ALGORITMA ENKRIPSI *ONE TIME PAD* UNTUK SISTEM PENGAMANAN *ACCESS DATABASE SERVER* MENGGUNAKAN BAHASA PEMPROGRAMAN VISUAL BASIC

SKRIPSI

Diajukan sebagai Salah Satu Syarat
untuk Memperoleh Gelar Sarjana Sains
Jurusan Matematika

oleh

Amelia Duwi Astutik

4150403009

JURUSAN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS NEGERI SEMARANG

2007

MOTTO DAN PERSEMBAHAN

Motto :

- Jenius adalah 1 % inspirasi dan 99 % keringat. Tidak ada yang dapat menggantikan kerja keras. Keberuntungan adalah sesuatu yang terjadi ketika kesempatan bertemu dengan kesiapan. - Thomas A. Edison

- Do all the goods you can, All the best you can, In all times you can, In all places you can, For all the creatures you can...

Persembahan :

Dengan mengucap syukur kepada Allah, kupersembahkan skripsi ini untuk:

- Ibu, Bapak, Kakak-Adikku, terimakasih tak terhingga atas apa yang telah diberikan,
- Almamaterku..

KATA PENGANTAR

Alhamdulillah, segala puji bagi Allah Rabb Penguasa Alam, penulis panjatkan atas kekuatan lahir dan batin yang dilimpahkan, sehingga penulis memiliki kemampuan untuk menyelesaikan skripsi yang berjudul: "Algoritma Enkripsi *One Time Pad* Untuk Sistem Pengamanan *Access Database Server* Menggunakan Bahasa Pemprograman Visual Basic".

Adapun tujuan penyusunan skripsi ini adalah dalam rangka menyelesaikan Studi Strata 1 (S1) untuk mencapai gelar Sarjana Sains pada Program Studi Matematika, Jurusan Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Negeri Semarang.

Pada kesempatan ini penulis mengucapkan terima kasih kepada:

1. Prof. Dr. H. Sudijono Sastroatmodjo, M.Si., Rektor Universitas Negeri Semarang, yang telah memberi kesempatan kepada penulis untuk menimba ilmu di Universitas Negeri Semarang.
2. Drs. Kasmadi Imam S., M.S, Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Negeri Semarang, yang memberi ijin penelitian dan kemudahan dalam penyusunan skripsi.
3. Drs. Supriyono, M.Si., Ketua Jurusan Matematika Universitas Negeri Semarang, yang telah membantu kelancaran administrasi penyusunan skripsi.
4. Endang Sugiharti, S.Si, M.Komp., Dosen Pembimbing Utama yang telah berkenan meluangkan waktu untuk memberikan bimbingan dan arahan kepada penulis.

5. Drs. Khaerun, M. Si., Pembimbing Pendamping yang telah banyak membantu dan dengan sabarnya memberikan bimbingan pada penulis.
6. Semua pihak yang telah memberikan bantuan hingga terselesaikannya penulisan skripsi ini.

Akhirnya penulis berharap semoga Allah SWT memberikan balasan yang lebih baik atas keikhlasan semua pihak yang tidak dapat disebutkan satu-persatu, yang telah membantu dalam penyusunan skripsi ini. Semoga skripsi ini dapat bermanfaat.

Semarang, Agustus 2007

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

ABSTRAK

Amelia Duwi Astutik. 2007. **Algoritma Enkripsi *One Time Pad* Untuk Sistem Pengamanan *Access Database Server* Menggunakan Bahasa Pemprograman Visual Basic.** Program Studi Matematika. Jurusan Matematika. Fakultas MIPA. Universitas Negeri Semarang.

Seperti kita ketahui bahwa *access database server* dengan menggunakan bahasa pemrograman Visual Basic saat ini banyak digunakan dalam kehidupan sehari-hari. Akan tetapi selama ini penggunaan *database access* belum pernah diberi pengamanan data. Ini berarti bahwa setiap orang dapat dengan mudahnya melihat informasi yang ada di dalamnya. Untuk itu agar datanya tidak diketahui oleh pihak-pihak yang tidak berkepentingan, kita harus berusaha menyiasati cara mengamankan informasi yang dimiliki. Perlindungan terhadap kerahasiaan data sekarang ini semakin meningkat, salah satu caranya dengan dengan penyandian data atau enkripsi.

Pada penulisan skripsi ini, penyandian data dibuat dengan menggunakan algoritma *One Time* Pad (OTP) yang terkenal sederhana dan '*unbreakable'*, untuk sistem pengamanan *access database server* pada bahasa pemprograman Visual Basic, khususnya untuk *database* MIPA *Connect* Universitas Negeri Semarang.

Prinsip enkripsi pada algoritma OTP ini adalah dengan mengkombinasikan masing-masing karakter pada plainteks dengan satu karakter pada kunci. Oleh karena itu, panjang kunci setidaknya harus sama dengan panjang plainteks. Untuk membangkitkan aliran kunci, dilakukan proses *linear feedback shift register* (LFSR) atau yang biasa disebut register geser dengan umpan balik linier. Bit-bit keluaran proses LFSR digunakan sebagai kunci baru proses enkripsi dekripsi. Fungsi untuk mengenkrip pada algoritma OTP hanyalah meng-XOR-kan plainteks dengan kunci yang telah disiapkan untuk menghasilkan cipherteks. Sedangkan fungsi untuk mendekrip tinggal meng-XOR-kan cipherteks dengan kunci yang sudah disepakati.

Kata kunci : Enkripsi, *One Time Pad, Database,* Visual Basic.

BAB I

PENDAHULUAN

A. Latar Belakang

Sistem basis data menempati posisi penting dalam masyarakat berbasis informasi dan pengetahuan. Setiap sistem besar di mana kita sehari-hari berinteraksi atau bahkan bergantung, semacam sistem perbankan memiliki basis data sebagai intinya. Dapat dikatakan bahwa sistem basis data telah merupakan pokok penunjang perkembangan teknologi informasi, serta merupakan kerangka utama beroperasinya sistem berbasis komputer. Sangat sulit dipisahkan operasi sistem berbasis komputer dan sistem basis data. Dapat dikatakan keberadaan sistem berbasis komputer menandakan keberadaan sistem basis data. Di masa datang, kebergantungan pada kebenaran dan efisiensi sistem basis data akan semakin meningkat.

Salah satu hal yang penting dalam penggunaan basis data adalah menjamin kerahasiaan data. Seperti yang dikatakan oleh Munir bahwa masalah keamanan (*security*) pada komputer menjadi isu penting pada era teknologi informasi sekarang ini. Banyak kejahatan *cyber* yang pernah kita dengar dari media massa. Pelaku kejahatan memanfaatkan celah keamanan yang ada untuk dimasuki dan melakukan manipulasi (Munir, 2006 : iii).

Informasi yang merupakan hasil pengolahan dari data, mempunyai nilai yang berbeda bagi setiap orang. Seringkali sebuah informasi menjadi

sangat berharga, dan tidak semua orang diperkenankan untuk mengetahuinya.

Kerahasiaan atau lebih dikenal dengan istilah *"confidentiality"*, dapat diartikan sebagai perlindungan terhadap data dalam sistem informasi perusahaan, sehingga tidak dapat diakses oleh orang yang tidak berhak. Banyak yang beranggapan bahwa tipe perlindungan seperti ini hanya penting untuk kalangan militer dan pemerintahan, dimana mereka perlu merahasiakan rencana dan data penting. Akan tetapi kerahasian juga sangat penting bagi kalangan bisnis yang perlu melindungi rahasia dagang mereka dari kompetitor, atau untuk mencegah akses terhadap data-data yang dianggap sensitif oleh orang-orang yang tidak berhak di dalam. Alasan lain diperlukannya pengamanan basis data adalah berlakunya Undang-Undang yang mengatur perihal kerahasiaan data pelanggan yang biasa disimpan pada basis data perusahaan. Salah satu contohnya adalah peraturan HIPAA (*Helath Insurance Portability and Accountability Act*) yang menstandarkan keamanan data medis dan data individual lainnya.

Karena alasan tersebut dan isu seputar privasi yang semakin diperhatikan akhir-akhir ini, telah memaksa badan pemerintahan dan perusahaan swasta sekalipun untuk menjaga kerahasiaan informasi secara lebih baik lagi, demi melindungi data-data pribadi yang disimpan dalam sistem informasi badan pemerintahaan atau perusahaan swasta tersebut.

Secara garis besar, terdapat dua tujuan dari pengamanan basis data yaitu melindungi kerahasiaan data dan menjamin integritas data. Kerahasian

harus terdefinisi dengan baik, dan prosedur untuk menjaga kerahasiaan informasi harus diterapkan secara berhati-hati, khususnya untuk komputer yang bersifat *standalone* atau tidak terhubung ke jaringan. Aspek penting dari kerahasiaan adalah pengidentifikasian atau otentikasi terhadap *user*. Identifikasi positif dari setiap *user* sangat penting untuk memastikan efektivitas dari kebijakan yang menentukan siapa saja yang berhak untuk mengakses data tertentu.

Salah satu cara untuk mengamankan data pada basis data adalah dengan menggunakan teknik kriptografi yang diterapkan pada data tersebut. Pengamanan menggunakan kriptografi memerlukan banyak pertimbangan dan strategi. Perlindungan terhadap kerahasiaan data dengan menggunakan kriptografi sekarang ini semakin meningkat, salah satu caranya dengan penyandian data atau enkripsi. Metode enkripsi adalah suatu metode manipulasi data dengan mengkodekan atau meyembunyikan data aslinya, sehingga data yang bisa dibaca dan dimengerti oleh siapapun (*plaintext/cleartext*) menjadi data yang tidak bisa dibaca dan dimengerti dengan jelas. Pengkodean data menjadi data enkripsi dapat menghindari pemanipulasian dan perusakan data melalui jaringan dalam upaya melindungi basis data.

Seperti kita ketahui bahwa *access database server* dengan menggunakan bahasa pemprograman Visual Basic saat ini banyak digunakan dalam kehidupan sehari-hari. Akan tetapi selama ini penggunaan *database access* belum pernah diberi pengamanan data. Ini berarti bahwa

setiap orang dapat dengan mudahnya melihat informasi yang ada di dalamnya. Untuk itu agar datanya tidak diketahui oleh pihak-pihak yang tidak berkepentingan, kita harus berusaha menyiasati cara mengamankan informasi yang dimiliki. Seperti yang telah dipaparkan di atas, salah satu cara mengamankan basis data adalah dengan penyandian data atau enkripsi.

Ada beberapa algoritma enkripsi yang biasa digunakan seperti DES, *Triple* DES, *Blowfish*, IDEA dan sebagainya. Algoritma-algoritma tersebut begitu rumit dan sulit dimengerti dengan dalih 'faktor keamanan', katanya semakin sulit suatu algoritma dimengerti, maka semakin aman. Namun bagi para pengguna mereka tidak memikirkan seberapa sulit algoritma dan aplikasinya, yang mereka inginkan adalah menjaga kerahasiaan data.

Ada 2 syarat untuk mengimplementasikan suatu sistem enkripsi yang aman (http://www.topsecretcripto.com). Pertama, kunci harus dipilih secara acak. Kedua, panjang kunci harus sama dengan panjang plainteks yang akan dienkripsikan. Jika kedua syarat dipenuhi, tidak masalah seberapa kompleks algoritma enkripsinya. Bahkan semakin sederhana semakin baik, karena semakin sederhana suatu algoritma, maka akan semakin sedikit proses komputasinya dan semakin sedikit waktu yang dibutuhkan untuk mengeksekusinya. Kesederhanaan itulah yang ditawarkan oleh algoritma *One Time Pad* (OTP), algoritma kriptografi yang secara teori dan praktik aman dari tangan-tangan penyadap, dan dikenal dengan sebutan '*unbreakable chiper*' (Munir, 2006 : 93).

Skema enkripsi yang akan dibangun di sini menerapkan teknik pada kriptografi modern, yang menganut kerahasiaan pada kunci (*key*). Pada penulisan skripsi ini, penyandian data dibuat dengan menggunakan algoritma OTP untuk sistem pengamanan *access database server* pada bahasa pemprograman Visual Basic.

B. Permasalahan

Berdasarkan latar belakang yang telah dijelaskan di atas, masalah yang akan diangkat dalam penulisan skripsi ini adalah bagaimana algoritma OTP untuk sistem pengamanan *access database server* pada bahasa pemprograman Visual Basic?

C. Pembatasan Masalah

Pada penelitian kali ini penulis membatasi ruang lingkup penulisan hanya pada seputar enkripsi untuk sistem pengamanan *access database server* pada *database* MIPA *Connect* Universitas Negeri Semarang.

D. Tujuan Penelitian

Dengan melihat permasalahan yang ada maka penelitian ini dilakukan dengan tujuan agar dapat mengetahui algoritma OTP untuk sistem pengamanan *access database server* pada bahasa pemprograman Visual Basic

E. Manfaat Penelitian

Penelitian ini diharapkan dapat memberikan manfaat sebagai berikut.

1. Bagi mahasiswa, menambah pengetahuan dan pemahaman mengenai terapan salah satu mata kuliah yang telah ditempuh dan menambah pengetahuan tentang kriptografi
2. Bagi pembaca, sebagai sumbangan pemikiran dan informasi khususnya bagi yang ingin melakukan penelitian sejenis.

F. Sistematika Skripsi

Penulisan skripsi ini secara garis besar dibagi menjadi 3 bagian, yaitu bagian awal, bagian isi dan bagian akhir.

Bagian awal memuat halaman judul, abstrak, halaman pengesahan, halaman motto dan persembahan, kata pengantar, daftar isi, daftar tabel, daftar gambar dan daftar lampiran.

Bagian isi terdiri dari 5 bab. Adapun 5 bab bagian isi tersebut adalah sebagai berikut.

Bab I. Pendahuluan

Bab ini berisi tentang latar belakang masalah, permasalahan, tujuan penelitian, manfaat penelitian dan sistematika skripsi.

Bab II. Landasan Teori

Landasan teori merupakan teori yang mendasari pemecahan dari permasalahan yang disajikan. Pada bab ini dibagi menjadi

beberapa sub bab yaitu Basis Data, Kriptografi, *One Time Pad* (OTP), Sistem Bilangan dan Operator Logika *Exclusive OR* (XOR).

Bab III. Metode Penelitian

Bab ini berisi tentang tahap-tahap penulisan skripsi yaitu penemuan masalah, perumusan masalah, studi pustaka, pemecahan masalah dan penarikan simpulan.

Bab IV. Hasil Penelitian dan Pembahasan

Bab ini berisi tentang hasil penelitian dan pembahasan mengenai algoritma OTP untuk sistem pengamanan *access database server* pada bahasa pemprograman Visual Basic serta bahasa pemprograman Visual Basic untuk algoritma OTP.

Bab V. Penutup

Bab ini berisi simpulan dan saran yang berkaitan dengan penelitian.

Pada bagian akhir skripsi, berisi daftar pustaka dan lampiran-lampiran yang mendukung isi skripsi.

BAB II

LANDASAN TEORI

A. Basis Data

1. Data dan Informasi

Data adalah fakta mengenai objek, orang dan lain-lain. Data dinyatakan dengan nilai angka, deretan karakter atau simbol (Kadir, 1999 : 8). Sejumlah penulis menggunakan data untuk menyatakan nilai-nilai yang secara aktual terkandung dalam basis data.

Informasi adalah hasil analisis dan sintesis terhadap data, dengan kata lain informasi dapat dikatakan sebagai data yang telah diorganisasikan ke dalam bentuk yang sesuai dengan kebutuhan seseorang, entah itu manajer, staf maupun orang lain dalam suatu perusahaan.

Dalam era informasi, informasi menjadi sumber penting untuk melakukan pengambilan keputusan. Informasi dapat mengurangi ketidakpastian dan mempermudah pengambilan keputusan.

2. Pengertian Basis Data

Basis data (*database*) adalah kumpulan informasi yang disimpan di dalam komputer secara sistematik sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi dari basis data tersebut. Perangkat lunak yang digunakan untuk mengelola dan memanggil kueri (*query*) basis data disebut sistem

manajemen basis data (*database management system*, DBMS). Sistem basis data dipelajari dalam ilmu informasi (Wikipedia Indonesia).

Istilah basis data banyak menimbulkan interpretasi yang berbeda. Chou dalam Kadir (1999 : 9) mendefinisikan basis data sebagai kumpulan informasi bermanfaat yang diorganisasikan ke dalam tata cara yang khusus.

Menurut Fabbri dan Schwab, basis data adalah sistem berkas terpadu yang dirancang terutama untuk meminimalkan pengulangan data. Menurut Date, sistem basis data pada dasarnya adalah sistem komputerisasi yang tujuan utamanya adalah memelihara dan membuat informasi tersebut tersedia saat dibutuhkan (Kadir, 1999 : 9).

Dari beberapa pengertian tersebut dapat disimpulkan bahwa basis data adalah sistem berkas terpadu yang dirancang untuk memelihara informasi, meminimalkan pengulangan data dan dapat membuat informasi tersebut dapat tersedia setiap saat dibutuhkan.

3. Sistem Basis Data

Suatu basis data dapat dibuat dan dipelihara dengan cara manual atau dengan menggunakan komputer. Suatu basis data yang berbasis komputer dibuat dan dipelihara oleh sekumpulan program aplikasi yang ditulis secara khusus untuk menyelesaikan masalah tertentu atau dengan menggunakan suatu Sistem Manajemen Basis Data (*Database Management System*).

Sistem Manajemen Basis Data merupakan suatu perangkat lunak yang terdiri atas sekumpulan program untuk mengelola dan memelihara data di dalam suatu struktur yang digunakan oleh banyak aplikasi, bebas terhadap media penyimpanan dan metode akses (Riyanto, 2004 : 5).

Sistem basis data terdiri atas beberapa komponen, yaitu data, perangkat keras, perangkat lunak dan pengguna yang meliputi pemprogram aplikasi, pengguna akhir dan administrator basis data.

4. Integritas dan Keamanan

a. Integritas Data

Integritas konstrain memberikan jaminan bahwa perubahan yang dilakukan terhadap basis data tidak menghasilkan hilangnya konsistensi data. Integritas konstrain juga mencegah terjadinya suatu kerusakan basis data akibat adanya kejadian yang bersifat asidental, seperti terjadinya *crashed* pada saat proses transaksi ataupun kesalahan *logic* yang merusak asumsi yang berakibat mengganggu lingkungan basis data.

Integritas data adalah jaminan konsistensi data terhadap semua status konstrain yang diberlakukan terhadap data tersebut, sehingga memberikan jaminan keabsahan data itu sendiri (Riyanto, 2004 : 92). Adapun beberapa integritas data meliputi integritas entitas, integritas referensial, konstrain domain dan *enterprise constraint*.

b. Keamanan Basis Data

Keamanan basis data adalah pemberian perlindungan basis data terhadap ancaman dan gangguan baik yang bersifat teknis maupun administrasi (Riyanto, 2004 : 94).

Gangguan terhadap basis data sangat bervariasi, dimana dapat meliputi *hardware*, *software*, manusia dan data. Secara keseluruhan, gangguan baik fisik maupun nonfisik meliputi pencurian, hilangnya kerahasiaan, kehilangan integritas dan kehilangan kemampuan.

Untuk memberikan perlindungan keamanan basis data, diantaranya dapat dilakukan dengan pemberian otoritas terhadap pengguna dalam melakukan akses objek yang meliputi tabel basis data, *view*, aplikasi, prosedur atau objek lainnya dalam sistem. Adapun aspek-aspek keamanan data adalah sebagai berikut.

(1) Kerahasiaan (*confidentiality*), adalah layanan yang ditujukan untuk menjaga agar pesan tidak dapat dibaca oleh pihak-pihak yang tidak berhak.

(2) Integritas data (*data integrity*), adalah layanan yang menjamin bahwa pesan masih asli/ utuh atau belum pernah dimanipulasi selama pengiriman.

(3) Otentikasi (*authentication*), adalah layanan yang berhubungan dengan identifikasi, baik mengidentifikasi kebenaran pihak-pihak yang berkomunikasi (*user authentication*) maupun

mengidentifikasi kebenaran sumber pesan (*data origin authentication*).

(4) Nirpenyangkalan (*non-repudation*), adalah layanan untuk mencegah entitas yang berkomunikasi melakukan penyangkalan, yaitu pengirim pesan menyangkal melakukan pengiriman atau penerima pesan menyangkal telah menerima pesan.

5. *Database access*

Relational Database Management System (RDBMS) yang terkenal di Windows adalah *database access* dan SQL Server. *Database access* adalah RDBMS yang sangat sederhana dan mudah digunakan, setiap *database* disimpan dalam satu file tersendiri yang memiliki ekstensi .mdb

Gambar 2.1. Contoh sebuah file *Database* .mdb Access

Karena model penyimpanan *database access* adalah berbasis *file*, maka kemampuan penyimpanan dari *database access* terbatas pada besar 2 GB. Untuk aplikasi-aplikasi yang sederhana hanya memerlukan sedikit data hal ini tidak bermasalah, bahkan dalam kenyataan jarang sekali aplikasi sederhana memerlukan data lebih dari 2 GB. Apabila data yang dibutuhkan memang sangat besar, maka *database access* pun masih dapat digunakan dengan cara memecah *database*, yaitu dengan

jalan membuat banyak file *database* untuk menampung datanya, sehingga data yang banyak dapat disimpan dengan merata ke dalam *database-database* yang ada.

Yang dimaksud *database access* bukanlah Microsoft Access yang merupakan bagian dari produk Microsoft Office di Windows, akan tetapi hanya file .mdb yang dapat dibuat secara interaktif melalui Microsoft Access.

Selain dapat dibuat dengan aplikasi Microsoft Access, *database access* juga dapat dibuat menggunakan fitur Visual Data Manager di Visual Basic 6.0, akan tetapi *database access* yang dibuat hanya memiliki versi Access97.

B. Kriptografi

1. Definisi Kriptografi

Kriptografi (*cryptography*) berasal dari bahasa Yunani: "*cryptos*" artinya "*secret*" (rahasia), sedangkan "*gráphein*" artinya "*writing*" (tulisan). Jadi kriptografi berarti "*secret writing*" (tulisan rahasia) (Munir, 2006 : 2).

Dalam buku-buku yang lama (sebelum tahun 1980-an) menyatakan bahwa kriptografi adalah ilmu dan seni untuk menjaga kerahasiaan pesan dengan cara menyandikannya ke dalam bentuk yang tidak dapat dimengerti lagi maknanya.

Menurut Menezes, kriptografi adalah ilmu yang mempelajari teknik-teknik matematika yang berhubungan dengan aspek keamanan informasi seperti kerahasiaan, integritas data serta otentikasi. Menurut Schneier, kriptografi adalah ilmu sekaligus seni untuk menjaga keamanan pesan (*message*) (Munir, 2006 : 2).

2. Terminologi dalam Kriptografi

a. Pesan, plainteks dan chiperteks

Pesan adalah data atau informasi yang dapat dibaca dan dimengerti maknanya. Nama lain untuk pesan adalah plainteks (*plaintext*) atau teks-jelas (*cleartext*). Pesan dapat berupa data atau informasi yang dikirim (melalui kurir, saluran komunikasi data, dsb) atau yang disimpan di dalam media perekaman (kertas, *storage*, dsb).

Agar pesan tidak dapat dimengerti maknanya oleh pihak lain, maka pesan disandikan ke bentuk lain. Bentuk pesan yang tersandi disebut cipherteks (*ciphertext*) atau kriptogram (*cryptogram*). Cipherteks harus dapat ditransformasi kembali menjadi plainteks. Sebagai contoh plainteks, uang disimpan di balik buku X, maka cipherteksnya adalah j&kloP#d$gkh*7h^”tn%6^klp..t@.

b. Pengirim dan penerima

Komunikasi data melibatkan pertukaran pesan antara dua entitas. Pengirim (*sender*) adalah entitas yang mengirim pesan kepada entitas lainnya. Penerima (*receiver*) adalah entitas yang

menerima pesan. Entitas dapat berupa orang, mesin, kartu kredit dan sebagainya.

c. Enkripsi dan dekripsi

Proses menyandikan plainteks menjadi chiperteks disebut enkripsi (*encryption*) atau *enchipering* (standard nama menurut ISO 7498-2). Proses mengembalikan cipherteks menjadi plainteksnya disebut dekripsi (*decryption*) atau *deciphering* (standard nama menurut ISO 7498-2) (Munir, 2006 : 4).

Enkripsi adalah sebuah proses yang melakukan perubahan sebuah kode yang bisa dimengerti menjadi sebuah kode yang tidak bisa dimengerti (tidak terbaca). Enkripsi dapat diartikan sebagai kode atau *chipper* (Wahana Komputer, 2003 : 43).

Di bidang kriptografi, enkripsi ialah proses mengamankan suatu informasi dengan membuat informasi tersebut tidak dapat dibaca tanpa bantuan pengetahuan khusus. Dikarenakan enkripsi telah digunakan untuk mengamankan komunikasi di berbagai negara, hanya organisasi-organisasi tertentu dan individu yang memiliki kepentingan yang sangat mendesak akan kerahasiaan yang menggunakan enkripsi. Di pertengahan tahun 1970-an, enkripsi kuat dimanfaatkan untuk pengamanan oleh sekretariat agen pemerintah Amerika Serikat pada domain publik, dan saat ini enkripsi telah digunakan pada sistem secara luas, seperti Internet *e-commerce*, jaringan telepon bergerak dan ATM pada bank.

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.

d. *Chiper* dan kunci

Algoritma kriptografi disebut juga *chiper* yaitu aturan untuk *enchipering* dan *dechipering*, atau fungsi matematika yang digunakan untuk enkripsi dan dekripsi.

Kriptografi modern mengatasi masalah keamanan algoritma kriptografi dengan penggunaan kunci. Kunci (*key*) adalah parameter yang digunakan untuk transformasi *enchipering* dan *dechipering*. Kunci biasanya berupa *string* atau deretan bilangan.

e. Sistem kriptografi

Kriptografi membentuk sebuah sistem yang dinamakan sistem kriptografi. Menurut Schneier dalam Munir (2006 : 7), sistem kriptografi (*cryptosystem*) adalah kumpulan yang terdiri atas algoritma kriptografi, semua plainteks dan chiperteks yang mungkin serta kunci.

f. Penyadap

Penyadap (*eavesdropper*) adalah orang yang mencoba menangkap pesan selama ditransmisikan. Tujuan penyadap adalah

untuk mendapatkan informasi sebanyak-banyaknya mengenai sistem kriptografi yang digunakan untuk berkomunikasi dengan maksud untuk memecahkan chiperteks. Nama lain penyadap adalah *enemy*, *adversary*, *intruder*, *interceptor*, *bad guy*.

g. Kriptanalisis dan kriptografi

Kriptanalisis (*cryptanalysis*) adalah ilmu dan seni untuk memecahkan chiperteks menjadi plainteks tanpa mengetahui kunci yang diberikan. Pelakunya disebut kriptanalis. Kriptologi (*cryptology*) adalah studi mengenai kriptografi dan kriptanalisis.

3. Konsep Matematis Algoritma Kriptografi

Konsep matematis yang mendasari algoritma kriptografi adalah relasi antara dua buah himpunan yaitu himpunan yang berisi elemen-elemen plainteks dan himpunan yang berisi chiperteks. Enkripsi dan dekripsi merupakan fungsi yang memetakan elemen-elemen antara kedua himpunan tersebut.

Misalkan P menyatakan plainteks dan C menyatakan chiperteks, maka fungsi enkripsi E memetakan P ke C,

$$E(P) = C$$

Dan fungsi dekripsi D memetakan C ke P,

$$D(C) = P$$

Karena proses enkripsi kemudian dekripsi mengembalikan pesan ke pesan asal maka persamaan berikut harus benar,

$$D(E(P)) = P$$

Dengan menggunakan kunci K, maka fungsi enkripsi dan dekripsi menjadi,

$$E_K(P) = C$$

$$D_K(C) = P$$

dan kedua fungsi ini memenuhi:

$$D_K(E_K(P)) = P$$

Gambar berikut memperlihatkan skema enkripsi dan dekripsi dengan menggunakan kunci.

Gambar 2.2. Skema enkripsi dan dekripsi dengan kunci

4. Tujuan Kriptografi

Selain untuk menjaga kerahasiaan (*confidentiality*) pesan, kriptografi juga digunakan untuk menangani masalah keamanan yang mencakup tiga hal berikut.

a. Keabsahan pengirim (*user authentication*)

   Hal ini berkaitan dengan keaslian pengirim. Dengan kata lain, masalah ini dapat diungkapkan sebagai pertanyaan: “Apakah pesan yang diterima benar-benar berasal dari pengirim yang sesungguhnya?”

b. Keaslian pesan (*message authentication*)

Hal ini berkaitan dengan keutuhan pesan (*data integrity*). Dengan kata lain, masalah ini dapat diungkapkan sebagai pertanyaan: "Apakah pesan yang diterima tidak mengalami perubahan (modifikasi)?"

c. Anti-penyangkalan (*nonrepudiation*)

Pengirim tidak dapat menyangkal (berbohong) bahwa dialah yang mengirim pesan.

C. *One Time Pad* (OTP)

1. Sejarah OTP

OTP ditemukan pada tahun 1917 oleh G. Vernam dan Major Joseph Mauborgne. OTP sering disebut "*Vernam Cipher*". OTP merupakan algoritma yang relatif gampang untuk dipelajari dan sudah dinyatakan oleh para ahli kriptografi sebagai "*perfect encryption algorithm*".

2. Tinjauan Umum

Algoritma OTP merupakan algoritma berjenis *symmetric key* yang artinya bahwa kunci yang digunakan untuk melakukan enkripsi dan dekripsi merupakan kunci yang sama. Dalam proses enkripsi, algoritma ini menggunakan cara *stream cipher* dimana *cipher* berasal dari hasil XOR antara bit plainteks dan bit *key*.

Algoritma ini sering digunakan dalam proses enkripsi (termasuk pemrosesan transaksi online menggunakan kartu kredit) karena prosesnya yang relatif mudah.

Prinsip enkripsi pada algoritma ini adalah dengan mengkombinasikan masing-masing karakter pada plainteks dengan satu karakter pada kunci. Oleh karena itu, panjang kunci setidaknya harus sama dengan panjang plainteks. Secara teori, adalah hal yang tak mungkin untuk mendekripsi chiperteks tanpa mengetahui kuncinya. Sebab jika kunci yang digunakan salah, akan diperoleh hasil yang salah juga, atau bukan plainteks yang seharusnya. Kemudian setiap kuncinya hanya boleh digunakan untuk sekali pesan. Pengambilan kunci harus dilakukan secara acak supaya tidak dapat diterka lawan dan jumlah karakter kunci harus sebanyak jumlah karakter pesan.

Fungsi untuk mengenkrip hanyalah meng-XOR-kan plainteks dengan kunci yang telah disiapkan untuk menghasilkan cipherteks.

$$c = p \text{ XOR } k$$

Sedangkan fungsi untuk mendekrip tinggal meng-XOR-kan cipherteks dengan kunci yang sudah disepakati.

$$p = c \text{ XOR } k$$

D. Sistem Bilangan

1. Bilangan Biner

Sistem bilangan biner atau sistem bilangan basis dua adalah sebuah sistem penulisan angka dengan menggunakan dua simbol yaitu 0 dan 1. Sistem bilangan biner modern ditemukan oleh Gottfried Wilhelm Leibniz pada abad ke-17 (Wikipedia Indonesia).

Perbedaan mendasar dari metoda biner dan desimal adalah berkenaan dengan basis. Jika desimal berbasis 10 (X_{10}) berpangkatkan 10^x, maka untuk bilangan biner berbasiskan 2 (X_2) menggunakan perpangkatan 2^x.

Komputer memproses data atau program dari memori komputer berupa sejumlah bilangan biner yang menyatakan dalam keadaan hidup atau mati (*on or off*) dengan angka 1 dan 0. Sehingga semua yang diproses komputer hanya angka 0 dan 1, sehingga sistem biner (bilangan berdasar 2) sangatlah penting. Cara mengkonversi bilangan biner ke bilangan desimal adalah dengan mengalikan dua dengan pangkat N (suku ke-N).

2. Sistem Heksa Desimal

Heksadesimal atau sistem bilangan basis 16 adalah sebuah sistem bilangan berbasiskan 16. Simbol yang digunakan dari sistem ini adalah 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A(=10), B(=11), C(=12), D(=13), E(=14) dan F(=15) (Wikipedia Indonesia).

Menurut Abe Poetra (2003 : 13), bilangan heksadesimal atau sering disingkat menjadi heks ini adalah bilangan berbasis enam belas.

3. Aritmetika Modulo

Misalkan *a* adalah bilangan bulat dan *m* adalah bilangan bulat > 0. Operasi *a* mod *m* (dibaca "*a* modulo *m*") memberikan sisa jika *a* dibagi dengan *m*. bilangan *m* disebut modulus atau modulo, dan hasil aritmetika modulo *m* terletak di dalam himpunan {0, 1, 2, …, *m*-1}.

Adapun notasi dari modulo adalah sebagai berikut.

a mod m = r sedemikian sehingga a = mq + r, dengan $0 \leq r < m$.

Beberapa contoh operasi dengan operator modulo:

(i). 23 mod 5 = 3 (23 = 5 . 4 + 3)

(ii). 27 mod 3 = 0 (27 = 3 . 9 + 0)

(iii). 6 mod 8 = 6 (6 = 8 . 0 + 6)

(iv). 0 mod 12 = 0 (0 = 12 . 0 + 6)

E. Operator Logika *Exclusive OR* (XOR)

Operator merupakan simbol yang sering digunakan dalam ekspresi yang berguna untuk menghubungkan variabel. Banyak operator yang digunakan di dalam pemprograman komputer, ada operator penjumlahan (+), pengurangan (-), perkalian (*), pembagian (/), ada juga NOT, OR, dan XOR.

Berbeda dengan operator yang lain, operator XOR memiliki keistimewaan yaitu sebuah nilai yang dihasilkan dari operasi XOR akan

mengembalikan nilai awalnya bila di XOR dengan nilai yang sama. Operator XOR akan menghasilkan TRUE hanya apabila salah satu *operand*-nya bernilai TRUE.

```
TRUE    XOR  FALSE  = TRUE
FALSE   XOR  TRUE   = TRUE
TRUE    XOR  TRUE   = FALSE
FALSE   XOR  FALSE  = FALSE
```

Operator XOR juga melakukan perbandingan pada posisi bit dalam dua ekspresi numerik dan mengatur hubungan bit yang hasilnya sesuai tabel berikut.

Tabel 2.1. Hubungan bit operator XOR

Ekspresi 1	Ekspresi 2	Hasil
0	0	0
0	1	1
1	0	1
1	1	0

BAB III

METODE PENELITIAN

A. METODE PENGUMPULAN DATA

Penulis melakukan pengumpulan data pada warnet MIPA *Connect* Universitas Negeri Semarang. Data yang dikumpulkan berupa data-data mahasiswa *member* MIPA *Connect* Universitas Negeri Semarang serta data-data lain yang mendukung dalam pembuatan *database*.

B. TAHAPAN PENELITIAN

Langkah-langkah penelitian yang dilakukan adalah sebagai berikut.

1. Tahap pengumpulan bahan-bahan yang terkait dengan sistem pengamanan *access database server* menggunakan algoritma OTP. Sumber-sumber diperoleh dari buku literatur, artikel-artikel yang terkait, serta penelitian-penelitian yang mendukung penelitian ini.
2. Tahap penyusunan kembali semua bahan yang telah diperoleh dan disusun sesuai dengan prosedur penggunaan algoritma OTP dalam pengamanan *access database server*. Dalam tahap ini penulis melakukan penyusunan algoritma OTP untuk pengamanan *access database server* dengan menggunakan bahasa pemprograman Visual Basic.

C. METODE ANALISIS DATA

Tahapan analisis data penelitian yang terkumpul yang akan dilakukan penulis adalah sebagai berikut.

1. Reduksi data

   Dari data yang terkumpul, perlu pemuatan rangkuman data yang inti, yaitu data yang diperlukan sehingga tetap dalam data. Proses ini memerlukan pembuangan data yang tidak diperlukan dalam proses analisis selanjutnya.

2. Penyusunan data

   Rangkuman data yang diperoleh disusun berdasarkan indikator yang dipakai dalam penelitian ini, sehingga memudahkan untuk melaksanakan tahapan analisis berikutnya. Indikator yang digunakan di sini adalah pemilihan data-data sensitif yang memerlukan perlindungan data.

3. Pembuatan *database*

   Setelah penyusunan dilakukan maka tahapan selanjutnya adalah pembuatan *database*. Dari data-data yang telah diperoleh, disusun suatu *database* yang telah diberi pengamanan data khususnya pada data mahasiswa *member* MIPA *Connect* Universitas Negeri Semarang.

BAB IV

HASIL PENELITIAN DAN PEMBAHASAN

A. Struktur Kerja *One Time Pad*

Cipher yang tidak dapat dipecahkan dikatakan memiliki tingkat kerahasiaan yang sempurna (*perfect secrecy*). Satu-satunya algoritma kriptografi sempurna, aman dan tidak dapat dipecahkan adalah *One Time Pad* (Munir, 2006: 93).

Seperti yang telah diungkapkan sebelumnya, bahwa *One Time Pad* merupakan algoritma yang relatif gampang untuk dipelajari dan sudah dinyatakan oleh para ahli kriptografi sebagai "*perfect encryption algorithm*" atau sering disebut "*Vernam Cipher*". OTP ditemukan pada tahun 1917 oleh G. Vernam dan Major Joseph Mauborgne. OTP termasuk cipher aliran (*stream cipher*), yaitu cipher yang berasal dari hasil XOR antara setiap bit plainteks dengan setiap bit kuncinya.

One Time Pad (*pad* = kertas bloknot) adalah kertas yang berisi deretan karakter-karakter kunci yang berisi huruf-huruf yang tersusun acak. Satu pad hanya digunakan sekali (*one time*) saja untuk mengenkripsi pesan, setelah itu *pad* yang telah digunakan dihancurkan supaya tidak dipakai kembali untuk mengenkripsi pesan yang lain.

1. Proses Enkripsi Dekripsi

Prinsip enkripsi pada algoritma ini adalah dengan mengkombinasikan masing-masing karakter pada plainteks dengan satu karakter pada kunci. Oleh karena itu, panjang kunci setidaknya harus sama dengan panjang plainteks. Enkripsi dapat dinyatakan sebagai penjumlahan modulo 26 dari satu karakter plainteks dengan satu karakter kunci OTP:

$$c_i = (p_i + k_i) \bmod 26$$

Yang dalam hal ini, p_i adalah plainteks ke-i, dan c_i adalah huruf cipherteks ke-i. Panjang kunci sama dengan panjang plainteks, sehingga tidak ada kebutuhan mengulang penggunaan kunci selama proses enkripsi.

Setelah pengirim mengenkripsikan pesan dengan kunci, ia menghancurkan kunci tersebut. Penerimaan pesan menggunakan kunci yang sama untuk mendekripsikan karakter-karakter cipherteks menjadi karakter-karakter plainteks dengan persamaan:

$$p_i = (c_i + k_i) \bmod 26$$

Angka 26 muncul karena sistemnya menggunakan abjad. Artinya hanya abjad A – Z saja yang dapat dikodekan dengan sistem seperti ini. Bila diinginkan pengkodean sebarang data, baik teks, gambar, suara maupun video, maka OTP ini diperluas dengan penggunaan sistem bilangan biner. Semua tipe data dapat dianggap sebagai data biner. Dan karena bilangan biner hanya mengenal 0 dan 1, maka basis 26 diubah

menjadi basis 2 (Kurniawan, 2004: 82). Penjumlahan modulo 2 ini dinyatakan dengan XOR. Dan inilah yang sering digunakan dalam sistem digital sekarang ini. Cipherteks diperoleh dengan melakukan penjumlahan modulo 2 satu bit plainteks dengan satu bit kunci:

$$c_i = (p_i + k_i) \bmod 2$$

yang dalam hal ini, p_i : bit plainteks, k_i : bit kunci, c_i : bit chiperteks.

Plainteks diperoleh dengan melakukan penjumlahan modulo 2 satu bit chiperteks dengan satu bit kunci:

$$p_i = (c_i + k_i) \bmod 2$$

Mengingat operasi penjumlahan modulo 2 identik dengan operasi bit dengan operator XOR, maka persamaan enkripsi dapat ditulis sebagai:

$$c_i = p_i \oplus k_i$$

dan proses dekripsi menggunakan persamaan:

$$p_i = c_i \oplus k_i$$

Pada proses *chipering*, bit hanya mempunyai dua buah nilai, sehingga proses enkripsi hanya menyebabkan dua keadaan pada bit tersebut, berubah atau tidak berubah. Dua keadaan tersebut ditentukan oleh kunci enkripsi yang disebut aliran kunci (*keystream*). Aliran kunci dibangkitkan dari sebuah pembangkit yang dinamakan pembangkit aliran kunci (*keystream generator*). Aliran kunci di-XOR-kan dengan aliran bit-bit plainteks p_1, p_2, ..., p_i, untuk menghasilkan aliran bit-bit chiperteks:

$$c_i = p_i \oplus k_i$$

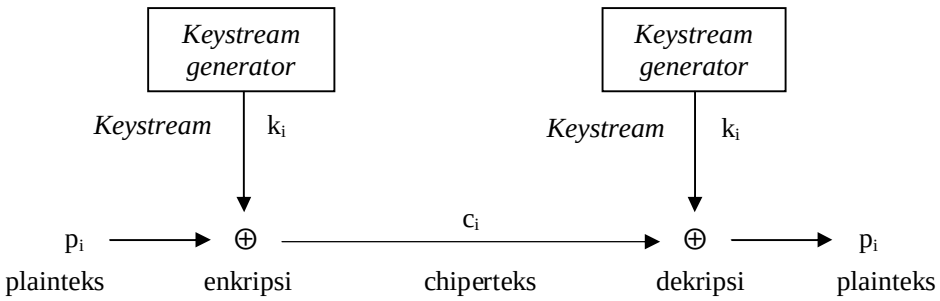
Di sisi penerima, bit-bit chiperteks di-XOR-kan dengan aliran kunci yang sama untuk menghasilkan bit-bit plainteks:

$$p_i = c_i \oplus k_i$$

karena proses enkripsi dua kali berturut-turut menghasilkan kembali plainteks semula.

$$c_i \oplus k_i = (p_i \oplus k_i) \oplus k_i = p_i \oplus (k_i \oplus k_i) = p_i \oplus 0 = p_i$$

Gambar 4.1 memperlihatkan skema global algoritma OTP. Pembangkit aliran kunci menghasilkan elemen bit kunci k_i yang kemudian di-XOR-kan dengan bit plainteks menghasilkan bit chiperteks c_i. Di sisi penerima, pembangkit yang sama digunakan untuk menghasilkan aliran kunci yang sama untuk selanjutnya di-XOR-kan dengan chiperteks c_i dan memberikan kembali plainteks p_i semula.



Gambar 4.1 Konsep algoritma OTP

Seperti kita ketahui bahwa untuk merancang *unbreakable* cipher, ada dua syarat yang harus dipenuhi, yaitu kunci harus dipilih secara acak (setiap kunci harus mempunyai peluang yang sama untuk terpilih) dan

panjang kunci harus sama dengan panjang plainteks yang akan dienkripsikan.

Kedua syarat tersebut dapat menyebabkan plainteks sama belum tentu dienkripsi menjadi cipherteks yang sama. Dengan kata lain kriptanalisis akan mendapatkan hasil bahwa sebuah cipherteks yang didekripsikannya mungkin menghasilkan beberapa plainteks bermakna. Hasil ini akan membingungkannya dalam menentukan plainteks mana yang benar.

Sebagai contoh, dipunyai sebuah plainteks yaitu RUSDI dan memiliki sebuah kunci yaitu CRASH. Perlu diingat bahwa panjang kunci harus sama dengan plainteks dan sebaiknya tidak ada karakter yang diulang. Pertama kita harus mendapatkan kode ASCII dari plainteks kemudian diubah ke bentuk biner.

Tabel 4.1 Kode ASCII dan notasi biner plainteks

Karakter	ASCII	Notasi biner
R	82	0101 0010
U	85	0101 0101
S	83	0101 0011
D	68	0100 0100
I	73	0100 1001

Hal yang sama juga harus dilakukan pada kunci yang dipilih.

Tabel 4.2 Kode ASCII dan notasi biner kunci

Karakter	ASCII	Notasi biner
C	67	0100 0011
R	82	0101 0010
A	65	0100 0001
S	83	0101 0011
H	72	0100 1000

Setelah itu masing-masing karakter di XOR-kan dengan kunci.

Tabel 4.3 Hasil proses XOR plainteks dan kunci

Cipherteks		
0001 0001	=	Ctrl-Q
0000 0111	=	Ctrl-G
0001 0010	=	Ctrl-R
0001 0111	=	Ctrl-W
0000 0001	=	Ctrl-A

Proses dekripsi pesan juga melakukan operasi yang sama yaitu XOR antara Cipher dengan kunci.

Tabel 4.4 Kode ASCII dan notasi biner dekripsi data

Plainteks			Kunci			Cipherteks		
Ctrl-Q	=	0001 0001	C	=	0100 0011	R	=	0101 0010
Ctrl-G	=	0000 0111	R	=	0101 0010	U	=	0101 0101
Ctrl-R	=	0001 0010	A	=	0100 0001	S	=	0101 0011
Ctrl-W	=	0001 0111	S	=	0101 0011	D	=	0100 0100
Ctrl-A	=	0000 0001	H	=	0100 1000	I	=	0100 1001

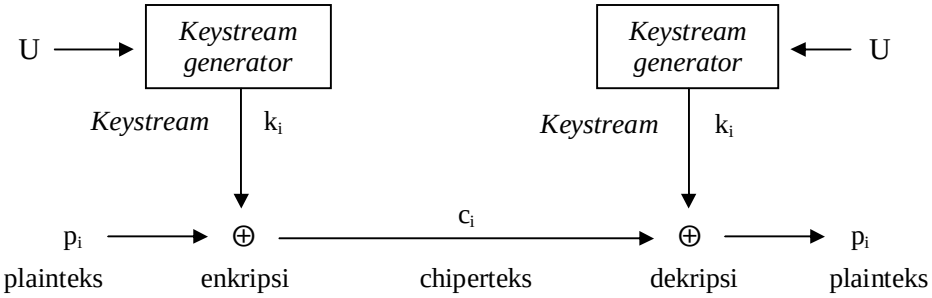
Sistem OTP tidak dapat dipecahkan karena:

a. Barisan kunci acak yang ditambahkan ke pesan plainteks yang tidak acak menghasilkan cipherteks yang seluruhnya acak
b. Beberapa barisan kunci yang digunakan untuk mendekripsi cipherteks mungkin menghasilkan pesan-pesan plainteks yang mempunyai makna, sehingga kriptanalis tidak punya cara untuk menentukan plainteks mana yang benar (Munir, 2006: 95).

Misalkan pada contoh di atas, jika kita mendapatkan sebuah cipher yaitu 0001 0001 maka kita tidak akan pernah bisa memastikan bahwa plainteks-nya adalah R. Sebab bila kuncinya bernilai 0100 1011 maka akan diperoleh plainteks 0101 1010 yang sama dengan huruf Z. Dan bila kunci bernilai 0101 0001, maka akan diperoleh plainteks @ yang bernilai 0100 000. Ini berarti bahwa cipherteks Ctrl-Q bisa memiliki plainteks yang tidak dapat ditentukan bila kita tidak mengetahui kuncinya.

2. Pembangkit Aliran Kunci

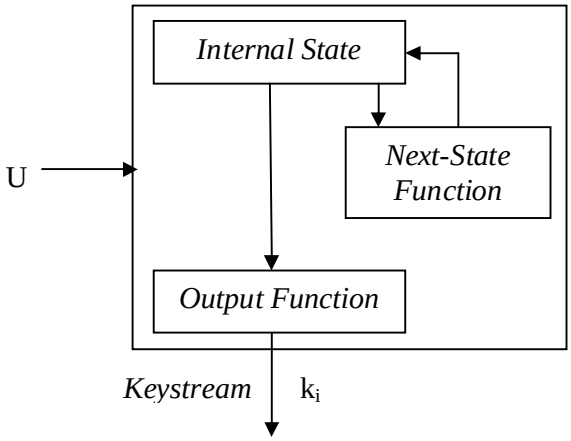
Pembangkit aliran bit kunci diimplementasikan sebagai prosedur algoritmik, sehingga bit-bit kunci (*keystream*) dapa dibangkitkan secara simultan oleh pengirim dan penerima pesan. Prosedur algoritmik tersebut menerima masukan sebuah kunci U. Keluaran dari prosedur merupakan fungsi dari U (Gambar 4.2). Pembangkit harus menghasilkan bit-bit kunci yang kuat secara kriptografi.



Gambar 4.2 *Enchipering* dengan pembangkit aliran kunci yang bergantung pada kunci U

Karena pengirim dan penerima harus menghasilkan bit-bit kunci yang sama, maka keduanya harus memiliki kunci U yang sama. Kunci U ini harus dijaga kerahasiaannya. Algoritma OTP menggunakan kunci U yang relatif pendek untuk membangkitkan bit-bit kunci yang panjang.

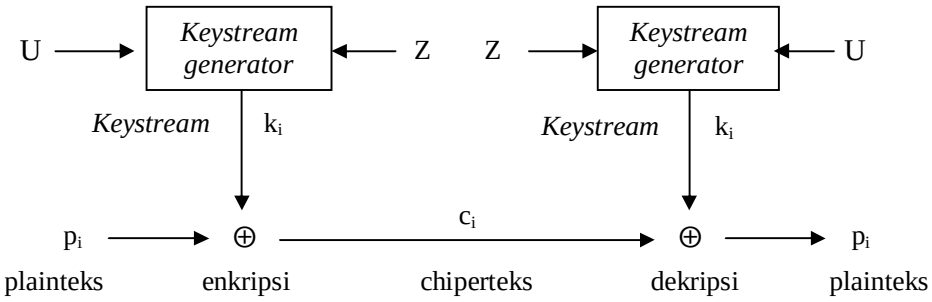
Pada gambar 4.3, aliran kunci dihasilkan sebagai fungsi status berikutnya (*next state function*) dan fungsi luaran (*output function*) yang menghasilkan bit-bit aliran kunci.



Gambar 4.3 Proses di dalam pembangkitan kunci

Karena U adalah besaran yang konstan, maka aliran bit-bit kunci yang dihasilkan pada setiap iterasi tidak berubah jika bergantung hanya pada U. Ini berarti pembangkit aliran kunci tidak boleh dimulai dengan kondisi awal yang sama supaya tidak menghasilkan kembali bit-bit kunci yang sama pada setiap iterasi. Oleh karena itu, beberapa pembangkit

aliran kunci menggunakan umpan yang disimbolkan dengan Z, seperti dalam gambar 4.4



Gambar 4.4 *Enchipering* dengan pembangkit bit aliran kunci yang bergantung pada kunci U dan umpan Z

Dengan demikian, bit-bit kunci K dapat dinyatakan sebagai hasil dari fungsi *g* dengan parameter kunci U dan masukan umpan Z:

$K = g_U(Z)$

Sehingga proses enkripsi dan dekripsi didefinisikan sebagai

$C = P \oplus K = P \oplus g_U(Z)$

$P = C \oplus K = C \oplus g_U(Z)$

Nilai Z yang berbeda-beda pada setiap iterasi menghasilkan bit-bit kunci yang berbeda pula.

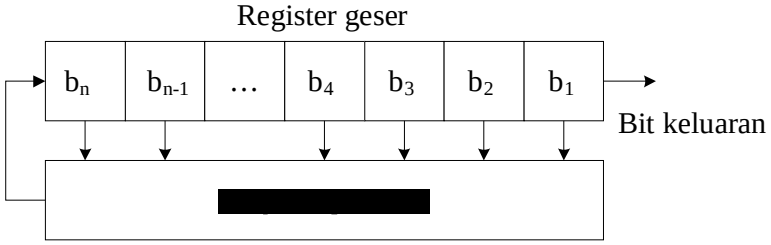
Sampai saat ini, ada beberapa metode yang dikenal dapat digunakan untuk membangkitkan bilangan acak pada OTP seperti LCG (*Linear Congruental Generators*), LFSR (*Linear Feedback Shift Register*), *Generator Geffe* (yang menggunakan 3 atau lebih LFSR), dan metode-metode yang lain. Metode pembangkitan bilangan acak yang

cukup familiar adalah LFSR, sebab LFSR digunakan pada kriptografi dan teori pengkodean serta telah digunakan militer ketika dimulainya penggunaan alat elektronik (Amri, 2006: 4).

B. *Linear Feedback Shift Register* (LFSR)

Pembangkit aliran kunci yang sering digunakan dalam kriptografi adalah LFSR atau yang bisa disebut register geser dengan umpan balik linier. Kriptografi sandi rahasia berbasis register geser telah digunakan militer sejak permulaan penggunaan peralatan elektronik. Register geser umpan balik (*feedback shift register*) atau FSR terdiri atas dua bagian:

1. Register geser, yaitu barisan bit-bit (b_n, b_{n-1}, ..., b_4, b_3, b_2, b_1) yang panjangnya n (disebut juga register geser n-bit)
2. Fungsi umpan balik, yaitu fungsi yang menerima masukan dari register geser dan mengembalikan nilai fungsi ke register geser.

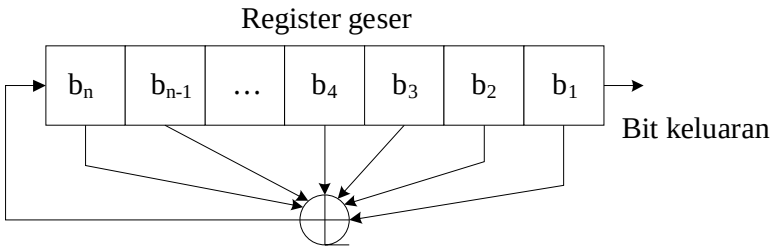


Gambar 4.5 Bagian-bagian FSR

Tiap kali sebuah bit dibutuhkan, semmua bit di dalam register digeser 1 bit ke kanan. Bit paling kiri (b_n) dihitung sebagai fungsi bit-bit lain

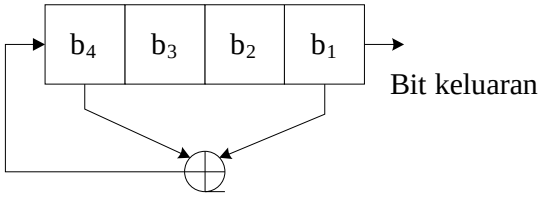
di dalam register tersebut. Keluaran dari register geser adalah 1 bit, yaitu bit b_1 yang tergeser. Bit keluaran ini nantinya akan menjadi kunci enkripsi.

Periode register geser adalah panjang barisan keluaran sebelum ia berulang kembali. Contoh register umpan balik (*feedback shift register*) adalah *linear feedback shift register* (LFSR). Fungsi umpan baliknya adalah peng-XOR-an bit-bit tertentu dalam register.



Gambar 4.6 LFSR sederhana

LFSR n-bit mempunyai $2^n - 1$ status internal (keadaan isi register). Secara teoritis LFSR dapat membangkitkan $2^n - 1$ barisan bit acak semu sebelum perulangan. Jadi periode maksimal LFSR adalah $2^n - 1$. Misal, dibuat LFSR sepanjang 8 bit, maka periode maksimumnya adalah $2^8 - 1 = 255$.



Gambar 4.7 LFSR 4-bit

Gambar 4.7 di atas adalah contoh LFSR 4-bit, yang dalam hal ini fungsi umpan balik meng-XOR-kan b_4 dengan b_1 dan menyimpan hasilnya di b_4:

$$b_4 = f(b_1, b_4) = b_1 \oplus b_4$$

Sebagai contoh, jika register diinisialisasi dengan 1001 (inisialisasi ini bisa sama dengan kunci atau diambil dari kunci), maka isi register (menyatakan status atau *state*) dan bit keluaran sebelum berulang kembali adalah:

Tabel 4.5 Isi register dan bit keluaran LFSR 4-bit

Tahap ke-	b4	b3	b2	b1	Bit Keluaran
0	1	0	0	1	1
1	0	1	0	0	0
2	0	0	1	0	0
3	0	0	0	1	1
4	1	0	0	0	0
5	1	1	0	0	0
6	1	1	1	0	0
7	1	1	1	1	1
8	0	1	1	1	1
9	1	0	1	1	1
10	0	1	0	1	1
11	1	0	1	0	0
12	1	1	0	1	1
13	0	1	1	0	0
14	1	0	1	1	1

Barisan bit-bit keluaran, yang merupakan bit-bit acak adalah:

```
1 0 0 1 0 0 0 1 1 1 1 0 1 0 1 ...
```

Dari LFSR di atas yang hanya sepanjang 4 bit, periodenya akan berulang ketika pergeseran ke-15, karena periode maksimumnya adalah $2^4 - 1 = 15$.

C. Algoritma OTP untuk Sistem Pengamanan *Access Database Server*

Semua yang bersifat mengamankan data menggunakan metode tertentu merupakan inti dari kriptografi, yaitu menjamin kerahasiaan (*confidentiality*) informasi dengan menggunakan enkripsi atau penyandian. Keutuhan (*integrity*) atas basis data dilakukan dengan menggunakan algoritma OTP. Begitu pula dengan jaminan atas identitas dan keabsahan (*authenticity*) pihak-pihak yang mengakses sistem *database*.

Enkripsi pada level basis data dilakukan pada saat data ditulis dan dibaca dari basis data. Enkripsi ini dilakukan pada kolom-kolom tabel basis data. Pemilihan *field* yang sensitif untuk diproteksi merupakan langkah pertama yang harus dilakukan dalam menerapkan proses enkripsi dekripsi.

Dalam penelitian kali ini, penulis mencoba menerapkan proses enkripsi dekripsi pada sistem basis data MIPA*Connect*. Enkripsi ini dilakukan pada *field password* member MIPA*Connect*. Seperti kita ketahui bahwa data tersebut merupakan data yang berkaitan dengan kepentingan member, di mana setiap member memiliki hak untuk dijaga kerahasiaan datanya.

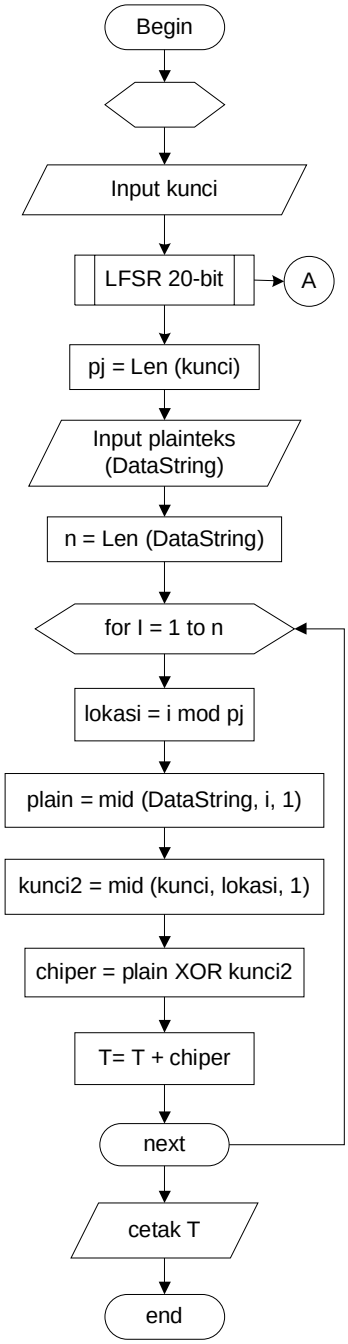
Sebelum membuat diagram blok tentang alur enkripsi data dengan menggunakan algoritma OTP, terlebih dahulu disusun algoritma yang

menunjang adanya enkripsi data menggunakan pengembangan dari algoritma OTP tersebut. Algoritma ini nantinya akan sangat berguna sekali pada waktu implementasai pada program.

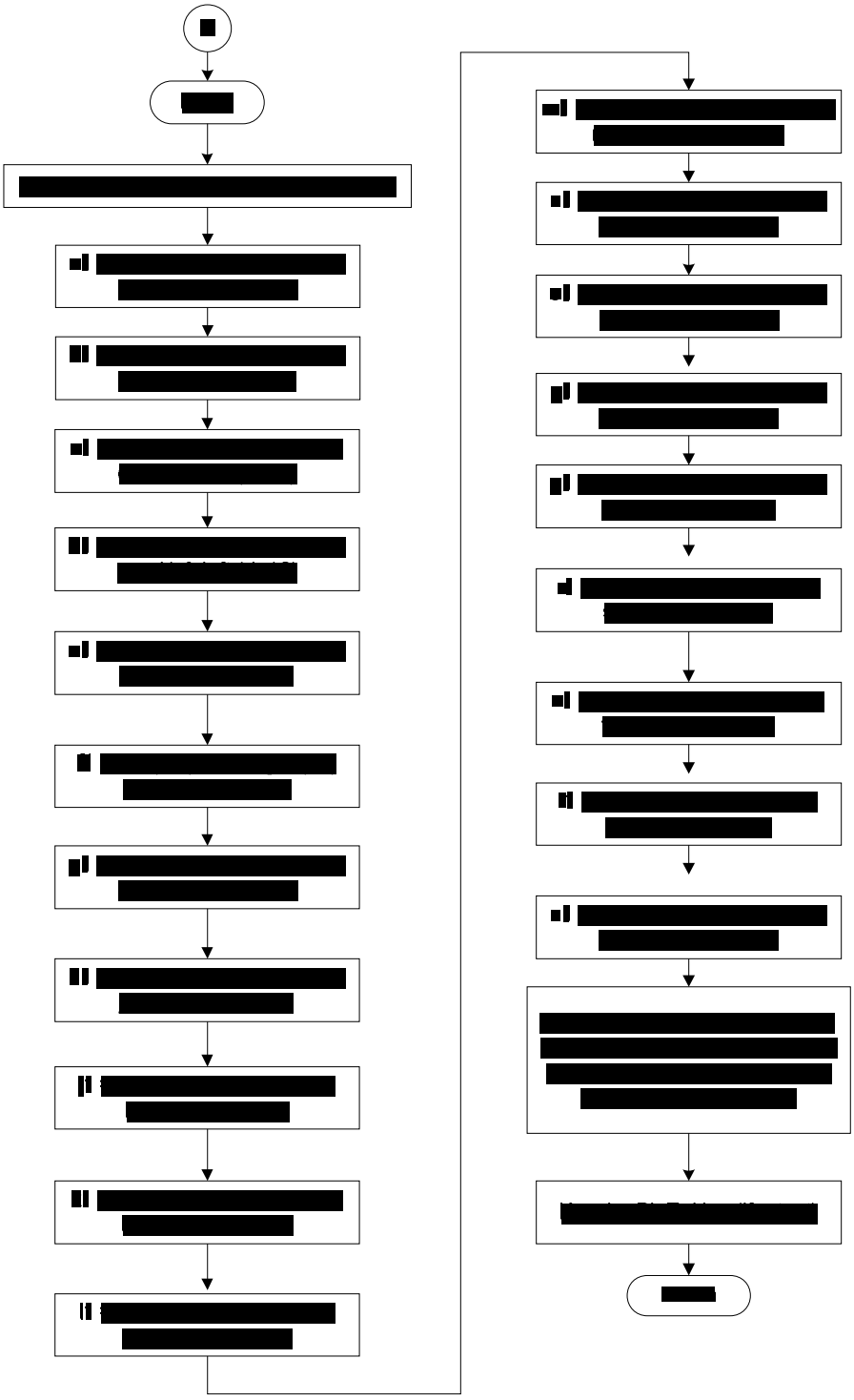
Adapun algoritmanya adalah sebagai berikut.

1. Awalnya masukkan kunci yang akan digunakan untuk melakukan enkripsi dekripsi data.
2. Untuk pembangkit acak kunci dilakukan proses LFSR 20-bit dengan proses sebagai berikut.
   a. b_1 sampai b_{20} diisi oleh bit-bit kunci
   b. Tahap pertama, b_1 dan b_{20} akan di-XOR-kan
   c. b_1 sampai b_{20} digeser ke kanan sepanjang 1 bit
   d. Bit pertama, b_1 akan dijadikan bit keluaran
   e. Bit hasil XOR antara b_1 dan b_{20} akan dimasukkan ke b_{20}
3. Bit-bit keluaran digunakan sebagai kunci baru proses enkripsi dekripsi
4. Kemudian diperiksa jumlah karakter dari kunci baru serta jumlah karakter dari string atau plainteks yang akan dienkripsikan.
5. Lakukan *looping* sebanyak jumlah karakter plainteks
6. Di dalam *looping* dilakukan beberapa operasi sebagai berikut.
   a. Modulokan karakter pertama dengan panjang kunci
   b. Karakter pertama plainteks di-XOR-kan dengan nilai hasil modulo dari proses sebelumnya, karakter berikutnya sampai yang terakhir juga akan di-XOR dengan nilai hasil modulo dari proses sebelumnya.

Adapun diagram blok dari enkripsi dengan menggunakan algoritma OTP dapat dilihat pada gambar 4.8 berikut ini.



Gambar 4.8 Diagram blok enkripsi



Gambar 4.9 Diagram blok LFSR 20-bit

Untuk dekripsi, proses yang sama dilakukan kembali, hanya saja plainteks yang digunakan adalah chiperteks hasil enkripsi sebelumnya. Sedangkan kunci untuk mendekripsikan sama dengan kunci yang digunakan pada saat mengenkripsikan.

D. Bahasa Pemprograman Visual Basic Untuk Algoritma OTP

Visual Basic adalah salah satu *software* untuk membuat program yang cukup sederhana tetapi memiliki banyak cakupan yang dikerjakan, karena Visual Basic dapat mengakses banyak *software* seperti Excel, Access dan sebagainya. Hal ini disebabkan karena Microsoft Visual Basic adalah bahasa pemprograman yang bekerja dalam lingkup Microsoft Windows (Alam, 1999 : 1).

Visual Basic merupakan bahasa pemrograman yang secara cepat dan mudah dapat digunakan untuk membuat aplikasi pada Microsoft Windows. Visual Basic dapat memanfaatkan kemampuan Microsoft Windows secara optimal. Kemampuannya dapat dipakai untuk merancang program aplikasi yang berpenampilan seperti program aplikasi lainnya yang berbasis Microsoft Windows.

Kesederhanaan Visual Basic terletak pada kemudahan membuat bahasa pemprograman dan bentuk tampilan yang dikehendaki. Bahasa pemprograman Visual Basic mampu menambah sendiri sebagian kode program secara otomatis ke dalam program sehingga pekerjaan *programmer* menjadi semakin mudah. Visual Basic tidak akan banyak menyulitkan kita

dalam membangun sebuah aplikasi, sekalipun kita seorang pemula. Hal inilah yang menyebabkan program ini sangant diminati di seluruh dunia oleh para pengguna jasa komputer, karena bahasanya yang mudah dan fitur-fiturnya yang familiar.

Karena alasan-alasan tersebut, maka pada penelitian ini program dibuat adalah untuk mengamankan *database access* dengan menggunakan algoritma OTP pada bahasa pemprograman Visual Basic, khususnya enkripsi untuk sistem pengamanan *access database server* pada *database* MIPA *Connect* Universitas Negeri Semarang.

Berikut ini program utama proses enkripsi dekripsi basis data dengan menggunakan algoritma OTP yang telah dibahas pada bahasan sebelumnya dengan menggunakan bahasa pemprograman Visual Basic.

```
Sub Translate()  'Encrypt/Decrypt Password
Dim i As Integer
Dim lokasi As Integer

‘input kunci diubah menjadi biner untuk proses LFSR
biner = HexToBin(InputBox("Kunci", "Kunci"))

‘proses LFSR 20-bit

a = biner

a1 = Left(a, 1) Xor Right(a, 1)
If a1 = 0 Then
  b = Format("0", a1 & Left(a, 19))
Else: b = Format(a1 & Left(a, 19))
End If

b1 = Left(b, 1) Xor Right(b, 1)
If b1 = 0 Then
  c = Format("0", b1 & Left(b, 19))
Else: c = Format(b1 & Left(b, 19))
End If
```

```
c1 = Left(c, 1) Xor Right(c, 1)
If b1 = 0 Then
 d = Format("0", c1 & Left(c, 19))
Else: d = Format(c1 & Left(c, 19))
End If

d1 = Left(d, 1) Xor Right(d, 1)
If d1 = 0 Then
 e = Format("0", d1 & Left(d, 19))
Else: e = Format(d1 & Left(d, 19))
End If

e1 = Left(e, 1) Xor Right(e, 1)
If e1 = 0 Then
 f = Format("0", e1 & Left(e, 19))
Else: f = Format(e1 & Left(e, 19))
End If

f1 = Left(f, 1) Xor Right(f, 1)
If b1 = 0 Then
 g = Format("0", f1 & Left(f, 19))
Else: g = Format(f1 & Left(f, 19))
End If

g1 = Left(g, 1) Xor Right(g, 1)
If g1 = 0 Then
 h = Format("0", g1 & Left(g, 19))
Else: h = Format(g1 & Left(g, 19))
End If

h1 = Left(h, 1) Xor Right(h, 1)
If h1 = 0 Then
 j = Format("0", h1 & Left(h, 19))
Else: j = Format(h1 & Left(h, 19))
End If

j1 = Left(j, 1) Xor Right(j, 1)
If j1 = 0 Then
 k = Format("0", j1 & Left(j, 19))
Else: k = Format(j1 & Left(j, 19))
End If

k1 = Left(k, 1) Xor Right(k, 1)
If k1 = 0 Then
 l = Format("0", k1 & Left(k, 19))
Else: l = Format(k1 & Left(k, 19))
End If

l1 = Left(l, 1) Xor Right(l, 1)
If l1 = 0 Then
 m = Format("0", l1 & Left(l, 19))
```

```
Else: m = Format(l1 & Left(l, 19))
End If

m1 = Left(m, 1) Xor Right(m, 1)
If m1 = 0 Then
 n = Format("0", m1 & Left(m, 19))
Else: n = Format(m1 & Left(b, 19))
End If

n1 = Left(n, 1) Xor Right(n, 1)
If n1 = 0 Then
 o = Format("0", n1 & Left(n, 19))
Else: o = Format(n1 & Left(n, 19))
End If

o1 = Left(o, 1) Xor Right(o, 1)
If o1 = 0 Then
 p = Format("0", o1 & Left(o, 19))
Else: p = Format(o1 & Left(o, 19))
End If

p1 = Left(p, 1) Xor Right(p, 1)
If p1 = 0 Then
 q = Format("0", p1 & Left(p, 19))
Else: q = Format(p1 & Left(p, 19))
End If

q1 = Left(q, 1) Xor Right(q, 1)
If q1 = 0 Then
 r = Format("0", q1 & Left(q, 19))
Else: r = Format(q1 & Left(q, 19))
End If

r1 = Left(r, 1) Xor Right(r, 1)
If b1 = 0 Then
 s = Format("0", r1 & Left(r, 19))
Else: s = Format(r1 & Left(r, 19))
End If

s1 = Left(s, 1) Xor Right(s, 1)
If s1 = 0 Then
 t = Format("0", s1 & Left(s, 19))
Else: t = Format(s1 & Left(s, 19))
End If

t1 = Left(t, 1) Xor Right(t, 1)
If b1 = 0 Then
 u = Format("0", t1 & Left(t, 19))
Else: u = Format(t1 & Left(t, 19))
End If
```

```
u1 = Left(u, 1) Xor Right(u, 1)
If u1 = 0 Then
 v = Format("0", u1 & Left(u, 19))
Else: v = Format(b1 & Left(u, 19))
End If

If a1 = 0 Then
    lfsr.Text = Format("0", a1 & b1 & c1 & d1 & e1 & f1
    & g1 & h1 & j1 & k1 & l1 & m1 & n1 & o1 & p1 & q1 &
    r1 & s1 & t1 & u1)
Else
    lfsr.Text = Format(a1 & b1 & c1 & d1 & e1 & f1 & g1
    & h1 & j1 & k1 & l1 & m1 & n1 & o1 & p1 & q1 & r1 &
    s1 & t1 & u1)
End If

lfsr.Text = ChkForBinary(lfsr.Text)

‘bit-bit hasil LFSR diubah menjadi Hex

If lfsr.Text <> "" Then
    kunci = BinToHex(lfsr.Text)
End If

‘proses enkripsi/dekripsi

Temp$ = ""
pj = Len(kunci)
For i = 1 To Len(DataString)
       lokasi = (i Mod pj)
       'Gunakan logika XOR utk kombinasi enkrip/dekrip
       plain = Asc(Mid(DataString, i, 1))
       kunci2 = Asc(Mid(kunci, i, 1))
       chiper = Chr(plain Xor kunci2)
       Temp$ = Temp$ + chiper
Next i
End Sub
```

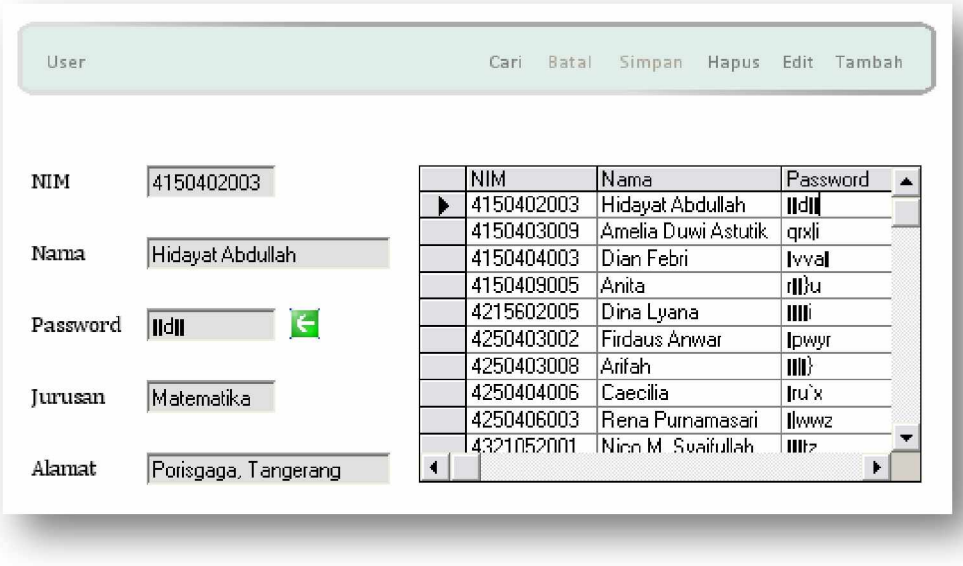
Cara kerja dari fungsi di atas adalah awalnya kita masukkan kunci yang akan digunakan untuk melakukan enkripsi dekripsi data. Untuk pembangkit acak kunci dilakukan proses LFSR 20-bit, yaitu b_1 sampai b_{20} diisi oleh bit-bit kunci, tahap pertama b_1 dan b_{20} akan di-XOR-kan, b_1 sampai b_{20} digeser ke kanan sepanjang 1 bit, bit pertama akan dijadikan bit keluaran sedangkan bit hasil XOR antara b_1 dan b_{20} akan dimasukkan ke b_{20}.

Begitu seterusnya sampai pergeseran ke-20. Bit-bit keluaran digunakan sebagai kunci baru proses enkripsi dekripsi.

Setelah itu diperiksa jumlah karakter dari kunci baru serta jumlah karakter dari string atau plainteks yang akan dienkripsikan. Kemudian dilakukan *looping* sebanyak jumlah karakter plainteks. Di dalam *looping* dilakukan beberapa operasi, yaitu karakter pertama dimodulokan dengan panjang kunci baru, kemudian karakter pertama plainteks di-XOR-kan dengan nilai hasil modulo dari proses sebelumnya, karakter berikutnya sampai yang terakhir juga akan di-XOR dengan nilai hasil modulo dari proses sebelumnya. Hasilnya adalah karakter yang sudah di acak (chiperteks).

Misalnya dimasukkan data string "CON4N" ke dalam fungsi tersebut, serta diberikan input kunci “12345” maka hasilnya adalah karakter acak “zw”.

Berikut ini hasil uji coba enkripsi dengan menggunakan algoritma OTP dalam pemprograman Visual Basic.



Gambar 4.10 Hasil enkripsi pada Visual Basic

Terlihat bahwa pada tabel di atas dalam kolom *password* telah terenkripsi. Jika kita ingin mengetahui plainteks dari *password* tersebut, kita harus memasukkan kunci enkripsinya terlebih dahulu, di mana kunci ini tidak semua orang bisa mengetahuinya. Ini berarti data member, khususnya data *password* telah diproteksi, sehingga kerahasiaan data member dapat terjaga.

BAB V

PENUTUP

A. Simpulan

Dalam penelitian ini memberikan simpulan yang mengindikasikan diperlukannya pengamanan data dengan menggunakan teknik kriptografi. Salah satunya adalah *One Time Pad*. Prinsip enkripsi pada algoritma ini adalah dengan mengkombinasikan masing-masing karakter pada plainteks dengan satu karakter pada kunci. Oleh karena itu, panjang kunci setidaknya harus sama dengan panjang plainteks. Untuk membangkitkan aliran kunci, dilakukan proses *linear feedback shift register* (LFSR) atau yang biasa disebut register geser dengan umpan balik linier. Bit-bit keluaran proses LFSR digunakan sebagai kunci baru proses enkripsi dekripsi. Fungsi untuk mengenkrip pada algoritma OTP hanyalah meng-XOR-kan plainteks dengan kunci yang telah disiapkan untuk menghasilkan cipherteks. Sedangkan fungsi untuk mendekrip tinggal meng-XOR-kan cipherteks dengan kunci yang sudah disepakati.

B. Saran

Mekanisme enkripsi yang digunakan dalam penelitian kali ini memang masih sederhana, akan tetapi diharapkan dapat berguna sebagai langkah awal untuk masuk ke dunia kriptografi, khususnya dalam implementasi pengamanan basis data dengan menggunakan bahasa

pemprograman Visual Basic. Untuk kedepannya, diharapkan penelitian ini dapat dikembangkan, digunakan serta diterapkan pada bidang-bidang kehidupan yang lain yang lebih kompleks.

DAFTAR PUSTAKA

Crasher. 2006. *Algoritma Enkripsi One Time Pad*. http://www.code_attack.com.

Hariyanto, Bambang. 2004. *Sistem Manajemen Basis Data*. Bandung: Penerbit Informatika.

Kadir, A. 1999. *Konsep dan Tuntunan Praktis Basis Data*. Yogyakarta: Penerbit Andi.

Kurniawan, Yusuf. 2004. *Kriptografi Keamanan Internet dan Jaringan Komunikasi*. Bandung: Penerbit Informatika.

Munir, Rinaldi. 2006. *Kriptografi*. Bandung: Penerbit Informatika.

Nugroho, Adi. 2004. *Konsep Pengembangan Sistem Basis Data*. Bandung: Penerbit Informatika.

One Time Pad. http://www.topsecretcripto.com

Riyanto, Djalal. 2004. *Buku Ajar Basis Data*. Semarang: Universitas Diponegoro.

Simple Encryption Mechanism. http://bulsara.host.sk/index.php?p=2017&d=4021

Wahana Komputer. 2003. *Memahami Model Enkripsi dan Security Data*. Yogyakarta: Andi Offset.

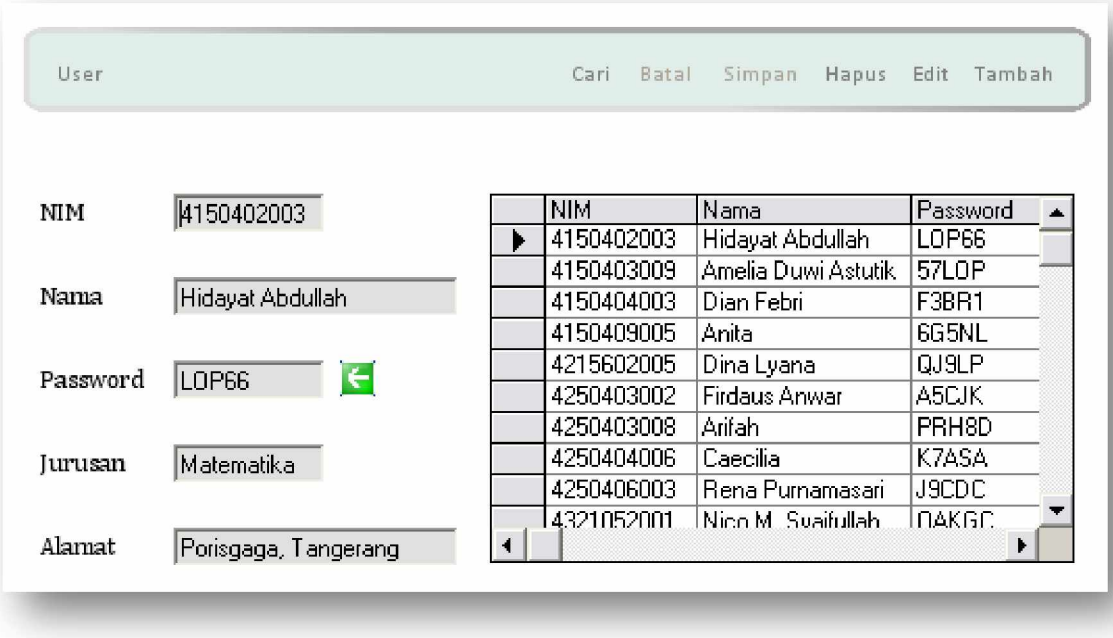
Wikipedia Indonesia. http://id.wikipedia.org/.

Proses Replikasi Data Enkripsi Antara Server Utama dan Firewall. http://keudekupi.com/index.php?option=com_content&task=view&id=30&Itemid=32

Lampiran 1. Tabel ASCII (*American Standard Code for Information Interchange*)

binary	MSN	0000		0001		0010		0011		0100		0101		0110		0111	
LSN	hex	0		1		2		3		4		5		6		7	
0000	0	NUL	0	DLE	16	SP	32	0	48	@	64	P	80	`	96	p	112
			00		10		20		30		40		50		60		70
0001	1	SOH	1	XON	17	!	33	1	49	A	65	Q	81	a	97	q	113
			01	(DC1)	11		21		31		41		51		61		71
0010	2	STX	2	DC2	18	"	34	2	50	B	66	R	82	b	98	r	114
			02		12		22		32		42		52		62		72
0011	3	ETX	3	XOFF	19	#	35	3	51	C	67	S	83	c	99	s	115
			03	(DC2)	13		23		33		43		53		63		73
0100	4	EOT	4	DC4	20	$	36	4	52	D	68	T	84	d	100	t	116
			04		14		24		34		44		54		64		74
0101	5	ENQ	5	NAK	21	%	37	5	53	E	69	U	85	e	101	u	117
			05		15		25		35		45		55		65		75
0110	6	ACK	6	SYN	22	&	38	6	54	F	70	V	86	f	102	v	118
			06		16		26		36		46		56		66		76
0111	7	BEL	7	ETB	23	'	39	7	55	G	71	W	87	g	103	w	119
			07		17		27		37		47		57		67		77
1000	8	BS	8	CAN	24	(	40	8	56	H	72	X	88	h	104	x	120
			08		18		28		38		48		58		68		78
1001	9	HT	9	EM	25	)	41	9	57	I	73	Y	89	i	105	y	121
			09		19		29		39		49		59		69		79
1010	A	LF	10	SUB	26	*	42	:	58	J	74	Z	90	j	106	z	122
			0A		1A		2A		3A		4A		5A		6A		7A
1011	B	VT	11	ESC	27	+	43	;	59	K	75	[	91	k	107	{	123
			0B		1B		2B		3B		4B		5B		6B		7B
1100	C	FF	12	FS	28	,	44	<	60	L	76	\	92	l	108	|	124
			0C		1C		2C		3C		4C		5C		6C		7C
1101	D	CR	13	GS	29	-	45	=	61	M	77	]	93	m	109	}	125
			0D		1D		2D		3D		4D		5D		6D		7D
1110	E	SO	14	RS	30	.	46	>	62	N	78	^	94	n	110	~	126
			0E		1E		2E		3E		4E		5E		6E		7E
1111	F	SI	15	US	31	/	47	?	63	O	79	(SPACE)	95	0	111	DEL	127
			0F		1F		2F		3F		4F		5F		6F		7F

Lampiran 2. Tampilan *Database* Sebelum Dienkripsi



Lampiran 3. Tampilan *Database* Setelah Dienkripsi dengan Menggunakan Algoritma *One Time Pad.*