahwa informasi betul-betul asli, orang yang mengakses atau memberikan informasi adalah betul-betul orang yang dimaksud,

atau *server* yang kita hubungi adalah betul-betul *server* yang asli.

Masalah pertama, membuktikan keaslian dokumen, dapat dilakukan dengan teknologi *watermarking* dan *digital signature*. *Watermarking* juga dapat digunakan untuk menjaga "*intelectual property*", yaitu dengan menandai dokumen atau hasil karya dengan "tanda tangan" pembuat. Masalah kedua biasanya berhubungan dengan akses kontrol, yaitu berkaitan dengan pembatasan orang yang dapat mengakses informasi.

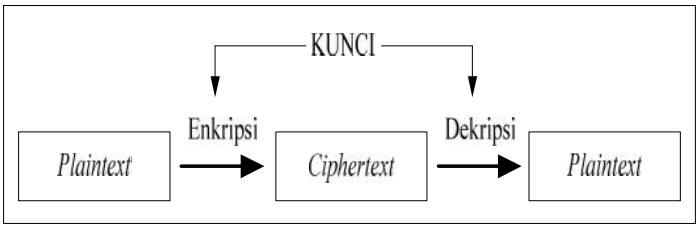
Dalam hal ini pengguna harus menunjukkan bukti bahwa memang dia adalah pengguna yang sah, misalnya dengan menggunakan *password* Aspek / servis dari *security biometric* (ciri-ciri khas orang), dan sejenisnya.

2.2 Algoritma Kriptografi

2.2.1 *Symmetric Algorithms*

Algoritma kriptografi simetris atau disebut juga algoritma kriptografi konvensional. Algoritma ini menggunakan kunci yang sama untuk proses enkripsi dan proses dekripsi.

Algoritma kriptografi simetris dibagi menjadi 2 kategori yaitu algoritma aliran (*Stream Ciphers*) dan algoritma blok (*Block Ciphers*). Pada algoritma aliran, proses penyandiannya berorientasi pada satu bit atau satu *byte* data. Sedang pada algoritma blok, proses penyandiannya berorientasi pada sekumpulan *bit* atau *byte* data (per blok). Contoh algoritma kunci simetris yang terkenal adalah DES (*Data Encryption Standard*).



Gambar 2.2 Prosedur kerja algoritma simetris

2.2.2 *Asymmetric Algorithms*

Algoritma kriptografi asimetris adalah algoritma yang menggunakan kunci yang berbeda untuk proses enkripsi dan dekripsinya. Algoritma ini disebut juga algoritma kunci umum (*publik key algorithm*) karena kunci untuk enkripsi dibuat umum (*publik key*) atau dapat diketahui oleh setiap orang, tapi kunci untuk dekripsi hanya diketahui oleh orang yang berwenang mengetahui data yang disandikan atau sering disebut kunci pribadi (*private key*). Contoh algoritma terkenal yang menggunakan kunci asimetris adalah RSA dan ECC.

Syarat – syarat yang harus dipenuhi oleh suatu algoritma *publik-key* yaitu (Stalling, 1995):

1. Mudah secara komputasi bagi suatu pihak B untuk mengkonstruksi sepasang kunci asimetris (kunci publik KU, kunci pribadi KR).
2. Mudah secara komputasi bagi pengirim A, dengan memiliki kunci publik B dan pesan yang ingin dienkripsi, M, untuk menghasilkan *ciphertext* (C).

3. Mudah secara komputasi bagi penerima B untuk mendekripsi *ciphertext* yang dihasilkan dengan menggunakan kunci pribadinya untuk mengembalikan pesan aslinya.
4. Tidak bisa secara komputasi bagi pihak ketiga untuk memperoleh kunci pribadi KRb hanya dengan mengetahui kunci publik KUb.
5. Tidak bisa secara komputasi bagi pihak ketiga untuk mengembalikan data asli M hanya dengan mengetahui kunci publik KUb dan *ciphertext* C.

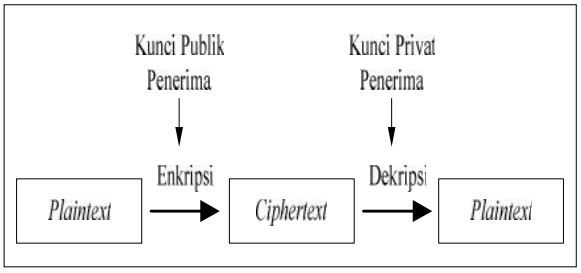
Walaupun bukanlah suatu keharusan bagi semua aplikasi publik-key, namun persyaratan keenam bisa ditambahkan :

6. Fungsi enkripsi dan dekripsi bisa diterapkan dengan urutan yang dibalik.

Kegunaan dari persyaratan keenam adalah untuk penerapan tanda tangan digital (*digital signature*) yang digunakan memecahkan isu otentikasi (*authentication*) dalam masalah keamanan data.

Menurut Stalling (Stalling, 1995), proses enkripsi publik-key sederhana melibatkan empat tahap berikut :

1. Setiap *user* di dalam jaringan membuat sepasang kunci untuk digunakan sebagai kunci enkripsi dan dekripsi dari pesan yang akan diterima.
2. *User* kemudian mempublikasikan kunci enkripsinya dengan menempatkan kunci publiknya ke tempat umum. Pasangan kunci yang lain tetap dijaga kerahasiaannya.
3. Jika *user* A ingin mengirimkan sebuah pesan ke *user* B, ia akan mengenkripsi pesan tersebut dengan menggunakan kunci publik *user* B.
4. Pada saat *user* B ingin mendeskripsikan pesan dari *user* A, ia akan menggunakan kunci pribadinya sendiri. Tidak ada pihak lain yang bisa mendekripsi pesan itu karena hanya B sendiri yang mengetahui kunci pribadi B.



Gambar 2.3 Prosedur kerja algoritma asimetris

2.3 Algoritma Rivest-Shamir-Adleman (RSA)

RSA merupakan salah satu teknik enkripsi dan dekripsi dengan menggunakan dua buah kunci. Kunci-kunci tersebut diperoleh dari hasil perhitungan eksponensial, perkalian, pembagian, penjumlahan dan pengurangan. Perhitungan dilakukan terhadap dua buah bilangan prima.

Walaupun RSA cenderung aman bukan berarti tidak bisa dilakukan "*attack*" terhadap enkripsinya. Didukung perkembangan *hardware* komputer yang semakin cepat maka semakin terbuka kemungkinan memecahkan enkripsi RSA. Pada tahun 1977 Rivest, Shamir dan Adleman mempublikasikan tantangan memecahkan enkripsi

RSA yang memakai 129 digit bilangan bulat. Tantangan ini diharapkan bisa bertahan dari "*attack*" untuk waktu yang lama. Tetapi pada tahun 1994 tantangan ini dipecahkan dengan menggunakan komputer yang kekuatan komputasinya berimbang dengan komputer untuk membuat film animasi "Toy Story" (kumpulan, 87 unit komputer dual prosesor, 30 unit komputer empat procesor, 100Mhz SPARC*station*).

Dibawah ini diterangkan beberapa kemungkinan melakukan "*attack*" terhadap RSA:

1. Cara untuk memecahkan enkripsi RSA oleh penyerang adalah mencari kunci pribadi berdasarkan kunci publik. Jalan menuju itu adalah melakukan pemfaktoran bilangan **n** dari kunci publik. Kemudian dari **n** bisa didapatkan **p** dan **q**, bersama dengan **e** maka bisa didapatkan **d**. Masalahnya adalah memfaktorkan bilangan **n**. Apabila bilangan yang dipakai cukup besar maka akan memperkecil penyerangan dengan cara ini.
2. Cara lain adalah mencari cara untuk menghitung **akar pangkat e mod n**, dari persamaan **c=m^e**, **c** akar pangkat **e**, maka akan didapatkan **m**. Tetapi faktor dari **n** tidak bisa diketahui. Sampai sekarang belum ada metode yang diketahui untuk penyerangan yang dilakukan dengan cara ini.
3. Cara pemecahan enkripsi RSA lainnya adalah dengan menebak sebagian dari kata atau *Single Message Attack*, kemudian kata tersebut dienkripsi dengan menggunakan kunci publik dan membandingkan hasilnya dengan data asli yang terenkripsi. Cara ini memang bisa dilakukan, tetapi masih bisa ditangkal dengan menyusupi bit-bit *random* dalam pesan.
4. Cara "*attack*" yang paling baik yang dikenal adalah GNFS (*General Number Field Sieve*), caranya adalah memfaktorkan **n** ke bilangan prima **p** dan **q,** sama seperti ide no.1. Tetapi waktu yang dibutuhkan untuk RSA dengan besar kunci 1024 bit, sekitar 3x10^11 MIPS-year(MIPS-*year*, komputasi 1 juta instruksi perdetik selama setahun), atau dengan komputer 300 MIPS membutuhkan 2^30 tahun komputer. Kecepatan masih bisa ditingkatkan dengan *hardware* yang lebih baik.

Jadi banyak cara yang bisa dilakukan untuk memecahkan enkripsi RSA, walaupun ada cara yang belum dapat dilakukan saat ini. Semakin meningkatnya daya komputasi komputer dari waktu-kewaktu harus diakui semakin memperbesar kemungkinan memecahkan enkripsi RSA. Tetapi RSA masih bisa tetap digunakan karena digit bilangan bulat yang dipakai masih dapat diperbesar dan disesuaikan dengan teknologi yang ada sekarang. Semakin besar nilai kunci yang

digunakan RSA, maka semakin aman juga teks yang terenkripsi tersebut.

Dengan demikian kekuatan dari algoritma RSA tergantung pada kesulitan pemfaktoran p dan q dari n (Menezes, 1996). Saat ini, tidak ada algoritma yang diketahui dapat memfaktorkan perkalian dari dua bilangan prima untuk nilai yang sangat besar (ratusan digit desimal) dengan cepat. Rekor tercepat dalam memfaktorkan perkalian dua bilangan prima dicatat pada tanggal 22 Agustus 1999 oleh sebuah tim yang dipimpin oleh Herman te Riele di Amsterdam. Bilangan yang berhasil difaktorkan adalah bilangan 512-bit (155-digit decimal) yang merupakan perkalian dua bilangan prima 78-digit desimal. Total waktu yang diperlukan adalah 7,4 bulan dengan menggunakan algoritma *General Number Field Sieve*.

Menanggapi hal ini RSA *Laboratories* pada FAQ versi 4.1 menyarankan kunci RSA yang digunakan adalah minimal 1024-bit yang dengan menggunakan teknologi sekarang masih diperlukan waktu bertahun – tahun untuk memfaktorkannya.

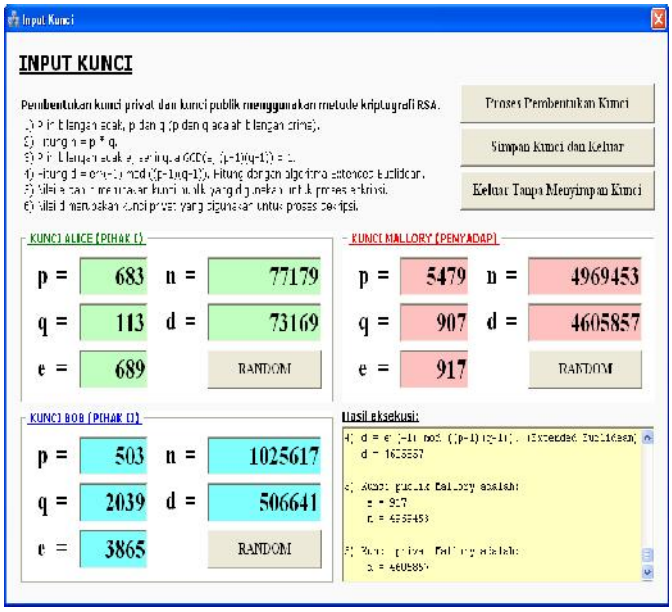
III. HASIL DAN PEMBAHASAN

3.1 Algoritma

Algoritma untuk merancang perangkat lunak simulasi pemanfaatan metode *interlock protocol* untuk mengatasi *man-in-the-middle-attack* terbagi menjadi 3 (tiga) bagian, yaitu:

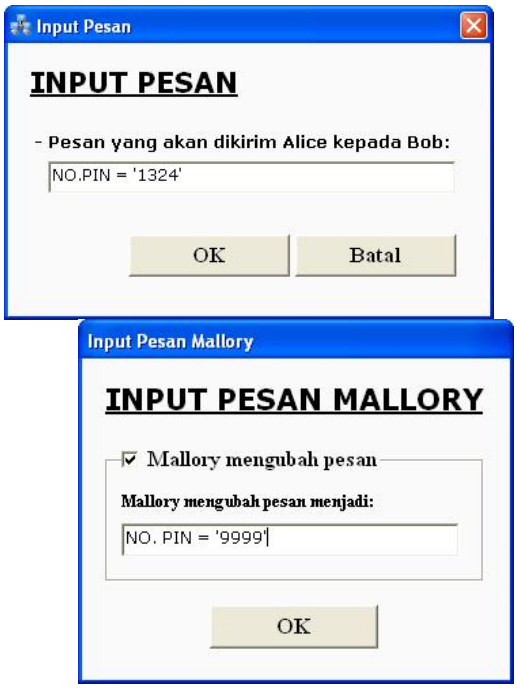
1. Algoritma Kunci Publik Rivest, Shamir dan Adleman (RSA).
2. Algoritma Proses Kerja Man-in-the-Middle-Attack.
3. Algoritma Fungsi-Fungsi Pembantu.

Misalkan, diambil contoh *input* kunci seperti terlihat pada gambar 3.1 berikut.



Gambar 3.1 Contoh *input* kunci

Misalkan, Alice mengirimkan pesan kepada Bob dan Mallory mengubah pesan Alice. Contoh *input* pesan terlihat pada gambar 3.1 berikut.



Gambar 3.2 Contoh *input* pesan

Tampilan proses simulasi terlihat pada gambar 3.2 berikut.

Gambar 3.3 Contoh tampilan *Form* Utama

IV. KESIMPULAN DAN SARAN

4.1 Kesimpulan

Setelah menyelesaikan perangkat lunak simulasi pemanfaatan metode *interlock protocol* untuk mengatasi *man-in-the-middle-attack*, penulis menarik kesimpulan sebagai berikut:

1. Perangkat lunak mensimulasikan proses kerja *man-in-the-middle-attack* sebagai salah satu bentuk penyerangan terhadap metode kriptografi publik dan proses kerja *interlock protocol* untuk mengatasinya, sehingga perangkat lunak dapat digunakan untuk mendukung proses belajar mengajar, terutama dalam mata kuliah Kriptografi.
2. Dengan menggunakan *interlock protocol*, walaupun kunci publik pihak penerima dan pengirim didapatkan dan diganti oleh penyadap, tetapi penyadap tidak dapat menjalankan prosedur *man-in-the-middle-attack* untuk melihat dan mengubah pesan. Hal ini dikarenakan pesan terenkripsi terbagi menjadi dua bagian pada variasi pertama dan terdapat fungsi *hash* untuk memverifikasi keaslian pesan pada variasi kedua.

4.2 Saran

Penulis ingin memberikan beberapa saran yang mungkin dapat membantu dalam pengembangan perangkat lunak ini yaitu :

1. Perangkat lunak ini dapat dikembangkan dengan menambahkan algoritma kunci publik lainnya, seperti: metode Rabin, ElGamal dan LUC.
2. Perangkat lunak dapat dikembangkan dengan menambahkan fitur *multimedia*, yaitu dengan menambahkan animasi yang lebih baik dan suara yang mendukung proses simulasi.

DAFTAR PUSTAKA

Schneier, B, 1996, *Applied Crytography*, *Second Edition*, John Willey and Sons Inc. Canada.

Kurniawan, J., 2004, Kriptografi, Keamanan Internet dan Jaringan Komunikasi, Informatika, Bandung.

Stallings, W., *Cryptography and Network Security Third Edition*, *Prentice Hall*.

Putar, R., 2005, *The Best Source Code Visual Basic*, PT. Elex Media Komputindo, Jakarta.

Suryokusumo.A, 2001, *Microsoft Visual Basic 6.0*, PT. Elex Media Komputindo, Jakarta.

Novian.A, 2004, Panduan MS. Visual Basic 6, Andi, Yogyakarta.

Supardi.Y, 2006, Microsoft Visual Basic 6.0 Untuk Segala Tingkat, PT. Elex Media Komputindo, Jakarta.

en.wilkipedia.org/wiki/man in the middle attack

www.computerhope.com/jargon/m/mitma.htm

www.cs.umu.se/education/examina/rapporter/mattiasericsson.pdf

www.ouah.org/mitmbrief.htm

www.cs.steven.edu/~swetzel/publications/mim.pdf