JURIKOM (Jurnal Riset Komputer), Vol. 7 No. 3, Juni 2020
e-ISSN 2715-7393 (Media Online), p-ISSN 2407-389X (Media Cetak)
DOI 10.30865/jurikom.v7i3.2126
Hal 358-364

Perancangan Aplikasi Penyandian Pesan Chat Client dan Server Berdasarkan Algoritma Spritz

Widodo Arif Prabowo, Mesran, Siti Nurhabibah Hutagalung

Program Studi Teknik Informatika, STMIK Budi Darma, Medan, Indonesia
Email: [1]widodoarif795@gmail.com, [2]mesran.skom.mkom@gmail.com, [3]siti.nurhabibah@stmik-budidarma.ac.id
Submitted 21-04-2020; Accepted 27-05-2020; Published 14-06-2020

Abstrak

Berkembangnya dunia Komputer pada era saat ini maka kebutuhan untuk mengamankan pesan/data dan untuk keamanan dalam berkomunikasi sangatlah penting. Penulis melakukan penelitian ini bertujuan untuk mengamankan pesan chatting dari client kepada server dan sebaliknya. Dengan teknik kriprografi ini digunakan untuk mengkonversi pesan chatting kedalam bentuk kode-kode tertentu dengan maksud agar pesan chatting tidak dapat dilihat/dibaca oleh siapa pun kecuali orang-orang yang memiliki hak untuk membaca pesan chatting tersebut. Pada penelitian ini, dirancang aplikasi penyandian Pesan Chatting Client dan Server yang mengimplementasikan cara kerja enkripsi dan dekripsi denganmenggunakan Algoritma Spritz. Hasil dari penelitian ini adalah aplikasi penyandian pesan chatting client dan server dengan algoritma Spritz yang dapat menyandikan pesan chatting yang bersifat rahasia. Aplikasi ini dibuat dengan menggunakan Microsoft Visual Basic 2008.

Kata Kunci: Client dan Server, Enkripsi, Dekripsi, Spritz.

Abstract

The development of the computer world in the current era, the need to secure messages / data and for security in communication is very important. The author conducted this research aims to secure chat messages from client to server and vice versa. With this cryptographic technique used to convert chat messages into the form of certain codes with the intention that chat messages cannot be seen / read by anyone except people who have the right to read the chat messages. In this study, the Client and Server Message Chat encoding application is designed which implements the workings of encryption and decryption using the Spritz Algorithm. The results of this study are chat client and server message encoding applications with the Spritz algorithm that can encode confidential chat messages. This application was created using Microsoft Visual Basic 2008.

Keywords: Client and Server, Encryption, Decryption, Spritz.

1. PENDAHULUAN

Pertukaran data melalui jaringan komputer, sangat mungkin dilakukan karena tentunya akan mempercepat dan memudahkan proses pertukaran data. Sebagai contoh adalah pertukaran data yang dilakukan oleh sebuah client yang ditujukan kepada server. Banyak keuntungan yang dapat diperoleh dengan melakukan pertukaran data melalui media jaringan komputer yaitu untuk mengefisienkan waktu mengingat pentingnya data yang di distribusikan tersebut sampai pada penerimanya tepat waktu. Disamping banyaknya keuntungan yang diperoleh, ada juga bahaya yang muncul dalam proses pertukaran data melalui jaringan komputer, salah satunya adalah pencurian data yang dilakukan pihak ketiga yang tidak bertanggung jawab yang bertujuan untuk kepentingan pribadi. Berdasarkan pengiriman data melalui jaringan komputer tersebut tidak ada pengamanan terhadap isi dari pesan itu sendiri, sehingga pada saat proses pengiriman pesan, seseorang dengan mudah dapat mencuri data dan langsung dapat mengetahui isi dari pesan tersebut. Salah satu cara yang digunakan pada pengamanan data ialah menggunakan kriptografi yaitu dengan menyandikan isi informasi (plaintext) tersebut menjadi isi yang tidak dimengerti melalui proses enkripsi dan dekripsi sehingga usaha-usaha untuk memecah kode kriptografi secara tidak sah menjadi lebih sulit.

Ada bagusnya pesan chat client dan server dapat bersifat rahasia sehingga tidak semua pihak dapat melihat atau membaca pesan chat tersebut. Hal tersebut membuat keamanan dari pesan menjadi sangat penting dalam proses pengiriman pesan. Dalam penyaluran informasi atau pesan antara client dan server dapat dilakukan baik melalui media internet dengan fasilitas media chatting seperti yahoo messenger, facebook maupun melalui pesan chat lainnya yang banyak disediakan oleh internet.

Algoritma Spritz adalah pembaruan dari algoritma RC4 yang dilakukan oleh Ron Rivest dan Jacob Schultz pada tahun 2014. Spritz sebagai varian dari enkripsi RC4 termasuk pesan atau data satu per satu menggunakan enkripsi transformasi waktu yang tergantung relatif singkat. Penambahan elemen yang relatif prima untuk nilai N dari algoritma generasi pseudo random adalah perbedaan dengan algoritma RC4. Selain stream cipher, algoritma spritz juga dapat digunakan sebagai fungsi hash dan Kode Otentikasi Pesan (MAC) dengan menggunakan fungsi spons dalam mengamankan data. Prosedur utama dari algoritma spritz sebagai stream cipher terdiri dari tiga proses: Algoritma Key Scheduling (KSA), Algoritma Pseudo Random Generation (PRGA) dan proses enkripsi atau dekripsi. Key Scheduling Algorithma (KSA) proses penjadwalan kuncia dalah proses yang dilakukan untuk membuat tabel S-Box (array S) dan permutasi tabel dalam array S. Panjang array yang diperlukan adalah 256 yang dimulai dari indeks 0 hingga 255. Tujuan dari KSA adalah proses darinilai array permutasi sebanyak 256 kali yang diinisialisasi dengan variable i dan j dengan tipe integer. Algoritma Pseudo Random Generation (PRGA). Nilai dari w adalah variable baru yang ditambahkan ke algoritma spritz yang sesuai dengan algoritma RC4. Nilai dari variable i, j, k dan z mulai dari 0 dan akan berubah sesuai hasil pada setiap iterasi. Proses ini melibatkan larik nilai-nilai S yang telah diubah dalam proses KSA[1].

JURIKOM (Jurnal Riset Komputer), Vol. 7 No. 3, Juni 2020
e-ISSN 2715-7393 (Media Online), p-ISSN 2407-389X (Media Cetak)
DOI 10.30865/jurikom.v7i3.2126
Hal 358-364

Pada penelitian sebelumnya oleh Taronisokhi Zebua dengan judul Encoding the Record Database of Computer Based Test Exam Based on Spritz Algorithm disimpulkan bahwa algoritma spritz sangat efektif untuk menyandikan record database dan mampu untuk menyandikan pesan[1]. Pada tahun 2016 oleh Gratia Vintana dan Mardi Hardjianto dengan judul Security Chatting Berbasis Dekstop dengan Enkripsi Caesar Cipher Key Random disimpulkan bahwa algoritma Caesar Cipher dapat menyandikan pesan chat [2].

2. METODE PENELITIAN

2.1 Kriptografi

Kriptografi pada awalnya dijabarkan sebagai ilmu yang mempelajari bagaimana menyembunyikan pesan. Namun pada pengertian modern kriptografi adalah ilmu yang bersandarkan pada teknik matematika untuk berurusan dengan keamanan informasi seperti kerahasiaan, keutuhan data dan otentikasi entitas.
Jadi pengertian kriptografi modern adalah tidak saja berurusan hanya dengan penyembunyian pesan namun lebih pada sekumpulan teknik yang menyediakan keamanan informasi. Kriptografi klasik umumnya merupakan teknik penyandian dengan kunci simetrik dan menyembunyikan pesan yang memiliki arti ke sebuah pesan yang nampaknya tidak memiliki arti dengan metode pergantian huruf (substitusi) atau transpose (pertukaran tempat)[2].

2.2 Algoritma Kriptografi Spritz

Algoritma *Spritz* adalah pembaruan dari algoritma RC4 yang dilakukan oleh Ron Rivest dan Jacob Schultz pada tahun 2014. *Spritz* sebagai varian dari *enkripsi* RC4 termasuk pesan atau data satu per satu menggunakan *enkripsi* transformasi waktu yang tergantung relatif singkat. Penambahan elemen yang relatif prima untuk nilai N dari algoritma *generasi pseudo random* adalah perbedaan dengan algoritma RC4. Selain *stream cipher*, algoritma *spritz* juga dapat digunakan sebagai fungsi *hash* dan Kode Otentikasi Pesan (MAC) dengan menggunakan fungsi *spons* dalam mengamankan data. Prosedur utama dari algoritma *spritz*s ebagai *stream cipher* terdiri dari tiga proses: Algoritma *Key Scheduling* (KSA), *Algoritma Pseudo-Random Generation* (PRGA) dan proses *enkripsi* atau *dekripsi*[1].

1. *Key Scheduling Algorithm* (KSA)
   Proses penjadwalan kunci adalah proses yang dilakukan untuk membuat tabel S-Box (*array* S) dan permutasi tabel dalam *array* S. Panjang *array* yang diperlukan adalah 256 yang dimulai dari indeks 0 hingga 255. Tujuan dari KSA adalah proses dari nilai array permutasi sebanyak 256 kali yang diinisialisasi dengan variabel i dan j dengan tipe integer.
   Pseudo-code dari KSA adalah:
   untuk i = 0 hingga N - 1
   S [i] = i
   next i
   i, j = 0
   untuk i = 0 ke N - 1
   j = (j + S [i] + K [i mod *Key.length*]) mod N
   swap (S [i], S [j])
   j = j
   next i
   mana N adalah ukuran array yang akan dimutasi, yaitu 0 - 255.
2. *Algoritma Pseudo Random Generation* (PRGA)
   Proses *algoritma generator pseudo random* dilakukan untuk mendapatkan nomor kunci baru elemen polos. Nilai dari w adalah variabel baru yang ditambahkan ke algoritma *spritz* yang sesuai dengan algoritma RC4. Nilai dari variabel i, j, k dan z mulai dari 0 dan akan berubah sesuai hasil pada setiap iterasi. Proses ini melibatkan larik nilai-nilai S yangtelah diubah dalam proses KSA.
   Pseudo-code PRGA adalah:
   i = 0 ke plain.length
   i = (i + w) mod N
   j = (k + S [j + S [i]]) mod N
   k = (i + k + S [j]) mod N
   swap S [i], S [j]
   z = (S [j + S [i + S [z + k]]]) mod N
   output z
   next i di
   mana w adalah nilai integer yang relatif prima dengan N dan nilai i, j, k, z mulai dari 0.

2.3 Penyandian Pesan Chatting

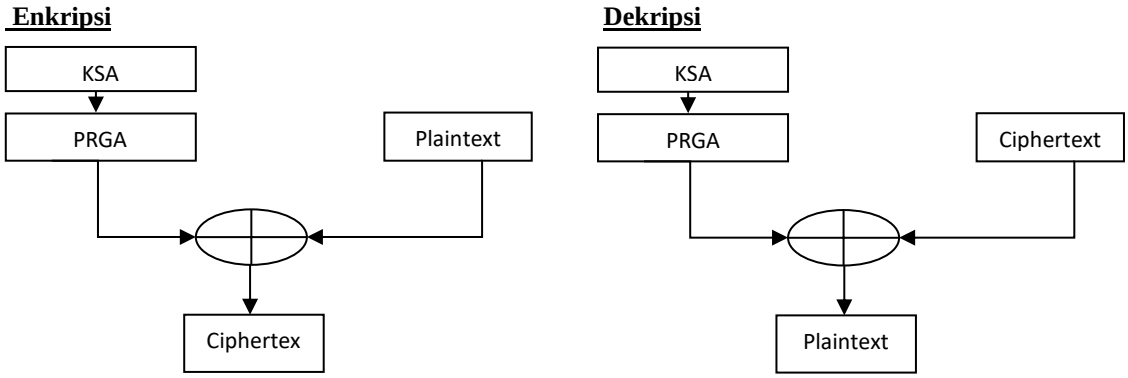
Dalam Buku Pengantar Ilmu Komunikasi, pesan yang dimaksud dalam proses komunikasi adalah sesuatu yang disampaikan pengirim kepada penerima. Pesan dapat disampaikan dengan cara tatap muka atau media komunikasi. Isinya bisa berupa ilmu pengetahuan, hiburan dan informasi. Pesan pada umumnya bersifat abstrak. Untuk membuatnya konkret agar dapat

JURIKOM (Jurnal Riset Komputer), Vol. 7 No. 3, Juni 2020
e-ISSN 2715-7393 (Media Online), p-ISSN 2407-389X (Media Cetak)
DOI 10.30865/jurikom.v7i3.2126
Hal 358-364

dikirim dan diterima oleh komunikan, manusia dengan akal budinya menciptakan sejumlah lambang komunikasi berupa suara, mimik, gerak-gerik, bahas lisan, dan bahasa tulisan.[5].

3. HASIL DAN PEMBAHASAN

Masalah keamanan *chatting* harus menjadi hal yang penting untuk dilihat, mengingat saat sekarang pengguna terhadap aplikasi *chatting* terutama antara *client* dan *server*. Penyandian berdasarkan teknik kriptografi memberikan hasil yang signifikan untuk mengurangi penyalahgunaan isi dari *chatting* tersebut. Pengamanan pesan *chatting client* dan *server* berdasarkan *algoritma spritz* meliputi tahap, yaitu proses *Key Scheduling Algorithm* (KSA), proses *Pseudo Random Generation Algorithm* (PRGA), proses *enkripsi* dan *dekripsi.* Berikut diagram alur kerja sistem algoritma *spritz*:



Gambar 1. Skema *Enkripsi* dan *Dekripsi*

Proses penyelesaian dari kriptografi metode *spritz* dapat dibagi menjadi tiga bagian, yaitu :

1. *Key Scheduling Algorithm* (KSA)
2. *Pseudo Random Generation Algorithm* (PRGA)
3. *Enkripsi* dan *Dekripsi*.

Proses *Enkripsi Algoritma Spritz*
Plaintext : WIDOD
Key : 25256
Proses *Enkripsi*: Pertama harus membuat inisialisasi panjang kunci 5 bit dengan: S [0] = 0
S [1] = 1
S [2] = 2
S [3] = 3
S [4] = 4
Kunci: 25256 diinisialisasikan menjadi
K0 = [2]
K1 = [5]
K2 = [2]
K3 = [5]
K4 = [6]
Inisialisasi i dan j dengan 0 kemudian dilakukan *Key Scheduling Algorithm* (KSA) agar tercipta *state array* yang acak. Penjelasan iterasi lebih lanjut dapat dijelaskan sebagai berikut:
Proses *Key Scheduling Algorithm* (KSA) :
Langkah pertama:
i = 0
j = (j + S [i] + k [i] mod 5
= (j + S [0] + k [0] mod 5)
= (0 +0+ 2) mod 5
j = 2
Swap S [0], S [2] sehingga menghasilkan array:

2	1	0	3	4

Langkah Kedua
i = 1
j = (j + S [i] + k [i] mod 5)
= (j + S [1] + k [1] mod 5)

JURIKOM (Jurnal Riset Komputer), Vol. 7 No. 3, Juni 2020
e-ISSN 2715-7393 (Media Online), p-ISSN 2407-389X (Media Cetak)
DOI 10.30865/jurikom.v7i3.2126
Hal 358-364

= (2 + 1 + 5) mod 5
j = 3
Swap S [1], S [3] sehingga menghasilkan array:

2	3	0	1	4

Sampai Langkah Ke 4

Setelah melakukan *Key Scheduling Algorithm* (KSA), akan dilakukan *Pseudo Random Generation Algorithm* (PRGA). PRGA akan dilakukan sebanyak 5 kali dikarenakan *plainteks* yang akan di *enkripsi* berjumlah 5 karakter. Hal ini disebabkan karena dibutuhkan 1 kunci dan 1 kali pengoperasian XOR untuk setiap-setiap karakter pada *plainteks.*

Berikut adalah tahapan penghasilan kunci *enkripsi* dengan *Pseudo Random Generation Algorithm* (PRGA). Hasil *Array* setelah di KSA:

4	1	2	3	0

Langkah pertama
i = (0+4) mod 5
= 4
j = (0+S[4]) mod 5
= (0+0) mod 5
= 0
Swap (S[4], S[0])
Array S

0	1	2	3	4

K0 = S [(S[4] + S[0]) mod 5]
= S [(4+0) mod 5)]
= S [(4 mod 5)]
= S[4]
= 4
K0 = 00000100
Langkah kedua
i = (4+4) mod 5
= 3
j = (0+ S[3] mod 5)
= (0+3) mod 5
= 3
Swap (S[3], S[3])
Array S

0	1	2	3	4

K1 = S [(S [3] + S [3] mod 5)]
= S [(3+3) mod 5)]
= S [(6 mod 5)]
= S [1]
= 1
K1 = 00000001
Sampai Langkah Ke 4

Setelah menemukan kunci untuk setiap karakter, maka dilakukan operasi XOR antara karakter pada *plaintext* dengan kunci yang dihasilkan. Berikut adalah kode *Ascii* pada setiap karakter *plaintext* dan kunci yang dihasilkan dari proses PRGA:

W = 01010111 **K0** = 00000100
I = 01001001 **K1** = 00000001
D = 01000100 **K2** = 00000000
O = 01001111 **K3** = 00000000
D = 01000100 **K4** = 00000011

Kemudian nilai *Array* S di *XOR* dengan nilai biner *plaintext* :

Plaintext = **W**
Biner Plaintext = 01010111
Key = 00000100 ⊕
Biner Chipertext = 01010011
Chipertext = **S**

Plaintext = **I**
Biner Plaintext = 01001001
Key = 00000001 ⊕
Biner Chipertext = 01001000

JURIKOM (Jurnal Riset Komputer), Vol. 7 No. 3, Juni 2020
e-ISSN 2715-7393 (Media Online), p-ISSN 2407-389X (Media Cetak)
DOI 10.30865/jurikom.v7i3.2126
Hal 358-364

Chipertext = **H**
Sehingga menghasilkan **Chipertext: SHDOG**
Chipertext : SHDOG
Key : 25256

Proses Dekripsi *Spritz*
Pertama harus membuat inisialisasi panjang kunci 5 bit dengan: S [0] = 0
S [1] = 1
S [2] = 2
S [3] = 3
S [4] = 4
Kunci: 25256 diinisialisasikan menjadi
K0 = [2]
K1 = [5]
K2 = [2]
K3 = [5]
K4 = [6]

Inisialisasi i dan j dengan 0 kemudian dilakukan *Key Scheduling Algorithm* (KSA) agar tercipta *state array* yang acak. Penjelasan iterasi lebih lanjut dapat dijelaskan sebagai berikut:

Proses *Key Scheduling Algorithm* (KSA) :
Langkah pertama:
i = 0
j = (j + S [i] + k [i] mod 5
= (j + S [0] + k [0] mod 5)
= (0 +0+ 2) mod 5
j = 2
Swap S [0], S [2] sehingga menghasilkan array:

2	1	0	3	4

Langkah Kedua
i = 1
j = (j + S [i] + k [i] mod 5)
= (j + S [1] + k [1] mod 5)
= (2 + 1 + 5) mod 5
j = 3
Swap S [1], S [3] sehingga menghasilkan array:

2	3	0	1	4

Setelah melakukan *Key Scheduling Algorithm* (KSA), akan dilakukan *Pseudo Random Generation Algorithm* (PRGA). PRGA akan dilakukan sebanyak 5 kali dikarenakan *plainteks* yang akan di *enkripsi* berjumlah 5 karakter. Hal ini disebabkan karena dibutuhkan 1 kunci dan 1 kali pengoperasian XOR untuk setiap-setiap karakter pada *plainteks*.

Berikut adalah tahapan penghasilan kunci *enkripsi* dengan *Pseudo Random Generation Algorithm* (PRGA). Hasil *Array* setelah di KSA:

4	1	2	3	0

Langkah pertama
i = (0+4) mod 5
= 4
j = (0+S[4]) mod 5
= (0+0) mod 5
= 0
Swap (S[4], S[0])
Array S

0	1	2	3	4

K0= S [(S[4] + S[0]) mod 5]
= S [(4+0) mod 5)]
= S [(4 mod 5)]
= S[4]
= 4
K0 = 00000100
Langkah kedua
i = (4+4) mod 5
= 3
j = (0+ S[3] mod 5)
= (0+3) mod 5

JURIKOM (Jurnal Riset Komputer), Vol. 7 No. 3, Juni 2020
e-ISSN 2715-7393 (Media Online), p-ISSN 2407-389X (Media Cetak)
DOI 10.30865/jurikom.v7i3.2126
Hal 358-364

= 3
Swap (S[3], S[3])
Array S

0	1	2	3	4

K1 = S [(S [3] + S [3] mod 5)]
= S [(3+3) mod 5)]
= S [(6 mod 5)]
= S [1]
= 1
K1 = 00000001
Sampai Langkah ke 4

Setelah menemukan kunci untuk setiap karakter, maka dilakukan operasi XOR antara karakter pada *plaintext* dengan kunci yang dihasilkan. Berikut adalah kode *Ascii* pada setiap karakter *plaintext* dan kunci yang dihasilkan dari proses PRGA:

S = 01010011 **K0** = 00000100
H = 01001000 **K1** = 00000001
D = 01000100 **K2** = 00000000
O = 01001111 **K3** = 00000000
G = 01000111 **K4** = 00000011

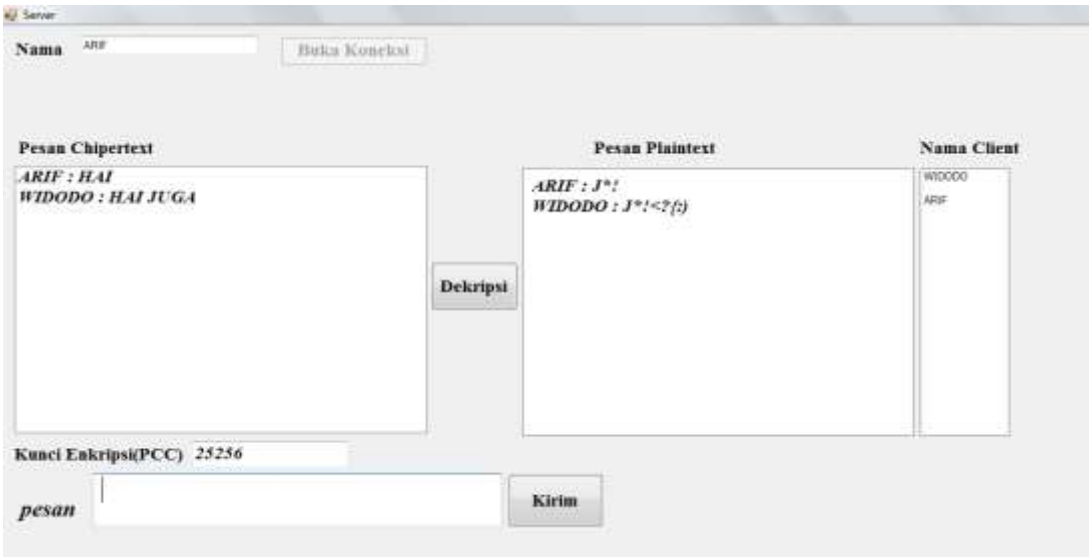
Kemudian nilai *Array* S di *XOR* dengan nilai biner *plaintext* :

Chipertext = **S**
Biner Chipertext = 01010011
Key = 00000100 ⊕
Biner Plaintext = 01010111
Plaintext = W
Chipertext = **H**
Biner Chipertext = 01001000
Key = 00000001 ⊕
Biner Plaintext = 01001001
Plaintext = **I**

Sehingga menghasilkan **Plaintext : WIDOD**.

3.2 Pengujian Aplikasi

Berikut pengujian program jika program dijalankan maka tampilan yang muncul adalah tampilan output program seperti *ciphertext* dari pesan yang mau dikirim dari *server* ke *client* ataupun sebaliknya. dapat dilihat pada gambar 3 dibawah.



Gambar 3. *Enkripsi Spritz Form Server*

4. KESIMPULAN

Berdasarkan hasil studi literatur, analisis, perancangan, implementasi, dan pengujian sistem ini, maka kesimpulan yang didapat adalah Proses penyandian pesan chat Client dan Server dapat dilakukan dengan algoritma spritz sehingga pesan yang dikirim dan diterima tidak dapat dibaca dan dimengerti oleh sembarang pihak. Aplikasi penyandian pesan chat Client dan

Server dibangun menggunakan Visual Studio 2008. Cara kerja algoritma spritz dalam proses penyandian pesan dalam penyisipan kata kunci yang ingin disisipkan dapat merubah pesan chat menjadi karakter – karakter symbol dan huruf yang tidak sesuai dengan pesan yang dikirim oleh pengirimProses penyandian pesan chat Client dan Server dapat dilakukan dengan algoritma spritz sehingga pesan yang dikirim dan diterima tidak dapat dibaca dan dimengerti oleh sembarang pihak. Aplikasi penyandian pesan chat Client dan Server dibangun menggunakan Visual Studio 2008. Cara kerja algoritma spritz dalam proses penyandian pesan dalam penyisipan kata kunci yang ingin disisipkan dapat merubah pesan chat menjadi karakter – karakter symbol dan huruf yang tidak sesuai dengan pesan yang dikirim oleh pengirim

REFERENCES

[1] T. Zebua, "Encoding the Record Database of Computer Based Test Exam Based on Spritz Algorithm," LONTAR KOMPUTER, vol. 9, no. April, pp. 52–62, 2018.
[2] G. Vintana and M. Hardjianto, "Security Chatting Berbasis Desktop dengan Enkripsi Caesar Cipher Key Random," Jurnal TICOM, vol. 5, no. 1, pp. 25–30, 2016.
[3] N. Dengen and H. R. Hatta, "Perancangan Sistem Informasi Terpadu Pemerintah Daerah Kabupaten Paser," Jurnal Informatika Mulawarman, vol. 4, no. 1, pp. 47–54, 2009.
[4] M. M. Amin, "IMPLEMENTASI KRIPTOGRAFI KLASIK PADA KOMUNIKASI KOMUNIKASI BERBASIS TEKS," Jurnal Pseudocode, vol. III, no. September, pp. 129–136, 2016.
[5] R. R. A. Gurning, "Perancangan aplikasi pengamanan pesan dengan algoritma caesar chiper," Jurnal Pelita Informatika Budi Darma, 2014.
[6] R. Sadikin, Kriptografi Untuk Keamanan Jaringan dan Implementasinya dalam Bahasa Java. Yogyakarta: Andi, 2012.
[7] P. G. Situmorang, H. K. Siburian, A. Fau, and S. Aripin, "PERANCANGAN APLIKASI PENYANDIAN PESAN CHATTING CLIENT DAN SERVER DENGAN ALGORITMA RC5," vol. I, pp. 100–106, 2017.
[8] T. Zebua, "ANALISA DAN IMPLEMENTASI ALGORITMA TRIANGLE CHAIN PADA PENYANDIAN RECORD DATABASE," Pelita Informatika Budi Darma, vol. 3, no. 2, pp. 37–49, 2013.
[9] W. A. Prabowo, A. Fitri, and R. I. Harahap, "Penyandian File Word Berdasarkan Algoritma Rivest Code 5 (RC5)," no. 1, pp. 47–56, 2018.
[10] A. Kadir, Pengenalan Sistem Informasi. 2003.
[11] S. Dharwiyanti and R. S. Wahono, "Pengantar Unified Modeling LAnguage (UML)," IlmuKomputer.com, pp. 1–13, 2003.
[12] R. Priyanto, Visual Basic.NET 2008, 1st ed. Yogyakarta: Andi Publisher, 2009.
[13] S. Sophian, "Jurnal Edik Informatika Pengimplementasian dan perancangan Sistem Informasi Pengontrolan Pemakaian Ruang Inap Jurnal Edik Informatika," Pengimplementasian Dan perancangan Sistem Informasi Pengontrolan Pemakaian Ruang Inap Puskesmas Muaro Kiawai Kec. Gunung Tuleh Jambi, vol. 3, pp. 1–11, 2015.