PERANCANGAN MEDIA PENGIRIMAN PESAN TEKS DENGAN PENYANDIAN PESAN MENGGUNAKAN ALGORITMA RC4 BERBASIS WEB

Henri Pandiangan, Salomo Sijabat

Program Studi Teknik Informatika
STMIK Pelita Nusantara Medan, Jl. Iskandar Muda No 1 Medan, Sumatera Utara 20154, Indonesia
henri.pandiangan@gmail.com, slm.jabat@gmail.com

Abstrak

Penelitian ini membahas tentang Implementasi algoritma RC4 Untuk Aplikasi Enkripsi dan Dekripsi Pesan Teks yang dikirim melalui jaringan LAN maupun internet melalui jaringan. Algoritma RC4 merupakan salah satu algoritma kunci simetris berbentuk stream cipher yang memproses unit atau input data, pesan atau informasi pada satu saat. Unit atau data pada umumnya sebuah byte atau bahkan bit (byte dalam hal RC4). Algoritma ini tidak harus menunggu sejumlah input data, pesan atau informasi tertentu sebelum diproses atau menambahkan byte tambahan untuk mengenkrip. RC4 mempunyai sebuah S-Box, S0,S1,……,S255, yang berisi permutasi dari bilangan 0 sampai 255. Menggunakan dua buah indeks yaitu i dan j di dalam algoritmanya. Indeks i digunakan untuk memastikan bahwa suatu elemen berubah, sedangkan indeks j akan memastikan bahwa suatu elemen berubah secara random. Intinya, dalam algoritma enkripsi metode ini akan membangkitkan pseudo random byte dari key yang akan dikenakan operasi Xor terhadap plaintext untuk menghasilkan ciphertext. Dan untuk menghasilkan plaintext semula, maka ciphertext nya akan dikenakan operasi Xor terhadap pseudo random bytenya (Kurniawan,Yusuf, 2004).

Keywords : *Algoritma RC4, Chatin*

I. PENDAHULUAN

Aplikasi Chating merupakan suatu jenis dari perkembangan suatu teknologi. Dengan kecanggihan teknologi saat ini, fungsi aplikasi chating tidak hanya sebagai alat komunikasi biasa, tetapi manusia juga dapat mengirimkan foto dan lain-lainya. Dampak yang ditimbulkan dari Chating mungkin tidak disadari sama sekali. Selain memudahkan dalam berkomunikasi sebagai dampak positif yang manusia dapatkan, terdapat pula dampak negatif yang manusia dapatkan sebagai akibat menggunakan *Chating* ini.

Kerahasiaan dan keamanan saat melakukan pertukaran data adalah hal yang sangat penting dalam komunikasi data, baik untuk tujuan keamanan bersama, maupun untuk privasi individu. Mereka yang menginginkan agar datanya tidak di ketahui oleh pihak - pihak yang sama sekali tidak berkepentingan selalu berusaha menyiasati cara mengamankan informasi yang akan dikomunikasikannya. Perlindungan terhadap kerahasiaan datapun meningkat, salah satu caranya dengan menyandikan data atau enkripsi.

Skema enkripsi yang akan dibangun pada skripsi ini menerapkan teknik pada kriptografi modern, yang menganut kerahasiaan pada kunci (*key*), sehingga keamanan enkripsi hanya tergantung pada kunci (*key*) dan tidak tergantung apakah algoritmanya diketahui orang atau tidak.

Secara umum dalam proses enkripsi-dekripsi dikenal dua macam cipher berdasarkan cara kerja penyandiannya, yaitu Stream cipher adalah suatu sistem dimana proses enkripsi dan dekripsinya dilakukan dengan cara bit per bit. Pada sistem ini aliran bit kuncinya dihasilkan oleh suatu pembangkit bit acak. Aliran kunci ini dikenakan operasi XOR dengan aliran bit-bit dari plaintext untuk menghasilkan aliran bit-bit ciphertext (Kurniawan,Yusuf, 2004).

Algoritma RC4 merupakan salah satu algoritma kunci simetris berbentuk stream cipher yang memproses unit atau input data, pesan atau informasi pada satu saat. Unit atau data pada umumnya sebuah byte atau bahkan bit (byte dalam hal RC4). Algoritma ini tidak harus menunggu sejumlah input data, pesan atau informasi tertentu sebelum diproses atau menambahkan byte tambahan untuk mengenkrip. RC4 mempunyai sebuah S-Box, S0,S1,……,S255, yang berisi permutasi dari bilangan 0 sampai 255. Menggunakan dua buah indeks yaitu i dan j di dalam algoritmanya. Indeks i digunakan untuk memastikan bahwa suatu elemen berubah, sedangkan indeks j akan memastikan bahwa suatu elemen berubah secara random. Intinya, dalam algoritma enkripsi metode ini akan membangkitkan pseudo random byte dari key yang akan dikenakan operasi Xor terhadap plaintext untuk menghasilkan ciphertext. Dan untuk menghasilkan plaintext semula, maka ciphertext nya akan dikenakan operasi Xor terhadap pseudo random bytenya (Kurniawan,Yusuf, 2004).

II. KRIPTOGRAFI

Kriptografi memiliki sejarah yang panjang dan mengagumkan. Penulisan rahasia ini dapat dilacak kembali ke 3000 tahun SM saat digunakan oleh bangsa Mesir. Mereka menggunakan hieroglyphcs untuk menyembunyikan tulisan dari mereka yang tidak diharapkan. Hieroglyphcs diturunkan dari bahasa Yunani hieroglyphica yang berarti ukiran rahasia. Hieroglyphs berevolusi menjadi hieratic, yaitu *stylized script* yang lebih mudah untuk digunakan. Sekitar 400 SM, kriptografi militer digunakan oleh bangsa Spartan dalam bentuk sepotong papirus atau perkamen dibungkus dengan batang kayu. Sistem ini disebut Scytale.

Sekitar 50 SM, Julius Caesar, kaisar Roma, menggunakan *cipher substitusi* untuk mengirim pesan ke Marcus Tullius Cicero. Pada *cipher* ini, huruf-huruf alfabet disubstitusi dengan huruf-huruf yang lain pada alfabet yang sama. Karena hanya satu alfabet yang digunakan, cipher ini merupakan substitusi monoalfabetik. *Cipher* semacam ini mencakup penggeseran alfabet dengan 3 huruf dan mensubstitusikan huruf tersebut. Substitusi ini kadang dikenal dengan C3 (untuk Caesar menggeser 3 tempat). Secara umum sistem cipher Caesar dapat ditulis sebagai berikut :

$Zi = Cn(Pi)$

Dimana Zi adalah karakter-karekter ciphertext, Cn adalah transformasi substitusi alfabetik, n adalah jumlah huruf yang digeser, dan Pi adalah karakter-karakter *plaintext*. Disk mempunyai peranan penting dalam kriptografi sekitar 500 th yang lalu. Di Italia sekitar tahun 1460, Leon Battista Alberti mengembangkan disk *cipher* untuk enkripsi. Sistemnya terdiri dari dua disk konsentris. Setiap disk memiliki alfabet di sekelilingnya, dan dengan memutar satu disk berhubungan dengan yang lainnya, huruf pada satu alfabet dapat ditransformasi ke huruf pada alfabet yang lain.

Bangsa Arab menemukan cryptanalysis karena kemahirannya dalam bidang matematika, statistik, dan lingiustik. Karena setiap orang muslim harus menambah pengetahuannya, mereka mempelajari peradaban terdahulu dan mendekodekan tulisan-tulisannya ke huruf-huruf Arab. Pada tahun 815, Caliph al-Mamun mendirikan *House of Wisdom* di Baghdad yang merupakan titik pusat dari usaha-usaha translasi. Pada abad ke-9, filsuf Arab al-Kindi menulis risalat (ditemukan kembali th 1987) yang diberi judul "*A Manuscript on Deciphering Cryptographic Messages*". Pada 1790, Thomas Jefferson mengembangkan alat enkripsi dengan menggunakan tumpukan yang terdiri dari 26 disk yang dapat diputar secara individual. Pesan dirakit dengan memutar setiap disk ke huruf yang tepat dibawah batang berjajar yang menjalankan panjang tumpukan disk. Kemudian, batang berjajar diputar dengan sudut tertentu, A, dan huruf-huruf dibawah batang adalah pesan yang terenkripsi. Penerima akan menjajarkan karakter-karakter *cipher* dibawah batang berjajar, memutar batang kembali dengan sudut A dan membaca pesan *plaintext*.

Sistem disk digunakan secara luas selama perang sipil US. *Federal Signal Officer* mendapatkan hak paten pada sistem disk mirip dengan yang ditemukan oleh Leon Battista Alberti di Italia, dan dia menggunakannya untuk mengkode dan mendekodekan sinyal-sinyal bendera diantara unit-unit. Sistem Unix menggunakan *cipher substitusi* yang disebut ROT 13 yang menggeser alfabet sebanyak 13 tempat. Penggeseran 13 tempat yang lain membawa alfabet kembali ke posisi semula, dengan demikian mendekodekan pesan. Mesin kriptografi mekanik yang disebut Hagelin Machine dibuat pada tahun 1920 oleh Boris Hagelin di Scockholm, Swedia. Di US, mesin Hagelin dikenal sebagai M-209. Pada tahun 20-an, Herbert O. Yardley bertugas pada organisasi rahasia US MI-8 yang dikenal sebagai "*Black Chamber*". MI-8 menjebol kode-kode sejumlah negara. Selama konferensi Angkatan Laut Washington tahun 1921-1922, US membatasi negosiasi dengan Jepang karena MI-8 telah memberikan rencana negosiasi Jepang yang telap disadap kepada sekretaris negara US. Departemen negara menutup MI-8 pada tahun 1929 sehingga Yardley merasa kecewa. Sebagai wujud kekecewaanya, Yardley menerbitkan buku *The American Black Chamber*, yang menggambarkan kepada dunia rahasia dari MI-8. Sebagai konsekuensinya, pihak Jepang menginstal kode-kode baru. Karena kepeloporannya dalam bidang ini, Yardley dikenal sebagai "Bapak Kriptografi Amerika".

II.1. KRIPTOGRAFI DAN SISTEM INFORMASI

Keamanan telah menjadi aspek yang sangat penting dari suatu sistem informasi. Sebuah informasi umumnya hanya ditujukan bagi segolongan tertentu. Oleh karena itu sangat penting untuk mencegahnya jatuh kepada pihak-pihak lain yang tidak berkepentingan. Untuk melaksanakan tujuan tersebutlah dirancang suatu sistem keamanan yang berfungsi melindungi sistem informasi (Rinaldi, 2011).

Salah satu upaya pengamanan sistem informasi yang dapat dilakukan adalah kriptografi. Kriptografi sesungguhnya merupakan studi terhadap teknik matematis yang terkait dengan aspek keamanan suatu sistem informasi, antara lain seperti kerahasiaan, integritas data, otentikasi, dan ketiadaan penyangkalan. Keempat aspek tersebut merupakan tujuan fundamental dari suatu sistem kriptografi(Rinaldi, 2011).

1. Kerahasiaan (*confidentiality*)

Kerahasiaan adalah layanan yang digunakan untuk menjaga informasi dari setiap pihak yang tidak berwenang untuk mengaksesnya. Dengan demikian informasi hanya akan dapat diakses oleh pihak-pihak yang berhak saja.

2. Integritas data (*data integrity)*

Integritas data merupakan layanan yang bertujuan untuk mencegah terjadinya pengubahan informasi oleh pihak-pihak yang tidak berwenang. Untuk meyakinkan integritas data ini harus dipastikan agar sistem informasi mampu mendeteksi terjadinya manipulasi data. Manipulasi data yang dimaksud di sini meliputi penyisipan, penghapusan, maupun penggantian data.

3. Otentikasi (*authentication*)

Otentikasi merupakan layanan yang terkait dengan identifikasi terhadap pihak-pihak yang ingin mengakses sistem informasi (*entity authentication*) maupun keaslian data dari sistem informasi itu sendiri (*data origin authentication*).

4. Ketiadaan penyangkalan (*non-repudiation*)

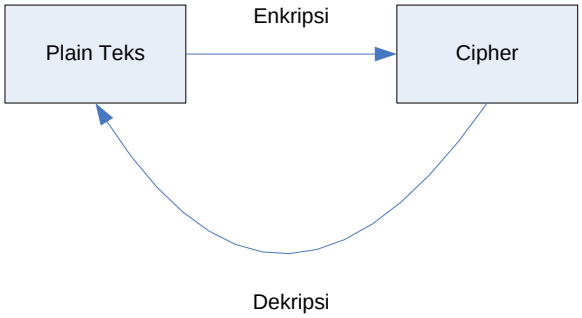
Ketiadaan penyangkalan adalah layanan yang berfungsi untuk mencegah terjadinya penyangkalan terhadap suatu aksi yang dilakukan oleh pelaku sistem informasi.

II.2. MEKANISME KRIPTOGRAFI

Suatu sistem kriptografi (kriptosistem) bekerja dengan cara menyandikan suatu pesan menjadi suatu kode rahasia yang dimengerti oleh pelaku sistem informasi saja. Pada dasarnya mekanisme kerja semacam ini telah dikenal sejak jaman dahulu. Bangsa Mesir kuno sekitar 4000 tahun yang lalu bahkan telah mempraktekkannya dengan cara yang sangat primitive (Rinaldi, 2011).

Dalam era teknologi informasi sekarang ini, mekanisme yang sama masih digunakan tetapi tentunya implementasi sistemnya berbeda. Sebelum membahas lebih jauh mekanisme kriptografi modern, berikut ini diberikan beberapa istilah yang umum digunakan dalam pembahasan kriptografi (Rinaldi, 2011).

1. *Plaintext*
   Plaintext (message) merupakan pesan asli yang ingin dikirimkan dan dijaga keamanannya. Pesan ini tidak lain dari informasi tersebut.
2. *Chipertext*
   Chipertext merupakan pesan yang telah dikodekan (disandikan) sehingga siap untuk dikirimkan.
3. *Chiper*
   Chiper merupakan algoritma matematis yang digunakan untuk proses penyandian plaintext menjadi ciphertext.
4. *Enkripsi*
   Enkripsi (*encryption*) merupakan proses yang dilakukan untuk menyandikan plaintext sehingga menjadi chipertext.
5. *Dekripsi*
   Dekripsi (*decryption*) merupakan proses yang dilakukan untuk memperoleh kembali plaintext dari chipertext.
6. *Kriptosistem*
   Kriptosistem merupakan sistem yang dirancang untuk mengamankan suatu sistem informasi dengan memanfaatkan kriptografi. Urutan-urutan proses kriptografi dapat digambarkan sebagai berikut.



Gambar 1 : Mekanisme kriptografi *(Rinaldi, 2011)*

Prosesnya pada dasarnya sangat sederhana. Sebuah plaintext (m) akan dilewatkan pada proses enkripsi (E) sehingga menghasilkan suatu ciphertext (c). Kemudian untuk memperoleh kembali plaintext, maka ciphertext (c) melalui proses dekripsi (D) yang akan menghasilkan kembali plaintext (m). Secara matematis proses ini dapat dinyatakan sebagai,

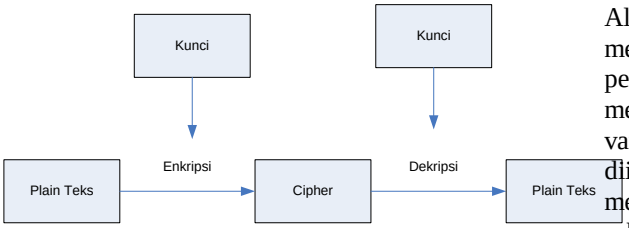
a) $E(m) = c$

$$D(c) = m$$

$$D(E(m)) = m$$

Kriptografi sederhana seperti ini menggunakan algoritma penyandian yang disebut *cipher*. Keamanannya bergantung pada kerahasiaan algoritma penyandian tersebut, karena itu algoritmanya harus dirahasiakan. Pada kelompok dengan jumlah besar dan anggota yang senantiasa berubah, penggunaannya akan menimbulkan masalah. Setiap ada anggota yang meninggalkan kelompok, algoritma harus diganti karena anggota ini dapat saja membocorkan algoritma (Rinaldi, 2011).

Kriptografi modern selain memanfaatkan algoritma juga menggunakan kunci (*key*) untuk memecahkan masalah tersebut. Proses enkripsi dan dekripsi dilakukan dengan menggunakan kunci ini. Setiap anggota memiliki kuncinya masing-masing yang digunakan untuk proses enkripsi dan dekripsi yang akan dilakukannya. Dengan demikian ada sedikit perubahan yang harus dilakukan pada mekanisme yang digambarkan pada gambar diatas menjadi seperti gambar dibawah berikut ini.



Gambar 2.: Kriptografi berbasis kunci *(Rinaldi, 2011)*

Mekanisme kriptografi seperti ini dinamakan kriptografi berbasis kunci. Dengan demikian kriptosistemnya akan terdiri atas algoritma dan kunci, beserta segala plaintext dan ciphertextnya.
Persamaan matematisnya menjadi seperti berikut,

b) $E_e(m) = c$
$D_d(c) = m$
$D_d(E_e(m)) = m$

dengan,
e = kunci enkripsi
d = kunci dekripsi

III. RC4

RC4 merupakan salah satu jenis *stream cipher*, yaitu memproses unit atau input data, pesan atau informasi pada satu saat. Unit atau data pada umumnya sebuah *byte* atau bahkan kadang kadang bit (*byte* dalam hal RC4). Dengan cara ini enkripsi atau dekripsi dapat dilaksanakan pada panjang yang variabel. Algoritma ini tidak harus menunggu sejumlah input data, pesan atau informasi tertentu sebelum diproses, atau menambahkan byte tambahan untuk mengenkrip. Contoh *stream cipher* adalah RC4, Seal, A5, Oryx, dan lain-lain. Tipe lainnya adalah *block cipher* yang memproses sekaligus sejumlah tertentu data (biasanya 64 bit atau 128 bit blok), contohnya : Blowfish, DES, Gost, Idea, RC5, Safer, Square, Twofish, RC6, Loki97, dan lain-lain (*Slamet Maryono, 2012)*

RC4 merupakan enkripsi *stream simetrik proprietary* yang dibuat oleh *RSA Data Security Inc* (RSADSI). Penyebarannya diawali dari sebuah source code yang diyakini sebagai RC4 dan dipublikasikan secara '*anonymously*' pada tahun 1994. Algoritma yang dipublikasikan ini sangat identik dengan implementasi RC4 pada produk resmi. RC4 digunakan secara luas pada beberapa aplikasi dan umumnya dinyatakan sangat aman. Sampai saat ini diketahui tidak ada yang dapat memecahkan/membongkarnya, hanya saja versi ekspor 40 bitnya dapat dibongkar dengan cara "*brute force*" (mencoba semua kunci yang mungkin). RC4 tidak dipatenkan oleh RSADSI, hanya saja tidak diperdagangkan secara bebas (*trade secret*) (*Slamet Maryono, 2012).*

Algoritma RC4 cukup mudah untuk dijelaskan. RC4 mempunyai sebuah *S-Box*, $S_0,S_1,...,S_{255}$, yang berisi permutasi dari bilangan 0 sampai 255, dan permutasi merupakan fungsi dari kunci dengan panjang yang variabel. Terdapat dua indeks yaitu i dan j, yang diinisialisasi dengan bilangan nol. Untuk menghasilkan random byte langkahnya adalah sebagai berikut (*Slamet Maryono, 2012)*:

$i = (i + 1) \bmod 256$
$j = (j + S_i) \bmod 256$
swap S_i dan S_j
$t = (S_i + S_j) \bmod 256$
$K = S_t$

Byte K di XOR dengan *plaintexs* untuk menghasilkan *cipherteks* atau di XOR dengan cipherteks untuk menghasilkan plainteks. Enkripsi sangat cepat kurang lebih 10 kali lebih cepat dari DES.

Inisialisasi S-Box juga sangat mudah. Pertama isi secara berurutan $S_0 = 0$, $S_1 = 1,...,S_{255} = 255$. Kemudian isi array 256 byte lainnya dengan kunci yang diulangi sampai seluruh array K_0, $K_1,...,K_{255}$ terisi seluruhnya. Set indeks j dengan nol, Kemudian lakukan langkah berikut :

for i = 0 to 255
$j = (j + S_i + Ki) \bmod 256$
swap S_i dan S_j

Salah satu kelemahan dari RC4 adalah terlalu tingginya kemungkinan terjadi tabel S-box yang sama, hal ini terjadi karena kunci user diulang-ulang untuk mengisi 256 bytes, sehingga 'aaaa' dan 'aaaaa' akan menghasilkan permutasi yang sama. Untuk mengatasi ini maka pada implementasinya nanti kita menggunakan hasil hash 160 bit SHA dari password kita untuk mencegah hal ini terjadi. Kekurangan lainnya ialah karena enkripsi RC4 adalah XOR antara data bytes dan *pseudo-random byte stream* yang dihasilkan dari kunci, maka penyerang akan mungkin untuk menentukan beberapa byte pesan orisinal dengan meng-XOR dua *set cipher byte*, bila beberapa dari pesan input diketahui (atau mudah untuk ditebak). Untuk mengatasinya pada aplikasinya kita menggunakan initialization vector (IV) yang berbeda-beda untuk setiap data, sehingga bahkan untuk file yang sama akan dihasilkan *ciphertext* yang berbeda. IV ini tidak perlu dirahasikan karena digunakan hanya agar setiap proses enkripsi akan menghasilkan *ciphertext* yang berbeda (*Slamet Maryono, 2012).*

Untuk lebih meningkatkan keamanan dari metoda ini dapat juga mengembangkan inisialisasi kunci yang baru yang kita sebut saja inisialisasi SK (*strengtened key*), pada proses ini kunci user di-*expand* hingga 260 byte (tetapi kemudian hanya 256 byte saja yang digunakan) dengan menggunakan SHA-1, caranya pertama kunci user dijadikan kunci, kemudian 1-20 byte pertama pada buffer diproses dengan SHA kemudian digestnya diletakan pada 20 byte pertama, kemudian diambil byte 1-40 diproses

dengan SHA dan hasilnya diletakan mulai pada byte 20, berikutnya byte 1-60 hasilnya diletakkan pada mulai byte 40, dan seterusnya. Kemudian buffer ini dienkrip dengan RC4, lalu buffer dijadikan kunci kembali, proses terakhir ini diulang sebanyak 16 kali untuk mencoba mencampur dengan baik sehingga dihasilkan kunci yang se-random mungkin.

Untuk lebih jelas tetang proses ini dapat dilihat pada listing. Penggunaan SHA pada proses inisialisasi kunci bukanlah hal yang baru, hal ini dapat dilihat pada proses inisialisasi kunci SEAL misalnya. Penggunaan proses primitif enkripsi pada inisialisasi kunci juga digunakan juga pada Blowfish ataupun Cobra-128. Secara teoritis dengan proses ini akan ekivalen dengan menggunakan kunci sebesar 2048 bit, walaupun penulis sendiri tidak yakin akan hal ini (mungkin pembaca ada yang bisa memberikan tanggapan). Metoda ini tampaknya sedikit lebih rumit dari pada inisialisasi kunci standar, tetapi pada Pentium 133 prosesnya hanya memerlukan waktu kurang sari 10ms saja. Metoda ini walaupun kami anggap lebih kuat, tetapi belum teruji sehingga dalam penerapan aplikasinya terdapat dua pilihan yaitu dengan metoda SK ini atau dengan metoda standard (*Slamet Maryono, 2012)*.

IV. ALGORITMA UMUM RC4

Algoritma RC4 adalah algoritma kriptografi simetrik. Disebut algoritma kriptografi simetrik karena menggunakan kunci yang sama untuk mengenkripsi ataupun mendekripsi suatu pesan, data, ataupun informasi. Kunci enkripsi didapat dari sebuat 256 bit *state-array (KSA)* yang diinisialisasi dengan sebuah *key* tersendiri dengan panjang 1-256 bit. Setelah itu, *state-array* tersebut akan diacak kembali dan diproses untuk menghasilkan sebuah kunci enkripsi yang akan di-XOR-kan dengan plainteks ataupun cipherteks. Secara umum, algoritma RC4 terbagi menjadi dua, inisalisasi *state-array* dan penghasilan kunci enkripsi serta pengenkripsiannya (Karina Novita Suryani, 2008). Adapun algoritma dari Enkripsi RC-4 adalah sebagai berikut ini :

Langkah 1:

```
Inisialisasi S-Box (Array S)
Jum = Len(Kunci)
i = 0
j  = 0
For  y=1 to Jum
            j = (j + S[i] + K [i mod jum]) mod
            jum
            Swap (S[i],S[j])
Next y
```

Langkah 2:

```
Lakukan Pengacakan S-Box
Jum = Len(Kunci)
i = 0
j  = 0
For  y=1 to Jum
            i = (i + 1) mod Jum
            j = (j + S[i]) mod Jum
            swap (S[i],S[j])
            Key(y) = S[(S[i]+S[j]) mod jum]
Next y
```

Langkah 3:

```
Lakukan Enkripsi
Jum = Panjang(PlainText)
i = 0
j  = 0
For  y=1 to Jum
   C(y) = Biner(P[y]) XOR Biner(Key[y])
Next y
```

Dan untuk melakukan Dekripsi maka dilakukan langkah sebagai berikut ini :

Langkah 1:

```
Inisialisasi S-Box (Array S)
Jum = Len(Kunci)
i = 0
j  = 0
For  y=1 to Jum
   j = (j + S[i] + K [i mod jum]) mod jum
   Swap (S[i],S[j])
Next y
```

Langkah 2:

```
Lakukan Pengacakan S-Box
Jum = Len(Kunci)
i = 0
j  = 0
For  y=1 to Jum
            i = (i + 1) mod Jum
            j = (j + S[i]) mod Jum
            swap (S[i],S[j])
            Key(y) = S[(S[i]+S[j]) mod jum]
Next y
```

Langkah 3:

```
Lakukan Enkripsi
Jum = Panjang(PlainText)
i = 0
j  = 0
For  y=1 to Jum
     P(y) = Biner(C[y]) XOR Biner(Key[y])
Next y
```

V. CONTOH ENKRIPSI PESAN DENGAN RC4

Untuk menunjukkan bagaimana RC4 bekerja pada tingkat dasar, mari kita state-array 4 bit. Hal ini dikarenakan akan sangat sulit menggambarkan proses RC4 secara manual dengan state-array 256 bit. Kali ini, kita akan mengenkripsi kata **HALO** dengan kunci **2573**. Pertama, kita menginisialisasi array S 4 bit sehingga terbentuk state-array S dan state-array K sebagai berikut,

Array S	**0**	**1**	**2**	**3**
Array K	**2**	**5**	**7**	**3**

Inisialisasi i dan j dengan 0 kemudian dilakukan KSA agar tercipta state-array yang acak. Penjelasan iterasi lebih lanjut dapat dijelaskan sebagai berikut.

Iterasi 1

i = 0
j = (0 + S[0] + K [0 mod 4]) mod 4
j = (j + S[i] + K[(i Mod 4)] Mod 4
= (0 + 0 + 2) mod 4 = 2
Swap (S[0],S[2])

Sehingga hasil array S adalah

Array S	2	1	0	3

Iterasi 2

i = 1
j = (2 + S[1] + K [1 mod 4]) mod 4
= (2 + 1 + 5) mod 4 =0
Swap (S[1],S[0])

Sehingga hasil array S adalah

Array S	1	2	0	3

Iterasi 3

i = 2
j = (0 + S[2] + K [2 mod 4]) mod 4
= (0 + 0 + 7) mod 4 = 3
Swap (S[2],S[3])

Sehingga hasil array S adalah

Array S	1	2	3	0

Iterasi 4

i = 3
j = (3 + S[3] + K [3 mod 4]) mod 4
= (3 + 0 + 3) mod 4 = 2
Swap (S[3],S[2])

Sehingga hasil array S adalah

Array S	1	2	0	3

Setelah melakukan KSA, akan dilakukan *Pseudo Random Generation Algorithm*(PRGA). PRGA akan dilakukan sebanyak 4 kali dikarenakan plainteks yang akan dienkripsi berjumlah 4 karakter. Hal ini disebakan karena dibutuhkan 1 kunci dan 1 kali pengoperasian XOR untuk tiap-tiap karakter pada plainteks. Berikut adalah tahapan penghasilan kunci enkripsi dengan PRGA.

Array S	1	2	0	3

Inisialisasi

i = 0
j = 0

Iterasi 1

i = (0 + 1) mod 4 = 1
j = (0 + S[1]) mod 4
= (0+ 2) mod 4 = 2
swap (S[1],S[2])

1	0	2	3

K1 = S[(S[1]+S[2]) mod 4]
= S[2 mod 4]
= S[2]
= 2
K1 = 00110010

Iterasi 2

i = (1 + 1) mod 4 = 2
j = (2 + S[2]) mod 4
= (2+ 2) mod 4 = 0
swap (S[2],S[0])

2	0	1	3

K2 = S[(S[2]+S[0]) mod 4]
= S[3mod 4]
= 3
K2 = 00110011

Iterasi 3

i = (2 + 1) mod 4 = 3
j = (0 + S[3]) mod 4 = (0+ 3) mod 4 = 3
swap (S[3],S[3])

2	0	1	3

K3 = S[(S[3]+S[3]) mod 4]
= S[6 mod 4]
= S[2]
=1
K3 = 00110001

Iterasi 4

i = (3 + 1) mod 4 = 0
j = (3 + S[0]) mod 4
= (3+ 1) mod 4 = 0
swap (S[0],S[0])

2	0	1	3

K1 = S[(S[0]+S[0]) mod 4]
= S[2 mod 4]
= S[2]
= 1
K1 = 00110001

Setelah menemukan kunci untuk tiap karakter, maka dilakukan operasi XOR antara karakter pada plaintext dengan kunci yang dihasilkan. Berikut adalah tabel ASCII untuk tiap-tiap karakter pada plaintks yang digunakan.

Tabel 1 Kode ASCII untuk setiap karakter plainteks yang digunakan

HURUF	KODE ASCII (8 Bit)
H	**01001000**
A	**01000001**
L	**01001100**
O	**01001111**

Proses XOR dari kunci bisa dilihat pada tabel 2.

Tabel 2 Proses XOR kunci enkripsi dengan plainteks padaEnkripsi

	H	**A**	**L**	**O**
Plainteks	01001000 (72)	01000001 (69)	01001100 (76)	01001111 (79)
Key	00110010 (50)	00110011 (51)	00110001 (49)	00110001 (49)
Cipherteks	01001010 (122) (z)	01000010 (118) (v)	01001110 (125) (})	01001101 (126) (~)

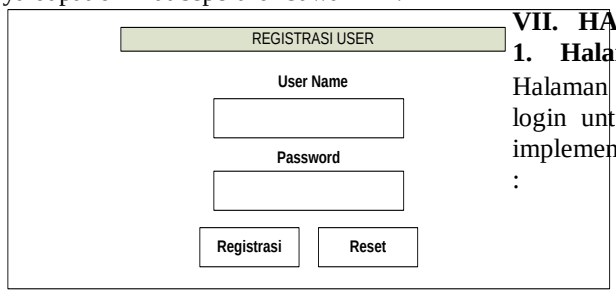
Dalam Proses pendekripsian dilakukan dengan proses XOR antara kunci dekripsi yang sama dengan kunci dekripsi dengan cipherteks yang dapat dilihat di tabel dibawah ini [9] :

Tabel 3 Proses XOR kunci dekripsi dengan cipherteks pada Dekripsi

Key	00110010 (50)	00110011 (51)	00110001 (49)	00110001 (49)
Cipherteks	01001010 (122) (z)	01000010 (118) (v)	01001110 (125) (})	01001101 (126) (~)
Plainteks	01001000 (72) (H)	01000001 (69) (E)	01001100 (76) (L)	01001111 (79) (O)

VI. PERANCANGAN SISTEM

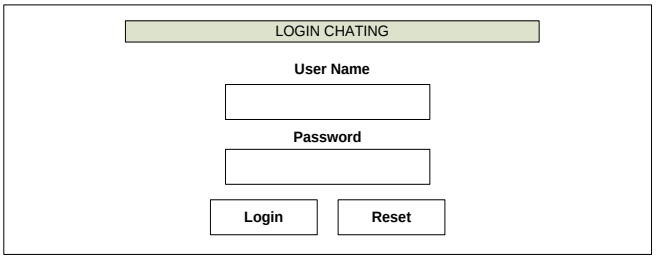
Program ini di gunakan untuk melakukan proses registrasi user untuk dapat melakukan chating dan gambarnya dapat di lihat seperti di bawah ini :



Gambar 3 : Halaman Registrasi User

1. Perancangan Login Chating

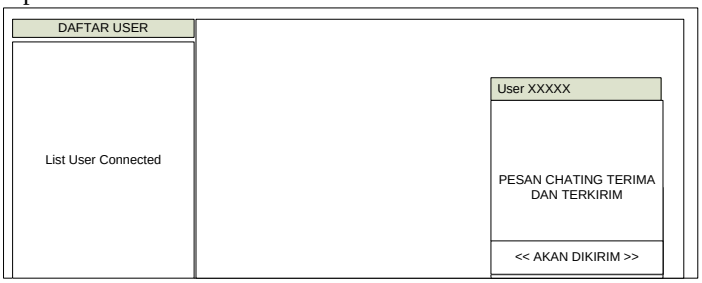
Program ini di gunakan untuk melakukan proses Login untuk dapat melakukan chating dan gambarnya dapat di lihat seperti di bawah ini :



Gambar 4 : Halaman Registrasi User

2. Perancangan Interface Chating

Program ini di gunakan untuk melakukan proses chating dengan computer yang sedang melakukan koneksi ke server dan gambarnya dapat di lihat seperti di bawah ini :



Gambar 5 : Sistem Utama Chating

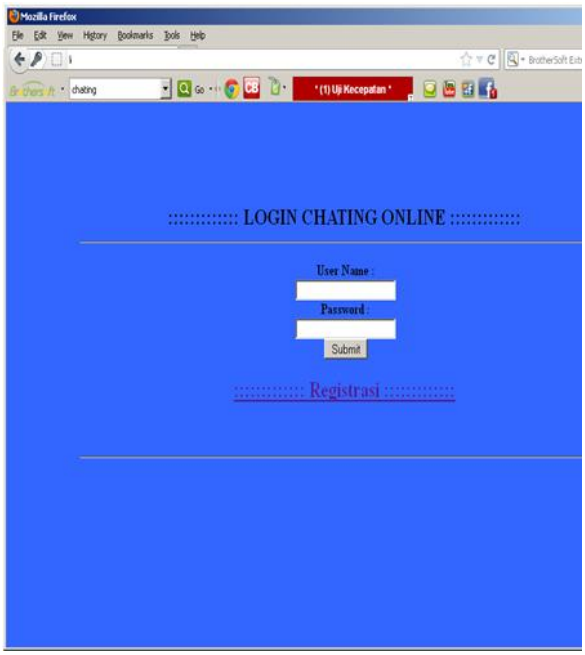
Keterangan :

a. User Connected, berguna untuk menampilkan semua user yang sedang melakukan koneksi ke server dari client
b. Pesan, Pesan yang sudah dikirim atau diterima
c. Kirim : Proses yang digunakan untuk melakukan pengiriman pesan

VII. HASIL

1. Halaman Login Chatting

Halaman ini digunakan untuk menampilkan user login untuk dapat masuk kedalam chatting, yang implementasinya terlihat seperti gambar dibawah ini :



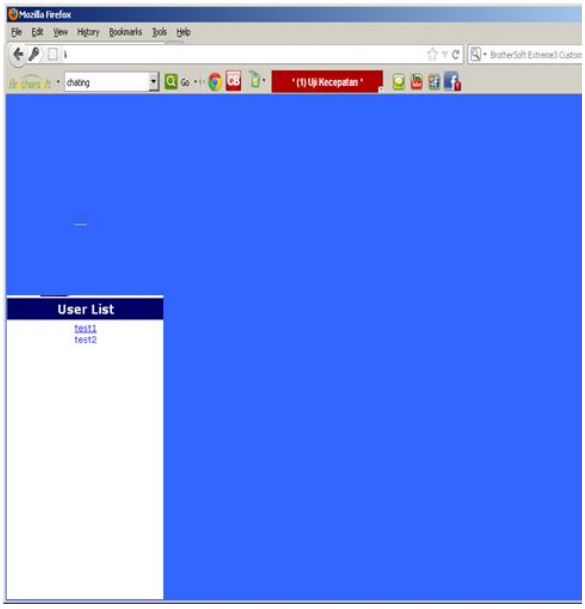
Gambar 6 : Menu Login

Algoritma :

1. Buka web browse
2. Ketik pada URL web browse "http://localhost/chat/
3. Maka akan tampil seperti gambar diatas
4. Untuk masuk ke sistem maka masukkan "user name" dan "password"
5. Lalu tekan tombol submit

2. Halaman Data User Pertama Setelah Login

Halaman ini digunakan untuk menampilkan menu utama setelah login, adapun gambar dari implementasi Halaman ini dapat dilihat pada gambar dibawah ini :



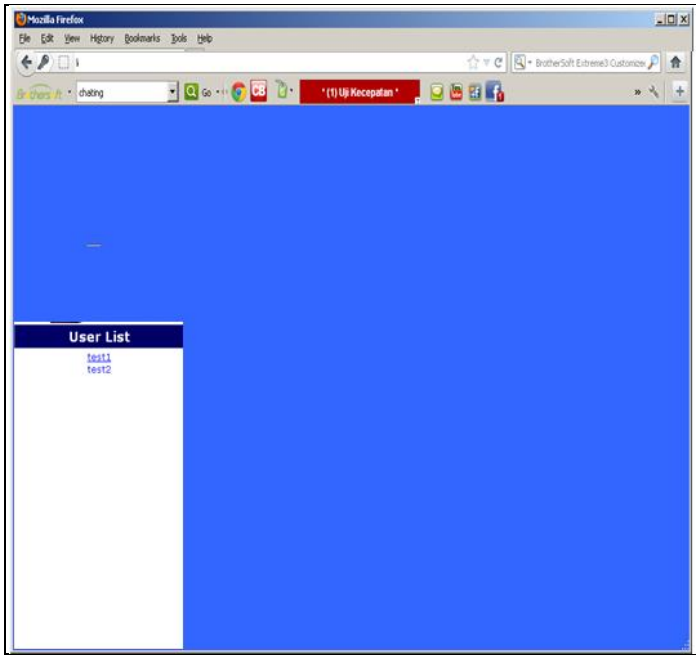
Gambar 7 :Data User Test 1

Algoritma :

1. Setelah login dilakukan maka akan tampil seperti gambar diatas
2. Untuk melakukan hubungan chating maka diklik salah satu nama user yang tercantum pada user list

3. Halaman Data User Kedua Setelah Login

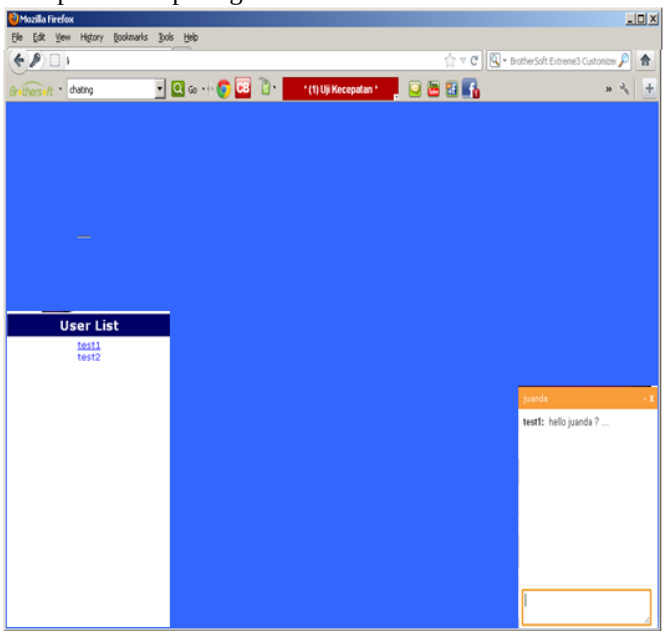
Halaman ini digunakan untuk menampilkan menu utama setelah login, adapun gambar dari implementasi Halaman ini dapat dilihat pada gambar dibawah ini :



Gambar 8 : Data User ke 2

4. Halaman Data Chating User-1

Halaman ini digunakan untuk menampilkan proses chating, adapun gambar dari implementasi Halaman ini dapat dilihat pada gambar dibawah ini :



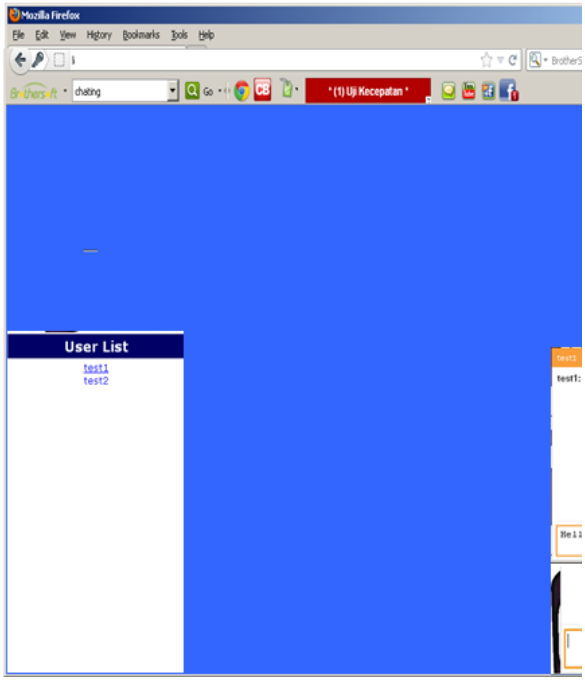
Gambar 9 : Halaman Proses Chating

Algoritma :

1. Setelah login dilakukan maka akan tampil seperti gambar diatas
2. Untuk melakukan hubungan chating maka diklik salah satu nama user yang tercantum pada user list
3. Maka akan tampil seperti gambar diatas

5. Halaman Data Chating User-2

Halaman ini digunakan untuk menampilkan proses chating, adapun gambar dari implementasi Halaman ini dapat dilihat pada gambar dibawah ini :



Gambar 10 : Halaman Proses Chating

Algoritma :

1. Setelah login dilakukan maka akan tampil seperti gambar diatas
2. Untuk melakukan hubungan chating maka diklik salah satu nama user yang tercantum pada user list
3. Maka akan tampil seperti gambar diatas

REFERENCES

Kristanto, A. 2008. *Perancangan Sistem Informasi dan Aplikasinya*. Yogyakarta: Gava Media.

Stream Cipher dan kelemahannya http://www.visumnews.com/download/jurnal/rc4.pdf

Slamet Mulyono, Enkripsi RC4 Stream Chiper sebagai security pada databaseaplikasi SIAK (Sistem Informasi Administrasi Kependudukan) http://www.mercubuana.ac.id/file/Jurnal%20Enkripsi%20RC4.pdf

Kurniawan,Yusuf. Kriptografi: Keamanan internet dan jaringan komunikasi. Informatika Bandung, Bandung, 2004

Sri Primaini Agustanti, Pengamanan Kunci Enkripsi One Time Pad (OTP) menggunakan Enkripsi RSA, 2010 http://sigma.ac.id/wp-content/uploads/2013/04/Jurnal-OTP-RSA.pdf

[12]. Aryus, Dony, 2006, Kritografi:Keamanan Data dan Komunikasi, *Yogyakarta, Graha Ilmu*