Pembuatan Aplikasi *Chat* yang Aman Menggunakan Protokol OTR

Ahmad 13512033
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13512033@std.stei.itb.ac.id

***Abstrak*— Aplikasi *Chat* merupakan aplikasi yang paling sering digunakan oleh pengguna internet. Melalui media ini terjadi komunikasi dua arah atau lebih dimana antar pengguna saling bertukar pesan melalui media internet. Ditinjau dari segi keamanan dan kerahasiaan pesan terdapat pertanyaan, bagaimanakah metode dan protokol yang paling baik untuk menjaga keamanan dan kerasahiaan pesan yang terkirim antar pengguna aplikasi. Protokol *Off-the-Record* menjadi protokol yang dipilih untuk mewujudkan fitur keamanan dan kerahasiaan bagi sisi pengguna.**

***Kata Kunci*— *Off-the-Record*, enkripsi, dekripsi, AES, MAC, Diffie-Hellman *Key Exchange*.**

I. PENDAHULUAN

Pada saat ini sebagai media yang dapat melakukan transfer informasi, data, hasil penelitian, teknologi internet telah berkembang sangat pesat dalam beberapa dekade terakhir. Internet sukses menjadi basis dari berbagai jenis komunikasi, baik dari segi *e-commerce*, *chatting* sampai ke berbagi musik dan video.

Seiring dengan berkembangnya jumlah populasi pengguna internet, bertambah besar pula tingkat kecemasan yang harus diperhatikan terhadap sekuritas data yang berjalan diatas media yang sangat besar skalanya ini. Komunikasi yang dapat kita nikmati dari *gadget* kita ini dapat dengan mudah dipantau oleh pihak ketiga manapun dan dapat digunakan untuk kepentingan sepihak.

Terdapat dua teknik umum yang biasa digunakan untuk melindungi informasi yang berjalan diatas medium internet. Teknik tersebut adalah dengan cara menggunakan *firewalls* atau dengan menggunakan kriptografi untuk melindungi informasi yang sedang berjalan. Sistem kriptografi umum yang biasa digunakan pada pertukaran informasi di internet biasanya adalah menggunakan SSL, PGP, atau S/MIME.

SSL menjadi sistem utama yang digunakan untuk melindungi komunikasi yang bertipe *e-commerce*. PGP dan S/MIME menjadi alat yang dipakai untuk melindungi komunikasi sosial, pada umumnya melindungi informasi yang dikandung melalui pesan elektronik / *email.* Semakin pesatnya tingkat penggunaan media komunikasi berbasis internet berupa media *chatting* meningkatkan urgensi untuk menciptakan sistem keamanan yang tepat guna bagi pesan yang dikirim ` *Off-the-Record*. Melalui literatur dan percobaan yang telah dilakukan, penulis menyatakan bahwa untuk menciptakan komunikasi yang *private* di dunia *online* tidak hanya dibutuhkan enkripsi yang dapat melindungi konten pesan yang tersedia di dalamnya, tetapi juga harus menyediakan *perfect-forward-secrecy* di dalamnya. Selain itu juga perlu disediakannya fitur untuk mengautentikasi lawan pembicara untuk meyakinkan bahwa lawan bicaranya adalah orang yang tepat.

Makalah ini akan mengkaji mengenai teori yang telah berkembang dan mendasari mengenai protokol OTR yang akan kami gunakan pada bagian 2. Setelah itu pada bagian 3 akan dijabarkan solusi yang ditawarkan untuk pembuatan aplikasi *chat* yang aman. Bagian 4 akan membahas mengenai implementasi yang telah dilakukan, serta bagaimana analisisnya terhadap implementasi tersebut. Pada akhirnya makalah ini akan ditutup dengan kesimpulan yang akan dicakup pada bagian 5.

II. DASAR TEORI

2.1 Tanda Tangan Digital

Tanda tangan digital adalah pesan elektronik yang secara unik digunakan untuk mengidentifikasi pengirim sebuah pesan. Untuk mengidentifikasi pengirim pesan, akan digunakan menyerupai teknik enkripsi kunci publik sehingga pengirim harus memiliki dua kunci, yaitu kunci pribadi dan kunci publik. Kunci pribadi tidak akan diberitahukan kepada siapapun nilainya, sedangkan kunci publik tersebut akan disebarluaskan nilainya kepada setiap orang.

Tanda tangan digital adalah pengganti tanda tangan secara manual yang bersifa elektronik dan mempunyai fungsi sama dengan tanda tangan manual. Tanda tangan

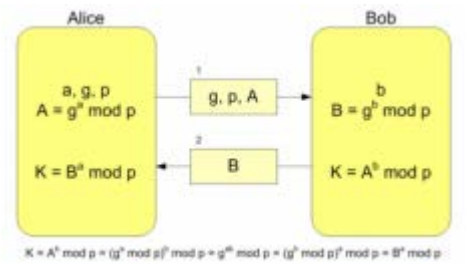
digital juga merupakan rangkaian bit yang diciptakan ndegan melakukan komunikasi elektronik melalui fungsi hash satu arah dan kemudian melakukan enkripsi pesan dengan kunci pribadi pengirim. Tanda tangan digital memiliki sifat yang unik untuk masing-masing dokumen, sehingga dengan adanya sedikit perubahan pada dokumen dapat menghasilkan tanda tangan digital yang jauh berbeda.

Cara yang digunakan untuk memastikan bahwa surat yang dikirm menggunakan tanda tangan digital telah terautentikasi dengan benar adalah dengan mengecek tanda tangan yang ada di dalam surat tersebut. Akan dilakukan tahap verifikasi tanda tangan yaitu dengan cara melakukan dekripsi serta hash terhadap pesan yang telah dikirim. Kemudian hasil perhitungan tadi dicek terhadap nilai dari tanda tangan yang tertera. Apabila memiliki nilai yang sama maka pesan berhasil terautentifikasi.

2.2 *Diffie-Hellman Key Exchange*

D-H *Key Exchange* merupakan protokol kriptografik yang memungkinkan kedua pihak yang bertukar informasi dapat secara bersama meciptakan sebuah kunci rahasia bersama melalui sebuah jalur komunikasi yang tak aman. Kunci hasil pertukaran ini akan digunakan untuk mengenkripsi pesan menggunakan kunci cipher simetris. Metode ini ditemukan pada tahun 1976 oleh Whitfield Diffie dan Martin Hellman.

Secara umum metode D-H *Key Exchange* dapat digambarkan oleh skema diagram sebagai berikut.



Gambar 2.1 Diagram D-H *Key Exchange*

Alice dan Bob sepakat menggunakan bilangan prima p dan bilangan acak g yang akan digunakan sebagai basis. Alice memilih satu bilangan natural acak a dan mengirim g^amod p ke Bob. Bob memilih satu bilangan natural acak b dan mengirim g^bmod p ke Alice. Alice menghitung $(g^a \text{mod } p)^b$ dan Bob menghitung $(g^b \text{mod } p)^a$. Alice dan Bob akan mendapatkan nilai yang sama dari hasil perhitungan.

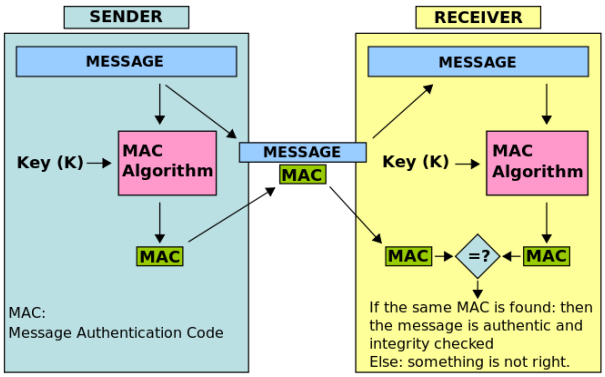
Nilai yang didapat dari perhitugnan tersebut adalah nilai dari kunci yang akan digunakan oleh Alice dan Bob untuk mengenkripsi dan mendekripsi pesan mereka. Unutk meningkatkan nilai sekuritas dari pesan yang dikirim maka Alice dan Bob harus memilih nilai a, b, dan p yang besar agar kunci mereka sulit dipecahkan oleh pihak ketiga.

Protokol D-H *Key Exchange* dinilai aman dari pihak ketiga apabila variabel g, a, b, dan p dipilih dengan baik. Tetapi metode ini rentan terhadap serangan bertipe *man in the middle attack*. Serangan ini memiliki pengertian bahwa terhadap pihak ketiga yang mengintersepsi nilai milik Alice yang dikirim ke Bob, dan mengirim nilai miliknya sendiri ke Bob dan vice versa.

Setelah proses ini pihak ketiga dapat dengan mudah mendekripsi pesan yang dikirim Alice, membaca ataupun bahkan mengubah nilainya. Kemudian pesan tersebut akan dienkripsi kembali ke Bob. Kelemahan ini terjadi karena metode pertukaran ini tidak bisa mengautentifikasi kedua pihak yang berkomunikasi. Agar lebih aman proses autentifikasi perlu diterapkan pada D-H *Key Exchange* untuk memastikan bahwa lawan bicara yang sedang berlangsung adalah orang yang tepat.

2.3 *Message Authentication Code*

Mekanisme yang menyediakan pemeriksaan integritas pesan berdasarkan atas kunci rahasia/privat biasa disebut juga dengan *Message Authentication Code* (MAC). Biasanya, MAC digunakan ketika kedua pihak membagi sebuah kunci rahasia/privat untuk melakukan autentikasi terhadap pesan yang ditransmiksikan antar pihak tersebut. Berikut merupakan skema umum dari MAC.



Gambar 2.2 Skema umum MAC

Ada beberapa macam MAC, salah satunya adalah *Keyed-hash Message Authentication Code* (HMAC). HMAC adalah teknik MAC yang memanfaatkan fungsi hash terhadap pesan dan kemudian mengenkripsi pesan dan tersebut dengan sebuah kunci private. mAC sendiri adalh teknik autentikasi pesan dengan membandingkan nilai *authentication tag* yng telah dihitung oleh pengirim dengan *authentication tag* yang dihitung sendiri oleh penerima.

HMAC didasari karena fungsi *hash* tidak memungkinkan pengunaan kunci ketika menghitung nilai

digest-nya. Secara umum algoritma HMAC dapat dijelaskan dengan persamaan berikut.

$$HMAC_K(m) = h((K \oplus opad) \square h((K \oplus ipad) \square m))$$

K adalah nilai yang menandakan kunci privat yan diketahui oleh pengirim dan penerima, h adalahhh fungsi hash yang digunakan, m adalah pesan yang akan diautentikasi, opad adalah 0x5c5c...5c dan ipad adalah 0x3636...36 dengan panjang yang sama.

2.4 Enkripsi dalam OTR

Dalam protokol OTR, dinginkan sebuah skema enkripsi yang cukup lunak. Skema enkripsi yang diinginkan ini harus bersifat *malleable* yang berarti dapat dengan mudah merubah nilai dari *ciphertext* sehingga dapat mempengaruhi dengan besar arti yang terkandung di dalam *plaintet*, bahkan ketika tidak diketahui kunci privatnya.

Sifat yang dimiliki oleh skema enkripsi bertipe block cipher sesuai dengan skema enkripsi yang diinginkan pada protokol OTR. Sangat sulit menghasilkan *ciphertext* yang berkorespondensi dengan *plaintext* yang berarti tanpa mengetahui kunci dari enkripsi terssebut. Bahkan apabila terdapat pihak ketiga yang menginterupsi pesan yang dikirim, dan berusaha untuk merubah nilai dari *ciphertext*, maka nilai tersebut akan kemungkinan besar terdekripsi menjadi nilai bit yang acak.

Tetapi dalam hal ini kita akan menggunakan stream cipher dibandingkan block cipher, karena pada kasus block cipher akan sulit dilakukan autentifikasi pesan karena kekompleksan pesan yang dihasilkan oleh block cipher apabila terdapat perubahan nilai pada *ciphertext*-nya. Pada OTR ingin dipastikan bahwa autentikasi pesan dapat dihasilkan sehingga melalui stream cipher dampak tersebut dapat dilihat dengan jelas.

III. PROTOKOL OFF-THE-RECORD

Pada bagian ini penulis akan menjabarkan mengenai protokol OTR yang akan digunakan sebagai **solusi** untuk mengembangkan aplikasi *chatting* yang aman. Penulis akan menjelesakan mengenai tahapan apa saja yang dibutuhkan untuk menciptakan protokol komunikasi yang aman sesuai dengan landasan teori yang telah mendasari di tahap sebelumnya.

3.1 Tahapan Enkripsi

Pertama-tama, harus dipastikan bahwa pesan yang dikirim telah terjaga dan bersifat privat, oleh karena itu pesan tersebut harus di enkripsi. Sesuai dengan ywbbang telah dibahas sebelumnya kita akan menggunakan tahapan enkripsi yang *malleable*. Algoritma stream cipher yang sesuai dan merupakan standar umum untuk enkripsi tersebut merupakan algoritma AES. Adapun untuk mendapatkan nilai dari kunci enkripsi-dekripsinya didapatkan menggunakan pertukaran kunci *Diffie-Hellman*.

Untuk menjamin bahwa nilai dari kuncinya sama dan memiliki usia yang muda maka Alice dan Bob dapat berulang kali membuat perjanjian *Diffie-Hellman* yang baru, tidak tergantung dengan nilai terdahulu yang dimiliki oleh kedua belah pihak. Pada titik ini, semua pesan lama yang telah terenkripsi akan mustahil dibaca lagi oleh Alice dan Bob karena mereka telah membuang kunci yang dimiliki olehnya. Sifat dari *perfect-forward-secrecy* berhasil didapat, seluruh pesan yang dienkripsi dengan nilai kunci sebelumnya sekarang tidak dapat dibaca.

Untuk memperbesar tingkat keamanan dari protokol ini, maka Alice dan Bob harus membuat perjanjian untuk *shared key* sebanyak mungkin. Hal ini mungkin dicapai karena perhitungan menggunakan algoritma *Diffie-Hellman* merupakan perhitungan yang mudah. Oleh karena itu hampir semua komputer akan dengan mudah melakukan komputasi untuk menentukan kunci baru setiap waktunya.

Tahapan enkripsi ini hanya akan dilakukan di sisi pengguna (*client)* tanpa melibatkan sisi dari dari *server* sama sekali. *Server* hanya bertindak sebagai perantara antar pengguna. Pada umumnya aplikasi berbasis chat hanya mengenkripsi pesan yang dikirim menggunakan internet dengan cara SSL saja, tetapi kepastian apakah di *server* data tersebut akan dibaca oleh pihak vendor atau tidak merupakan pertanyaan yang tidak terkuak jawabannya.

Oleh karena itu untuk aplikasi chat yang aman ini seluruh tahap dekripsi dan enkripsi hanya aka dilakukan di sisi pengguna saja, dan *server* hanya akan digunakan sebagai perantara antar kedua pihak yang sedan berkomunikasi.

3.2 Merubah Kunci

Pada tahap enkripsi telah disebutkan bahwa untuk meningkatkan tingkat sekuritas dari protokol ini dibutuhkan langkah untuk merubah kunci sesering mungkin. Langkah ini harus diatasi dengan baik karena terdapat kemungkinan ketika Alice sedang mengirim pesan kepada Bob, dan pada saat itu Alice sedang merubah kuncinya menjadi kunci yang baru.

Karena pesan yang lama baru sampai ketika Alice telah merubah kuncinya, maka Alice tidak sempat mendekripsi pesan yang dikirim oleh Bob menggunakan kunci yang baru saja dibentuknya.

Oleh karena itu, Alice harus mengingat kunci yang

lama sampoai Alice menerima pesan dari Bob bahwa sekarang adalah waktunya untuk menggunakan kunci yang baru dan melupakan kunci yang lama dengan asumsi bahwa seluruh pesan yang dikirm oleh Bob akan sampai secara terurut.

3.3 Autentifikasi

Berdasarkan pembahasan yang telah dibaha di bagian 2, telah disebutkan bahwa protokol OTR akan menggunakan MAC sebagai teknik untuk mengautentikasikan setiap pesan yang diterima. Untuk menghasilkan nilai kunci dari MAC, diterapkan fungsi hash kepada nilai dari kunci dekripsi. Hal ini menjamin bahwa siapapun yang dapat membaca pesan yang dikirm juga mampu menrubah dan memperbaharui nilai MAC.

Sebagai contoh, apabila Eve berhasil menemukan nilai dari kunci enkripsi dan dapat mendekripsi pesan, Eve tidak dapat meyakinkan kepada siapapun bahwa yang membaca pesan tersebut bukanlah Eve melainkan sebagai pembaca pesan aslinya.

Nilai kunci enkripsi yang diperoleh untuk MAC ini sendiri didapat menggunakan pertukaran pesan D-H yang juga harus diautentikasi nilainya menggunakan tanda tangan digital. Dapat dilihat bahwa pendekatan untuk autentifikasi ini cukup rumit karena digunakannya dua teknik autentifikasi yaitu menggunakan tanda tangan digital yang berujung pada penggunaannya teknik MAC.

IV. IMPLEMENTASI DAN ANALISIS

Aplikasi *Chatting* yang penulis kembangkan didasarkan pada arsitektur *server* berbasis *server-client*. Pengembangan aplikasi ini diprogram menggunakan bahasa pemrograman Java. Pada bagian 4 akan dijelaskan mengenai proses implementasi secara detail dari sisi *server* serta sisi *client*.

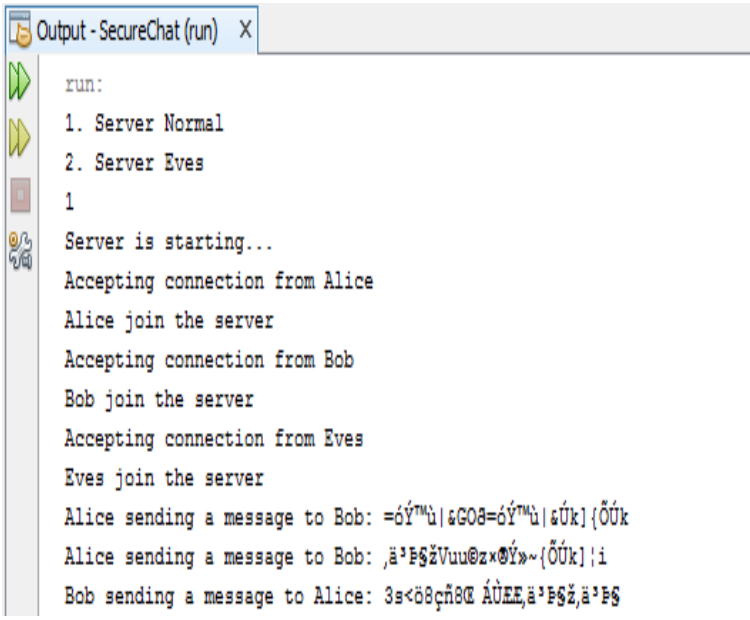
4.1 *Server*

Untuk mengembangkan *server* pada aplikasi chat ini penulis menggunakan Socket yang disediakan pada pustaka bahasa Java, dengan protokol untuk komunikasinya adalah protokol TCP/IP. Penulis memilih protokol komunikasi ini karena TCP/IP merupakan protokol yang *reliable, connection-oriented*, serta murah untuk digunakan.

Pada sisi pengembangan *server*, logika yang dipasang di *server* ada beberapa fitur. Berikut merupakan fitur yang diterapkan untuk aplikasi ini.

1. Menyimpan seluruh address dari *client*
2. Perantara komunikasi antara *client*-to-*client*
3. Menyimpan *Log* yang berisi tentang seluruh aktifitas yang ernah dijalankan oleh *client*.
4. Menyediakan fitur Eve untuk melakukan pengetesan terhadap *man in the middle attack.*

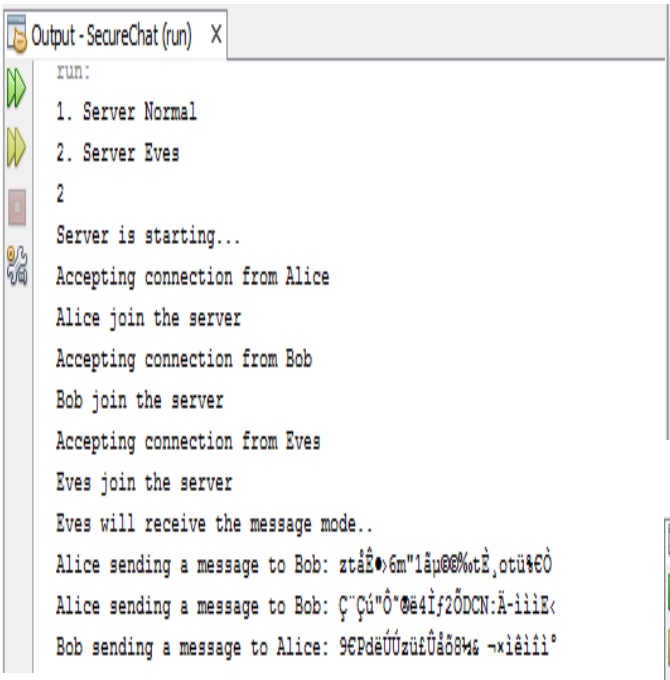
Karena menggunakan *protokol Off-the-Record*, maka *server* tidak perlu memperhatikan fitur keamanan dari segi enkripsi-dekripsi, serta autentikasi pesan. Seluruh fungsi yang mengatur kemanan dari pesan yang terkirim antar dua pihak disimpan di sisi logika tiap tiap *client*. Berikut merupakan contoh implementasi untuk sisi *server* dengan mode normal.



Gambar 4.1 Screenshot Implementasi *Server*

Pada gambar diatas terdapat dua mode untuk pengetesan *server*, yaitu mode Normal dan mode Eves. Pada mode Normal, *server* hanya bertindak sebagai perantara antara dua *client* yang sedang berkomunikasi, sedangkan pada mode Eves, diperbolehkan oleh serer kepada *client* Eves untuk menerima pesan yang dikirim antar *client* (Alice dan Bob).

Pada mode normal ditunjukkan bahwa setiap komunikasi antara *client* melalui perantara *server* memiliki pesan yang masih terenkripsi dengan *server* tidak mengetahui kunci dekripsi yang diterapkan. Hanya tiap *client* saja yang memilki *shared secret* tersebut. Sehingga dari sisi kerahasiaan, pesan sangat terjamin kontennya di setiap sisi *client*. Untuk mode Eves berikut adalah tampilan dari program.



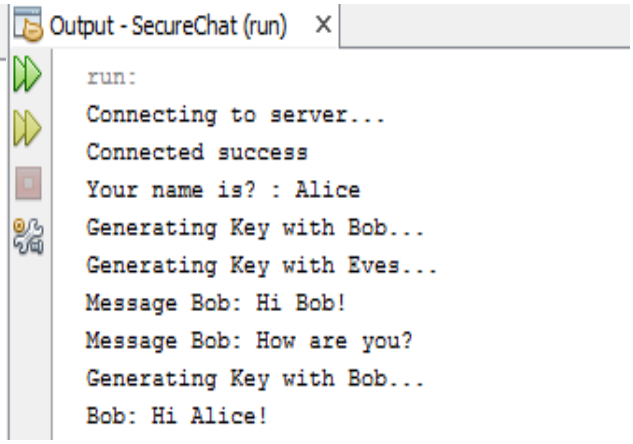
Gambar 4.2 Screenshot 2 Implementasi *Server*

Terdapat pula mode 2 yaitu mode Eves yang memberikan simulasi untuk *man in the middle attack*. Pada mode ini, data yang seharusnya hanya dikirim antara Alice dan Bob juga akan dikirm ke Eves sehingga dia dapat melakukan manipulasi dan terhadap data dan berusaha merusak autentikasi dari pesan. Penjelasan mengenai keberhasilan pembobolan dari keamanan inin akan dijelaskan di bagian *client*.

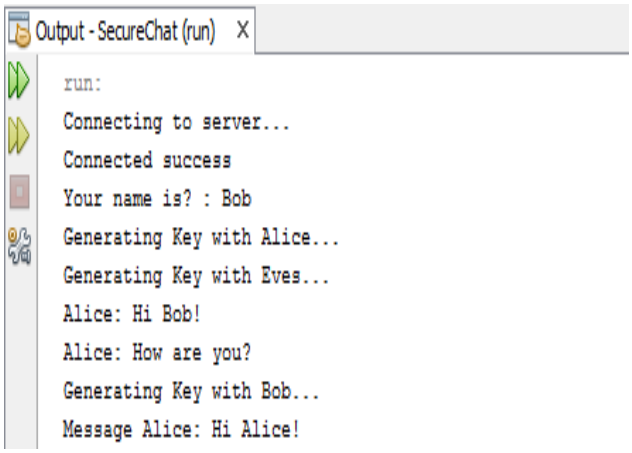
4.2 *Client*

Karena aplikasi *chatting* yang dikembangkan berdasarkan protokol OTR, maka keseluruhan fitur keamaanan yang bertipe enkripsi-dekripsi pesan, serta autentifikasi dari pengirim akan dilakukan di sisi *client*. Dalam pengimplementasian program chatting ini, digunakan bahasa pemrograman yang sama yaitu Java, serta untuk sisi jaringan digunakan Socket TCP/IP dengan alasan pengembangan yang sama dengan sisi *server*.

Karena aspek waktu pengerjaan dan fokus dari makalah ini, penulis menggunakan pustaka dan *open source program* untuk meninjau sisi fungsi kriptografi. Untuk menerapkan enkripsi dan dekripsi digunakan algoritma AES, kemudian algoritma RSA sebagai tanda tangan digital, dan terakhir SHA1-HMAC untuk penerapan autentikasi dengan MAC. Adapun untuk menghasilkan bilangan acak digunakan pustaka yang merupakaan bawaan dari bahasa Java.



Gambar 4.3 Screenshot Implementasi *Client* Alice



Gambar 4.4 Screenshot Implementasi *Client* Bob

Gambar diatas adalah implementasi program dari aspek pengguna. Pada awalnya *client* akan melakukan koneksi ke *server* dan mendaftarkan dirinya sebagai salah satu pengguna yang sedang aktif. Kemudian *server* akan memberikan notifikasi kepada setiap pengguna yang aktif bahwa Alice telah masuk ke *Server* dan siap untuk berkmunikasi. Pada tahap itu akan dihasilkan bilangan acak sebagai kunci dari tiap-tiap pengguna. Seluruh pengguna yang sedang aktif (Bob dan Eves) akan berkomunikasi dengan Alice dengan menerapkan perjanjian D-H *key exchange* untuk menentukan nilai kunci enkripsi-dekripsi untuk algoritma AES.

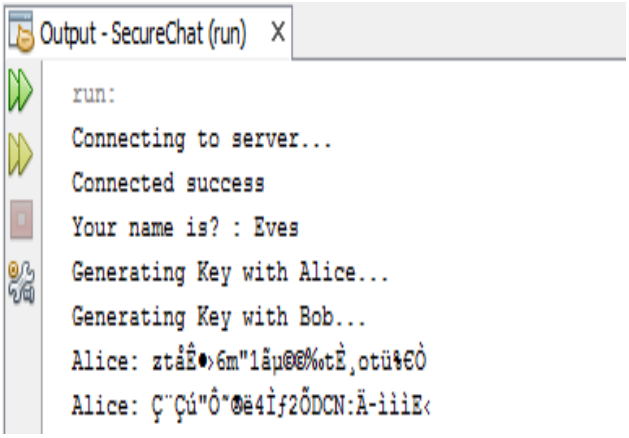
Setelah ditentukan nilai dari *shared secret key* untuk tiap-tiap pasang *client*, maka *client* dapat memulai komunikasi. Pada gambar ini ditunjukkan bahwa Alice mengirim pesannya kepada Bob sebanyak dua pesan, dimana setiap pesan tersebut akan dienkripsi menggunakan kunci yang diperoleh pada perjanjian di awal. Selain itu pesan akan diberikan *authentication tag* yaitu nilai dari MAC. Pesan dari Alice akan dikirim ke *server*.

Setelah pesan Alice sampai di *server*, maka pesan akan diteruskan ke Bob. Pesan yang sudah sampai di Bob akan

didekripsi menggunakan *shared secret key* antara Bob dan Alice. Selain itu Bob akan membaca *tag* yang telah ditandai dalam pesan dan melakukan konversi autentifikasi menggunakan metode MAC dari nilai pesan yang terkirim dari *server* oleh Alice. Apabila nilai MAC yang dihitung sama dengan nilai MAC yang terdapat pada pesan, maka dapat dibuktikan bahwa Bob telah terautentifikasi benar.

Untuk menjamin *perfect-forward-secrecy*, maka Alice dan Bob harus menentukan nilai baru dari kunci yang dimiliki oleh mereka, dalam program ini telah diatur bahwa tiap dua menit semenjak pertama kali dibentuknya kunci pertama maka akan dibentuk lagi kunci yang baru menggunakan perjanjian *Diffie-Hellman.*

Berikut merupakan implementasi program dari sisi Eves pada mode *server* 2 yang mensimulasikan *man in the middle attack.*



Gambar 4.4 Screenshot Implementasi *Client* Eves

Pada mode ini dapat dilihat bahwa Eves berhasil menyadap komunikasi antara Bob dan Alice, dimana seharusnya pesan yang dikirm hanya ditujukan kepada pihak Bob saja. Tetapi pada program dapat ditunjukkan karena menggunakan protokol OTR, pesan yang sampai ke pihak Eves masih terenkripsi dan tidak dapat didekripsi menggunakan kunci apapun yang dimiliki Eves saat itu. Aspek ini menunjukkan bagaimana amannya aplikasi *chatting* dengan protokol OTR ini.

Walaupun pada akhirnya Eves dapat menerima pesan yang berhasil didekripsi dengan cara apapun. Eves tidak dapat mengelabui pesan kepada Alice maupun Bob, karena ketika Eves mengirim nilai MAC kepada salah satu diantara mereka maka akan dengan pasti bahwa pesan tersebut terautentifikasi salah. Selanjutnya lagi apabila dilakukan pengecekan terhadap seluruh *shared-key* yang dimiliki Bob atau Alice maka dapat dibuktikan bahwa terdapat penyadap bernama Eves yang beusaha mengelabui pesan yang dikirim.

Aspek keamanan yang ditunjukkan melalui implementasi program *chatting* ini memberikan ekspektasi dan nilai keamanan yang tinggi bagi pihak *client.* Dapat dibuktikan dengan implementasi yang dilakukan pada screenshot program diatas. Tetapi terdapat biaya yang harus dibayar dengan melakukan pengimplementasian metode OTF ini.

Untuk menghasilkan nilai dari kunci yang selalu unik setiap waktunya dibutuhkan biaya yang cukup besar untuk komputasi apabila terdapat banyak *client* yang terkoneksi pada satu jaringan yang besar. Komunikasi yang terjadi akan sangat banyak apabila jumlah *client* besar, hal ini menyebabkan beratnya upaya di sisi *server* untuk menjamin komunikasi yang baik. Serta dibutuhkan jaringan yang lancar di sisi *client* agar dapat terus melakukan pengecekan dan pembaharuan nilai kunci terhadap semua *client* yang terdaftar dengannya.

V. SIMPULAN

Aplikasi *Chatting* membutuhkan biaya komunikasi yang cukup tinggi baik dari segi jaringan komunikasi serta aspek keamanan dan kerahasiaan yang harus terus terjaga. Protokol *Off-the-Record* menyediakan protokol yang menjamin aspek kerahasiaan informasi secara lengkap di sisi *client*. Aspek yang dicakup merupakan enkripsi serta dekripsi pesan dimana kunci hanya dimiliki oleh sisi *client*, serta autentifikasi pesan yang diterima oleh penerima dengan menggunakan MAC. Biaya pengimplementasian harus dibayar mahal menggunakan protokol OTF in karena akan terjadi banyak sekali jumlah konmunikasi antar *client* guna menciptakan saluran komunikasi yang aman dan bersifat rahasia.

REFERENCES

National Institute of Standars and Technology. Announcing the advanced encryption standard (AES). Federal Information Processing Standards Publication 197, November 2001.

Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2): 120-126, 1978.

B. Nikita, G. Ian, B. Eric. *Off –the-Record Communication, or Why Not To Use PGP*. October 28, 2004.

Munir,Rinaldi. *Algoritma RSA.* Bandung: Penerbit Informatika. 2015.

Munir,Rinaldi. *Algoritma Diffie-Hellman.* Bandung: Penerbit Informatika. 2015.

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, 11 Mei 2015

ttd

Ahmad 13512033