PENERAPAN SISTEM KEAMANAN MENGGUNAKAN CRYPTOGRAPHY PADA APLIKASI *CHATTING* DENGAN MEMODIFIKASI ALGORITMA *RIVEST SHAMIR ADLEMAN (RSA)*

Muttaqin, Haruno Sajati, Nurcahyani Dewi
Jurusan Teknik Informatika
Sekolah Tinggi Teknologi Adisutjipto
informatika@stta.ac.id.

ABSTRACT

Chatting application is an application which used for communication through local network or internet. It can use to make communication to be effective and easy to use but there are several shortages in delivery information process are data security when communication is going on. Using chrypthography with modificationRivest Shamir adleman (RSA) algorithm in encryption process and decryption on packet data which transmited is one of security system which can use in this application then the security of data packet secured. Result from implementation of chatting application which applies chrypthography with modification Rivest Shamir adleman (RSA) algorithmhave done in local network and internet able to run smoothly and have done by several testing cases based on certain scenario. Whereas process testing encryption and decryption using tools wireshark.

Keywords*: *Sercurity system, modification algorithm RSA, chatting application.

1. Pendahuluan

Penerapan kriptografi pada aplikasi *chatting*merupakan salah satu aspek keamanan yang digunakan untuk menjaga kerahasian dari informasi tersebut. Sehingga pengguna dikalangan umum yang menggunakan aplikasi tersebut menjadi lebih aman terhadap informasi yang ditransmisikan. mengingat keamanan informasi yang akan ditransmisikan melalui media jaringan *LAN/internet* dapat mengakibatkan proses manipulasi akan informasi tersebut maka untuk mengatasi hal tersebut yaitu membangun kriptografi dengan memodifikasi algoritma *RSA* yang diterapakan pada aplikasi *chatting*.

2. Metodologi

2.1 Tinjauan Pustaka

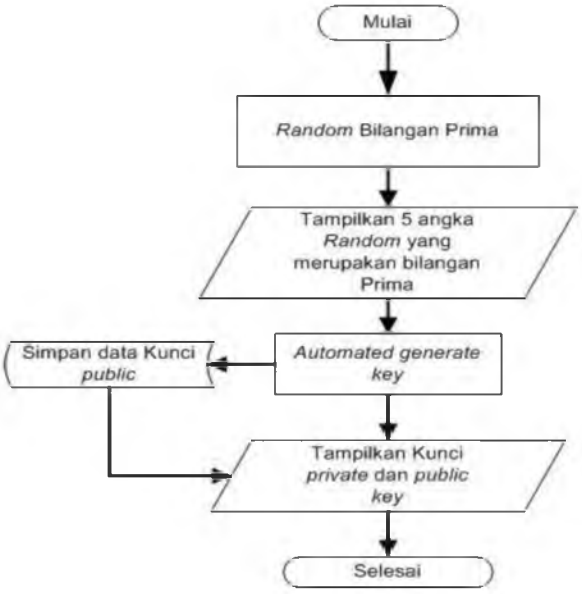
Kriptografi merupakan ilmu dan seni untuk menjaga keamanan pesan yang mempelajari teknik-teknik matematika yang berhubungan dengan aspek keamanan informasi seperti kerahasiaan, integritas, serta otentikasi. Jurnal ini menjelaskan tentang kriptografi kunci publik yang menggunakan sepasang kunci untuk enkripsi dan dekripsi. Aplikasi dari kunci publik antara lain kunci persetujuan (k*ey agreement)*, kerahasiaan data (*data encryption*) dan tanda tangan digital (*digital signature*). Contoh kriptografi kunci publik antara lain *RSA, DH* dan *DSA*. Penerapan kriptografi kunci publik membutuhkan pendukung yang dinamakan PKI *(Public Key Infrastructure)*. *PKI* adalah sebuah pengaturan yang menjamin penggunaan kunci

publik bagi pihak-pihak yang terlibat dengan sistem penggunaan sistem keamanan. Dengan *PKI* setiap pengguna dapat mengotentikasi satu sama lain (Lusiana, 2010).

Masalah keamanan, kerahasiaan, keaslian dan integritas data merupakan aspek-aspek penting yang perlu dilakukan untuk menjaga informasi dari pihak-pihak yang tidak memiliki otoritas atau hak akses. Untuk mengatasi hal ini, penulis mencoba mengimplementasikan konsep kriptografi pada sistem keamanan data pada jaringan komputer. Data-data elektronik dapat diamankan dengan cara mengubah data menjadi sandi-sandi yang tidak dimengerti. Banyak algoritma kriptografi yang bisa diterapkan untuk mengamankan data, namun pada kesempatan kali ini penulis akan merancang algoritma tersendiri untuk mengatasi masalah keamanan data pada jaringan komputer. Cara efektif untuk menyembunyikan data atau informasi adalah dengan cara enkripsi (Munawar, 2012).

2.2 *Flowchart* Modifikasi algoritma *RSA*

Bagan alir sistem modifikasi algoritma *RSA* dapat terlihat pada gambar 1. Dalam perancangan Modifikasi algoritma *RSA*, bagan alir sistem ini akan digambarkan menggunakan *flowchart*. Modifikasi algoritma *RSA* sangat sederhana jika dibandingkan dengan *flowchart* algoritma yang lainnya, keutamaan dari algoritma ini adalah terletak pada pembuatan kunci yang didasari rumus.



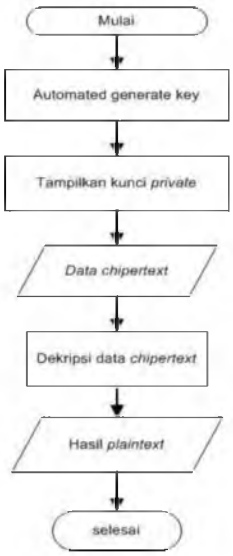
Gambar 1. *Flowchart* modifikasi algoritma *RSA*

2.3 *Flowchart* Enkripsi

Proses enkripsi data atau informasi yang akan dikirimkan terlihat pada Gambar 3. Dalam proses ini, akan digambar menggunakan *flowchart* sebagai bagan alir sistem.

2.4 *Flowchart* Dekripsi

Proses dekripsi akan dilakukan secara otomatis oleh sistem tanpa diketahui oleh *user*, terlihat pada Gambar 2. Proses ini, digambarkan menggunakan *flowchart* sebagai bagan alir sistem.

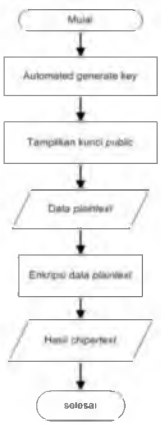


Gambar 2. *Flowchart* Dekripsi data *chipertext*

2.5 Bilangan Prima

Bilangan prima merupakan bilangan yang hanya mempunyai 2 (dua) *factorial* yaitu angka (1) dan bilangan itu sendiri, seperti contoh berikut ini :

1. Tentukan bilangan prima dari angka 17 ?
   Jawaban : 1 dan 17.
   Karena mencari *factorial* dari bilangan 17 adalah 1 dan 17, maka bilangan 17 merupakan bilangan prima.
2. Tentukan bilangan prima dari 1 – 20 ?
   Jawaban : 1,2,3,5,7,11,13,17
   Dari jawaban diatas bahwa terdapat 8 buah angka prima, jika difaktorial maka akan mendapatkan 1 dan bilangan itu sendiri.



Gambar 3. *Flowchart* enkripsi data *plaintext*

2.6 *Generate Key*

Dalam memodifikasi algoritma *RSA* terdapat konsep ataupun alur dalam proses pembuatan kunci. Pada saat kunci diciptakan menggunakan *random* bilangan prima yang sudah dibatasi dengan jumlah maksimal dari angka *random* tersebut adalah

seratus angka. Dari hasil *random* yang sudah diproses terdapat 5 buah angka prima yang akan dipakai dalam pembuatan kunci sebagai berikut:

1. Angka ke 1 : 17
2. Angka ke 2 : 23
3. Angka ke 3 : 13
4. Angka ke 3 : 11
5. Angka ke 5 : 7

Semua angka yang sudah dideklarasikan diatas digantikan dengan *variable* sehingga p = 17, x =23, q =13 , y = 11 dan n = 7

Dengan menerapkan *formula* yang diciptakan oleh Modifikasi algoritma RSA yaitu :

Formula : Modifikasi algoritma RSA

1. P ^ x mod n = (kunci *public* , n)
2. Q ^ y mod n = (kunci *private* , n)

a. 17 ^ 23 mod 7 = (kunci *public* , n)
b. 13 ^ 11 mod 7 = (kunci *private* , n)

Dengan adanya proses dari ke-2 bilangan yang disebutkan di atas maka perolehan hasil dari ke-2 angka tersebut sudah siap untuk diterapkan dalam proses enkripsi maupun dekripsi data sesuai dengan kunci yang telah ditentukan.

Dalam proses pertukaran kunci *public* maka setiap *user* yang *login* ke sistem maka akan dibangun 2 komponen utama yaitu kunci *public* dan kunci *private*. Kunci *public* akan disimpan ke dalam *database* pada *table* kriptografi, sehingga setiap *user* mendapatkan kunci *public user* yang lainnya sedangkan kunci *private* hanya terdapat di sisi *user* atau *client* saja, ini untuk mencegah adanya pihak ke-3 yang mendapatkan kunci *private* dalam proses pertukaran kunci *public* maupun kunci *private* yang ada.

2.7 Proses Enkripsi

Pada pembahasan sebelumnya yaitu menjelaskan tentang bagaimana kunci diciptakan dan tahapan proses yang dilaluinya berikut ini akan membahas tentang bagaimana enkripsi sebuah data yang sudah disediakan dalam bentuk *plaintext* menjadi *chipertext*.

Plain text adalah sebuah data yang masih dalam keadaan sempurna dan bisa dibaca oleh manusia layaknya sebuah tulisan yang dituliskan oleh manusia, yang menjadi perbedaannya adalah data tersebut ada pada jendala *editor* yang berada dalamkomputer sehingga bisa dilihat oleh siapanpun yang mengakses data tersebut. Sedangkan *chipertext* adalah kebalikan dari definisi *plaintext*, sehingga dapat didefinisikan sebagai data yang sudah dalam keadaan atau strukturnya sudah rusak (*broken)* yang hanya dapat dibaca oleh mesin komputer, dan data tersebut dapat disebarkan kemanapun dan kapanpun, sehingga data tersebut sudah aman dalam proses pembungkusan menggunakan modifikasi algoritma *RSA*.

***Formula* enkripsi**

EK = (kunci *public*, N) ➔ H + C + I

EK = Enkripsi
N = Angka dari algoritma *random*
H = Hasil dari Proses Perhitungan kunci *Public*
C = Karakter atau *symbol*
I = Pembacaan dari karakter awal ke (1) hingga sebanyak karakter

2.8 Proses Dekripsi

Proses dekripsi merupakan kebalikan dari proses enkripsi, proses ini sering juga dikenal dengan sebutan kriptonalisis yaitu proses pembongkaran sebuah data dari hasil enkripsi tanpa menggunakan kunci *private* yang sudah ditentukan oleh mesin sesuai yang telah dibuat menggunakan rumus modifikasi algoritma *RSA*.

Dalam proses dekripsi, membutuhkan kunci *private* untuk menyusun atau mendekrip kembali dari hasil enkripsi. Kunci *private* ini sudah tersedia di mesin *computer user* pada saat *login* ke dalam sistem sehingga *user* tidak perlu mengetahui tentang kunci yang telah dibuat.

***Formula* Dekripsi**

DK = (kunci *private*, N) = H - C + I

DK = Dekripsi
N = Angka prima dari algoritma *random*
H = Hasil dari Proses Perhitungan kunci *Public*
C = Karakter atau *symbol*
I = Pembacaan dari karakter awal (ke 1) hingga sebanyak karakter.

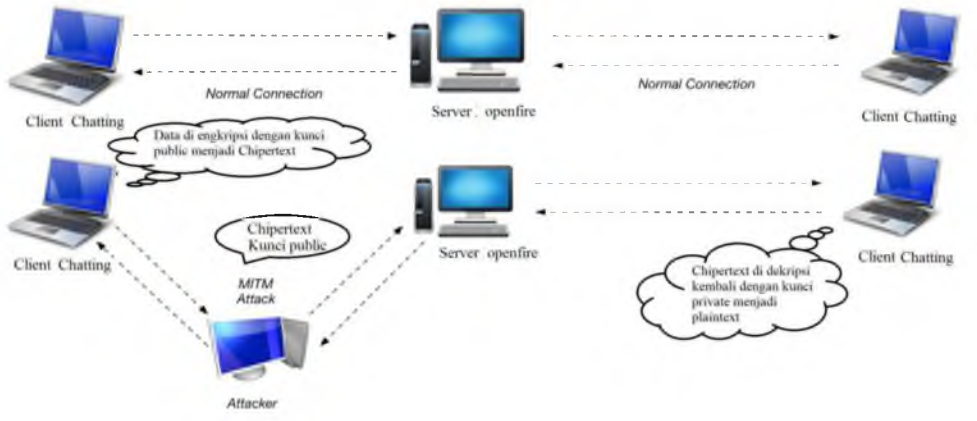
2.9 Analisa Sistem

Layanan aplikasi *chatting* yang beredar di *internet* menjadi salah satu cara untuk dapat terhubung antara satu *client* dengan *client* lainnya yang saling bertukaran informasi dengan sangat mudah dan cepat. Fasilitas ini juga sangat rentan terhadap proses *sniffing/spoofing* yang terjadi di kalangan masyarakat yang saat ini bukan hal biasa lagi yang memakai fasilatas *internet* di dunia maya.

Beberapa dari layanan aplikasi *chatting* sudah tersedia keamanan tersendiri oleh pihak yang menciptakan aplikasi tersebut, namun hanya sebatas untuk *protocol* yang memakai sistem *TLS/SSL* yang diamankan sedangkan paket data dikirimkan dalam jaringan masih tidak aman. Hal ini dapat ditangani dengan kolaborasi antara proses kriptografi dengan modifikasi algoritma *RSA* pada aplikasi *chatting* maka paket data yang dikirimkan akan dienkirpsi terlebih dahulu menggunakan kunci *public* sehingga jika di tengah–tengah dari proses pengiriman data terjadi *sniffing/spoffing*

maka paket data tersebut tidak dapat membuka ataupun memanipulasikan paket tersebut.

Pada saat data tersebut tiba di pihak *client* yang dituju, maka akan dilakukan proses dekripsi dengan menggunakan kunci *private* sehingga *client* dapat menerima data yang sama dengan yang dikirimkan seperti pada Gambar 4.



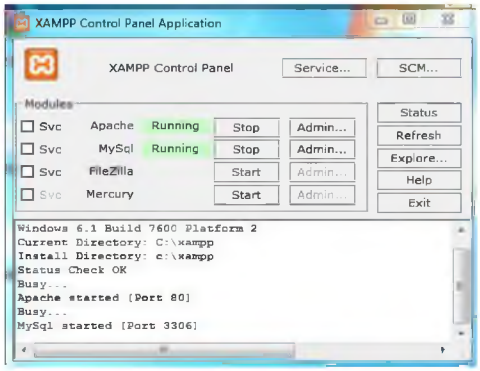
Gambar 4. Arsitektur Sistem *Sniffing / Spoffing*

Bagi pengguna, kemanan paket data yang kirimkan merupakan hal yang sangat penting dan menjadi kebutuhan utama untuk menjaga kerahasiaan data tersebut tetap utuh.pada jaringan *LAN/ internet* yang luas memimbulkan pemikiran seseorang dapat melakukan tindakan kejehatan (*cybercrime*) pada proses pertukaran paket data yang dilakukan oleh pihak lain. Sehingga, setiap paket data yang dikirimkan membutuhkan sistem keamanan untuk menjamin data tersebut sampai pada tujuan dan tidak terjadi manipulasi di tengah jalan.

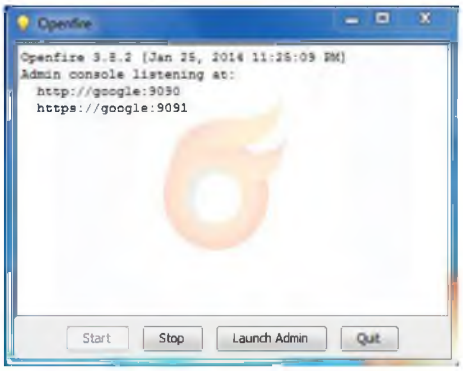
Oleh karena itu dibutuhkan sebuah sistem untuk mengamankan data tersebut dengan menggunakan aplikasi *chatting*yang menerapkan modifikasi algoritma *RSA*. Salah satu cara yaitu menggunakan teknik enkripsi dan dekripsi terhadap paket data tersebut. Sehingga pengguna dapat selalu aman dan tetap terjaga kerahasiaan data tersebut.

2.10 Konfigurasi *Openfire Server*

Dalam konfigurasi ini akan dilakukan proses menjalankan *openfire, database* dan men-*disable TLS/SSL* yang dapat mempengaruhi proses keamanan paket data yang akan ditransmisikan. Tampilan *database* dan *openfire server* saat dijalankan terlihat pada Gambar 5 dan 6.

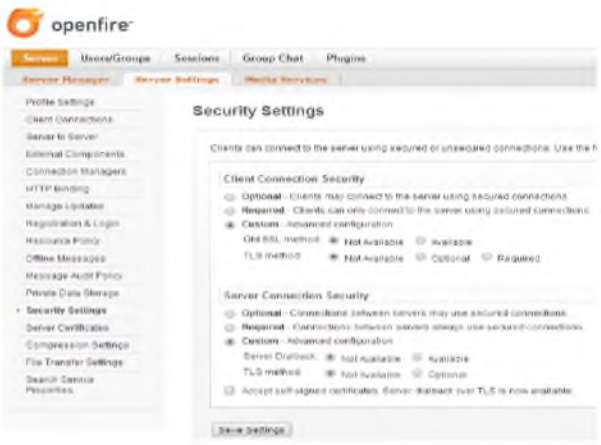


Gambar 5. *Database serverRunning*



Gambar 6. *Openfire serverRunning*

Setelah *openfire server* berjalan, selanjutnya adalah konfigurasi *TLS/SSL* untuk di-*disable*, melalui browser dengan mengakses URL :http://www.192.168.137.1:9090, Tampilan halaman *web server* terlihat pada Gambar 7.



Gambar 7. Konfigurasi TLS / SSL

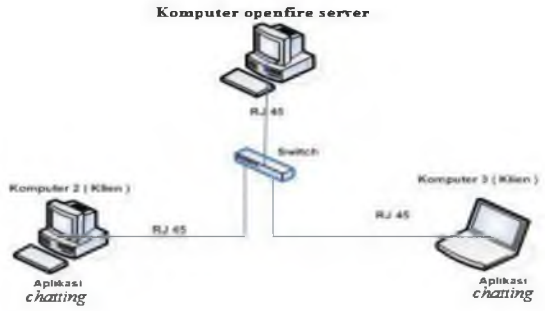
3. Hasil dan Pembahasan

Tahapan selanjutnya dari penelitian ini, uji fungsi meliputi tiga jenis pengujian, yaitu uji aplikasi *chatting* dengan menggunakan *LAN,* uji kriptografi dan pengujian berfungsi untuk mengetahui paket data yang dikirimkan (*TCP Follow Stream)* dari komputer 1 ke komputer 2 dengan menggunakan *tools Wireshark*.

3.1 Uji Aplikasi *Chatting* Pada Jaringan *LAN*

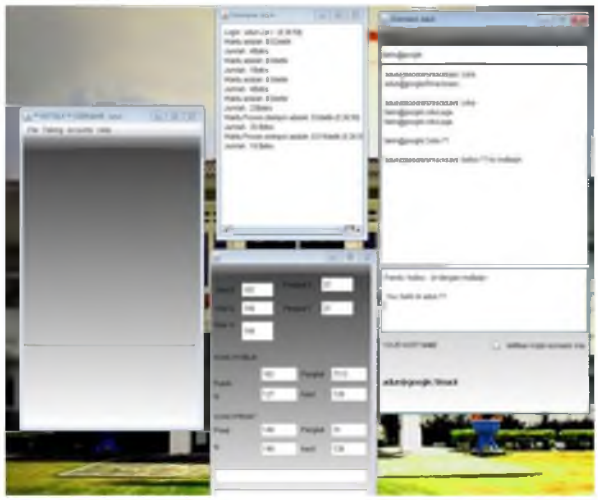
Uji Aplikasi *chatting* merupakan salah satu cara untuk mengetahui dan memastikan bahwa aplikasi *chatting* berjalan dengan lancar dengan menggunakan jaringan lokal dan menggunakan media tramsmisi kabel *LAN,* konektor RJ45 dan*switch.*Pengujian ini menggunakan 3 komputer yang sudah diberi alamat *IP* agar dapat saling terhubung. 1 buah komputer dijadikan server, dan 2 buah komputer dijadikan sebagai client. Setelah masing–masing komputer mempunyai alamat *IP*, selanjutnya ke – 3 komputer tersebut dihubungkan menggunakan switch dengan menggunakan media transmisi kabel *LAN* dan konektor RJ45.

Untuk memastikan ketiga komputer sudah saling terhubung dengan cara melakukan "ping" pada setiap komputer ke alamat ip komputer lainnya. Proses "*ping*" dilakukan dengan membuka jendela *commnd prompt (CMD),* Setelah *cmd* terbuka ketikan ping <spasi>*IP* komputer, misalnya "ping 192.168.134.1". Apabila jawaban yang di tampilkan di cmd dengan balasan "*replay*" maka komputer telah tersambung dengan komputer lainnya. Namun apabila mendapatkan balasan "*request time out*.

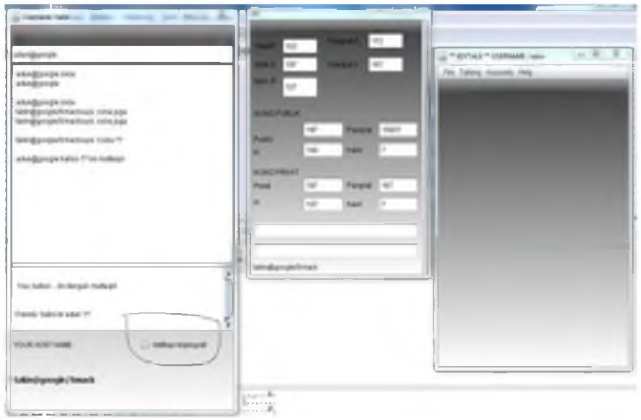


Gambar 8. Skema Uji Coba Aplikasi *Chatting*

Setelah semua komputer saling terhubung dalam jaringan *LAN,* maka selanjutnya yaitu melakukan proses uji coba aplikasi tersebut terlihat hasilnya pada gambar 9 dan 10.



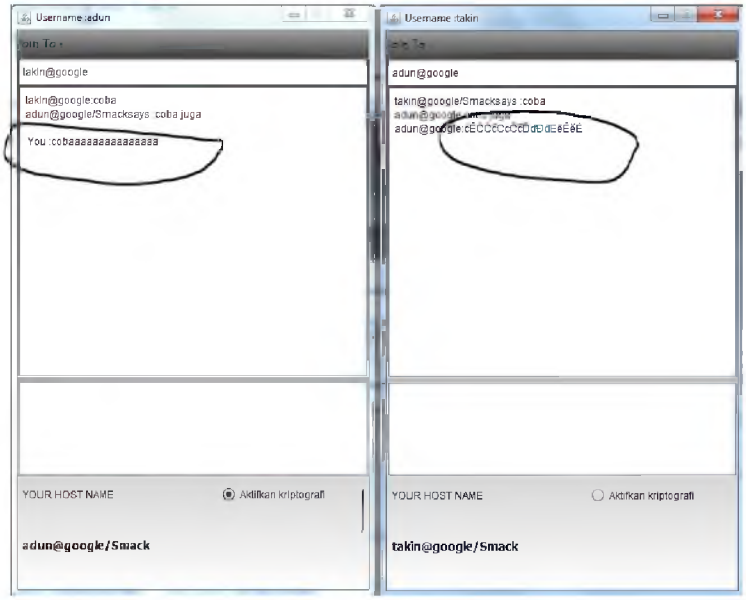
Gambar 9. Demo *Chatting* dengan *Login* adun



Gambar 10. Demo *Chatting* dengan *Login* takin

3.2 Uji Kriptografi

Uji kriptografi merupakan pengujian terhadap *file* yang akan diinputkan ke dalam aplikasi *chatting* yang sedang dijalankan. Pengujian ini dilkukan untuk mendapatkan data proses enkripsi, dekripsi dan waktu yang yang diperlukan ketika proses kriptografi sedang berjalan dari sebelum dan sesudah.



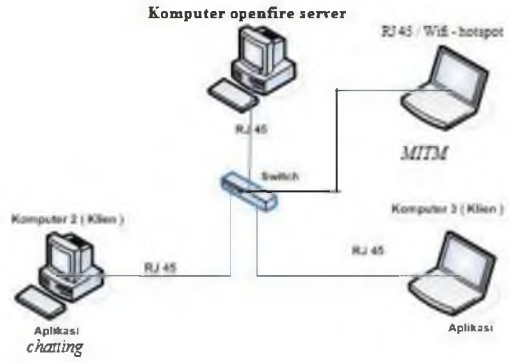
Gambar 11. Pengujian tidak aktif "*radio button*" pada Salah Satu *user*

Pada saat pesan (*message*) telah diterima maka proses dekripsi secara otomatis dengan menggunakan kunci *private* yang sudah di-*generate* oleh aplikasi pada saat *user login*. Jika terjadi pada salah satu aplikasi *chatting* yaitu *user* tidak mengaktifkan "*radio button*" maka pesan yang dikirimkan tidak dapat didekripsi kejadian tersebut terletak di sisi penerima sedangkan pada sisi pengirim tidak dapat dienkripsi sehingga yang akan di tampilkan pada layar *chatting* hanyalah sebuah *string (chipper)* yang tidak dapat dimengerti oleh manusia kecuali mesin komputer. Pada Gambar 11 menjelaskan bahwa pada *user* takin tidak mengaktifkan kriptografi yang telah tersedia dengan ini menandakan bahwa proses enkripsi berhasil dilakukan dan juga dapat mengetahui jumlah byte yang dikirimkan dan waktu proses enkripsi dan dekripsi yang dibutuhkan terlihat pada *form Log*.

3.3 Uji *TCP Follow Stream*

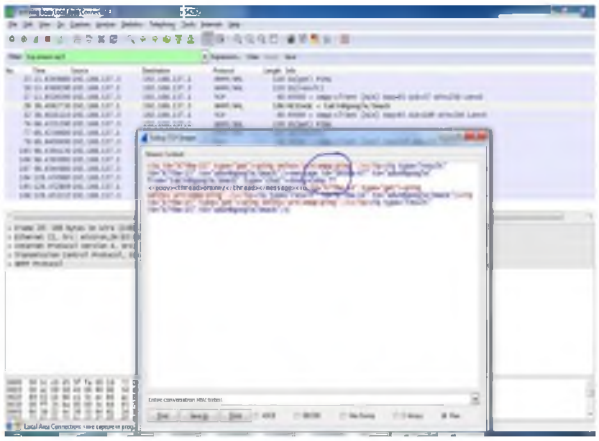
Uji ini dilakukan untuk mengetahui apakah paket data yang dikirimkan benar-benar merupakan hasil enkripsi (*chipper*) dan tidak bisa terbaca oleh serangan *sniffing/spoffing,* dengan menggunakan *tools wireshark* semua paket data yang dikirimkan dapat di-*filter* oleh komputer 4 yang tiba–tiba berdiri di antara komputer 2 dengan komputer 1 atau komputer 3 dengan komputer 1. Sehingga proses ini sering disebut dengan *MITM (Man in the middle attact).*

Dalam percobaan ini, komputer 4 hanya sekedar ingin mengetahui dan memastikan apakah pesan atau *file* yang dikirimkan melalui aplikasi *chatting* merupakan hasil dari enkripsi dan dapat didekripsi kembali oleh pihak penerima. Ilustrasi dari proses *man in the middle attack* (*MITM*) dapat terlihat pada Gambar 12.



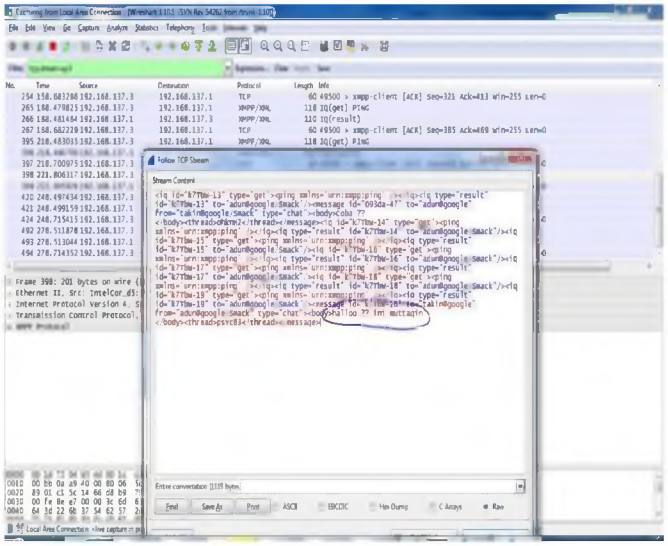
Gambar 12. Proses *MITM (man in the middle attack)*

Berikut ini adalah proses yang dilakukan oleh *tools wireshark* dalam menyaring semua paket data yang terkirim melalui jaringan yang terhubung pada komputer 2 dan 3 melalui *switch.*



Gambar 13. Proses *filter* tanpa kriptografi pada *user* takin

Pada Gambar 13 terlihat bahwa *wireshark* mampu menyadap atau menyaring informasi yang dikirimkan melalui aplikasi *chatting* ke alamat tujuan dengan tidak mengaktifkan kriptografi proses berlangsung dari *user* takin ke adun.



Gambar 14. Proses *filter* tanpa kriptografi pada *user* adun

Pada Gambar 14 terlihat bahwa pengiriman paket data dari *user* adun ke takin dapat disaring oleh *tools wireshark*, ini dapat menyebabkan lemahnya keamanan informasi yang ditransmisikan pada jaringan *LAN/internet*. Hal ini membuat setiap *user* akan terjadi kekhawatiran dalam proses pertukaran informasi pada jaringan *LAN/internet*.

3.4 Analisa Hasil Uji Fungsi

Dari pengujian yang telah dilakukan dari 3 jenis proses pengujian yaitu pengujian aplikasi *chatting* pada *LAN*, pengujian kriptografi dan pengujian *TCP follow stream* diperoleh analisa dari hasil uji fungsi tersebut. Uji coba yang pertama adalah uji aplikasi *chatting* pada *LAN*, dilakukan dengan menggunakan 3 buah komputer, 1 komputer dijadikan sebuah *server* dan 2 buah komputer dijadikan sebagai *client* yang dihubungkan dalam jaringan lokal dengan media trasmisi kabel *LAN*, konektor RJ45dan sebuah *switch* dengan kecepatan perpindahan paket yang berupa pesan singkat (*String*) 0.01 - 0.03 detik yang terbaca di dalam *form log*. Sedangkan untuk kecepatan transmisi paket data yang berupa *file* 0.03 – 0.10.Uji coba ini menghasilkan bahwa aplikasi *chatting* berjalan lancar dan sesuai dengan harapan.

Uji Coba selanjutnya yaitu uji kriptografi dengan modifikasi algoritma *RSA*, uji coba ini dilakukan pada saat *user* mengaktifkan "*radio button*" yang terdapat pada *form chatting* sehingga setiap pesan yang dikirimkan akan dirubah menjadi *chipper* terlebih dahulu selanjutnya baru dikirimkan ke alamat tujuan. Uji coba enkripsi paket data yang berupa *string* membutuhkan waktu 0.014 detik dengan jumlah data 25 bytes dan membutuhkan waktu untuk dekripsi 0.08 detik.Hasil ini di uji coba 1 x.

Dalam proses kriptografi initerlihat bahwa waktu yang dibutuhkan untuk enkripsi lebih lama dibandingkan dengan waktu dekripsi. Hal ini disebabkan karena proses enkripsi merupakan proses memperluas atau mengurainya menjadi lebih banyak dari data semula sehingga di sinilah membutuhkan waktu cukup lama. Sedangkan proses dekripsi merupakan proses menghimpun kembali menjadi lebih sempit atau memperkecil jumlah karakter yang ada pada *chipper*. Sehingga ini menyebabkan prosesnya menjadi setengah dari waktu enkripsi.

Uji coba selanjutnya adalah pada *TCP follow stream*, uji coba ini dilakukan adalah untuk memastikan bahwa paket data yang dikirimkan melalui komputer 2 ke komputer 3 merupakan asli dari sumber ke tujuannya. Selain itu paket data yang dikirimkan terbukti bahwa dengan menggunakan aplikasi *chatting*. *Tools* ini dapat membaca dari setiap pengiriman paket data yang dikirimkan melalui komputer 2 ke komputer 3 dan sebaliknya.

4. Kesimpulan dan Saran

Kesimpulan yang diperoleh dari hasil analisa pengujian aplikasi *chatting* adalah sebagai berikut :

1. Penerapan kriptografi pada aplikasi *chatting* dengan memodifikasi algoritma *RSA* telah berhasil rancang bangun dan diujikan pada jaringan komputer.
2. Penggunaan kriptografi dengan memodifikasi algoritma *RSA* terbukti memberikan keamanan terhadap serangan *sniffing/spoffing* yang terjadi pada jaringan LAN/*internet*.
3. Faktor keamanan informasi yang dikirimkan pada jaringan *LAN/internet* terbukti bahwa dapat di-*filter* oleh *tools wireshark* tetapi hanya terbaca berupa titik. Sehingga *tools wireshark* tidak mempunyai celah untuk melihat seluruh paket data yang dikirimkan.

Dalam aplikasi ini juga memiliki saran yang dapat digunakan sebagai pengembangan aplikasi ini selanjutnya, sebagai berikut:

1. Pada Aplikasi *chatting* dapat dikembangkan lagi pada sistem kriptografi yang membutuhkan waktu yang lama dalam proses enkripsi sedangkan dekripsi tidak.
2. Apabila Aplikasi *chatting* mengirimkan paket data yang berupa *file* membutuhkan sistem encode dan decode untuk merubah *file* tersebut kedalam bentuk *text* atau *string* yang akan digunakan pada proses kriptografi. Hal ini dapat dikembangkan agar tidak membutuhkan sistem encode dan decode pada proses pengiriman *file*.
3. Aplikasi *chatting* dapat dikembangkan lagi di dalam berbagai platform, misalnya dapat berjalan di sistem operasi linux, berbasis web bahkan berbasis *mobile (android, iphone, windows phone, Nokia X)*.

DAFTAR PUSTAKA

Aria, M., Rahajoeningroem, Tri. 2009. *Studi dan Implementasi Algoritma RSA untuk Pengamanan Data Transkip Akademik Mahasiswa*. Majalah ilmiah Unikom, Volume 8, Nomor 1. Bandung.

Ariyus, D., 2008. *Pengantar Ilmu Kriptografi Teori, Analisis, dan Implementasi*. Penerbit Andi Offset, Yogyakarta.

Listiyono, H., 2009. *Implementasi Algoritma Kunci Public Pada Algoritma RSA*. Jurnal Dinamikan Informatika Volume I, Nomor 2. Yogyakarta.

Lusiana, 2010. *Kriptografi Kunci Publik (Public Key Crytography)*. Jurnal Dinamika Informatika Volume 2, Nomor 1. Semarang.

Munawar, 2012. *Perancangan Algoritma Sistem Keamanan Data Menggunakan Metode Kriptografi Asimetris*. Jurnal Ilmiah Komputer dan Informatika Volume 1, Nomor 1. Bandung.

Sadikin, R., 2012. *Kriptografi Untuk Keamanan Jaringan*. Andi Offset. Yogyakarta.

Stallings, W., 2006. *Cryptography and Network Security*. Fourth Edition, Pearson Education. New Jersey.

Supriyanto, A., 2009. *Pemakaian Kriptografi Kunci Public untuk Proses Enkripsi dan Tandatangan Digital pada Dokumen e-mail*. Jurnal Dinamika Informatika Volume 1, Nomor 1. Semarang