ALGORITMA RC4 DALAM PROTEKSI TRANSMISI DAN HASIL QUERY UNTUK ORDBMS POSTGRESQL

Yuri Ariyanto

Jurusan Teknik Informatika, Fakultas Teknologi Informasi Institut Teknologi Adhi Tama Surabaya Jl. Arief Rachman Hakim 100 Surabaya Email: yuri bjn@yahoo.com

ABSTRAK: Pada penelitian ini dibahas mengenai bagaimana mengimplementasikan algoritma kriptografi RC4 dalam proteksi terhadap *query* dan hasil *query*, pengamanan dilakukan dengan cara melakukan enkripsi dan dekripsi selama keduanya berada di dalam jaringan. Pengimplementasian dari penelitian ini yaitu membangun sebuah software yang akan diletakkan di sisi client yang berfungsi mengakses database yang diletakkan di sisi server. Software yang dibangun memiliki fasilitas untuk mengenkripsi dan mendektipsi *query* dan hasil *query* yang dikirimkan dari client ke server dan juga sebaliknya. Dengan demikian tramsmisi *query* dan hasil *query* dapat terjamin keamanannya. Terjaminnya keamanan transmisi *query* dan hasil *query* dan hasil *query* dan hasil *query* dan hasil *query* yang ditransmisikan sehingga apabila terjadi penyadapan terhadap keduanya, penyadap tidak akan mengerti isi data tersebut. Kesimpulan dari penelitian ini yaitu software yang dibangun berhasil mengenkripsi query dan hasil *query* yang ditransmisikan antara aplikasi client dan server database.

Kata kunci: enkripsi, transmisi, RC4, query, database.

ABSTRACT: In this research will be worked through about how cryptography RC4's algorithm implementation in protection to query result and of query, security by encryption and descryption up to both is in network. Implementation of this research which is build software in client that function access databases that is placed by the side of server. Software that building to have facility for encryption and descryption query result and of query that is sent from client goes to server and. transmission query result and of query can be told to succeed if success software can encryption query result and of query which transmission so that in the event of scanning to both, scanning will not understand data content. Conclusion of this research that is woke up software succeed encryption query and result of query which transmission between application of client and of server databases.

Keywords: encryption, transmission, RC4, query, databases.

PENDAHULUAN

Dengan adanya kebutuhan database yang semakin besar dan kompleks, secara otomatis akan diikuti dengan kebutuhan akan keamanan terhadap data yang tersimpan dari berbagai ancaman yang dapat berupa pengaksesan, perubahan serta perusakan data oleh pihak/user yang tidak mempunyai kewenangan. Terdapat beberapa level keamanan pada database, diantaranya: keamanan sistem operasi, keamanan sistem manajemen database, keamanan fisik dan keamanan dari segi user/manusia.

Untuk menambah tingkat keamanan dapat dilakukan dengan cara mengimplementasikan kriptografi untuk melindungi *query* yang dikirimkan dan hasil *query* yang akan diterima selama keduanya berada dalam jaringan komputer. Algoritma kriptografi yang digunakan adalah RC4.

Algorima kriptografi Rivest Code 4 (RC4) merupakan salah satu algoritma kunci simetris yang dibuat oleh RSA Data Security Inc (RSADSI). Dipilihnya RC4 sebagai metode kriptografi untuk proteksi *query* dan hasil *query* berdasarkan pada beberapa hal, yaitu:

- 1. Pengamanan transmisi database memerlukan suatu proses yang cepat, maka algoritma kriptografi simetris adalah solusi yang tepat.
- 2. RC4 merupakan algoritma *stream cipher* yang paling tepat dibandingkan dengan algoritma *stream cipher* yang lain untuk masalah transmisi *query* database seperti ini. Hal ini dikarenakan RC4 memiliki proses enkripsi yang cukup sederhana dan hanya melibatkan beberapa operasi saja per *byte*-nya.

Menurut hasil pengujian, kecepatan algoritma kriptografi RC4 adalah 5380,035 Kbytes/detik pada Pentium1 33 memori 16 MB pada Windows 95.

Kecepatan dalam testing ini adalah kecepatan enkripsi di memori, pada kenyataanya proses enkripsi file melibatkan banyak faktor lain seperti interface IO, tipe hardisk, dan lain-lain sehingga pada kenyataannya kecepatan enkripsi lebih lambat dari hasil tersebut, karena dipengaruhi faktor lain tersebut. Hasil testing didapat dengan enkripsi 256 byte per blok sebanyak 20480 kali, atau setara dengan kurang lebih 5 MB data. Sebagai perbandingan, hasil testing dengan algoritma Blowfish pada jenis komputer yang sama yaitu 2300 KByte/detik pada 8 byte per blok [1, 2]. Jadi testing tersebut membuktikan bahwa RC4 sebagai algoritma yang cepat dalam pemrosesan proses enkripsi dan dekripsi.

Dengan penggunaan algoritma RC4, diharapkan dapat meningkatkan tingkat keamanan transmisi data tanpa mengurangi performasi database secara signifykan.

METODE

Algoritma RC4

Algoritma kriptografi Rivest Code 4 (RC4) merupakan salah satu algoritma kunci simetris dibuat oleh RSA Data Security Inc (RSADSI) yang berbentuk stream chipper [3]. Algoritma ini ditemukan pada tahun 1978 oleh Ronald Rivest dan menjadi simbol keamanan RSA (merupakan singkatan dari tiga nama penemu: Rivest Shamir Adleman). RC4 menggunakan panjang kunci dari 1 sampai 256 byte yang digunakan untuk menginisialisasikan tabel sepanjang 256 byte. Tabel ini digunakan untuk generasi yang berikut dari pseudo random yang menggunakan XOR dengan plainteks untuk menghasilkan cipherteks. Masing-masing elemen dalam tabel saling ditukarkan minimal sekali.

RC4 merupakan salah satu jenis *stream cipher* sehingga RC4 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) sehingga 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.

Algoritma RC4 menggunakan dua buah S-Box yaitu *array* sepanjang 256 yang berisi permutasi dari bilangan 0 sampai 255, dan S-Box kedua, yang berisi permutasi merupakan fungsi dari kunci dengan panjang yang variabel.

Cara kerja algoritma RC4 yaitu inisialisasi S-Box pertama, S[0],S[1],...,S[255], dengan bilangan 0 sampai 255. Pertama isi secara berurutan S [0] = 0,

S[1] = 1,..., S[255] = 255. Kemudian inisialisasi array lain (S-Box lain), misal array K dengan panjang 256. Isi array K dengan kunci yang diulangi sampai seluruh array K[0], K[1],...,K[255] terisi seluruhnya.

```
Proses inisialisasi S-Box (Array S) for i = 0 to 255 S[i] = i
```

Proses inisialisasi S-Box (Array K)
Array Kunci // Array dengan
panjang kunci "length".
for i = 0 to 255
K[i] = Kunci[i mod length]

Kemudian lakukan langkah pengacakan S-Box dengan langkah sebagai berikut:

```
i = 0; j = 0
for i = 0 to 255
{
j = (j + S[i] + K[i]) \mod 256
swap S[i] dan S[j]
```

Setelah itu, buat *pseudo random byte* dengan langkah sebagai berikut:

```
\begin{split} i &= (i+1) \bmod 256 \\ j &= (j+S[i]) \bmod 256 \\ swap S[i] \dim S[j] \\ t &= (S[i]+S[j]) \bmod 256 \\ K &= S[t] \end{split}
```

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

Berikut adalah implementasi algoritma RC4 dengan mode 4 byte (untuk lebih menyederhanakan). Inisialisasi S-Box dengan panjang 4 byte, dengan S[0]=0, S[1]=1, S[2]=2 dan S[3]=3 sehingga array S menjadi:

0 1 2 3

Inisialisasi 4 byte kunci array, Ki. Misalkan kunci terdiri dari 2 byte yaitu byte 1 dan byte 7. Ulang kunci sampai memenuhi seluruh array K sehingga array K menjadi:

1 7 1 7

Berikutnya mencampur operasi dimana kita akan menggunakan variabel i dan j ke index array S[i] dan K[i]. Pertama diinisialisasi i dan j dengan 0. Operasi Pencampuran adalah pengulangan rumusan (j + S[i]+ K[i]) mod 4 yang diikuti dengan penukaran S[i] dengan S[j]. Untuk contoh ini, karena kita mengguna-

kan array dengan panjang 4 byte maka algoritma menjadi:

$$\begin{aligned} & \text{for } i = 0 \text{ to } 4 \\ & j = (j + S[i] + K[i]) \text{ mod } 4 \\ & \text{swap } S[i] \text{ dan } S[j] \end{aligned}$$

Dengan algoritma seperti di atas maka dengan nilai awal i=0 sampai i=3 akan menghasilkan array S seperti di bawah ini:

iterasi pertama: i = 0, maka j = (j + S[i] + K[i]) mod

 $= (j + S[0] + K[0]) \mod 4$ = $(0 + 0 + 1) \mod 4$ = 1

Swap S[0] dan S[1] sehingga menghasilkan array S:

iterasi kedua: i = 1, maka $j = (j + S[i] + K[i]) \mod 4$ $= (j + S[1] + K[1]) \mod 4$ $= (1 + 0 + 7) \mod 4 = 0$

Swap S[1] dan S[0] sehingga menghasilkan array S:

iterasi ketiga:

$$i = 2$$
, maka
 $j = (j + S[i] + K[i]) \mod 4$
 $= (j + S[2] + K[2]) \mod 4$
 $= (0 + 2 + 1) \mod 4$
 $= 3$

iterasi keempat:

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

Berikutnya adalah proses enkripsi yaitu meng-XOR kan pseudo random byte dengan plainteks, misalnya plaintetks "HI". Plainteks terdiri dari dua karakter maka terjadi dua iterasi. Berikut iterasi 1:

Inisialisasi i dan j dengan i = 0; j = 0.

$$i = 0; j = 0$$

 $i = (i + 1) \mod 4$
 $= (0 + 1) \mod 4$
 $= 1$

dan

$$j = (j + S[i]) \mod 4$$

= $(0 + 2) \mod 4$
= 2

Swap S[i] dan S[j] yaitu S[1] dan S[2] sehingga array S menjadi

$$\begin{array}{c|ccccc}
 & 2 & 3 & 1 & 0 \\
t = (S[i] + S[j]) \mod 4 \\
 & = (3+1) \mod 4 \\
 & = 0 \\
K = S[t] = S[0] = 2
\end{array}$$

Byte K di-XOR-kan dengan plainteks "H". Kemudian iterasi 2:

$$i = 1; j = 2$$

 $i = (i + 1) \mod 4$
 $= (1 + 1) \mod 4$
 $= 2$
dan
 $j = (j + S[i]) \mod 4$
 $= (2 + 2) \mod 4 = 0$

Swap S[i] dan S[j] yaitu S[2] dan S[0] sehingga array S menjadi

$$\begin{array}{c|cccc}
 & 1 & 3 & 2 & 0 \\
t = (S[i] + S[j]) \mod 4 \\
 & = (2+1) \mod 4 \\
 & = 3 \\
K = S[t] = S[3] = 2 \\
\text{Byte K di-XOR-kan dengan plainteks "T"}.$$

Proses XOR *pseudo random byte* dengan plainteks, dapat dilihat pada Tabel 1.

Tabel 1. Proses XOR *pseudo random byte* dengan plainteks pada enkripsi

	Н	I
Plainteks	01001000	01001001
Pseudo random byte	00000010	00000010
Cipherteks	01001010	01001011

Pesan dikirim dalam bentuk cipherteks sehingga setelah sampai di penerima pesan dapat kembali diubah menjadi plainteks dengan meng-XOR-kan dengan kunci yang sama.

Pemrosesan pesan setelah sampai pada penerima dapat dilihat pada Tabel 2.

Tabel 2. Proses XOR *pseudo random byte* dengan cipherteks pada dekripsi

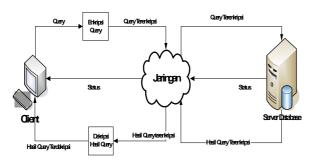
	Н	I
Plainteks	01001010	01001011
Pseudo random byte	00000010	00000010
Cipherteks	01001000	01001001

PERANCANGAN SISTEM UNTUK APPLIKASI PERCOBAAN

Rancangan Aplikasi Percobaan Secara Umum

Pada dasarnya software ini memiliki fungsi utama untuk melakukan proses enkripsi dan dekripsi data yang dikirimkan antara aplikasi client dan server database dan juga sebaliknya. Enkripsi dilakukan untuk query dan hasil query saja, dan algoritma yang digunakan adalah RC4. Software ini menerima inputan data berupa string query, user dapat menginputkan query sederhana seperti insert, update, delete. kemudian query ini dienkripsi oleh fungsi yang terdapat pada aplikasi client sebelum query tersebut dikirim ke server database. Data yang terenkripsi dikirim ke server database, kemudian setelah sampai di server database, data tersebut kembali didekripsi oleh function/stored procedure yang berada pada server database. Hasil query akan dikirim kembali ke aplikasi client dengan cara yang sama, yaitu data dienkripsi sebelum ditransmisikan dan kembali didekripsi sesampainya di tujuan.

Gambaran ranncangan aplikasi secara umum dapat dilihat pada Gambar 1.



Gambar 1. Gambaran Aplikasi secara umum

Perancangan Aplikasi Client

Pada aplikasi client terdapat dua proses yaitu login dan eksekusi query. Proses login yaitu login ke database server (Gambar 2). Proses eksekusi query dibedakan menjadi dua yaitu query yang me-retrieve data dan query yang tidak me-*retrieve* data.



Gambar 2 Blok Diagram Aplikasi Client

Perancangan modul pada Server Database

Pada server database akan dibuat dua *function* yaitu *function* enkripsi RC4 dan *function sql_runner*.

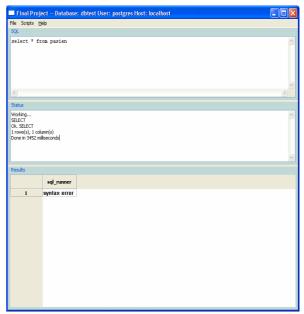
Kedua fungsi itu akan dibuat pada database yang akan diakses oleh aplikasi client. *Function* enkripsi RC4 berfungsi sebagai dekriptor untuk mendekripsi teks *query* sebelum dijalankan oleh *function sql_runner*. Di samping itu juga sebagai enkriptor untuk hasil *query* yang akan dikirimkan kepada aplikasi *client*.

IMPLEMENTASI APPLIKASI UNTUK PERCOBAAN

Aplikasi Client

Pada halaman *query editor* terdapat 3 panel, yaitu panel SQL, status dan result. Pada panel SQL terdapat text box multiline yang berguna bagi user untuk input sintaks *query* yang nantinya akan dieksekusi. Panel kedua yaitu panel status, pada panel ini terdapat text box *multiline* yang berfungsi untuk menampilkan status eksekusi *query*, apakah berhasil atau gagal. Serta dapat *menampilkan error* dari proses eksekusi *query*, dan juga untuk menampilkan lama proses eksekusi *query*.

Sedangkan pada panel ke tiga, terdapat grid yang berfungsi untuk menampilkan hasil eksekusi *query*. Tampilan aplikasi *client* dapat dilihat pada Gambar 3.



Gambar 3. Halaman Query Editor

Modul Enkripsi dan Dekripsi di Server Database

Implementasi fungsi enkripsi RC4

Fungsi RC4 yang berada pada *server* berguna untuk mendekripsi *query* yang dikirimkan oleh aplikasi *client* yang dalam keadaan terenkripsi, dan juga berguna untuk mengenkripsi hasil *query* sebelum dikirimkan kembali kepada *client*.

Implementasi fungsi sql runner

Fungsi ini berguna untuk memproses inputan *query* yang dikirimkan oleh *client*, nilai kembalian dari fungsi ini adalah berupa teks yang merupakan hasil eksekusi *query*. Sebelum teks yang berisi hasil *query* dikirimkan kembali kepada *client*, yang dilakukan adalah mengenkripsi hasil *query* tersebut yang nantinya akan didekripsi sesampainya di *client*.

UJICOBA DAN EVALUASI

Tahap uji coba dan evaluasi sistem dilakukan untuk mengetahui proses-proses didalamnya berjalan sesuai dengan yang diharapkan atau tidak. Proses-proses yang akan diuji coba dalam sistem adalah enkripsi, dekripsi dan keamanan *query* dan hasil *query* selama berada di jaringan.

Ujicoba Fungsionalitas

Pengujian dilakukan untuk memastikan bahwa sistem telah bekerja sesuai dengan tujuan pembuatannya. Proses enkripsi terjadi pada saat sebelum *query* dikirimkan oleh client ke server database, dan pada saat sebelum hasil *query* dikirimkan kembali oleh server ke client.

Pada pengujian ini, digunakan tools Wireshark yang berfungsi untuk memonitor transmisi *query* dan hasil *query*.

Ujicoba ini dilakukan untuk memastikan bahwa *query* dan hasil *query* dalam keadaan terenkripsi selama keduanya berada di jaringan.

0010 00 43 0020 01 19 0030 fe 1a	3 3f f7 40 00 80 06 9 05 8c 15 38 fe 51 a 30 d1 00 00 51 00	7d c7 c9 f0 08 00 45 00 37 51 c0 a8 01 03 c0 a8 cf e0 c2 f3 4d 4d 50 18 00 00 1a 73 65 6c 65 63 20 72 69 6d 69 6e 66 6f	.c?.@ 7Q
----------------------------------------	-------------------------------------------------------------------	-------------------------------------------------------------------------------------------------------------------------------------	----------

Gambar 4. Query sebelum dienkripsi

Pada gambar diatas terlihat sebelum diimplementasikan fungsi enkripsi, *query* masih dalam keadaan belum terenkripsi. Pada gamabr 5 hasil *query* yang dikirimkan oleh *server* database ke *client* yang masih dalam keadaan belum terenkripsi

Setelah diimplementasikan fungsi enkripsi menggunakan algoritma RC4 dalam proses pengiriman *query* dan hasil *query*, dapat dilihat pada gambar 6 parameter yang menjadi inputan dari fungsi sql_runner yang berada pada server database dalam keadaan terenkripsi.

Selain itu juga hasil *query* yang dikirimkan oleh server database ke *client* juga dalam keadaan terenkripsi, Gambar 7 yang memperlihatkan hasil *query* yang sudah dalam keadaan terkenkripsi.

0000							00								45		}	
010							80								C0		+@ r	
0020							c2								50		8 №	
0030							54		00	00					69		.WQT	
040							41		01				17			ff	id@A	
0050	ff						6e		6d	65	00	00	00	40	41	00	rna m	1e@A.
0060	02						ff		00	00	fe			72				
070							40		00	03				13	ff	ff	del@A .	
080							65		64		00				00	04	er c	
090							ff		ff	ff				70		6b		
00a0							41		05	00			17			ff	es@A	
0d00							73		7a	65	00			40		00	rsi z	
00c0			00				04			ff	ff			44	00	00		
00d0	00		00				00		31	00	00			49		74	.3 1	
00e0	65		73	65			00		4d	61	67		33		00	00	ense M	
00f0		03			35		00				32		00	00	02	32	525	
0100	36		00		00	31	00		00	00	00		33	00	00	00	6D1	
0110		4d				63			00	05	43				33	00	.Mavic	
120			03				00		00				00	00	00	03	598	
0130			30				00		00	06			00		36	00	700D8 .	
0140							73			79		00			48		odyss e	
0150			72				69				00				39		zard Lit e	
160							00			02		30		00		00	48	
170							02				00				75		A9 8	
180							39			20					49		M19A ,	
190							6d								00		(331 mm	
01a0							01							36		00	2980 .	
01b0							00			38				05		61	-::	
01c0							03			37				03		34	vic7 1	
01d0							32		00				36		00	00	132	
01e0							00							03	53	75	.6 1	
)1f0							69			6c				72	61	63	nSin g	
0200							34		00				33		00	00	k548 .	
0210	00		32				00						00	00	02	33	26D 5	
0220			00				75			00						63	8Sun .	
0230							68				03					00	h Which	
0240	00			32			00			36						00	32 2	
0250	06	00	00	00	02	33	30	00	00	00	04	41	6с	65	78	00	30	Alex.

Gambar 5. Hasil *Query* sebelum dienkripsi

0000	00 15 f2 05 29 e	14 00 1d 7d c7 c9 f0 0	B 00 45 00) }E.
		10 80 06 84 26 c0 a8 01	
0020	01 19 07 87 15 3	8 6d 02 fb 04 b7 21 3	
		0 51 00 00 00 36 73 6	
			3 27 49 6d / t sql_ru nner('Im 🛝
0050	4c 51 54 67 74 6	i8 39 2f 61 6b 63 2f 4i	2 75 70 47 (LQTgth9/ akc/NupG) DOANtUJP nb').
0060	44 6f 41 6e 74 5	is 4a 50 Ge 62 27 29 0i	O \ DoAntUJP nb').

Gambar 6. Query Terenkripsi

0000 00 1d 7d c7 c9 f0 00 15 f2 05 29 e4 08 00 45 00}							
0010 05 dc 99 f8 40 00 80 06 d7 b6 c0 a8 01 19 c0 a8	0000				05 29 e4	08 00 45 00	}E.
0030	0010						@
0040 5f 72 75 6e 6e 65 72 00 00 00 00 00 00 00 00 00 00 00 00 00							8! 5Em;P.
0050 00 19 ff							.;T#sql
0060 01 00 03 9e 9c 62 58 58 54 58 45 68 74 75 72 44							
0070 71 5a 72 74 35 75 69 6d 6e 53 58 35 4a 50 75 2b q2rt5uim nsx5jpu+ 0080 6d 6d 49 4e 48 6b 58 6c 6b 52 50 58 76 75 56 l 0000 6d 6d 49 4e 48 6b 58 6c 6b 52 50 58 76 75 56 l 0000 79 45 51 2b 61 61 51 6c 4f 4a 6a 6f 76 44 68 6e yEq4aql Ojjovbh 0000 35 65 5a 4e 41 79 6f 61 4b 78 4c 41 6b 4d 38 56 5EZNAYOA KKLAKMBV 0000 6a 58 6d 4a 30 69 59 6a 41 38 4e 59 76 37 39 66 jxmJ0ij ABNYV79f 0000 6a 47 76 6d 37 73 33 36 70 79 88 6d 6b 51 34 31 43 1 d5 mrM2u33p yxmRQ41c 0000 6a 44 77 6d 4a 72 52 38 56 76 6b 65 77 66 65 48 jpwmJrR8 VvkewfeH 00100 4e 57 4e 7a 73 33 6b 59 50 75 78 6f 6e 37 44 62 NoWnZs3kY PUXOn70b 0110 50 70 64 78 65 63 63 31 65 36 76 5a 55 50 49 43 protected e6vZUPIC 0120 6c 34 57 49 41 39 48 39 52 4d 70 6e 42 59 45 31 0130 49 6e 61 68 31 4a 38 78 79 37 4c 5a 33 44 77 53 Inah118x y7L230wS 0140 67 44 72 51 56 73 4f 74 2b 63 51 35 6f 34 34 31 gprQVsot +cQ50441 0150 33 50 47 66 69 78 71 49 68 70 47 4f 44 53 6c 4b 50 60 67 44 72 51 56 73 4f 74 2b 63 51 35 6f 34 34 31 gprQVsot +cQ50441 0150 33 50 47 66 69 78 71 49 68 70 47 4f 44 53 6c 4b 50 60 67 46 77 38 46 64 26 48 83 27 44 63 55 77 35 33 5a 4e 75 60 978 77 35 32 64 46 59 76 65 77 51 77 66 65 77 66 65 77 67 66 67 78 77 67 66 67 78 77 67 68 78 77 78 67 68 78 77 78 67 68 78 78 79 50 69 4c 5a 69 57 51 74 60 67 49 68 50 39 65 63 59 6a 61 77 69 78 79 50 69 4c 5a 69 57 51 74 60 67 49 68 50 39 65 63 59 6a 61 77 59 2f 4d 2b 74 48 6e 49 76 2b 30 orey9e6cy jayrR2L/ 0160 76 66 57 61 75 59 2f 4d 2b 74 48 6e 49 76 2b 30 orey9e6cy jayrR2L/ 0160 76 66 57 61 75 59 2f 4d 2b 74 48 6e 49 76 2b 30 orey9e6cy jayrR2L/ 0160 77 66 78 38 72 69 2f 6f 48 86 73 34 37 76 65 77 70 68 70 0rey9e6cy jayrR2L/ 0160 76 66 57 61 75 59 2f 4d 2b 74 48 6e 49 76 2b 30 orey9e6cy jayrR2L/ 0160 77 66 78 58 73 87 59 71 52 78 65 68 73 68 67 70 70 68 70 68 70 70 70 68 70 87 87 90 70 68 70 70 70 70 70 70 70 70 70 70 70 70 70							
0080 4F 44 4b 76 64 50 32 53 67 54 54 4a 4f 6f 73 6d 00KvdP2S gTTJOOSM 0090 6d 46 49 4e 48 8b 58 6c 6b 52 50 58 76 57 55 61 mFINHKX1 KRPXWMUA 0000 6d 58 6d 4a 30 69 59 6a 41 38 4e 59 76 37 39 66 ymmly 79 6d 4a 30 69 59 6a 41 38 4e 59 76 37 39 66 ymmly 79 6d 4a 30 69 59 6a 41 38 4e 59 76 37 39 66 ymmly 79 6d 6d 4a 30 69 59 6a 41 38 4e 59 76 37 39 66 ymmly 79 6d 6d 58 6d 4a 72 52 38 56 76 6b 65 77 66 65 48 ymmly 79 6d 6d 58 6d 59 39 6d 53 33 37 07 79 88 6d 6b 13 43 31 43 56 fmmly 239 ymmly 6d 12 0000 6a 44 77 6d 4a 72 52 38 56 76 6b 65 77 66 65 48 ymmly 8 fmmly							
0000 6d 46 49 4e 48 8b 58 6c 6b 52 50 58 76 57 55 61 0000 79 45 51 2b 61 61 51 6c 4f 4a 6a 6f 76 44 68 6e 0000 35 65 5a 4e 41 79 6f 61 4b 78 4c 41 6b 4d 38 56 0000 6a 78 6d 4a 30 69 59 6a 41 38 4e 59 76 37 39 66 0000 6a 72 4d 32 75 33 33 70 79 58 6d 6b 51 34 31 43 00000 6a 72 4d 32 75 33 33 70 79 58 6d 6b 51 34 31 43 00000 6a 74 77 6d 4a 72 52 38 56 76 6b 65 77 66 65 48 00000 6a 44 77 6d 4a 72 52 38 56 76 6b 65 77 66 65 48 00000 6a 44 77 6d 78 65 63 63 31 65 36 76 5a 55 50 49 43 0100 4e 57 4e 7a 73 33 6b 59 50 75 78 8f 6e 37 4d 62 0100 50 70 64 78 65 63 63 31 65 36 76 5a 55 50 49 43 0120 6c 34 57 49 41 39 48 39 52 4d 70 6e 42 59 4b 53 0130 49 6e 61 68 31 4a 38 78 79 37 4c 5a 33 4d 77 53 0130 49 6e 61 68 31 4a 38 78 79 37 4c 5a 33 4d 77 53 0140 67 4d 72 51 56 73 4f 7d 2b 63 51 35 6f 34 33 1 0500000000000000000000000000000000000							
0000 79 45 51 2b 61 61 51 6c 4f 4a 6a 6f 76 44 68 6e yEQ+aaQl OjjovDhn 00b0 35 6f 5a 4e 41 79 6f 61 4b 78 4c 41 6b 4d 38 66 yEQ+aaQl OjjovDhn 00b0 35 6f 5a 6d 4a 30 69 59 6a 41 38 4e 59 76 37 39 66 yEQ+aaQl OjjovDhn 00b0 62 72 4d 32 75 33 33 70 79 58 6d 6b 51 34 31 43 yED yEQ+aaQl OjjovDhn 00b0 62 72 4d 32 75 33 56 76 6b 65 77 66 66 48 yED yEQ+aaQl OjjovDhn 00b0 6a 44 77 6d 4a 72 52 38 56 76 6b 65 77 66 66 48 yED							ODKVaPZS gTTJOOSM
0000 35 65 5a 4e 41 79 6f 61 4b 78 4c 41 6b 4d 38 56 5 2 2 2 2 3 4 32 75 33 33 70 79 58 6d 6b 51 34 31 43 3 50 5 2 3 2 3 3 70 79 58 6d 6b 51 34 31 43 3 3 3 70 79 58 6d 6b 51 34 31 43 3 3 70 79 58 6d 6b 51 34 31 43 3 70 79 58 6d 6b 51 34 31 43 70 70 70 70 70 70 70 70 70 70 70 70 70							
0000 6a 58 6d 4a 30 69 59 6a 41 38 4e 59 76 37 39 66							
0000 62 72 4d 32 75 33 33 70 79 58 6d 6b 51 34 31 43 6mA2u33p yxmkQ41c 00e0 6a 44 77 6d 4a 72 52 38 56 76 6b 65 77 66 65 48 10mBrR8 vvkewFeH 00f0 53 55 54 73 41 30 51 4d 77 7a 61 69 75 77 67 43 57 57 57 86 76 62 37 44 62 20 20 20 20 20 20 20 20 20 20 20 20 20							
00e0 6a 44 77 6d 4a 72 52 38 56 76 6b 65 77 66 65 48 jpwmjrR8 vvkewfeH 00f0 53 35 54 73 41 30 51 4d 777 ac 16 6 57 77 67 43 STSADQM wzaiwgc 0100 4e 57 4e 7a 73 33 6b 59 50 75 78 6f 6e 37 44 62 90 0100 4e 76 4e 7a 7a 33 6b 59 50 75 78 6f 6e 37 44 62 90 0110 50 70 64 78 65 63 63 31 65 36 76 5a 55 50 49 43 9pdxecc1 e6vZUPIC 0120 6c 34 57 49 41 39 48 39 52 4d 70 6e 42 59 4b 53 14whA9H9 RMpnBYKS 0130 49 6e 61 68 31 44 38 78 78 79 37 4c 5a 33 44 77 53 1mahlask y7Lz3Dws 0140 67 44 72 51 56 73 4f 74 2b 63 51 35 6f 34 34 31 gprqysot +cQ50441 0150 35 50 47 56 69 78 71 49 68 70 47 4f 44 53 6c 4b 59 60 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14 50 14							DOMOUSE ANNIVIAL
0000 4e 57 4e 7a 73 33 6b 59 50 75 78 6f 6e 37 4d 62 NMNZS3kY PUXON70b 1010 50 70 64 78 65 63 33 4d 50 50 50 75 78 6f 6e 37 4d 62 NMNZS3kY PUXON70b 1010 50 70 64 78 65 63 63 31 65 36 76 5a 55 50 49 43 Ppdxecc1 e6v2UPIC 0120 6c 34 57 49 41 39 48 39 52 4d 70 6e 42 59 4b 53 14WIA919 RMPIBYKS 0130 49 6e 61 68 31 4a 38 78 79 37 4c 5a 33 44 77 53 Inahlasx y7L23DWS 0140 67 44 72 51 56 73 4f 74 2b 63 51 35 6f 34 34 31 gprQvSot +cQ50441 0150 35 50 47 56 69 78 71 49 68 70 47 4f 44 53 6c 4b 59C3VBIC 0150 37 73 43 6c 42 4e 48 32 74 46 35 77 35 33 5a 4e 7sclBNH2 tF5W53ZN 0170 38 61 42 72 36 6d 42 6b 37 59 33 78 42 44 5a 6c 88Br6mek 7Y3XBDZ1 0180 55 41 73 32 4d 4d 59 55 47 62 4d 53 44 69 46 70 UAS2MMYU GMSD1FD 0190 2f 76 6b 57 2b 43 77 4f 50 58 73 23 39 38 55 66 (Wkw+cwo Pxs298UF 01a0 62 2b 6f 4e 77 69 78 79 50 69 4c 5a 69 57 51 74 browniny Pilziwot 01b0 76 66 57 61 75 59 2f 4d 2b 74 48 6e 49 76 2b 30 0FY96Cf 13prRZL/01d0 74 61 30 4f 48 57 59 71 52 7a 62 55 68 7a 6b 67 100 0100 74 61 30 4f 48 57 59 71 52 7a 62 55 68 7a 6b 67 76 87 70 878 170 170 170 170 170 170 170 170 170 170							
0100 4e 57 4e 7a 73 33 6b 59 50 75 78 6f 6e 37 44 62 MWNZS3kY PUXON7Ďb 0110 50 70 64 78 65 63 63 31 65 36 76 5a 55 50 49 43 POROMECCL e6VZUPTS 0120 6c 34 57 49 41 39 48 39 52 4d 70 6e 42 59 4b 53 14WIA949 RMPNBYKS 0130 49 6e 61 68 31 4a 38 78 79 37 4c 5a 33 44 77 53 Inahl3kx y7LZ3bws 0140 67 44 72 51 56 73 4f 74 2b 63 51 35 6f 34 34 31 porovsot+cq5o441 0150 35 50 47 56 69 78 71 49 68 70 47 4f 44 53 6c 4b 5PGV3xql hpgoDs1k 0160 37 73 43 6c 42 4e 48 32 74 46 35 77 35 33 5a 4e 7sc BNH2 tF\$W53ZN 0170 38 61 42 72 36 6d 42 6b 37 59 33 78 42 44 5a 6c 4b 5PGV3xql hpgoDs1k 0160 55 41 73 32 4d 4d 59 55 47 62 4d 53 4d 69 46 70 47 4f 44 69 46 70 47 4f 46 94 67 04 74 67 47 47 47 47 49 68 74 47 47 47 47 49 68 74 47 47 47 47 49 68 74 47 47 47 47 49 68 74 47 47 47 49 68 74 47 47 47 49 68 74 47 47 47 47 49 68 74 47 47 47 49 68 74 47 47 49 68 74 47 47 47 49 68 74 47 47 47 49 68 74 47 47 49 68 74 47 47 47 49 68 74 47 47 49 48 74 58 47 47 47 47 49 68 74 47 47 49 48 74 58 47 47 47 49 48 74 47 47 49 48 74 58 47 47 47 49 48 74 58 47 47 47 49 48 74 47 47 49 48 74 47 47 49 48 74 47 47 49 48 74 49 48 74 78 48 74 49 48 74 78 49 49 49 49 49 49 49 49 49 49 49 49 49							
0110 50 70 64 78 65 63 63 31 65 36 76 5a 55 50 49 43 Pydxeccl e6vzUPIC 0120 6c 34 57 49 41 39 48 39 52 47 70 6e 42 59 4b 53 Pydxa94 RMpnBYKS 0130 49 6e 61 68 31 48 38 78 79 37 4c 5a 33 44 77 53 Inahl18x 97L230wS 0140 67 44 72 51 56 73 4f 74 2b 63 51 35 6f 34 34 31 gprQvsot +cQ5o441 0150 35 50 47 56 69 78 71 49 68 70 47 4f 44 53 6c 4b 59cpixq1 hg6oDSIK 0160 37 73 43 6c 42 4e 48 32 74 46 35 77 35 33 5a 4e 75 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 12 75 1							
0120 6c 34 57 49 41 39 48 39 52 4d 70 6e 42 59 4b 53							
0130							
0140 67 44 72 51 56 73 4f 74 2b 63 51 35 6f 34 34 31 gprQysot 4cQ50441 0150 35 50 47 56 69 78 71 49 68 70 47 4f 44 53 6c 4b 55 60 73 35 61 42 76 69 78 71 49 68 70 46 35 77 35 33 5a 4c 56 60 75 60 78 71 60 70 38 61 42 72 36 6d 42 6b 37 59 33 78 42 44 5a 6c 8aBr6mBk 7y3xBD21 0180 55 41 73 32 4d 4d 59 55 47 62 4d 53 44 69 46 70 UASZMMYU GBMSD1Fp 0190 2f 76 66 57 2b 43 77 4f 50 58 73 32 39 38 55 66 (Vkb4Fcwb PxS29BUf 0140 76 66 57 61 75 59 2f 4d 2b 74 48 6e 49 76 2b 30 vfWaUy/M +tHnIV+0 0160 76 66 57 61 75 59 2f 4d 2b 74 48 6e 49 76 2b 30 vfWaUy/M +tHnIV+0 0160 67 46 63 39 65 36 63 59 6a 61 79 46 52 5a 4c 2f 0FY966r jayFRZL/0160 77 6f 2b 38 72 69 2f 6f 48 68 73 44 37 76 65 79 70 FY966r jayFRZL/0160 37 6f 2b 38 72 69 2f 6f 48 68 73 34 37 76 65 79 70 FY966r jayFRZL/0160 37 6f 2b 38 72 69 2f 6f 48 68 73 34 37 76 65 79 70 FY966r jayFRZL/0160 53 79 75 55 6a 51 2b 47 31 63 53 57 75 52 56 34 2yUujq4g 1cSwuV4 0200 42 55 4e 57 61 41 6d 48 63 39 62 32 6f 68 72 72 BUNMAAMH C9b2ohrr							
0150 35 50 47 56 69 78 71 49 68 70 47 4f 44 53 6c 4b							
0160 37 73 43 6c 42 4e 48 32 74 46 35 77 35 33 5a 4e 75clBnH2 tf5w53ZN 0170 38 61 42 72 36 6d 42 6b 37 99 33 78 42 44 5a 6c 8aBr6mBk 7Y3xBDZl 0180 55 41 73 32 4d 44 59 55 47 62 4d 53 44 69 46 70 UA5ZMMYLG GMSD1FD 0190 2f 76 6b 57 2b 43 77 4f 50 58 73 32 39 38 55 66 (wk+cwo pxs298UF 01a0 62 2b 6f 4e 77 69 78 79 50 69 4c 5a 69 57 51 74 b-toNnivy PiLZiNQt 01b0 76 66 57 61 75 59 2f 4d 2b 74 48 6e 49 76 2b 30 vfwauy/M t+Iniv+0 01c0 6f 46 59 39 65 36 63 59 6a 61 79 46 52 5a 4c 2f 0FY966cY jayFRZL/ 01d0 74 61 30 4f 48 57 59 71 52 7a 62 55 68 7a 6b 67 taOhwyq Rzbuhzkg 01e0 37 6f 2b 38 72 69 2f 6f 48 68 73 44 37 76 65 79 70+8R1/0 HhsD7vey 01f0 5a 79 75 55 6a 51 2b 47 31 63 53 57 75 52 56 34 2yJuljqkg 1cSwurv4 0200 42 55 4e 57 61 41 6d 48 63 39 62 32 6f 68 72 72 BUNWAAMH C9bZohrr	0150		56 69 78	71 49 68			
0180 55 41 73 32 4d 4d 59 55 47 62 4d 53 44 69 46 70 UAS2MMYU GbMSD1FP 0190 2f 76 6b 57 2b 43 77 4f 50 58 73 32 39 38 55 66 /vkw+cwo Pxs298Uf 01a0 62 2b 6f 4e 77 69 78 79 50 69 4c 5a 69 57 51 74 b+onwixy PiLziwqt 01b0 76 66 57 61 75 59 2f 4d 2b 74 48 6e 49 76 2b 30 vFWaury/M + +HnIV+0 01c0 6f 46 59 39 65 36 63 59 6a 61 79 46 52 5a 4c 2f 0FY9e6cY jayFRZL/01d0 74 61 30 4f 48 37 59 71 52 7a 62 55 68 7a 6b 67 70+Br1/O Hnsb7vey 01f0 5a 79 75 55 6a 51 2b 47 31 63 53 57 75 52 56 34 Zyuljqea IcSWuRV4 0200 42 55 4e 57 61 41 6d 48 63 39 62 32 6f 68 72 72 BUNWAAMH C9b2ohrr	0160	37 73 43	6c 42 4e	48 32 74	46 35 77	' 35 33 5a 4e	
0190 2f 76 6b 57 2b 43 77 4f 50 58 73 32 39 38 55 66	0170	38 61 42	72 36 6d	42 6b 37	59 33 78	3 42 44 5a 6c	8aBr6mBk 7Y3xBDZl
01a0 62 2b 6f 4e 77 69 78 79 50 69 4c 5a 69 57 51 74 beonwixy Pilziwqt 01b0 76 66 57 61 75 99 2f 4d 2b 74 48 6e 49 76 2b 30 of 94 66 59 39 65 36 35 9 6a 61 79 46 25 2a 4c 2f 0FY986c7 jayfRZL/01d0 74 61 30 4f 48 57 59 71 52 7a 62 55 68 7a 6b 67 ta00Hw7q Rzbuhzkg 01e0 37 6f 2b 38 72 69 2f 6f 48 68 73 44 37 76 65 79 70+RFi/0 Hbs07vey 01f0 5a 79 75 55 6a 51 2b 47 31 63 53 57 75 52 56 34 2yuJuJqkg 1cSwurv4 0200 42 55 4e 57 61 41 6d 48 63 39 62 32 6f 68 72 72 BUNMSAMH c9b2ohrr	0180						UAs2MMYU GbMSDiFp
01b0 76 66 57 61 75 59 2f 4d 2b 74 48 6e 49 76 2b 30 VfWauY/M +tHnIv+0 01c0 6f 46 59 39 65 36 63 59 6a 61 79 46 52 5a 4c 2f 0FY9e6cY jayFRZL/ 01d0 74 61 30 4f 48 37 59 71 52 7a 62 55 68 7a 6b 67 ta00hwYq RzbUhzkg 01e0 37 6f 2b 38 72 69 2f 6f 48 68 73 44 37 76 65 79 7o+8ri/o HhsD7vey 01f0 5a 79 75 55 6a 51 2b 47 31 63 53 57 75 52 56 34 Zyuljq-G 1cSwuRV4 0200 42 55 4e 57 61 41 6d 48 63 39 62 32 6f 68 72 72 BUNWAAMH C9b2ohrr	0190			77 4f 50			/vkw+cwo PXs298Uf
01c0 6f 46 59 39 65 36 63 59 6a 61 79 46 52 5a 4c 2f oFY9e6cY jayFRZL/ 01d0 74 61 30 4f 48 37 59 71 52 7a 62 55 68 7a 6b 67 ta0/bwyq Rzbuhzkg 01e0 37 6f 2b 38 72 69 2f 6f 48 68 73 44 37 76 65 79 70+Brijo HhsD7vey 01f0 5a 79 75 55 6a 51 2b 47 31 63 53 57 75 52 56 34 ZyuUjq+G 1cSwuRv4 0200 42 55 4e 57 61 41 6d 48 63 39 62 32 6f 68 72 72 BUNMARMH C0b2ohrr							
01d0 74 61 30 4f 48 57 59 71 52 7a 62 55 68 7a 6b 67 ta00Hwrq Rzbuhzkg 01e0 37 6f 2b 38 72 69 2f 6f 48 68 73 44 37 76 65 79 7048r1/o Hhsp7vey 01f0 5a 79 75 55 6a 51 2b 47 31 63 53 57 75 52 56 34 zyuujq\edga (25wukya) 48 63 39 62 32 6f 68 72 72 BUNWAAMH C9b2ohrr							
01e0 37 6f 2b 38 72 69 2f 6f 48 68 73 44 37 76 65 79 70+8ri/o Hhso7vey 01f0 5a 79 75 55 6a 51 2b 47 31 63 53 57 75 52 56 34 2yuUjQ+G 1cSwuRV4 0200 42 55 4e 57 61 41 6d 48 63 39 62 32 6f 68 72 72 BUNWAAMH c9b2ohrr							
01f0 5a 79 75 55 6a 51 2b 47 31 63 53 57 75 52 56 34 Zyuuljoung 1cswurv4 0200 42 55 4e 57 61 41 6d 48 63 39 62 32 6f 68 72 72 BUNWAAMH c9b2ohrr							
0200 42 55 4e 57 61 41 6d 48 63 39 62 32 6f 68 72 72 BÚNWáAmH c9b2ohrr							

Gambar 7. Query TerenkripsiEvaluasi

Evaluasi

Ujicoba Peforma

Ujicoba berikut bertujuan untuk mengetahui kemampuan sistem dalam melakukan proses enkripsi dan dekripsi. Pengujian dilakukan dengan menghitung waktu yang dibutuhkan dari mulai pengiriman *query* hingga hasil *query* diterima kembali oleh client.

<u>Ujicoba Tanpa Menggunakan Fungsi Enkripsi dan</u> Dekripsi

Uji coba ini dilakukan untuk mengetahui seberapa cepat pemrosesan *query* tanpa dilakukan pengimplementasian fungsi enkripsi RC4 pada sisi *client* dan *server* database.

Tabel 3. Hasil Pengujian Tanpa Menggunakan Fungsi Enkripsi dan Dekripsi

Uji Coba Ke	Banyaknya record	Ukuran Data	Waktu
1	100	5730 Bytes	30 milliseconds
2	200	12468 Bytes	63 milliseconds
3	300	18692 Bytes	77 milliseconds
4	400	25363 Bytes	92 milliseconds
5	500	31249 Bytes	125 milliseconds
6	600	38127 Bytes	141 milliseconds
7	700	44183 Bytes	155 milliseconds
8	800	50236 Bytes	187 milliseconds

Dapat dilihat pada Tabel 3 bahwa semakin panjang kunci yang dibentuk maka semakin lama watu yang diperlukan untuk pemrosesan *query*.

Ujicoba Menggunakan Fungsi Enkripsi dan Dekripsi

Uji coba ini dilakukan untuk mengetahui seberapa cepat pemrosesan *query* dengan melakukan pengimplementasian fungsi enkripsi RC4 pada sisi client dan server database. Ujicoba ini dilakukan dengan menggunakan panjang kunci yang berbeda-beda, hal ini dilakukan untuk mengetahui apakah panjang kunci mempengaruhi secara signifikan terhadap performa enkripsi dan dekripsi.

Uji coba yang pertama yaitu menggunakan modul enkripsi yang dibangun dengan bahasa pemrograman C, dengan panjang kunci enkripsi sepanjang 8, 20, dan 30 karakter. Hasil pengujian dapat dilihat pada Tabel 4, 5, dan 6.

Dari ketiga tabel dapat disimpulkan bahwa panjang kunci dapat mempengaruhi kecepatan dari enkripsi dan dekripsi tapi dengan selisih waktu yang sangat kecil, sehingga perbedaan panjang kunci tidak mempengaruhi performa dari fungsi enkripsi dan dekripsi secara keseluruhan.

Tabel 4. Hasil Uji Coba Enkripsi dan Hasil Enkripsi Menggunakan kunci dengan panjang 8 karakter

Uji Coba Ke	Banyaknya record	Ukuran Data	Waktu
1	100	27741 Bytes	156 milliseconds
2	200	56909 Bytes	328 milliseconds
3	300	85451 Bytes	483 milliseconds
4	400	114603 Bytes	657 milliseconds
5	500	142711 Bytes	811 milliseconds
6	600	172121 Bytes	983 milliseconds
7	700	200327 Bytes	1157 milliseconds
8	800	228659 Bytes	1328 milliseconds

Tabel 5. Hasil Uji Coba Enkripsi dan Hasil Enkripsi Menggunakan kunci dengan panjang 20 karakter

Uji Coba Ke	Banyaknya record	Ukuran Data	Waktu
1	100	27741 Bytes	125 milliseconds
2	200	56909 Bytes	297 milliseconds
3	300	85451 Bytes	483 milliseconds
4	400	114603 Bytes	655 milliseconds
5	500	142711 Bytes	796 milliseconds
6	600	172121 Bytes	985 milliseconds
7	700	200327 Bytes	1156 milliseconds
8	800	228659 Bytes	1282 milliseconds

Tabel 6. Hasil Uji Coba Enkripsi dan Hasil Enkripsi Menggunakan kunci dengan panjang 30 karakter

Uji Coba Ke		Ukuran Data	Waktu
1	100	27741 Bytes	155 milliseconds
2	200	56909 Bytes	280 milliseconds
3	300	85451 Bytes	483 milliseconds
4	400	114603 Bytes	657 milliseconds
5	500	142711 Bytes	828 milliseconds
6	600	172121 Bytes	983 milliseconds
7	700	200327 Bytes	1141 milliseconds
8	800	228659 Bytes	1328 milliseconds

Uji coba yang kedua yaitu menggunakan modul enkripsi yang dibangun dengan bahasa pemrograman python, dengan panjang kunci enkripsi sepanjang 8, 20 dan 30 karakter. Hal ini lakukan untuk mengetahui apakah ada perbedaan waktu yang signifikan dengan menggunakan modul yang dibangun dengan bahasa pemrograman C, dikarenakan postgresql dibangun dengan bahasa C. Hasil pengujian dapat dilihat pada Tabel 7, Tabel 8, dan Tabel 9.

Tabel 7. Hasil Uji Coba Enkripsi dan Hasil Enkripsi Menggunakan kunci dengan panjang 8 karakter

Uji Coba Ke	Banyaknya record	Ukuran Data	Waktu
1	100	27742 Bytes	203 milliseconds
2	200	56910 Bytes	437 milliseconds
3	300	85452 Bytes	641 milliseconds
4	400	114604 Bytes	891 milliseconds
5	500	142712 Bytes	1108 milliseconds
6	600	172122 Bytes	1328 milliseconds
7	700	200328 Bytes	1546 milliseconds
8	800	228660 Bytes	1608 milliseconds

Tabel 8. Hasil Uji Coba Enkripsi dan Hasil Enkripsi Menggunakan kunci dengan panjang 20 karakter

Uji Coba	Banyaknya	Ukuran Data	Waktu
Ke	record		
1	100	27742 Bytes	202 milliseconds
2	200	56910 Bytes	437 milliseconds
3	300	85452 Bytes	656 milliseconds
4	400	114604 Bytes	922 milliseconds
5	500	142712 Bytes	1093 milliseconds
6	600	172122 Bytes	1344 milliseconds
7	700	200328 Bytes	1547 milliseconds
8	800	228660 Bytes	1750 milliseconds

Tabel 9. Hasil Uji Coba Enkripsi dan Hasil Enkripsi Menggunakan kunci dengan panjang 30 karakter

Uji Coba Ke	Banyaknya record	Ukuran Data	Waktu
Ne			
1	100	27742 Bytes	203 milliseconds
2	200	56910 Bytes	437 milliseconds
3	300	85452 Bytes	657 milliseconds
4	400	114604 Bytes	891 milliseconds
5	500	142712 Bytes	1093 milliseconds
6	600	172122 Bytes	1233 milliseconds
7	700	200328 Bytes	1562 milliseconds
8	800	228610 Bytes	1766 milliseconds

KESIMPULAN

Berdasarkan hasil ujicoba yang dilakukan selama implementasi software dapat diperoleh hasil sebagai berikut:

- Software berhasil mengamankan transmisi *query* dan hasil *query* selama keduanya berada di jaringan dengan cara mengenkripsi keduanya menggunakan algoritma RC4.
- 2. Software pada sisi client dapat melakukan operasi *insert, update, delete* dengan cara menginputkan *query* pada text box yang disediakan.
- 3. Penerapan algoritma untuk mengenkripsi *query* dan hasil *query* secara langsung berpengaruh terhadap waktu eksekusi dari *query*, hal ini dapat dilihat pada hasil uji coba.

DAFTAR PUSTAKA

- Alfred J. Menezes, Paul C. van Oorschot, Scott A. Vanstone, "HANDBOOK of APPLIED CRYP-TOGRAPHY", Asiacrypt/Auscrypt Proceedings, 1997.
- 2. Stallings, W., 2005, Cryptography and Network Security Principles and Practices, 4th Edition", Prentice Hall.
- 3. Sukmawan, B., 1998, RC4 Stream Cipher, http://www.bimacipta.com/rc4.htm, diakses 23 April 2009.
- 4. Fathansyah, 1999, *Basis Data*, Informatika, Bandung.
- 5. Munir, R., 2006, Kriptografi, Informatika, Bandung.
- 6. PostgreSQL 8.3.7 Documentation
- 7. Python 2.5 Documentation