Pengamanan Pesan Komunikasi Menggunakan Algoritma Rsa, Rabbin Miller Dan Fungsi Sha-1 Serta Penanganan Man In The Middle Attack Dengan Interlock Protocol

Sony Bahagia Sinaga
AMIK STIEKOM Sumatera Utara, Jl. A.H Nasution No. 19, Sumatera Utara, Indonesia
e-mail : sony@amikstiekomsu.ac.id

Abstrak

Komunikasi informasi saat ini telah berkembang dengan luas. Komunikasi informasi baik dalam bentuk saling bertukar pesan, transaksi sampai pertukaran informasi rahasia telah menggunakan teknologi komputer dan jaringan global yang luas dan kompleks. Luasnya komunikasi informasi memanfaatkan teknologi komputer dan jaringan global seperti saat sekarang ini menimbulkan munculnya pihak-pihak yang ingin mencuri informasi tersebut. *Man-in-the-middle-attack* sebagai salah satu bentuk penyerangan terhadap metode kriptografi publik dan proses kerja *interlock protocol* untuk mengatasinya, 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. Untuk melakukan enkripsi dan dekripsi dari pesan komunikasi menggunakan algoritma RSA, Rabin Miller dan Fungsi SHA-1 untuk menambah dan meningkatkan keamanan dari pesan tersebut.

Kata kunci : *Keamanan, Pesan, Komunikasi, RSA, Rabbin Miller, SHA-1*

Abstract

The current information communication has grown broadly. Communication of information whether in the form of exchanging messages, transactions to the exchange of confidential information has been using the computer technology and global network is wide and complex. The extent of information communication utilizing computer technology and global networks such as today has led to the emergence of parties who want to steal the information. Man-in-the-middle-attack as a form of attack on public cryptographic methods and interlock protocol work processes to overcome them, by using the interlock protocol, even though the public key of the recipient and the sender is obtained and replaced by the bugper, but the tappers cannot perform the procedure man-in-the-middle-attack to see and change messages. This is because the encrypted message is divided into two parts in the first variation and there is a hash function to verify the authenticity of the message on the second variation. To encrypt and decrypt the communication messages using the RSA algorithm, Rabin Miller and the SHA-1 function to increase and increase the security of the message.

Keywords: Security, Message, Communication, RSA, Rabbin Miller, SHA-1

1. PENDAHULUAN

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.

Berdasarkan dari latar belakang maka permasalahan dapat dirumuskan antara lain :

1. Mengamankan pesan komunikasi menggunakan algoritma RSA, Rabbin Miller dan SHA-1
2. Bagaimana proses perancangan dan pengembangan aplikasi pengamanan pesan komunikasi.

Berdasarkan perumusan maka dapat diambil tujuan dan manfaat, antara lain :

1. Untuk pengamanan pesan komunikasi
2. Merancang aplikasi pengamanan pesan komunikasi dengan menggunakan algoritma RSA, Rabbin Miller dan Fungsi SHA-1.

Adapun manfaat dari penelitian ini antara lain :

1. Mengatasi permasalahan dalam pencurian data rahasia.
2. Pemahaman proses terjadinya *man-in-the-middle attack* dan proses pencegahan dengan *interlock protocol.*
3. Pencegahan dalam duplikasi serta pengambilan pesan.

Berdasarkan tujuan dan manfaat penelitian maka dapat diambil suatu batasan dari pengamanan pesan antara lain :

1. Proses enkripsi dan dekripsi menggunakan algoritma RSA.
2. Pembangkit kunci menggunakan algoritma Rabbin Miller.

Perangkat lunak yang digunakan adalah visual basic.Net 2008.

2. METODOLOGI PENELITIAN

II.1. Kriptografi

Kriptografi (*cryptography*) berasal dari bahasa Yunani: "*cryptos"* artinya "*secret*" (rahasia), sedangkan "*graphein*" artinya "*writing*" (tulisan). Jadi, kriptografi berarti "*secret writting*" (tulisan rahasia). Ada beberapa defenisi kriptografi yang telah dikemukakan didalam berbagai literatur. Kriptografi adalah ilmu dan seni untuk menjaga kerahasian pesan dengan cara menyandikannya ke dalam bentuk yang tidak dapat dimengerti lagi maknanya. Tujuan dari kriptografi adalah memberikan keamanan dalam bentuk / *privacy, data integrity, authentication, non-repudiation.* Pada bidang kriptografi terdapat dua proses utama yaitu enkripsi dan dekripsi. Enkripsi adalah proses menyandikan plainteks atau pesan menjadi cipherteks. Dekripsi adalah proses mengembalikan cipherteks menjadi plainteks semula.

Pada kriptografi modern proses enkripsi dan dekripsi biasanya dilakukan dengan kunci yang dipilih oleh pelaku komunikasi ataupun dapat dibangkitkan secara acak. Algoritma kriptografi modern seperti DES, AES, dan IDEA merupakan algoritma kriptografi modern yang sangat rumit dan kompleks. Algoritma kriptografi modern secara umum memiliki daya tahan yang cukup tinggi terhadap serangan, namun memiliki alur proses yang rumit serta membutuhkan sumber daya yang relative besar. Suatu pesan yang tidak disandikan disebut sebagai *plaintext* ataupun dapat disebut juga sebagai *cleartext*. Proses yang dilakukan untuk mengubah *plaintext* ke dalam *ciphertext* disebut *encryption* atau *encipherment*. Sedangkan proses untuk mengubah *ciphertext* kembali ke *plaintext* disebut *decryption* atau *decipherment*. Beberapa istilah yang terdapat dalam kriptografi antara lain :

1. Pesan, Plainteks dan Cipherteks

Pesan (*message*) adalah data atau informasi yang dibaca dan dimengerti maknanya. Nama lain untuk pesan adalah plainteks (*plaintext*) atau teks jelas (*cleartext*). Pesan dapat berupa data atau informasi yang dikirim (melalui kurir, saluran telekomunikasi, dan sebagainya) atau yang disimpan di dalam media perekaman (kertas, *storage,* dan sebagainya). Pesan yang tersimpan tidak hanya berupa teks, tetapi juga dapat berbentuk citra (*image*), suara/bunyi (audio), dan video, atau berkas biner lainnya.

Agar pesan tidak dapat dimengerti maknanya oleh pihak lain, maka pesan perlu disandikan ke bentuk lain yang tidak dapat dipahami. Bentuk pesan yang tersandi disebut **cipherteks** (*ciphertext*) atau **kriptografi** (*cryptogram*). Cipherteks harus dapat ditransformasikan kembali menjadi plainteks semula agar pesan yang diterima bisa dibaca. Gambar berikut memperlihatkan contoh-contoh dua buah plainteks, masing-masing berupa teks dan gambar, serta cipherteks yang berkoresponden.

2. Pengirim dan Penerima

Komunikasi data melibatkan pertukaran pesan antara dua entitas. **Pengirim** (*sender*) adalah entitas yang mengirim pesan kepada entitas lainnya. **Penerima** (*receiver*) adalah entitas yang menerima pesan. Entitas disini dapat berupa orang, mesin (komputer), kartu kredit, dan sebagainya. Jadi, orang bisa bertukar pesan dengan orang lainnya (contoh: Alice berkomunikasi dengan Bob), sedangkan di dalam jaringan komputer mesin (komputer) berkomunikasi dengan mesin (contoh: mesin ATM berkomunikasi dengan komputer *server* di bank).

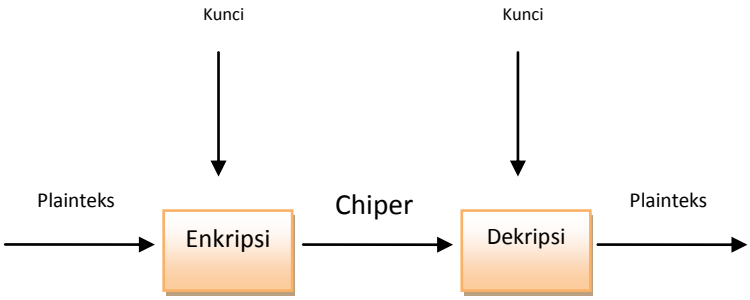
3. Enkripsi dan Dekripsi

Proses menyandikan plainteks menjadi cipherteks disebut **enkripsi** (*encryption*) atau *enciphering* (standart nama menurut ISO 7498-2). Sedangkan proses mengembalikan cipherteks menjadi plainteks semula dinamakan **dekripsi** (*decryption*) atau *deciphering* (standart nama menurut ISO 7498-2). Enkripsi dan dekripsi dapat diterapkan bagaikan pada pesan yang dikirim maupun pada pesan tersimpan. Istilah *encryption of data in motion* mengacu pada enkripsi pesan yang ditransmisikan melalui saluran komunikasi, sedangkan istilah *encryption of data at-rest* mengacu pada enkripsi dokumen yang disimpan didalam *storage*. Contoh *encryption of data in motion* adalah pengiriman nomor PIN dari mesin ATM ke komputer *server* dikantor bank pusat. Contoh *encryption of data at-rest* adalah enkripsi *file* basis data didalam *harddisk.*

4. Cipher dan Kunci

Algoritma kriptografi disebut juga *cipher* yaitu aturan untuk *enchipering* dan *dechipering,* atau fungsi matematika yang digunakan untuk enkripsi dan dekripsi. Beberapa *cipher* memerlukan algoritma yang berbeda untuk *enchipering* dan *dechipering.*

Konsep matematis yang mendasari algoritma kriptografi adalah relasi antara dua buah himpunan yang berisi elemen-elemen plainteks dan himpunan yang berisi cipherteks. Enkripsi dan dekripsi merupakan fungsi yang memetakan elemen-elemen antara kedua himpunan tersebut. Misalkan *P* menyatakan plainteks dan *C* menyatakan cipherteks, maka fungsi enkripsi *E* memetakan *P* ke *C.*



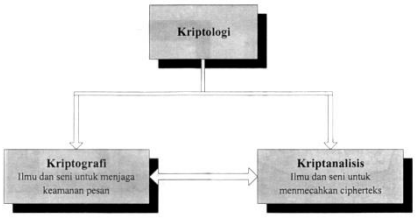
Gambar 1. Skema Enkripsi dan Dekripsi

Keamanan algoritma sering kriptografi diukur dari banyaknya kerja (*work*) yang dibutuhkan untuk memecahkan cipherteks menjadi plainteksnya tanpa mengetahui kunci yang digunakan. Kerja ini dapat diekivalenkan dengan waktu, memori, uang, dan lain-lain. Semakin banyak kerja yang diperlukan, yang berarti juga semakin lama waktu yang dibutuhkan, maka semakin kuat algoritma kriptografi tersebut, yang berarti semakin aman digunakan untuk menyandikan pesan. Jika keamanan kriptografi ditentukan dengan menjaga kerahasiaan algoritmanya, maka algoritma kriptografinya dinamakan algoritma *restricted*. Algoritma *restricted* biasanya digunakan oleh sekelompok orang untuk bertukar pesan satu sama lain. Mereka membuat suatu algoritma enkripsi dan algoritma enkripsi tersebut hanya diketahui oleh anggota kelompok itu saja. Tetapi, algoritma *restricted* tidak cocok lagi saat ini, sebab setiap kali ada anggota kelompok keluar, maka algoritma kriptografi harus diganti lagi. Kunci biasanya berupa *string* atau deretan bilangan. Dengan menggunakan kunci *K,* maka fungsi enkripsi dan dekripsi dapat ditulis sebagai :

$$E_k(P)=C \text{ dan } D_k©=P$$

Dan kedua fungsi ini memenuhi

$$D_x(E_k(P))=P$$



Gambar 2. Diagram Bidang Kriptografi

II.2. Algoritma 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.

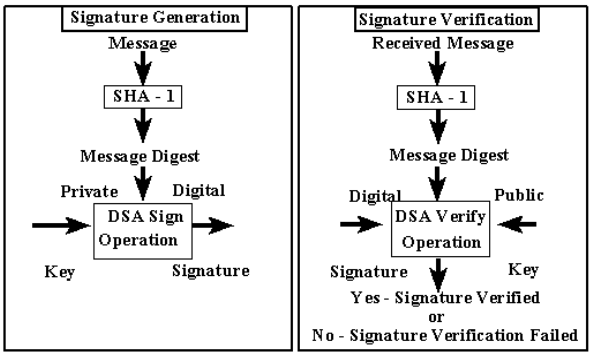
Jadi banyak cara yang bisa dilakukan untuk memecahkan enkripsi RSA, walaupun ada cara yang belum dapat dilakukan saat ini. Semakin meningkatnya daya komputasi komputer dari waktu-kewaktu harus diakui semakin memperbesar kemungkinan memecahkan enkripsi RSA. Tetapi RSA masih bisa tetap digunakan karena 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.

II.3. Fungsi SHA-1

Secure Hash Algorithm, SHA-1 ini dikembangkan oleh NIST (*National Institute of Standard and Technology*). SHA-1 dapat diterapkan dalam penggunaan *Digital Signature Algorithm* (DSA) yang dispesifikasikan dalam *Digital Signature Standard* (DSS) dan SHA tersebut dapat diterapkan untuk aplikasi federal. Untuk suatu pesan yang panjangnya < 2 ^ 64, SHA-1 akan menghasilkan keluaran sebanyak 160 bit dari pesan tersebut dan pesan keluaran itu disebut *message digest*. Panjang jarak *message digest* dapat berkisar antara 160 sampai 512 bit tergantung algoritmanya. Berdasarkan cirinya SHA-1 dapat digunakan dengan algoritma kriptografi lainnya seperti *Digital Signature Algorithms* atau dalam generasi angka yang acak (*bits*).



Gambar 3. Bentuk Penggunaan SHA-1

SHA-1 dikatakan aman karena proses SHA-1 dihitung secara infisibel untuk mencari pesan yang sesuai untuk menghasilkan *message digest* atau dapat juga digunakan untuk mencari dua pesan yang berbeda yang akan menghasilkan *message digest* yang sama. Menurut jenisnya SHA dapat dispesifikasikan menjadi 4 bagian yaitu: SHA-1, SHA-256, SHA-384, dan SHA-512. Berikut ini merupakan daftar-daftar properti dari keempat SHA.

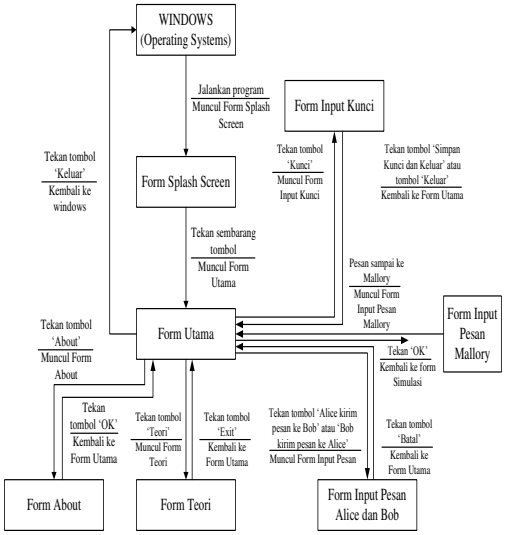
Tabel 1. Daftar Property Keempat SHA

Algorithm	Message Size (bits)	Block Size (bits)	Word Size (bits)	Message Digest Size (bits)	Security[2] (bits)
SHA-1	$<2^{64}$	512	32	160	80
SHA-256	$<2^{64}$	512	32	256	128
SHA-384	$<2^{128}$	1024	64	384	192
SHA-512	$<2^{128}$	1024	64	512	256

3. HASIL DAN PEMBAHASAN

III.1. Hasil

Dalam proses ini, diasumsikan terdapat dua pihak yang sedang berkomunikasi (Alice dan Bob) dan satu pihak sebagai penyadap (Mallory) yang berada di tengah-tengah saluran komunikasi. Sebelum proses simulasi dijalankan, *user* harus meng-*input* kunci privat dan publik Alice, Bob dan Mallory pada *form* Input Kunci. *Input* kunci juga dapat dihasilkan secara acak oleh komputer. Pengacakan menggunakan *random number generator* yang terdapat dalam *syntax* Microsoft Visual Basic 6.0. Setelah itu, ditampilkan proses simulasi penyadapan dan penukaran kunci yang dilakukan oleh Mallory terhadap Bob dan Alice. Selanjutnya, *user* dapat meng-*input* sendiri pesan yang akan dikirim Bob kepada Alice atau sebaliknya dan pesan samaran yang dibuat oleh Mallory. Dalam proses simulasi ini, *user* juga dapat memilih untuk menggunakan opsi untuk menggunakan *interlock protocol* atau tidak. Mengatasi masalah *man-in-the-middle-attack* dengan *interlock protocol* ini terbagi lagi menjadi dua cara, yaitu: pisahkan hasil enkripsi menjadi dua bagian atau gunakan fungsi *hash* sebagai bagian pertama dan pesan terenkripsi menjadi bagian kedua. Kedua cara ini akan mengatasi tindakan penyamaran pesan oleh Mallory.



Gambar 5. State Transition Diagram (STD) Perangkat Lunak

III.2. Pembahasan

Proses-proses yang terjadi di dalam perangkat lunak, yaitu:

1. Pertama, *user* harus meng-*input* kunci publik dan privat Bob, Alice dan Mallory. *Input* kunci adalah:
   a. Nilai p, untuk menghitung nilai n (p adalah bilangan prima).
   b. Nilai q, untuk menghitung nilai n (q adalah bilangan prima)..
   c. Nilai n, dihitung dengan rumus n = p x q. (n adalah kunci publik untuk enkripsi).
   d. Nilai e, sebagai kunci publik untuk proses enkripsi. (GCD(e, (p-1)(q-1)) = 1).
   e. Nilai d, sebagai kunci privat untuk proses dekripsi, dihitung dengan rumus $d = e^{-1} \bmod ((p-1)(q-1))$. Hitung dengan algoritma Extended Euclidean.

   Input kunci ini juga dapat dihasilkan secara acak (*random*).
2. *User* dapat memulai proses simulasi *man-in-the-middle-attack*. Proses ini mensimulasikan terjadinya penyadapan dan penggantian kunci oleh Mallory, pihak yang berada di tengah saluran komunikasi Alice dan Bob. Proses yang terjadi adalah:
   a. Alice mengirimkan kunci publiknya (KP_A) kepada Bob.
   b. Mallory menyadap kunci publik Alice (KP_A) dan menggantinya dengan kunci publiknya sendiri (KP_M).
   c. Bob menerima 'kunci publik Alice' yang sebenarnya adalah kunci publik Mallory (KP_M).
   d. Bob mengirimkan kunci publiknya (KP_B) kepada Alice.
   e. Mallory menyadap kunci publik Bob (KP_B) dan menggantinya dengan kunci publiknya sendiri (KP_M).
   f. Alice menerima 'kunci publik Bob' yang sebenarnya adalah kunci publik Mallory (KP_M).

   Sampai di sini, kunci publik Alice dan kunci publik Bob telah disadap oleh Mallory. Alice dan Bob tidak menyangka bahwa kunci yang dimiliki adalah kunci publik Mallory.
3. Selanjutnya, *user* dapat mensimulasikan pengiriman pesan Alice kepada Bob dan Bob kepada Alice. Misalkan, Alice mengirimkan pesan kepada Bob.
   a. Alice mengenkripsi pesan yang akan dikirimkan kepada Bob dengan 'kunci publik Bob' (yang sebenarnya adalah kunci publik Mallory).

b. Mallory menyadap pesan dari Alice. Karena kunci publik yang digunakan untuk proses enkripsi sebenarnya adalah kunci publik Mallory, maka Mallory dapat mendekripsi pesan Alice dengan kunci privatnya dan melihat pesan yang dikirimkan Alice. Di sini, Mallory memiliki kekuasaan penuh untuk melihat dan mengubah pesan yang dikirim Alice. Mallory kemudian mengenkripsi kembali pesan dengan kunci publik Bob yang sebenarnya.
c. Pesan diteruskan kepada Bob.
d. Bob menerima pesan. Bob mendekripsi pesan dengan kunci privatnya. Bob tidak akan menyadari bahwa pesan sebenarnya telah dibaca dan bahkan diubah oleh Mallory.

4. *User* juga dapat mensimulasikan proses solusi dengan menggunakan *interlock protocol*. Terdapat dua buah variasi *interlock protocol*, yaitu:
   a. Variasi pertama adalah: ketika Alice akan mengirimkan pesan kepada Bob, Alice mengenkripsi pesannya dan membagi hasil enkripsi menjadi 2 bagian. Kedua bagian dikirimkan secara terpisah dalam satu jangka waktu. Ketika Mallory menerima bagian pertama dari pesan, Mallory tidak akan dapat membaca pesan. Mallory harus menerima kedua bagian sekaligus untuk dapat membaca pesan, dan ketika Mallory menerima bagian kedua dari pesan, Mallory sudah terlambat untuk mengubah pesan karena bagian pertama telah atau sudah harus diterima Bob. Apabila tidak, maka ini akan menimbulkan kecurigaan karena *range* waktu yang tidak wajar.

Variasi kedua adalah: Alice mengenkripsi pesannya dan menggunakan fungsi *hash* SHA-1 untuk menghasilkan nilai *hash* dari pesan terenkripsi. Nilai hash dikirimkan sebagai bagian pertama, dan pesan terenkripsi dikirimkan sebagai bagian kedua. Ketika Bob menerima kedua bagian pesan, Bob melakukan verifikasi apakah nilai *hash* bagian kedua yang diterima sama dengan bagian pertama. Apabila tidak sama, maka dapat dipastikan ada yang mengubah atau memodifikasi pesan.

4. KESIMPULAN

Jadi Dapat Disimpulkan Kesimpulan :

1. Proses enkripsi dan dekripsi menggunakan algoritma RSA.
2. Pembangkit kunci menggunakan algoritma Rabbin Miller.

5. SARAN

Saran yang diusulkan untuk penelitian selanjutnya dapat dikembangkan Kriptografi dan anlogritma RSA untuk kasus yang lain

DAFTAR PUSTAKA

[1] Rinaldi Munir. 2006. “Kriptografi”. Bandung : Informatika Bandung.

[2] Dony Ariyus. 2008. “Pengantar Ilmu Kriptografi Teori, Analisis, dan Implementasi”. Yogyakarta : Andi OFFSET.

[3] Dony Ariyus. 2009. “Keamanan Multimedia”. Yogyakarta : Andi OFFSET.

[4] Al-Bahra Bin Ladjamuddin. 2005.“Analisis Dan Desain Sistem Informasi”.Yogyakarta : Graha Ilmu.

[5] Jogiyanto. 2005. “Pengenalan Komputer”. Yogyakarta : Andi OFFSET.

[6] Djon Irwanto. 2006. “Perancangan *Object Oriented software* dengan UML”. Yogyakarta: Andi OFFSET.

[7] A.M. Hirin. 2011. “VB.NET 2010”. Jakarta : Prestasi Pustakaraya.