RANCANG BANGUN APLIKASI ENKRIPSI DATABASE MYSQL DENGAN ALGORITMA BLOWFISH

Tetuko Pambudi Nusa
D3 Manajemen Informatika, Fakultas Teknik, Universitas Negeri Surabaya, tetukopambudinusa@gmail.com

Anita Qoiriah
D3 Manajemen Informatika, Fakultas Teknik, Universitas Negeri Surabaya, anitaqoi@yahoo.com

Abstrak

Untuk menjaga keamanan data ataupun informasi yang tersimpan dalam *database* MySQL adalah dengan menggunakan enkripsi. Ada banyak algoritma enkripsi yang ada dan salah satunya adalah algoritma *Blowfish*. Algoritma *Blowfish* merupakan algoritma modern kunci simetris berbentuk *chiperblock*. Enkripsi dilakukan dengan menggunakan kunci tertentu, sehingga menghasilkan *chipertext* yang tidak bisa dibaca. *Chipertext* tersebut dapat dikembalikan seperti semula jika didekripsi menggunakan kunci yang sama.
Algoritma *Blowfish* memiliki 16 putaran dan masukan berupa data 64 bit. Bagi data 64 bit tersebut menjadi 2 bagian XL dan XR yang masing-masing 32 bit, selanjutnya lakukan operasi XL = XL xor Pi dan XR = F(XL) xor XR, kemudian tukar XL dan XR, lakukan proses sebanyak 16 kali. Pada proses ke-17 lakukan operasi untuk XR = XR xor P17 dan XL = XL xor P18, kemudia satukan kembali XL dan XR sehingga menjadi 64 bit kembali sehingga menghasilkan *chipertext*.
Dari penggunaan algoritma Blowfish ini membuat data dari *database* MySQL yang meliputi *database*, *table*, *field*, dan *record* tidak dapat terbaca karena telah terenkripsi, sehingga hanya *user* tertentu yang dapat membaca isi dari *database* dengan cara mendekripsinya.

Kata Kunci: Algoritma *Blowfish*, enkripsi, dekripsi, kunci simetris.

PENDAHULUAN

Database adalah sebuah bagian terpenting dalam suatu sistem informasi. Dengan adanya *database*, semua jenis data penting yang kita miliki dapat tersimpan dengan rapi dan dapat diakses kapan saja untuk mendapatkan informasi yang kita butuhkan**.**

Ada berbagai macam program manajemen data (*database*) yang didesain memiliki tampilan yang mudah digunakan (*User Friendly*) sehingga pengguna tidak perlu bingung dengan antar muka program *database* tersebut. Namun ada juga program *database* yang tanpa antar muka, dan hanya menggunakan *command text* atau *console* sebagai sarana komunikasi dengan seorang penggunanya (*User*), salah satu program *database* yang tidak memiliki *interface* adalah MySQL. Meskpun tidak memiliki *interface* sendiri, MySQL memiliki kelebihan daripada program manajemen data yang lain. Dengan membuat sebuah *interface* untuk MySQL maka kekurangan dari program tersebut telah berkurang, dan data yang terdapat didalam *database* tersebut dapat diakses kapan saja. Dengan mudahnya akses untuk melihat isi dari sebuah *database*, maka tingkat keamanan data akan menjadi berkurang. Untuk itu diperlukan suatu enkripsi untuk menjaga keamanan data tersebut.

Algoritma *Blowfish* adalah sebuah algoritma enkripsi simetris yang berarti bahwa algoritma ini menggunakan kunci yang sama baik untuk melakukan enkripsi dan dekripsi. Dengan menggunakan algoritma blowfish maka keamananpun akan lebih meningkat.

Berdasarkan latar belakang tersebut, maka dapat dirumuskan permasalahan, Bagaimanakah cara merancang, dan mengimplementasikan metode *blowfish* untuk mengamankan *database* MySQL sehingga hanya pengguna tertentulah yang dapat melihat isi dari *database* MySQL tersebut?

Tujuan dari pembuatan apliksai ini adalah untuk mengenkripsi data yang ada dalam database MySQL sehingga tidak semua orang dapat mengaksesnya.

Serta manfaat pembuatan aplikasi ini adalah Keamanan informasi dalam sebuah *database* akan lebih terjamin

LANDASAN TEORI

MySQL

SQL yang merupakan kepanjangan dari Structured Query Language. SQL merupakan bahasa terstruktur yang khusus digunakan untuk mengolah *database*. SQL pertama kali didefinisikan oleh American National Standards Institute (ANSI) pada tahun 1986. MySQL adalah sebuah sistem manajemen *database* yang bersifat open source. MySQL adalah pasangan serasi dari PHP. MySQL dapat digunakan untuk membuat dan mengola

database beserta isinya. Kita dapat memanfaatkan MySQL untuk menambahkan, mengubah dan menghapus data yang berada dalam *database*. MySQL merupakan sisitem manajemen *database* yang bersifat relational. Artinya data-data yang dikelola dalam *database* akan diletakkan pada beberapa tabel yang terpisah sehingga manipulasi data akan menjadi jauh lebih cepat. MySQL dapat digunakan untuk mengelola *database* mulai dari yang kecil sampai dengan yang sangat besar. MySQL juga dapat menjalankan perintah-perintah Structured Query Language (SQL) untuk mengelola *database-database* yang ada di dalamnya.

Enkripsi

Di bidang kriptografi, enkripsi ialah proses mengamankan suatu informasi dengan membuat informasi tersebut tidak dapat dibaca tanpa bantuan pengetahuan khusus (Equan, 2011).

Enkripsi dapat digunakan untuk tujuan keamanan, tetapi teknik lain masih diperlukan untuk membuat komunikasi yang aman, terutama untuk memastikan integritas dan autentikasi dari sebuah pesan. Contohnya, *Message Authentication Code* (MAC) atau *digital signature*. Penggunaan yang lain yaitu untuk melindungi dari analisis jaringan komputer.

Algoritma Blowfish

Blowfish merupakan algoritma kunci simetrik cipher blok yang dirancang pada tahun 1993 oleh Bruce Schneier untuk menggantikan DES. Pada saat itu banyak sekali rancangan algoritma yang ditawarkan, namun hampir semua terhalang oleh paten atau kerahasiaan pemerintah Amerika. Schneier menyatakan bahwa *blowfish* bebas paten dan akan berada pada domain publik. Dengan pernyataan Schneier tersebut *blowfish* telah mendapatkan tempat di dunia kriptografi, khususnya bagi masyarakat yang membutuhkan algoritma kriptografi yang cepat, kuat, dan tidak terhalang oleh lisensi (Schneier, 1996).

Bagian-Bagian dalam Algoritma *Blowfish*

Blowfish termasuk dalam enkripsi block Chiper 64-bit dengan panjang kunci minimal 32-bit sampai 448-bit. Algoritma *Blowfish* terdiri atas dua bagian (Schneier, 1996). :

1) Key-Expansion

Berfungsi merubah kunci (Minimum 32-bit, Maksimum 448-bit) menjadi beberapa array subkunci r(*subkey*) dengan total 4168 byte (Sutanto, 2009).

2) Enkripsi Data

Terdiri dari iterasi fungsi sederhana (*Feistel Network*) sebanyak 16 kali putaran. Setiap putaran terdiri dari permutasi kunci-dependent dan substitusi kunci dan datadependent. Semua operasi adalah penambahan (*addition*) dan XOR pada variabel 32-bit. Operasi tambahan lainnya hanyalah empat penelusuran tabel (*table lookup*) array berindeks untuk setiap putaran.

Proses enkripsi data terjadi pada jaringan feistel (*Feistel Network*), mengandung fungsi pengulangan sederhana sebanyak enam belas kali. Setiap iterasi, terdiri dari sebuah permutasi yang tidak bergantung pada kunci dan sebuah substitusi yang tidak bergantung pada data dan kunci. Semua operasi merupakan penambahan dan XOR pada *word* 32-bit. Operasi penambahan yang dilakukan hanya merupakan empat indeks *array data lookup* pada setiap iterasi. Input adalah elemen 64-bit, X.

Alur Algoritma Blowfish

Untuk alur algoritma enkripsi dengan metode *Blowfish* dijelaskan sebagai berikut.

1. Bentuk inisial array P sebanyak 18 buah (P1,P2,……P18) masing-masing bernilai 32-bit. Array P terdiri dari delapan belas kunci 32-bit subkunci :
2. P1,P2,……,P18
3. Bentuk S-box sebanyak 4 buah masing-masing bernilai 32-bit yang memiliki masukan 256. Empat 32-bit S-box masing-masing mempunyai 256 entri :
   a. $S_{1,0},S_{1,1},\ldots\ldots\ldots\ldots,S_{1,255}$
   b. $S_{2,0},S_{2,1},\ldots\ldots\ldots\ldots,S_{2,255}$
4. $S_{3,0},S_{3,1},\ldots\ldots\ldots\ldots,S_{3,255}$
5. $S_{4,0},S_{4,1},\ldots\ldots\ldots\ldots,S4,255$
6. Plainteks yang akan dienkripsi diasumsikan sebagai masukan, Plainteks tersebut diambil sebanyak 64-bit, dan apabila kurang dari 64-bit maka kita tambahkan bitnya, supaya dalam operasi nanti sesuai dengan datanya.
7. Hasil pengambilan tadi dibagi 2, 32-bit pertama disebut XL, 32-bit yang kedua disebut XR.
8. Selanjutnya lakukan operasi XL = XL xor Pi dan XR = F(XL) xor XR.
9. Hasil dari operasi di atas ditukar XL menjadi XR dan XR menjadi XL.
10. Lakukan sebanyak 16 kali, perulangan yang ke-16 lakukan lagi proses penukaran XL dan XR.
11. Pada proses ke-17 lakukan operasi untuk XR = XR xor P17 dan XL = XL xor P18.
12. Proses terakhir satukan kembali XL dan XR sehingga menjadi 64-bit kembali.

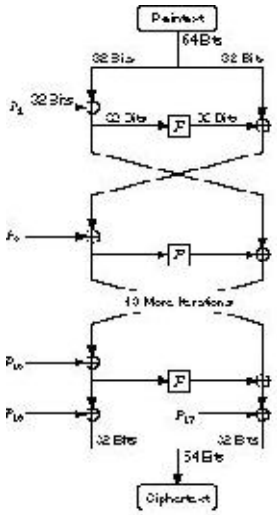
Diagram algroritma *Blowfish* seperti Gambar 1.

: **Penghitungan Subkunci Algoritma *Blowfish***

a. Pertama-tama inilialisasi *P-array* dan kemudian empat *S-box* secara berurutan dengan string yang tetap. String ini terdiri atas digit hexadesimal dari Pi.

Digit heksadesimal bilangan pi merupakan deret bilangan pi dalam bentuk heksadesimal ***(n*** = 3,243Sa8885a308d3 13 198a2e03707344 A4093822 ...) yang dapat dibanglatkan oleh formula Bailey-Borwein-Plouffe (Finch, 1995).

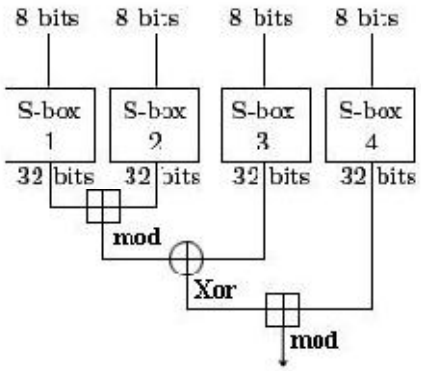
b. XOR P1 dengan 32-bit pertama kunci, XOR P2 dengan 32-bit kedua dari kunci dan seterusnya untuk setiap bit dari kunci (sampai P18). Ulangi terhadap bit kunci sampai seluruh P-array di XOR dengan bit kunci.
c. Enkrip semua string nol dengan algoritma *Blowfish* dengan menggunakan subkunci seperti dijelaskan pada langkah (a) dan (b).



Gambar 1. Diagram Proses *Blowfish* (Sutanto, 2009)**.**

d. Ganti P1 dan P2 dengan keluaran dari langkah (c).
e. Enkrip keluaran dari langkah (c) dengan algoritma *Blowfish* dengan subkunci yang sudah dimodifikasi.
f. Ganti P3 dan P4 dengan keluaran dari langkah (e).
g. Lanjutkan proses tersebut, ganti seluruh elemen dari P-array, kemudian seluruh keempat S-box berurutan, dengan keluaran yang berubah secara kontiyu dari algoritma *Blowfish* (Guritman, 2003).

Berikut Gambar 2 adalah fungsi F dalam *Blowfish* :



Gambar 2. Fungsi F (Feistel).

Secara keseluruhan terdapat 521 iterasi atau putaran yang dibutuhkan untuk membangkitkan seluruh subkunci yang dibutuhkan (Guritman, 2003). Aplikasi kemudian dapat menyimpan subkunci yang telah dihasilkan. Proses pembangkitan kunci ini tidak perlu selalu dilakukan setiap saat.

Untuk dekripsi sama persis dengan enkripsi, kecuali array P (P1,P2,…….,P18) digunakan dengan urutan terbalik (Sitinjak, 2010).

METODE REKAYASA

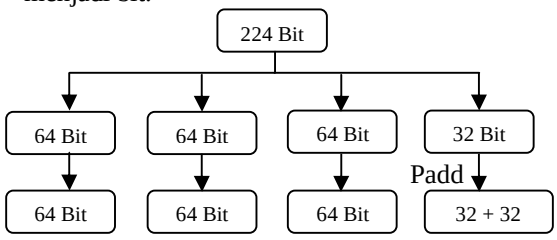
Ruang Lingkup Sistem

Aplikasi yang dibuat ini adalah sebuah sistem yang digunakan untuk melakukan enkripsi *database* dari MySQL dengan algoritma *Blowfish.* Enkripsi pada *database* ini meliputi *database*, tabel, *field*, dan *record* sehingga tidak semua orang dapat mengakses isi dari suatu *database* yang terdapat dalam MySQL

Alur Enkripsi

Untuk alur algoritma enkripsi dengan metode *Blowfish* dijelaskan sebagai berikut.

1) Sebelum melakukan enkripsi, algoritma *blowfish* akan menghitung *sub-key* yang akan digunakan dengan langkah sebagai berikut :
   a) Inisialisasi delapan belas *p-array* dan empat *s-array* dengan string yang sudah pasti yang terdiri dari heksadesimal pi
   b) Setelah inisialisasi, XOR P1 dengan 32 bit kunci pertama, XOR P2 dengan 32 bit kunci ke dua, dan seterusnya hingga P18.
   c) Enkrip semua string nol dengan algoritma *blowfish* dengan menggunakan subkunci seperti dijelaskan di langkah (1) dan (2)
   d) Ganti P1 dan P2 dengan keluaran dari langkah (3)
   e) enkrip keluaran dari langkah (3) dengan *p-array* dan *s-array* (s-box). P1 dan P2 yang sudah diganti dengan hasil dari langkah (3) itu dienkripsi dengan algoritma *blowfish*.
   f) ganti P3 dan P4 dengan keluaran dari langkah (5). Hasil dari langkah (5) digunakan untuk menggantikan P3 dan P4.
2) Setelah selesai melakukan perhitungan *sub-key* kemudian lakukan pengubahan terhadap teks menjadi bit.



Gambar 3. Proses Pemecahan Bit dan Penambahan Bit

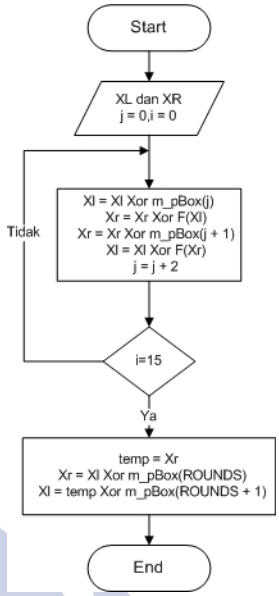
3) Setelah dilakukan pemecahan bit, maka proses selanjutnya adalah melakukan enkripsi oleh algoritma *blowfish* seperti Gambar 1.
Plaintext (teks jelas sebelum terenkripsi) diambil sebesar 64 bit lalu dibagi 2 menjadi 32 bit, 32 bit pertama disebut XL, dan 32 bit kedua disebut XR. Selanjutnya lakukan operasi XL = XL xor P1 dan XR = XR xor F(XL), lakukan proses tersebut sebanyak 16 kali, pada proses ke-16 lakukan penukaran XL dan XR. Dan pada P17, dan P18 dilakukan penyatuan XL dan XR yang telah ditukar tersebut menjadi *Chipertext* (*plaintext* yang telah terenkripsi)..

Untuk dekripsi sama persis dengan enkripsi, kecuali array P (P1,P2,……,P18) digunakan dengan urutan terbalik (Sitinjak, 2010).

4) Setelah *plaintext* terenkripsi, proses selanjutnya adalah dekripsi. Dekripsi adalah proses mengembalikan *chipertext* menjadi *plainteks* yang prosesnya sebagai berikut :
Chipertext diambil sebesar 64 bit. Dengan membalikkan 18 *subkey* untuk medekripsi. Proses dekripsi sama dengan proses enkripsi hanya saja masukkan awal nya saja yang berbeda. Pada proses dekripsi P18 diproses terlebih dahulu. P18 xor XL kemudian gunakan fungsi F pada XL lalu xor dengan XR, F(XL) xor XR lalu ditukar tempat, dan seterusnya. Dan diakhiri pada P1 dan P2 tidak dilakukan penukaran melainkan proses penyatuan untuk mendapatkan *plaintext* dari *chipertext* yang telah ter dekripsi adalah memeriksa apakah *plaintext* yang dihasilkan terjadi penambahan bit (*padding*) atau tidak pada saat enkripsi. Jika terjadi penambahan bit, maka bit yang sudah ditambahkan tersebut harus dilepas. Sistem akan menghitung penambahan bit yang terjadi sebelum enkripsi dan setelah *plaintext* terbentuk lalu sistem akan

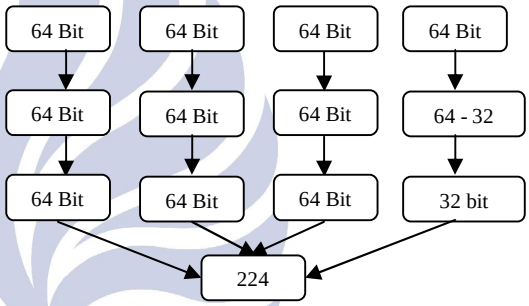
Setelah menghasilkan *plaintext*, proses selanjutnya mengurangi bit yang sudah ditambahkan itu. Setelah dilepas penambahan bit tersebut, maka blok-blok bit tersebut disatukan dan menghasilkan *plaintext*

5) Proses yang terjadi pada fungsi feistel (F( )) merupakan proses s-box. Pada proses ini terjadi penelusuran tabel. Dari Gambar 6 masukkan yang sebesar 32 bit (diambil dari XL) tersebut dibagi menjadi 4 bagian sebesar 8 bit. 8 bit ini akan digunakan untuk indeks, dimana indeks tersebut sebagai penunjuk pada tabel. Jadi 8 bit tersebut sebagai penunjuk digit heksadesimal pada tabel pi



Gambar 4. Alur Enkripsi

.



Gambar 5. Penyatuan Bit

6) (π), dengan kode heksadesimal *s-box*
Dalam proses *s-box* terjadi proses perhitungan bit. Dimana hasil penelusuran tabel *s-box* yang pertama akan ditambahkan dengan hasil *s-box* yang kedua, kemudian hasil penjumlahan di XOR dengan hasil *s-box* ketiga, hasilnya kemudian ditambahkan dengan *s-box* yang keempat maka akan menghasilkan output sebesar 32 bit.

HASIL DAN PEMBAHASAN

Aplikasi enkripsi *database* MySQL ini akan terbagi menjadi dua *user*, *user* biasa dan administrator. Seorang *user* biasa hanya dapat mengakses satu *database* yang menjadi miliknya yang telah didaftarkan. Administrator memiliki hak penuh untuk mengakses seluruh *database* yang terdapat dalam MySQL.

Aplikasi ini terdiri dari beberapa bagian penting yaitu login user, backup *database*, *restore database*, enkripsi *database*. *Database* yang terenkripsi meliputi *database* itu sendiri, tabel, dan *record* yang terdapat dalam *database* tersebut. Jadi aplikasi ini akan memuat seluruh nama *database* yang terdapat di MySQL lalu mengenkripsi nya dengan algoritma *blowfish*. Nama

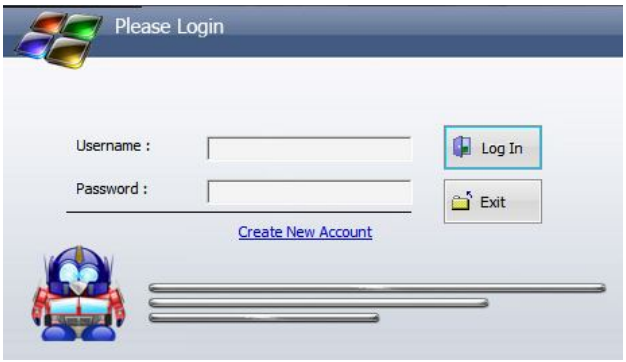
database yang sudah terkumpul lalu akan dihitung dan diterjemahkan menjadi byte. 1 byte sama dengan 8 bit.

Implementasi Program

Dalam aplikasi enkripsi *database* MySQL dengan algoritma *Blowfish* ini, data yang dibutuhkan adalah data yang terdapat di server MySQL, data tersebut berupa daftar *database*

1) Halaman *form login*

Pada halaman *form login* ini, digunakan untuk login sesuai dengan jenis *user*

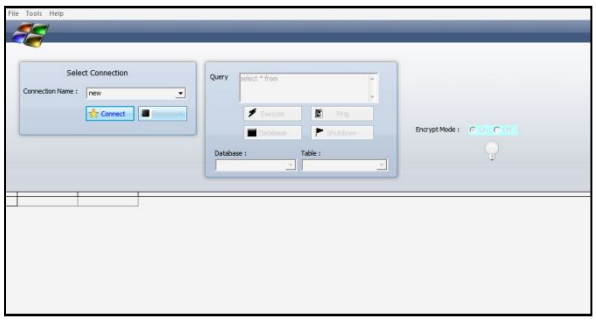


Gambar 6. *Form Login*

Pengguna perlu mengisikan *username* dan *password* agar dapat mengakses ke halaman selanjutnya.

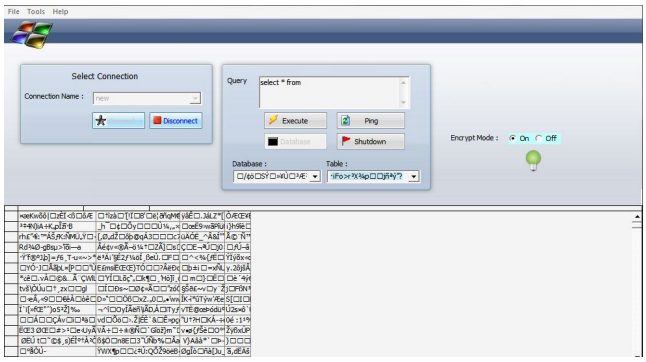
2) Halaman Utama

Pada Gambar 7 *form* utama ini tersedia menu-menu yang dapat digunakan untuk mengakses *database* dari MySQL yang langsung dalam keadaan terenkripsi, sehingga itu akan menjadi keamanan kedua setelah *login*. Setelah *form* utama muncul, maka hal pertama yang dilakukan adalah memilih koneksi pada kolom *connection name*, setelah ditekan tombol konek maka secara otomatis aplikasi akan tersambung ke *server* MySQL dan mengumpulkan semua *database* yang ada pada MySQL ke kolom *database* yang terdapat dalam aplikasi.



Gambar 7. *Form* Utama (administrator)

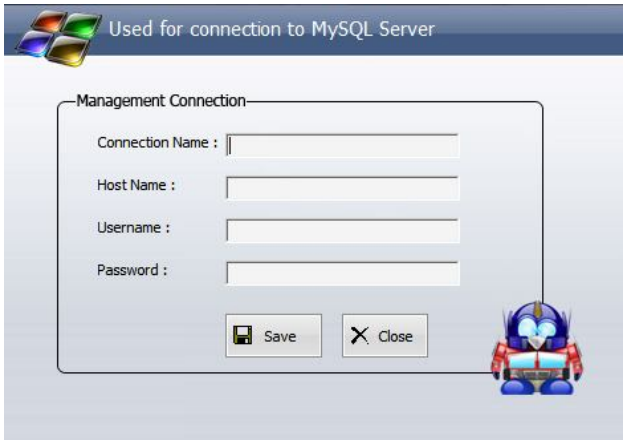
Setelah terkoneksi maka tampilan form nya seperti Gambar 8 di bawah ini.



Gambar 8. Form Utama setelah terenkripsi (administrator)

3) Form connection Manager

Form ini berfungsi untuk menambahkan data untuk koneksi ke MySQL, sehingga user tidak perlu terus mengetikkan nama *host*, *username*, dan *password*.

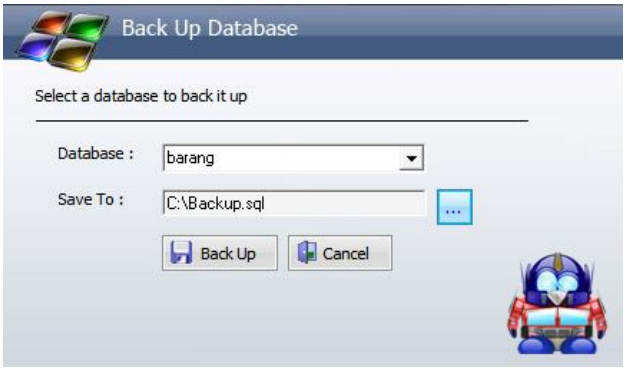


Gambar 9. *Form Connection Manager*

Setelah user mengisi form seperti Gambar 24 di atas maka data yang diisikan tersebut akan tersimpan dalam *database* yang kemudian akan digunakan untuk koneksi ke MySQL.

4) Form Backup Database

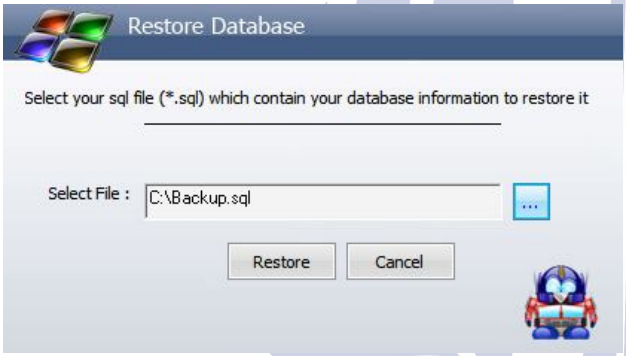
Database yang terdapat dalam MySQL akan tampil semua dalam aplikasi, setelah tampil maka *database* tersebut dapat dibuat salinan (*Backup*) dengan menggunakan fasilitas yang terdapat pada form ini. Dengan memilih nama *database* dari kolom *drop down database*, lalu memilih tempat dimana salinan tersebut akan disimpan maka salinan dari *database* tersebut akan tersimpan dengan ekstensi .sql.



Gambar 10. Form Backup Database

5) Form Restore Database

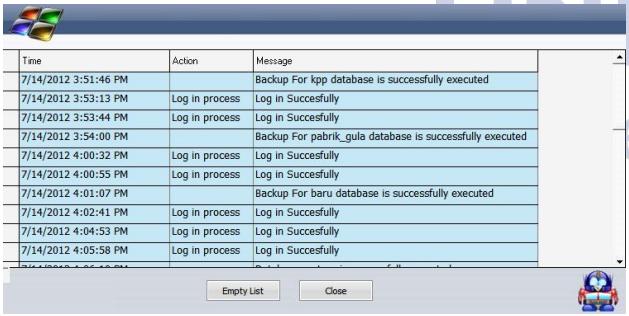
Database yang telah dibuat salinan dengan ekstensi .sql, maka dapat pula dikembalikan (*restore*), sehingga suatu saat *database* tersebut hilang maka dapat kita kembalikan dengan salinan yang telah dibuat.



Gambar 11. Form Restore Database

6) Form History

Semua kegiatan yang dilakukan dengan aplikasi ini akan dicatat dalam *history* lalu menampilkannya pada *form history* ini.



Gambar 12. Form History

4. PENUTUP

Simpulan

Simpulan yang dapat diambil dari pembuatan aplikasi *enkripsi database MySQL dengan algoritma Blowfish* adalah:

Salah satu cara yang dapat dilakukan untuk meningkatkan pengamanan data yaitu dengan menggunakan algoritma *Blowfish*. Algoritma *Blowfish* ini dapat digunakan untuk mengamankan data dalam *database* yang meliputi *database*, tabel, dan *record*. Dalam penelitian ini *database* yang akan diamankan datanya adalah MySQL.

Algoritma *Blowfish* dipilih dalam penelitian ini karena algoritma tersebut mampu bekerja pada komputer dengan spesifikasi minim, cepat. Dan mudah dimengerti.

Setelah dilakukan penelitian ini dapat diketahui bahwa enkripsi suatu *database* dapat dilakukan dengan menggunakan algoritma *Blowfish* dengan visual basic 6.0 sebagai bahasa pemrogramannya.

Saran

Untuk pembuatan rancang bangun aplikasi enkripsi *database* MySQL dengan algoritma *blowfish* diperlukan kelompok yang terdiri dari analis dan programmer. Analis untuk menganalisa alur dari algoritma blowfish. Programmer yang dibutuhkan *programmer level* menengah sampai profesional yang bisa membuat aplikasi berbasis *desktop*

DAFTAR PUSTAKA

Finch, S. 1995. *The Miraculous Bailey-Bonvein- Ploufe Pi'Algorithm. Mathsoft Engeering & Education, Inc.* (http://uauillac.inria.fr/aleo/bsolve/constant~~i~plouffe/ulouffe.html).

Sutanto, Candra Alim., 2009, Penggunaan Algoritma Blowfish Dalam Kriptografi, Program Studi Teknik Informatika Institut Teknologi Bandung.

Sitinjak, suriski, et al., 2010, Aplikasi Kriptografi File Menggunakan Algoritma Blowfish, Jurusan Teknik Informatika UPN "Veteran" Yogyakarta.

Schneier, Bruce, 1996, Applied Cryptography, Second Edition, John Wiley & Son, New York.