BULLETIN OF INFORMATION TECHNOLOGY (BIT)
Vol 5, No 1, Mare 2024 , Hal. 9 - 16
ISSN 2722-0524 (media online)
DOI 10.47065/bit.v5i1.1149
https://journal.fkpt.org/index.php/BIT

Pengamanan Perintah Koneksi ke Database MySQL Menggunakan Algoritma Caesar Cipher dan Algoritma Stout Codes

Surya Darma Nasution*

Fakultas Ilmu Komputer dan Teknologi Informasi, Program Studi Teknik Informatika, Universitas Budi Darma, Medan, Indonesia
Email: darmashadow@gmail.com

Abstrak- Sistem database MySQL merupakan salah satu sistem database yang banyak digunakan di dunia, baik untuk keperluan pribadi maupun perusahaan. Namun, semakin banyaknya serangan siber yang terjadi pada sistem database membuat keamanan data dalam sistem database semakin penting untuk diamankan. Serangan siber bisa melalui SQL Injection atau pun meretas melalui perintah koneksi ke database dengan menggunakan aplikasi yang databasenya ingin diretas. Hal tersebut yang menjadi fokus permasalahan, sehingga perlu dilakukan pengamanan koneksi ke database menggunakan algoritma kriptografi. Penelitian ini bertujuan untuk meningkatkan keamanan sistem database MySQL dengan mengimplementasikan teknik kriptografi Caesar Cipher dan Teknik kompresi Stout Codes pada perintah koneksi ke database melalui aplikasi client yang dibangun menggunakan bahasa pemrograman Visual Basic. Cara pengujian yang digunakan dalam penelitian ini adalah bersifat eksperimen, dimana sistem database MySQL diuji dengan menggunakan perintah koneksi biasa dan perintah koneksi yang telah dienkripsi dengan algoritma Caesar Cipher dan algoritma Stout Sodes. Hasil pengujian menunjukkan bahwa penggunaan sistem kripto-kompresi Caesar Cipher dan Stout Codes dapat meningkatkan keamanan sistem database MySQL dengan mempersulit akses oleh pihak yang tidak berwenang dengan persentase keberhasilan sebesar 100%. Dengan demikian, penelitian ini diharapkan dapat memberikan kontribusi pada peningkatan keamanan sistem database MySQL yang saat ini semakin rawan terhadap serangan siber.

Kata Kunci: Keamanan; Database; MySQL; Kriptografi; Kompresi Data

Abstract- The MySQL database system is one of the database systems that is widely used in the world, both for personal and corporate purposes. However, the increasing number of cyber attacks occurring on database systems makes it increasingly important to secure data security in database systems. Cyber attacks can be via SQL Injection or hacking via connection commands to the database using the application whose database you want to hack. This is the focus of the problem, so it is necessary to secure connections to the database using cryptographic algorithms. This research aims to improve the security of the MySQL database system by implementing the Caesar Cipher cryptography technique and the Stout Codes compression technique in connection commands to the database via a client application built using the Visual Basic programming language. The testing method used in this research is experimental, where the MySQL database system is tested using ordinary connection commands and connection commands that have been encrypted with the Caesar Cipher algorithm and the Stout Sodes algorithm. The test results show that the use of the Caesar Cipher and Stout Codes crypto-compression systems can increase the security of the MySQL database system by making access difficult for unauthorized parties with a success percentage of 100%. Thus, it is hoped that this research can contribute to improving the security of the MySQL database system which is currently increasingly vulnerable to cyber attacks.

Keywords: Security; Databases; MySQL; Cryptography; Data Compression

1. PENDAHULUAN

Keamanan data dalam sistem database menjadi salah satu hal yang sangat penting dalam era digital ini. Sistem database MySQL merupakan salah satu sistem database yang banyak digunakan di dunia, baik untuk keperluan pribadi maupun perusahaan. Namun, semakin banyaknya serangan siber yang terjadi pada sistem database membuat keamanan data dalam sistem database semakin penting. Serangan siber bisa melalui SQL Injection atau pun meretas melalui perintah koneksi ke database melalui aplikasi yang databasenya ingin diretas. Salah satu cara mendapatkan perintah koneksi ke database bisa dilakukan dengan software decompiler, hasil dari decompiler berupa source code aplikasi dan dari situ bisa mendapatkan perintah koneksi ke database MySQL. Salah satu cara untuk meningkatkan keamanan sistem database adalah dengan menggunakan teknik kriptografi pada perintah koneksi ke database MySQL.

Terdapat berbagai macam teknik kriptografi yang digunakan untuk meningkatkan keamanan data dalam sistem database, salah satunya adalah algoritma Caesar Cipher. Algoritma Caesar Cipher adalah salah satu teknik kriptografi yang sederhana namun efektif dalam memperbaiki keamanan data. Dalam algoritma Caesar Cipher, setiap karakter dalam data yang akan dienkripsi digeser sejumlah n posisi berdasarkan urutan abjad. Namun, penggunaan teknik ini dalam sistem database masih belum banyak digunakan, sehingga perlu dilakukan penelitian lebih lanjut untuk mengeksplorasi penggunaan teknik ini dalam meningkatkan keamanan sistem database. Kelemahan dari penggunaan algoritma caesar cipher ini yaitu karena algoritmanya yang sangat sederhana jadi sangat mudah unduk di dekripsi, untuk mengatasi hal tersebut maka dalam penelitian ini dilakukan teknik kripto-kompresi dimana setelah plainteks di enkrip maka setelah itu dilakukan proses kompresi. Algoritma kompresi yang akan diterapkan dalam penelitian ini adalah algoritma Stout Codes. Dalam penelitian terdahulu yang pernah dilakukan oleh Benni Purnama dan Hetty Rohayani.AH dengan memodifikasi algoritma kriptografi caesar cipher yang menghasilkan suatu cipherteks yang bisa terbaca sehingga pasti pihak lain atau kriptanalis tidak akan curiga dengan pesan yang telah dienkripsi [1]. Priya Verma, dkk juga melakukan penelitian tentang pengembangan caesar cipher untuk keamanan yang lebih baik, hasil yang didapat dalam penelitian ini yaitu dalam teknik yang dikembangkan berisi simbol, angka, huruf besar dan karakter huruf kecil yang meningkatkan kinerja sandi Caesar. Ukuran kunci yang digunakan 82, sehingga semakin sulit untuk mendekripsi pesan dan terbukti lebih aman [2]. Kemudian Anupama Mishra juga melakukan penelitian tentang peningkatan keamanan caesar cipher dengan menggunakan metode

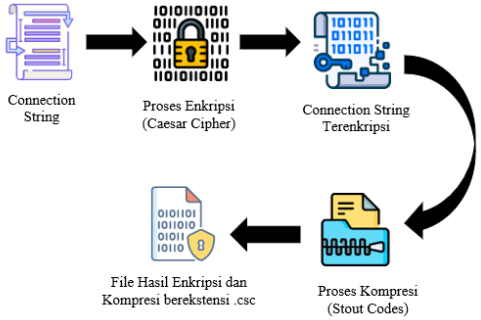
BULLETIN OF INFORMATION TECHNOLOGY (BIT)
Vol 5, No 1, Mare 2024 , Hal. 9 - 16
ISSN 2722-0524 (media online)
DOI 10.47065/bit.v5i1.1149
https://journal.fkpt.org/index.php/BIT

kriptografi yang berbeda, hasil yang dicapai dalam penelitian tersebut yaitu dengan mengkombinasikan dari dua teknik klasik ini menghasilkan sandi yang lebih aman dan kuat sehingga sangat sulit untuk dipecahkan[3].
Penelitian lainnya juga dilakukan oleh Fahrul Ikhsan Lubis, dkk tentang kombinasi dari modifikasi algoritma caesar cipher digabungkan dengan cipher transposisi, dalam tiga kali enkripsi pada percobaan yang dilakukan yaitu modifikasi caesar pada awalnya kemudian ciphertext yang dihasilkan akan dienkripsi dengan transposisi, dan terakhir, hasil dari transposisi akan dienkripsi lagi dengan modifikasi caesar kedua, demikian pula pada proses dekripsi tetapi prosesnya dilakukan secara terbalik. Hasil yang dicapai sangat bagus dan untuk mengatasinya ada banyak kemungkinan yang harus dilakukan oleh kritanalis [4].
Enas Ismael Imran dan Farah abdulameerabdulkareem pada penelitiannya tentang peningkatan caesar cipher untuk Keamanan Lebih Baik menunjukkan bahwa caesar cipher menjadi salah satu algoritma untuk mengenkripsi yang paling sederhana dan banyak teknik yang dapat digunakan untuk memperkuat bahkan melebihi apa yang dapat dicapai oleh algoritma caesar cipher tersebut [5]. O.E. Omolara, dkk dalam penelitiannya tentang pengembangan Hybrid Caesar Cipher dan Vigenere Cipher yang dimodifikasi untuk Komunikasi Data, pengujian yang dilakukan menghasilkan persentase keamanan yang tinggi sehingga menjadi sandi yang sangat kuat dan sulit dipecahkan [6].
Atish Jain, dkk dalam penelitiannya untuk meningkatkan keamanan penggantian caesar cipher menggunakan pendekatan acak, penelitian ini bertujuan untuk mengusulkan sebuah versi yang disempurnakan dari teknik substitusi sandi Caesar yang dapat mengatasi segala keterbatasan yang dihadapi klasik Caesar Cipher [7]. Imam Wahyu Utomo, dkk telah melakukan penelitian mengenai aplikasi kriptografi yang berbasis android dimana penelitian ini menggunakan penggabungan antara algoritma caesar cipher dan vigenere cipher. Hasil yang dicapai dalam penelitian tersebut adalah dalam melakukan proses enkripsi dan dekripsi tingkat kesuksesannya 75% [8].
Penelitian lainnya juga dilakukan oleh Primaningtyas Nur Arifah dan Windi Agustiar Basuki mengenai pengimplementasian algoritma kriptografi Caesar Chiper dengan menggunakan Matlab R2013a, dipenelitian ini dilakukan sedikit modifikasi dimana karakter yang diamankan selain huruf A-Z juga menambahkan karakter .,!? dan juga angka 0-9. Hasil dari penelitian ini dinilai dapat mempersulit pihak-pihak yang kurang bertanggung jawab untuk memecahkan sistem keamanan yang dibuat [9]. Sementara Muhammad Lutfi Wijaya, dkk melakukan penelitian tentang kriptografi dengan kompopsisi caesar cipher dan affine cipher untuk mengamankan pesan rahasia. Pada penelitian ini dilakukan kombinasi dengan melakukan dua kali proses enkripsi untuk mengamankan pesan rahasia, serta melakukan dua kali proses dekripsi untuk mengembalikan ke pesan semula [10].
Penerapan teknik Caesar Cipher dan Stout Codes pada perintah koneksi ke database MySQL memiliki beberapa keuntungan. Salah satunya adalah meningkatkan keamanan data dalam sistem database. Dengan mengenkripsi perintah koneksi ke database menggunakan Caesar Cipher dan Stout Codes, maka perintah koneksi tersebut akan sulit untuk dibaca oleh pihak yang tidak berwenang. Namun, penerapan teknik Caesar Cipher dan Stout Codes dalam sistem database MySQL juga memiliki beberapa kendala. Salah satunya dari kendala yang dihadapi adalah kompleksitas dalam implementasi teknik ini. Dalam mengimplementasikan teknik Caesar Cipher dan Stout Codes pada perintah koneksi ke database MySQL, perlu mempertimbangkan beberapa faktor seperti waktu proses dan efisiensi penyimpanan data. Selain itu, diperlukan pemahaman yang cukup dalam penggunaan algoritma Caesar Cipher dan algoritma Stout Codes.

2. METODOLOGI PENELITIAN

2.1 Konsep Pengamanan Yang Dilakukan

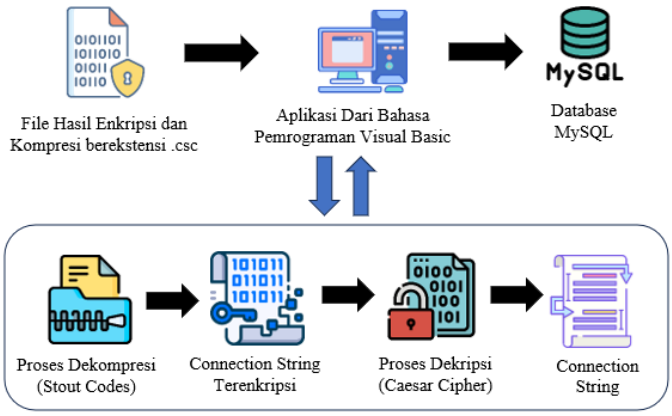
Dalam proses mengamankan perintah koneksi ke database MySQL hal yang pertama dilakukan adalah mengambil perintah koneksi atau yang biasa disebut dengan connection string pada bahasa pemrograman visual basic. Connection string tersebut yang nantinya akan dienkripsi menggunakan algoritma caesar cipher, dan hasil proses enkripsi tersebut akan diproses kembali menggunakan algoritma stout codes. Hasil dari proses enkripsi dan kompresi tersebut akan disimpan pada file yang berekstensi .csc, file tersebut yang nantinya akan digunakan untuk terhubung ke database. Untuk lebih jelasnya dapat dilihat pada gambar 1 berikut ini.



Gambar 1. Proses Pengamanan Perintah Koneksi Ke Database

BULLETIN OF INFORMATION TECHNOLOGY (BIT)
Vol 5, No 1, Mare 2024 , Hal. 9 - 16
ISSN 2722-0524 (media online)
DOI 10.47065/bit.v5i1.1149
https://journal.fkpt.org/index.php/BIT

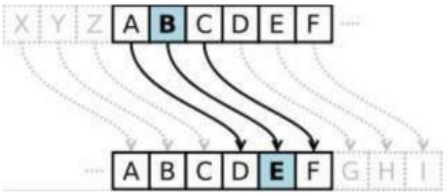
Pada aplikasi client yang dibuat dengan bahasa pemrograman visual basic hanya tinggal memanggil file .csc tersebut lalu akan melakukan dekompresi dan dekripsi sehingga akan menghasilkan connection string yang dapat terhubung ke database. Proses penggunaan file .csc tersebut dapat dilihat pada gambar 2.



Gambar 2. Proses Penggunaan File .csc

2.2 Algoritma Caesar Cipher

Metode penyandian yang paling terkenal dalam kriptografi klasik adalah Caesar Cipher, yang pertama kali digunakan oleh Julius Caesar untuk berkomunikasi dengan panglimanya [11][12]. Caesar Cipher merupakan suatu teknik kriptografi yang sangat sederhana dan umum digunakan, di mana setiap huruf pada teks aslinya digantikan oleh huruf lain dengan pergeseran sebesar nilai kunci tertentu[13][14][15]. Dalam konteks penelitian ini, nilai kunci Caesar Cipher yang digunakan adalah 3. Proses pergeseran karakter dalam Caesar Cipher dapat dilihat pada gambar 3.



Gambar 3. Proses Enkripsi Menggunakan Caesar Cipher

2.3 Algoritma Stout Codes

Kompresi merujuk pada tindakan mengompres atau mereduksi ukuran. Kompresi data merupakan proses pengkodean informasi dengan menggunakan bit lain yang memiliki nilai lebih rendah dibandingkan dengan representasi data yang belum dikodekan [16][17][18][19]. Dalam algoritma kompresi Stout Codes, metodenya sangat mirip dengan kode Elias omega dan Even-Rodeh. Codeword yang dihasilkan oleh algoritma Stout Code bergantung pada pilihan parameter "l" yang nilainya lebih besar atau sama dengan 2. Algoritma Stout Code diperkenalkan oleh Quentin Stout dan memiliki dua keluarga, yaitu rekursi [20][17]. Pada kelompok Rl, semakin banyak kelompok panjang yang dibaca sampai kelompok tersebut ditemukan diikuti oleh angka 0. Gunakan notasi L = 1 +[Log2 n] dan gambarkan representasi biner B(n, l) dari l bit bilangan bulat n. Sebagai contoh, B(12,5) = 01100. Untuk nilai l yang lebih besar atau sama dengan 2, definisikan prefiks sebagai berikut:

$$R_l\, n = \begin{cases} B(n,L), & for\ 0 \le n \le 2^l - 1, \\ R_l\,(L)B(n,L), & for\ n \ge 2^l \end{cases} \quad \text{..... (1)}$$

Kategori kedua dari sistem Stout memiliki prosedur yang serupa, namun dengan awalan yang berbeda dan ditandai sebagai Sl(n). Untuk nilai l yang kecil, kategori ini menampilkan beberapa peningkatan dibandingkan dengan sistem Rl. Terutama, mengurangi sedikit redundansi dalam kode Rl karena grup panjang tidak boleh nol (maka dari itu, kelompok panjang dalam kode omega menyandikan Li-1 dan bukan Li). Awalan Sl serupa dengan awalan Rl, dengan perbedaan bahwa kelompok panjang untuk Li disandikan sebagai Li −1 - l. Awalan Sl(n) didefinisikan secara rekursif oleh:

$$S_l\, n = \begin{cases} B(n,L), & for\ 0 \le n \le 2^l - 1, \\ R_l\,(L-1-l)B(n,L), & for\ n \ge 2^l \end{cases} \quad \text{..... (2)}$$

BULLETIN OF INFORMATION TECHNOLOGY (BIT)
Vol 5, No 1, Mare 2024 , Hal. 9 - 16
ISSN 2722-0524 (media online)
DOI 10.47065/bit.v5i1.1149
https://journal.fkpt.org/index.php/BIT

3. HASIL DAN PEMBAHASAN

3.1 Pengamanan Perintah Koneksi

Dalam penelitian ini perintah koneksi atau connection string yang akan diamankan yaitu "server=localhost; uid=root;pwd=;database=db_ujicoba", pengamanan perintah koneksi dibagi menjadi 2 (dua) bagian dimana bagian pertama mencakup proses enkripsi dan kompresi, serta bagian kedua yaitu proses dekompresi dan dekripsi. Langkah awal yang harus dilakukan untuk mengamankan perintah koneksi yaitu adalah melakukan enkripsi dengan caesar cipher dan kompresi stout codes.

3.1.1 Proses Enkripsi dan Kompresi

Caesar cipher sebagai algoritma clasik hanya bisa melakukan enkripsi hanya pada huruf saja, tetapi saat ini sudah banyak yang mengembangkannya sehingga dapat melakukan enkripsi pada karakter angka atau simbol. Pada penelitian ini masih menggunakan caesar cipher yang masih belum dikembangkan, sehingga hanya bisa melakukan enkripsi pada huruf dengan pergeseran karakter sebanyak 3 (tiga). Berikut proses enkripsi menggunakan algoritma caesar cipher :

s	e	r	v	e	r	=	l	o	c	a	l	h	o	s	t	;
v	h	u	y	h	u	=	o	r	f	d	o	k	r	v	w	;

u	i	d	=	r	o	o	t	;	p	w	d	=	;	d	a	t
x	l	g	=	u	r	r	w	;	s	z	g	=	;	g	d	w

a	b	a	s	e	=	d	b	_	u	j	i	c	o	b	a
d	e	d	v	h	=	g	e	_	x	m	l	f	r	e	d

Hasil dari proses enkripsi menggunakan caesar cipher yaitu "vhuyhu=orfdokrvw;xlg=urrw;szg=; gdwdedvh=ge_xmlfred". Dari hasil enkripsi tersebut, maka akan dilakukan proses kompresi menggunakan algoritma stout codes.

Langkah-langkah mengkompresi menggunakan stout codes yaitu:

1. Hitung frekuensi kemunculan setiap karakter, lalu urutkan dari yang paling tinggi ke yang paling rendah. Hasil dari langkah ini dapat dilihat pada tabel 1.

Tabel 1. Proses Menghitung Frekuensi dan Pengurutannya

n	Karakter	Frek
1	R	5
2	D	5
3	=	4
4	G	4
5	V	3
6	H	3
7	U	3
8	W	3
9	;	3
10	E	3

n	Karakter	Frek
11	O	2
12	F	2
13	X	2
14	L	2
15	Y	1
16	K	1
17	S	1
18	Z	1
19	_	1
20	M	1

2. Pembentukan codeword dan membuat stringbit
   Dalam melakukan pembentukan cordword, terlebih dahulu menentukan nilai *l*. Disini kita mengambil nilai *l*=2.
   a. Untuk nilai $0 \leq n \leq 2^l-1$
      Karena nilai *l*=2, maka $0 \leq n \leq 2^2-1= 3$
      1) n = 1, maka codeword yang dihasilkan berupa nilai biner dari n yaitu 1 dan diambil sebanyak *l* yaitu 2 sehingga codeword-nya adalah 01.
      2) n = 2, maka codeword yang dihasilkan berupa nilai biner dari n yaitu 10 dan diambil sebanyak *l* yaitu 2 sehingga codeword-nya adalah 10.
      3) n = 3, maka codeword yang dihasilkan berupa nilai biner dari n yaitu 11 dan diambil sebanyak *l* yaitu 2 sehingga codeword-nya adalah 11.
   b. Untuk nilai $n= 2^l$
      Karena nilai *l*=2, maka $n \geq 2^2 = 4$
      1) n = 4 dengan nilai biner 100, maka nilai L = 3 diambil dari panjang bit biner nilai n dan langkah selanjutnya mencari nilai R.

BULLETIN OF INFORMATION TECHNOLOGY (BIT)
Vol 5, No 1, Mare 2024 , Hal. 9 - 16
ISSN 2722-0524 (media online)
DOI 10.47065/bit.v5i1.1149
https://journal.fkpt.org/index.php/BIT

R_2(3-1-2) = 0, nilai binernya adalah 0 dan diambil sebanyak nilai *l* yaitu 2 sehingga nilainya dari R = 00.
R_l(L-1-l) B (n, L) = 00 100

2) n = 5 nilai binernya 101, maka nilai L = 3.
R_2(3-1-2) = 0, nilai binernya adalah 0 dan diambil sebanyak nilai *l* yaitu 2 sehingga nilainya dari R = 00.
R_l(L-1-l) B (n, L) = 00 101
3) n = 6 nilai binernya 110, maka nilai L = 3.
R_2(3-1-2) = 0, nilai binernya adalah 0 dan diambil sebanyak nilai *l* yaitu 2 sehingga nilainya dari R = 00.
R_l(L-1-l) B (n, L) = 00 110
4) n = 7 nilai binernya 111, maka nilai L = 3.
R_2(3-1-2) = 0, nilai binernya adalah 0 dan diambil sebanyak nilai *l* yaitu 2 sehingga nilainya dari R = 00.
R_l(L-1-l) B (n, L) = 00 111
5) n = 8 nilai binernya 1000, maka nilai L = 4.
R_2(4-1-2) = 1, nilai binernya adalah 1 dan diambil sebanyak nilai *l* yaitu 2 sehingga nilainya dari R = 01.
R_l(L-1-l) B (n, L) = 01 1000
6) n = 9 nilai binernya 1001, maka nilai L = 4.
R_2(4-1-2) = 1, nilai binernya adalah 1 dan diambil sebanyak nilai *l* yaitu 2 sehingga nilainya dari R = 01.
R_l(L-1-l) B (n, L) = 01 1001
7) n = 10 nilai binernya 1010, maka nilai L = 4.
R_2(4-1-2) = 1, nilai binernya adalah 1 dan diambil sebanyak nilai *l* yaitu 2 sehingga nilainya dari R = 01.
R_l(L-1-l) B (n, L) = 01 1010
8) n = 11 nilai binernya 1011, maka nilai L = 4.
R_2(4-1-2) = 1, nilai binernya adalah 1 dan diambil sebanyak nilai *l* yaitu 2 sehingga nilainya dari R = 01.
R_l(L-1-l) B (n, L) = 01 1011
9) n = 12 nilai binernya 1100, maka nilai L = 4.
R_2(4-1-2) = 1, nilai binernya adalah 1 dan diambil sebanyak nilai *l* yaitu 2 sehingga nilainya dari R = 01.
R_l(L-1-l) B (n, L) = 01 1100
10) n = 13 nilai binernya 1101, maka nilai L = 4.
R_2(4-1-2) = 1, nilai binernya adalah 1 dan diambil sebanyak nilai *l* yaitu 2 sehingga nilainya dari R = 01.
R_l(L-1-l) B (n, L) = 01 1101
11) n = 14 nilai binernya 1110, maka nilai L = 4.
R_2(4-1-2) = 1, nilai binernya adalah 1 dan diambil sebanyak nilai *l* yaitu 2 sehingga nilainya dari R = 01.
R_l(L-1-l) B (n, L) = 01 1110
12) n = 15 nilai binernya 1111, maka nilai L = 4.
R_2(4-1-2) = 1, nilai binernya adalah 1 dan diambil sebanyak nilai *l* yaitu 2 sehingga nilainya dari R = 01.
R_l(L-1-l) B (n, L) = 01 1111
13) n = 16 nilai binernya 10000, maka nilai L = 5.
R_2(5-1-2) = 2, nilai binernya adalah 10 dan diambil sebanyak nilai *l* yaitu 2 sehingga nilainya dari R = 10.
R_l(L-1-l) B (n, L) = 10 10000
14) n = 17 nilai binernya 10001, maka nilai L = 5.
R_2(5-1-2) = 2, nilai binernya adalah 10 dan diambil sebanyak nilai *l* yaitu 2 sehingga nilainya dari R = 10.
R_l(L-1-l) B (n, L) = 10 10001
15) n = 18 nilai binernya 10010, maka nilai L = 5.
R_2(5-1-2) = 2, nilai binernya adalah 10 dan diambil sebanyak nilai *l* yaitu 2 sehingga nilainya dari R = 10.
R_l(L-1-l) B (n, L) = 10 10010
16) n = 19 nilai binernya 10011, maka nilai L = 5.
R_2(5-1-2) = 2, nilai binernya adalah 10 dan diambil sebanyak nilai *l* yaitu 2 sehingga nilainya dari R = 10.
R_l(L-1-l) B (n, L) = 10 10011
17) n = 20 nilai binernya 10100, maka nilai L = 5.
R_2(5-1-2) = 2, nilai binernya adalah 10 dan diambil sebanyak nilai *l* yaitu 2 sehingga nilainya dari R = 10.
R_l(L-1-l) B (n, L) = 10 10100

Setelah selesai membentuk codeword, berikutnya mengganti karakter dengan codeword lalu menghasilkan string bit. Tabel 2 menunjukan penggantian karakter dengan codeword.

Tabel 2. Mengganti Karakter Dengan Codeword

n	Karakter	Frek	Codeword
1	R	5	01
2	D	5	10
3	=	4	11
4	G	4	00100

BULLETIN OF INFORMATION TECHNOLOGY (BIT)
Vol 5, No 1, Mare 2024 , Hal. 9 - 16
ISSN 2722-0524 (media online)
DOI 10.47065/bit.v5i1.1149
https://journal.fkpt.org/index.php/BIT

n	Karakter	Frek	Codeword
5	V	3	00101
6	H	3	00110
7	U	3	00111
8	W	3	011000
9	;	3	011001
10	E	3	011010
11	O	2	011011
12	F	2	011100
13	X	2	011101
14	L	2	011110
15	Y	1	011111
16	K	1	1010000
17	S	1	1010001
18	Z	1	1010010
19	_	1	1010011
20	M	1	1010100

Setelah mengganti karakter dengan codeword, maka yang harus dilakukan berikutnya menyusun sesuai kalimat yang akan dikompresi. Dari proses penyusunan dan pergantian bit didapatkan hasil dari string bitnya, yaitu : 00101001100011101111100110001111101101101011100100110111010000010010101100001100101110101110010010010011000100110101000101001101100100011010101001101110110101000111100111000101101010.

3. Penambahan padding dan flagging.
   Penambahan padding dan flagging dibutuhkan agar jumlah stringbit yang dihasilkan dari proses kompresi habis dibagi 8. Jumlah bit dari stringbit yaitu 236 dan tidak habis dibagi 8 dengan menyisakan angka 4, sehingga perlu ditambahkan padding dengan menambahkan 0 sebanyak 7 - n + "1" di ujung bit string, dimana n= sisa hasil bagi tersebut. Dengan menggunakan rumus tersebut maka 7-4+"1" = 0001. Penambahan flagging dilakukan dengan menggunakan rumus 9-n, sehingga 9-4 = 5 dinyatakan dalam bentuk bilangan biner 8-bit sehingga menghasilkan 00000101. Hasil dari string bit yang sudah ditambahkan flagging yaitu : 00101001100011101111100110001111101101101011100100110111010000010010101100001100101110101110010010011000111010101100001100110100011010010001001101100100100100110001001101010001010011011001000111010101001101110110101000111100111000101101010**000100000101**.

   Dari string bit tersebut dibagi per 8 bit, lalu dirubah ke karakter sehingga karakter yang dihasilkan yaitu :)Žù¶¹7A+ºñ3«Ñ¤M'LME6F©»Qçi. Hasil tersebut yang tersimpan dalam file berekstensi .csc.

3.1.2 Proses Dekompresi dan Dekripsi

Proses untuk mengembalikan perintah koneksi yang sudah diamankan agar kembali menjadi perintah koneksi yang dikenali oleh bahasa pemrograman yaitu dengan melakukan dekompresi terlebih dahulu, dan kemudian dilakukan dekripsi.

Langkah-langkah dekompresi algoritma stout code yaitu :

1. Ambil karakter yaang ingin di dekompresi, dalam hal ini adalah)Žù¶¹7A+ºñ3«Ñ¤M'LME6F©»Qçi dan rubah menjadi nilai biner. Hasil bilangan binernya yaitu 00101001100011101111100110001111101101101011100100110111010000010010101100001100101110101110010010011000111010101100001100110100011010010001001101100100100100110001001101010001010011011001000111010101001101110110101000111100111000101101010**000100000101**.
2. Hilangkan padding dan flagging dengan cara mengambil 8 bit nilai biner pada akhir stringbit, lalu ubah menjadi nilai desimal. Delapan bit terakhir adalah 00000101 dan jika diubah menjadi nilai desimal adalah 5 nyatakan dalam variabel n. Hilangkan bit pada akhit string bit sebanyak 7+n = 7+5 = 12 maka hilangkan 12 bit terakhir.
3. Baca nilai stringbit persatu bit dari kiri ke kanan lalu melakukan pencocokan dengan nilai codeword yang ada pada tabel 3.2. Jika terdapat nilai biner yang sama dengan codeword maka ganti nilai biner dengan karakter yang memiliki codeword tersebut.
4. Ulangi langkah ke 3 sampai tidak ada stringbit yang tersisa.
5. Hasil dari proses dekompresi pada langkah ke 3 dan langkah ke 4 yaitu "vhuyhu=orfdokrvw;xlg=urrw;szg=;gdwdedvh=ge_xmlfred".

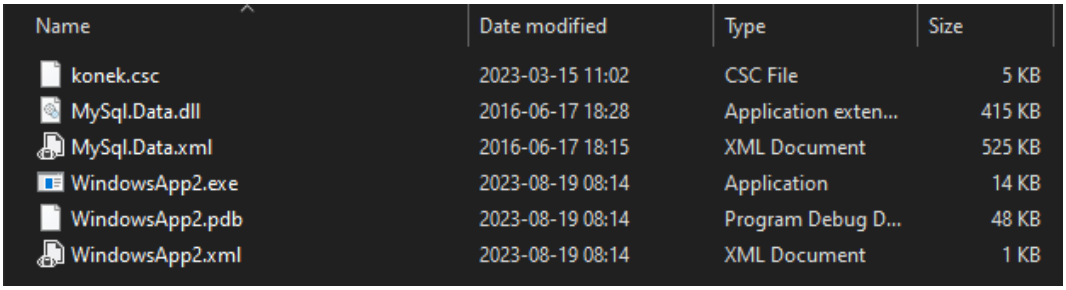
Berdasarkan dari hasil proses dekompresi tersebut, maka akan dilakukan proses dekripsi dengan algoritma caesar cipher. Untuk melakukan dekripsi dengan algoritma caesar cipher, cukup lakukan pengurangan karakter sebanyak 3 karakter sebelumnya atau dengan kata lain akan mundur sebanyak 3 karakter (selain simbol). Dari hasil proses dekripsi diatas maka perintah koneksi yang dihasilkan yaitu : "server=localhost;uid=root;pwd=;database=db_ujicoba". Dengan hasil tersebut dapat dikatakan bahwa proses dekompresi dan proses dekripsi telah berhasil mengembalikan perintah koneksi yang telah diamankan menjadi perintah koneksi yang dimengerti oleh bahasa pemrograman.

BULLETIN OF INFORMATION TECHNOLOGY (BIT)
Vol 5, No 1, Mare 2024 , Hal. 9 - 16
ISSN 2722-0524 (media online)
DOI 10.47065/bit.v5i1.1149
https://journal.fkpt.org/index.php/BIT

3.2 Pengujian

Pada bab sebelumnya telah dikatakan bahwa output dari perintah koneksi yang telah diamankan adalah 1 (satu) file berekstensi .csc yang berisi perintah koneksi yang telah diamankan dengan algoritma caesar cipher dan algoritma stout codes. Berikut penggunaan file .csc tersebut pada bahasa pemrograman visual basic:

1. Letakkan file .csc pada lokasi yang sama dengan file .exe dari aplikasi tersebut. Dapat dilihat pada gambar 3 bahwa aplikasi yang dibuat bernama WindowsApp.exe dan didalam folder yang sama terdapat file berekstensi .csc.



Gambar 3. Lokasi file csc

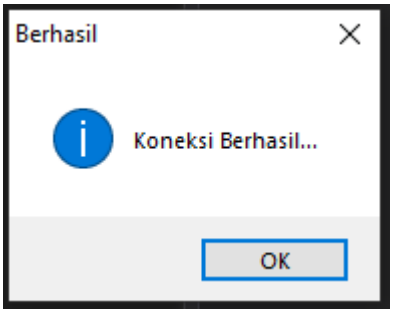
2. Pada kode program untuk memanggil perintah koneksi lakukan proses dekompresi dan dekripsi untuk mendapatkan perintah koneksi yang dapat dimengerti oleh bahasa pemrograman.

```
Sub BUKADB()
    Dim hasil_dekompresi, hasil_dekrip As String
    hasil_dekompresi = Dekompresi_SC(System.IO.File.ReadAllText(My.Application.Info.DirectoryPath & "\konek.csc"))
    hasil_dekrip = Dekrip_CS(hasil_dekompresi)
    Try
        db.ConnectionString = hasil_dekrip
        db.Open()
        MsgBox("Koneksi Berhasil...", MsgBoxStyle.Information, "Berhasil")
    Catch ex As Exception
        MsgBox("Tidak Terkoneksi Ke Server", vbExclamation, "Koneksi Gagal")
    End Try
End Sub
```

Gambar 4. Penggunaan File .csc Dalam Bahasa Pemrograman

Pada gambar 4 dapat dilihat bahwa langkah pertama yaitu dengan melakukan proses dekompresi terhadap file konek.csc, dan hasil dekompresi tersebut akan didekripsi. Hasil proses dekripsi tersebut yang digunakan dalam connection string untuk terkoneksi ke database.

Pengujian juga dilakukan apakah aplikasi yang dibuat dengan bahasa pemrograman dapat terhubung ke database dengan menggunakan perintah koneksi yang telah diamankan. Dalam proses pengujian ini didapatkan hasil bahwa aplikasi dapat terkoneksi ke database.



Gambar 5. Pengujian Koneksi Ke Database Telah Berhasil.

4. KESIMPULAN

Proses mengamankan perintah koneksi ke database MySQL pada aplikasi yang dibuat menggunakan bahasa pemrograman visual basic dengan melakukan proses enkripsi dengan algoritma caesar cipher terlebih dahulu dan hasil dari proses tersebut akan dikompresikan lagi dengan algoritma stout codes. Hasil penerapan algoritma caesar cipher dan algoritma stout codes untuk mengkompresi perintah koneksi ke database MySQL telah berhasil dilakukan, berdasarkan dari hasil pengujian aplikasi yang dibuat dapat terkoneksi ke database melalui perintah koneksi yang telah diamankan dan disimpan dalam file berkestensi .csc.

REFERENCES

[1] B. Purnama and A. H. H. Rohayani, “A New Modified Caesar Cipher Cryptography Method with LegibleCiphertext from a Message to Be Encrypted,” *Procedia Comput Sci*, vol. 59, no. Iccsci, pp. 195–204, 2015, doi: 10.1016/j.procs.2015.07.552.

BULLETIN OF INFORMATION TECHNOLOGY (BIT)
Vol 5, No 1, Mare 2024 , Hal. 9 - 16
ISSN 2722-0524 (media online)
DOI 10.47065/bit.v5i1.1149
https://journal.fkpt.org/index.php/BIT

[2] P. Verma, G. S. Gaba, and R. Miglani, "Diversified Caesar Cipher for Impeccable Security," *International Journal of Security and Its Applications*, vol. 11, no. 2, pp. 33–40, 2017, doi: 10.14257/ijsia.2017.11.2.04.

[3] A. Mishra, "Enhancing Security of Caesar Cipher Using Different Methods," *Int J Res Eng Technol*, vol. 02, no. 09, pp. 327–332, 2013, doi: 10.15623/ijret.2013.0209049.

[4] F. I. Lubis, H. F. S. Simbolon, T. P. Batubara, and R. W. Sembiring, "Combination of caesar cipher modification with transposition cipher," *Advances in Science, Technology and Engineering Systems*, vol. 2, no. 5, pp. 22–25, 2017, doi: 10.25046/aj020504.

[5] P. E. Ismael Imran and P. F. abdulameerabdulkareem, "Enhancement Caesar Cipher for Better Security," *IOSR J Comput Eng*, vol. 16, no. 3, pp. 01–05, 2014, doi: 10.9790/0661-16350105.

[6] O. E. Omolara, A. I. Oludare, and S. E. Abdulahi, "Developing a Modified Hybrid Caesar Cipher and Vigenere Cipher for Secure Data Communication," *Issn*, vol. 5, no. 5, pp. 2222–1719, 2014, [Online]. Available: www.iiste.org

[7] A. Jain, R. Dedhia, and A. Patil, "Enhancing the Security of Caesar Cipher Substitution Method using a Randomized Approach for more Secure Communication," *Int J Comput Appl*, vol. 129, no. 13, pp. 6–11, 2015, doi: 10.5120/ijca2015907062.

[8] I. W. Utomo, R. Latifah, and D. Risanty, "Aplikasi Kriptografi Berbasis Android Menggunakan Algoritma Caesar Cipher Dan Vigenere Cipher," *Jurnal Sistem Informasi, Teknologi Informasi dan Komputer*, vol. 9, no. 2, pp. 142–149, 2018.

[9] P. N. Arifah and W. A. Basuki, "Implementasi Kriptografi Caesar Chiper," *SEMINAR MATEMATIKA DAN PENDIDIKAN MATEMATIKA UNY 2017*, pp. 297–304, 2017.

[10] M. L. Wijaya, K. Yulianti, and H. S. Husain, "Caesar Cipher Dan Affine Cipher Untuk Mengubah Pesan Rahasia," *E u r e k a M a t i k a*, vol. 5, no. 1, pp. 30–45, 2017.

[11] S. K. Verma and D. B. Ojha, "An innovative Enciphering Scheme based on Caesar Cipher," *International Journal of Innovative Science, Engineering & Technology (IJSET)*, vol. 1, no. 5, pp. 25–27, 2014.

[12] F. N. Nife, "A New Modified Cesar Cipher Cryptographic Method Along With Rail Fence to Encrypt Message 1-Dynamic Key Generation," *Int J Adv Res (Indore)*, vol. 3, no. 2, pp. 331–335, 2015.

[13] Y. Dwi Putri, R. Rosihan, and S. Lutfi, "Penerapan Kriptografi Caesar Cipher Pada Fitur Chatting Sistem Informasi Freelance," *JIKO (Jurnal Informatika dan Komputer)*, vol. 2, no. 2, pp. 87–94, 2019, doi: 10.33387/jiko.v2i2.1319.

[14] B. Oktaviana and A. P. Utama Siahaan, "Three-Pass Protocol Implementation in Caesar Cipher Classic Cryptography," *IOSR J Comput Eng*, vol. 18, no. 04, pp. 26–29, 2016, doi: 10.9790/0661-1804032629.

[15] S. Y. Wulandari, "Cryptography : A Combination of Caesar and Affine Cipher to Conceal the Message," *PROC. INTERNAT. CONF. SCI. ENGIN.*, vol. 3, no. April, pp. 741–744, 2020.

[16] L. Marlina, A. Putera, U. Siahaan, H. Kurniawan, and I. Sulistianingsih, "Data Compression Using Elias Delta Code," *International Journal of Recent Trends in Engineering & Research*, vol. 3, no. 8, pp. 210–217, 2017, doi: 10.23883/IJRTER.2017.3406.TEGS6.

[17] S. D. Nasution, "Data Compression Using Stout Codes," *The IJICS (International Journal of Informatics and Computer Science)*, vol. 3, no. 1, pp. 28–33, 2019.

[18] S. D. Nasution, G. L. Ginting, M. Syahrizal, and R. Rahim, "Data Security Using Vigenere Cipher and Goldbach Codes Algorithm," *International Journal of Engineering Research & Technology (IJERT)*, vol. 6, no. 01, pp. 360–363, 2017.

[19] A. Singh and Y. Bhatnagar, "Enhancement of Data Compression Using Incremental Encoding," *Int J Sci Eng Res*, vol. 3, no. 5, pp. 1457–1465, 2012, [Online]. Available: http://www.ijser.org/paper/Enhancement-of-Data-Compression-Using-Incremental-Encoding.html

[20] D. Salomon and G. Motta, *HandBook Of Data Compression*. Springer, 2010.