IMPLEMENTASI ENKRIPSI DAN DEKRIPSI DENGAN METODE RC4 UNTUK PENGAMANAN DATA SISTEM INFORMASI

(Skripsi)

Oleh

Uli Sholihah Saragih

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2017

ABSTRAK

IMPLEMENTASI ENKRIPSI DAN DEKRIPSI DENGAN METODE RC4 UNTUK PENGAMANAN DATA SISTEM INFORMASI

Oleh

Uli Sholihah Saragih

Pada tahun 2015, Sistem Informasi Kegiatan Dosen (SIKSEN) Jurusan Ilmu Komputer Unila mengalami penyerangan data (*hack*) yang mengakibatkan data kegiatan dosen tidak dapat muncul di sistem informasi tersebut. Penyerangan data yang terjadi adalah salah satu akibat dari kurangnya keamanan data pada Sistem Informasi Jurusan Ilmu Komputer. Salah satu cara untuk mengamankan data adalah dengan menggunakan teknik kriptografi. Kriptografi merupakan cara yang dilakukan dengan menyandikan pesan asli menjadi pesan acak yang sulit dipahami. Didalam kriptografi terdapat dua proses utama yaitu enkripsi dan dekripsi. Enkripsi merupakan mengubah pesan asli menjadi pesan sandi, sedangkan dekripsi adalah proses pengembalian dari pesan sandi ke pesan aslinya. Metode yang dipakai adalah metode *Rivest Code* 4 (RC4) karena metode ini merupakan salah satu metode yang hasil penyandian (*ciphertext*) memiliki ukuran panjang karakter yang sama dengan pesan aslinya (*plaintext*). Metode ini terdiri dari tiga tahap utama yaitu KSA (*Key Schedulling Algorithm*), PRGA (*Pseudo Random Generation Algorithm*) dan proses XOR. Data yang digunakan dalam penelitian ini adalah data mahasiswa Jurusan Ilmu Komputer Unila yang berjumlah 970 data mahasiswa yang terdiri dari 25 kolom data. Pengujian dilakukan dengan menguji apakah jumlah karakter pesan asli sama dengan pesan sandinya dan berapa banyak data yang berhasil dienkripsi dan didekripsi. Hasil pengujian didapatkan bahwa 970 data berhasil dienkripsi dan 970 data berhasil didekripsi dengan jumlah karakter yang sama antara pesan asli dan pesan sandi.

Keyword: RC4, Enkripsi, Dekripsi, Kriptografi, *Database*

ABSTRACT

IMPLEMENTASI ENKRIPSI DAN DEKRIPSI DENGAN METODE RC4 UNTUK PENGAMANAN DATA SISTEM INFORMASI

Oleh

Uli Sholihah Saragih

In 2015, Lecturer Activity Information System (SIKSEN) Department of Computer Science Unila experiencing data attacks (hack) which resulted in data lecturer activities can not appear in the information system. Attacking the data that occurred is one result of the lack of data security in Information Systems Computer Science Department. One way to secure data is to use cryptographic techniques. Cryptography is the way it does by encoding the original message into an elusive random message. In cryptography there are two main processes namely encryption and decryption. Encryption represents converting the original message into a password message, while decryption is the process of returning the password message to the original message. The method used is Rivest Code 4 (RC4) because this method is one of the methods that the encoding result (ciphertext) has the same character length as the original message (plaintext). This method consists of three main stages: KSA (Key Scheduling Algorithm), PRGA (Pseudo Random Generation Algorithm) and XOR process. The data used in this study is the data of Unila Computer Science students, amounting to 970 student data consisting of 25 columns of data. Testing is done by testing whether the number of original message characters is the same as the password and how much data is successfully encrypted and decrypted. The test results obtained that 970 data successfully encrypted and 970 data successfully decrypted with the same number of characters between the original message and password.

Keyword: RC4, Encryption, Decryption, Cryptography, Database

IMPLEMENTASI ENKRIPSI DAN DEKRIPSI DENGAN METODE RC4 UNTUK PENGAMANAN DATA SISTEM INFORMASI

Oleh

Uli Sholihah Saragih

Skripsi
Sebagai Salah Satu Syarat untuk Memperoleh Gelar
SARJANA KOMPUTER

Pada

Program Studi Ilmu Komputer
Fakultas Matematika dan Ilmu Pengetahuan Alam

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2017

Judul Skripsi	**: IMPLEMENTASI ENKRIPSI DAN DEKRIPSI DENGAN METODE RC4 UNTUK PENGAMANAN DATA SISTEM INFORMASI**
Nama Mahasiswa	**: Uli Sholihah Saragih**
Nomor Pokok Mahasiswa	**: 1317051067**
Program Studi	**: S1 Ilmu Komputer**
Jurusan	**: Ilmu Komputer**
Fakultas	**: Matematika dan Ilmu Pengetahuan Alam**

MENYETUJUI

1. **Komisi Pembimbing**

Dwi Sakethi, S.Si., M.Kom
NIP. 19680611 199802 1 001

2. **Mengetahui,**

Ketua Jurusan Ilmu Komputer

Dr. Ir. Kurnia Muludi, M.S.Sc.
NIP. 19640616 198902 1 001

MENGESAHKAN

1. **Tim Penguji**

Ketua : Dwi Sakethi, S.Si., M.Kom

Penguji 1

Bukan Pembimbing : Wamiliana, Dra., MA, Ph.D

Penguji 2

Bukan Pembimbing : Febi Eka Febriansyah, M.T

2. **Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam**

Prof. Dr. Warsito, S.Si, DEA, Ph.D.

NIP. 19710212 1995121 001

Tanggal Lulus Ujian Skripsi : 23 Agustus 2017

PERNYATAAN

Saya yang bertandatangan di bawah ini, menyatakan bahwa skripsi saya yang berjudul "Implementasi Enkripsi dan Dekripsi dengan Metode RC4 untuk Pengamanan Data Sistem Informasi" ini merupakan hasil karya sendiri dan bukan hasil karya orang lain. Semua hasil tulisan yang tertuang dalam skripsi ini telah mengikuti kaidah penulisan karya ilmiah Universitas Lampung. Apabila di kemudian hari terbukti bahwa skripsi ini merupakan hasil salinan atau dibuat oleh orang lain, maka saya bersedia menerima sanksi berupa pencabutan gelar akademik yang telah saya terima.

Bandar Lampung, 23 Agustus 2017

Uli Sholihah Saragih

NPM. 1317051067

RIWAYAT HIDUP

Penulis dilahirkan di Metro, Lampung pada tanggal 30 Maret 1996, sebagai anak pertama yang kembar dari empat bersaudara, dari pasangan Bapak Hairul Syahri Saragih dan Ibu Rofingah. Penulis memulai pendidikan di TK Darul Falah, Purworejo, Kotagajah, Lampung Tengah pada Tahun 2000. Pada tahun 2001, penulis melanjutkan sekolahnya di SDN 2 Purworejo, Kotagajah untuk kelas 1 sampai dengan kelas 3 dan SD IT Bustanul Ulum, Terbanggi Besar untuk kelas 4 sampai dengan kelas 6. Pada tahun 2007, penulis menempuh pendidikan di SMPN 2 Kotagajah, Lampung Tengah dan aktif sebagai anggota organisasi English Club (EC) dan anggota altet olahraga di bidang Catur. Pada tahun 2010, penulis melanjutkan di SMAN 1 Kotagajah, Lampung Tengah dan selama menjadi siswa, penulis aktif sebagai anggota organisasi English Club (EC), anggota organisasi ROHIS (Rohani Islam) dan anggota Olimpiade di Bidang Studi Komputer .

Pada tahun 2013, penulis terdaftar sebagai mahasiswa Program Studi Ilmu Komputer Jurusan Ilmu Komputer FMIPA Universitas Lampung melalui jalur SBMPTN. Selama menjadi mahasiswa penulis pernah menjadi anggota Himpunan Mahasiswa Komputer (HIMAKOM) FMIPA Unila dan Bendahara Bidang KRT ROIS FMIPA Unila. Selain itu penulis juga pernah mengikuti beberapa kegiatan perkuliahan dikampus diantaranya:

1. Mengikuti kegiatan Karya Wisata Ilmiah (KWI) FMIPA Unila di Desa Mulyo Sari, Tanjung Sari pada tanggal 27 Januari 2014-01 Februari 2014.

2. Melaksanakan Kuliah Kerja Nyata (KKN) di Desa Gunung Tapa Tengah, Kecamatan Gedong Meneng, Tulang Bawang selama 60 hari pada tanggal 19 Januari 2016-18 Maret 2016.

3. Melaksanakan Kerja Praktik (KP) di Badan Ketahanan Pangan Daerah (BKPD) Provinsi Lampung selama 40 hari dari tanggal 18 Juni 2016-31 Agustus 2016.

PERSEMBAHAN

Kupersembahkan karya kecilku ini kepada :

Allah Subhanahu Wa Ta'ala,

sebagai bentuk penghambaanku dalam menuntut ilmu
yang tidaklah seberapa dibandingkan
Ilmu-Nya yang begitu luas.

Ummi dan Ayah tercinta,

(Ibu Rofingah dan Bapak Hairul Syahri Saragih)

Terima kasih atas segala kerja keras, dukungan dan doa yang tiada henti-hentinya demi tercapai semua cita-cita dan impianku.

Saudara-saudariku tersayang,

(Ahi Saragih, Iqlimah Saragih dan Mujahid Saragih)

Untuk semangat dan doa yang telah diberikan.

Serta...

Sahabat-sahabatku

MOTTO

إِنَّ مَعَ الْعُسْرِ يُسْرًا

“Sesungguhnya sesudah kesulitan itu ada kemudahan.”
(QS. Alam Nasyroh: 6)

لَا يُكَلِّفُ اللَّهُ نَفْسًا إِلَّا وُسْعَهَا لَهَا مَا كَسَبَتْ وَعَلَيْهَا مَا اكْتَسَبَتْ

“Allah tidak membebani seseorang melainkan sesuai dengan kesanggupannya.Ia mendapat pahala (dari kebajikan) yang diusahakannya dan ia mendapat siksa (dari kejahatan) yang dikerjakannya.”
(Q.S Al Baqarah: 286)

-Hidup adalah anugrah sekaligus tanggungjawab-
(Uli Sholihah Saragih)

-Jangan mudah putus asa, banyak berusaha dan jangan lupa selalu berdoa-
(Umi dan Ayah)

SANWACANA

Puji syukur penulis panjatkan kehadirat Allah SWT, atas segala rahmat, nikmat dan karunia-Nyalah penulis dapat menyelesaikan skripsi dengan judul "Implementasi Enkripsi dan Dekripsi dengan Metode RC4 untuk Pengamanan Data Sistem Informasi" sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer di Universitas Lampung.

Penulis menyadari skripsi ini dapat diselesaikan dengan baik tak terlepas dari bantuan dan partisipasi dari berbagai pihak. Oleh karena itu pada kesempatan ini penulis ingin menyampaikan rasa terima kasih yang sebesar-besarnya kepada:

1. Bapak Dwi Sakethi, S.Si., M.Kom selaku pembimbing. Terima kasih telah banyak mengarahkan dalam perbaikan skripsi ini agar menjadi lebih baik. Bukan hanya di bidang akademik, melalui kebiasaan dan pemikirannya juga telah mengajarkan nilai-nilai moral kehidupan. Terima kasih atas segala bimbingan, waktu yang diluangkan dan pelajaran hidupnya sehingga menjadi inspirasi dan pedoman bagi penulis.
2. Ibu Wamiliana, Dra., MA, Ph.D selaku pembahas pertama yang telah meluangkan waktu di tengah kesibukannya untuk memberikan coretan-coretan yang sangat membantu dalam perbaikan skripsi penulis. Terimakasih atas koreksi, saran dan nasihat yang ibu berikan kepada penulis selama ini.

3. Bapak Febi Eka Febriansyah, M.T selaku pembahas kedua yang telah yang telah membantu penulis untuk selalu lebih teliti dan cermat didalam mengerjakan skripsi. Terimakasih atas segala koreksi dan masukan kepada penulis agar selalu lebih baik kedepannya.
4. Bapak Rico Andrian, S.Si., M.Kom selaku pembimbing akademik penulis.
5. Bapak Dr. Ir. Kurnia Muludi, M.S.Sc. selaku Ketua Jurusan Ilmu Komputer FMIPA Universitas Lampung.
6. Bapak Prof. Dr. Warsito, S.Si, DEA, Ph.D. selaku Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam.
7. Bapak dan Ibu dosen jurusan Ilmu Komputer yang telah memberikan ilmu yang bermanfaat kepada penulis selama di bangku perkuliahan ini.
8. Kedua orangtua tercinta, Bapak Hi. Hairul Syahri Saragih, S.Pd dan Ibu Hj. Rofingah, S.Pd yang senantiasa berjuang tanpa lelah untuk masa depanku. Terima kasih umi, ayah atas semua pengorbanan, jerih payah, dukungan, semangat dan doa yang tiada henti kalian berikan untukku.
9. Saudara kembarku, Ahi Sholihin Saragih yang lebih duluan wisuda. Terimakasih atas semangat dan doa yang diberikan serta menjadi pendengar setia untuk segala cerita. Adik-adikku, Iqlimah Saragih dan Mujahidur Rahman Saragih yang telah memberikan canda tawa, semangat dan doa untuk keberhasilan kakaknya.
10. Teman-teman jurusan Ilmu Komputer angkatan 2013. Terimakasih atas kebersamaan selama diperkuliahan dalam suka maupun duka.
11. Rekan Rempong Unal yang selalu membantu selama kegiatan perkuliahan dan seminar: Berliana Yuni Sari, Lilis Oktaviani Sirait dan Anisa.

12. Keluarga ROIS FMIPA UNILA : Mb Jeje, Mb Taqiya, Mb Con, Mb Ruwaidah, Mb Nay, Mb Ria, Mb Kristy, Ukh Melita, Uut, Nurul, Fentry, Eria, Dini, Annisa, Erva, Ines, Suci, Sinta, Siar, Putri. Bisa menjadi bagian dari kalian sungguh pengalaman yang benar-benar berharga.
13. LQ An-Najm : Mb Shofi, Eria, Annisa, Citra, Fentry, Fitria dan Melina yang telah mengingatkan untuk selalu *istiqomah*, "Aku mencintai kalian karena Allah SWT".
14. Keluarga KKN Gunung Tapa Tengah 2016: Juliana Marbun, Dyah Ayu Kumala Dewi, Meti Fitriana, Bunda Marlyna, Bang Wahyu Gautama dan M.Abidin Amriansyah. Terima kasih atas doanya, pengalaman tak terlupakan selama 60 hari bersama kalian akan selalu ada, I am gonna miss you guys.
15. Keluarga kos Asrama Dania Putri : Intan, May, Ulvi, Anis, Mba (Adit, Reni, Renai, Ayu, Febri, Selvi, Dila, Nita, Novi, Adit, Widya, Evi, Alfi, Mb Mariska, Mb Novi) terimakasih atas kebersamaan kita selama tinggal di kos.
16. Sahabat Prilly: Rika dan Peti yang selalu ada dalam suka duka dan mendoakan penulis agar cepat wisuda. Semoga persahabatan kita langgeng terus.
17. Seluruh pihak yang telah telah membantu yang tidak dapat disebutkan satu persatu, atas peran dan dukungannya dalam penyelesaian skripsi ini.

Penulis menyadari bahwa skripsi ini masih jauh dari sempurna, akan tetapi semoga skripsi ini bermanfaat bagi semua civitas Ilmu Komputer Unila.

Bandar Lampung, 23 Agustus 2017

Uli Sholihah Saragih

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR KODE PROGRAM

BAB I

PENDAHULUAN

1.1 Latar Belakang

Saat ini data dan informasi menjadi hal yang penting. Berbagai perusahaan, organisasi, instansi, dan lainnya telah memanfaatkan teknologi *database* untuk menyimpan dan juga mengelola data organisasi atau perusahaannya. Keamanan terhadap data yang disimpan dalam *database* sudah menjadi hal yang sangat dibutuhkan. Namun keamanan pada *database* dengan pembatasan hak akses sudah tidak lagi dapat menjamin keamanan data karena kebocoran data dapat disebabkan oleh pihak-pihak yang langsung berhubungan dengan *database* itu sendiri seperti *administrator database*.

Pada tahun 2015, data kegiatan dosen di Sistem Informasi Jurusan Ilmu Komputer FMIPA Unila mengalami penyerangan data (*hack*) yang mengakibatkan data kegiatan dosen tidak dapat muncul di sistem informasi tersebut. Penyerangan data yang terjadi adalah salah satu akibat dari kurangnya keamanan data pada Sistem Informasi Jurusan Ilmu Komputer. Salah satu data lainnya yang terdapat pada Sistem Informasi Jurusan Ilmu Komputer adalah data mahasiswa Jurusan Ilmu Komputer. Data tersebut merupakan data penting yang kini perlu dijaga keamanannya.

Salah satu cara mengamankan sebuah data adalah dengan menggunakan kriptografi. Kriptografi adalah ilmu dan seni penyimpanan pesan, data, atau informasi secara aman. Dalam ilmu kriptografi terdapat proses enkripsi dan dekripsi. Enkripsi adalah proses mengubah suatu pesan asli (*plaintext*) menjadi suatu pesan dalam bahasa sandi (*ciphertext*) sedangkan dekripsi adalah proses mengubah pesan dalam suatu bahasa sandi kembali menjadi pesan asli.

Pada penelitian yang dilakukan oleh Jumrin (2016) yang berjudul "Aplikasi Sistem Keamanan Basis Data dengan Teknik Kriptografi RC4 *Stream Cipher*" dijelaskan bahwa hasil penelitian tersebut menunjukkan bahwa data pada tabel basis data dapat terenkripsi dengan menggunakan algoritma RC4 yang bersifat *stream cipher* yakni didekripsikan secara *byte* per *byte* dan proses enkripsi dan dekripsi jauh lebih cepat karena menggunakan kunci yang sama serta memiliki tingkat keamanan yang tinggi sesuai panjang kunci. Tetapi proses enkripsi pada penelitian ini tidak dapat dilakukan sekaligus dengan mengenkrip semua *field* yang ada. Selain itu proses enkripsi dilakukan dengan memilih *database* yang akan dienkripsi sehingga apabila terjadi penambahan atau pengubahan *database*, maka dilakukan kembali proses enkripsi sehingga hal ini menjadi tidak efektif.

Selanjutnya pada penelitian yang dilakukan oleh Arifyanto (2013) yang berjudul "Implementasi Enkripsi Basis Data Berbasis Web Dengan Algoritma *Stream Cipher* RC4", dijelaskan bahwa enkripsi hanya dilakukan sebatas *login* saja. Padahal apabila enkripsi *login* dapat dipecahkan, maka data yang tersimpan dalam *database* masih dapat diambil atau dibaca oleh orang lain.

RC4 merupakan algoritma kriptografi kunci simetris dan bersifat *stream cipher* sehingga panjang karakter hasil enkripsi (*ciphertext)* mempunyai panjang karakter yang sama dengan data asli (*plaintext*). Teknik kriptografi simetris dipilih karena dengan algoritma ini proses enkripsi dan dekripsi data dapat dilakukan dengan waktu yang lebih cepat dibandingkan dengan algoritma kriptografi asimetris.

Berdasarkan pada informasi yang dipaparkan, peneliti membuat sebuah penerapan enkripsi dan dekripsi dengan menggunakan metode RC4 pada data mahasiswa Jurusan Ilmu Komputer FMIPA Universitas Lampung dengan mengenkripsi dan mendekripsi isi data, bukan data *login* karena apabila *login* dapat dibobol maka data yang tersimpan dalam *database* masih dapat dibaca oleh orang lain. Kemudian cara kerja dari proses enkripsi dan dekripsi akan dibuat menjadi lebih mudah dari penelitian sebelumnya yaitu dengan mengenkripsi dan mendekripsi isi data secara keseluruhan sehingga tidak perlu harus memilih *database* dan kolom yang akan dienkripsi dan dekripsi. Implementasi ini menerapkan metode sistem enkripsi dan dekripsi dengan metode RC4 simetris dalam pengamanan data mahasiswa Jurusan Ilmu Komputer dengan judul "**Implementasi Enkripsi Dan Dekripsi Dengan Metode RC4 Untuk Pengamanan Data Sistem Informasi**".

1.2 Rumusan Masalah

Rumusan masalah pada penelitian ini adalah bagaimana mengenkripsi dan mendekripsikan sebuah *database* pada data sistem informasi yang dapat membantu keamanan *database* dengan menggunakan metode algoritma RC4 simetris yang bersifat *stream cipher* sehingga data hasil enkripsi (*ciphertext*) mempunyai ukuran yang sama dengan data asli (*plaintext*).

1.3 Tujuan

Tujuan penelitian ini adalah mengimplementasikan algoritma RC4 yang bersifat *stream cipher* pada data mahasiswa Jurusan Ilmu Komputer Fakultas MIPA Universitas Lampung.

1.4 Batasan Masalah

Batasan masalah pada penelitian ini yaitu sebagai berikut.

1. Enkripsi dan dekripsi sebuah *database* menggunakan metode algoritma RC4 yang bersifat stream *cipher* dan memiliki kunci simetris,
2. Enkripsi dan dekipsi dilakukan dengan menggunakan bahasa pemrograman PHP dan menggunakan aplikasi XAMMP,
3. Data yang akan dienkripsi adalah data mahasiswa Ilmu Komputer Unila yang berjumlah 970 data mahasiswa.

1.5 Manfaat

Manfaat yang akan didapat dari penelitian ini yaitu

1. Mengamankan data mahasiswa Jurusan Ilmu Komputer FMIPA Unila,
2. Menjamin kerahasiaan data,
3. Menambah pengetahuan tentang metode RC4,
4. Menghasilkan kode PHP metode RC4 yang dapat diimplementasikan untuk proses enkripsi dan dekripsi pada data lainnya.

BAB II

TINJAUAN PUSTAKA

2. 1 Kriptografi

Kriptografi penting dalam dunia teknologi informasi saat ini terutama dalam bidang komputer yang mempelajari teknik-teknik matematika yang berhubungan dengan aspek keamanan informasi. Kriptografi juga menjadi salah satu syarat penting dalam keamanan teknologi informasi dalam pengiriman pesan penting dan rahasia.

2.1.1 Pengertian Kriptografi

Kriptografi (*cryptography*) berasal dari bahasa Yunani: "*cryptos*" yang artinya "*secret*" (rahasia) dan "*graphein*" yang artinya "*writing*" (tulisan). Jadi kriptografi adalah ilmu dan seni untuk menjaga keamanan pesan (c*ryptography is the art and science of keeping message secure*) (Munir, 2008).

2.1.2 Tujuan Kriptografi

Salah satu upaya pengamanan sistem informasi yang dapat dilakukan adalah kriptografi. Kriptografi sesungguhnya merupakan studi terhadap teknik matematis yang terkait dengan aspek keamanan suatu sistem informasi, antara lain seperti kerahasiaan, integritas data, keaslian, dan ketiadaan penyangkalan. Keempat aspek

tersebut yang menjadi tujuan fundamental dari suatu sistem kriptografi (Munir, 2011).

1. Kerahasiaan (*confidentiality*)

   Kerahasiaan adalah layanan yang digunakan untuk menjaga informasi dari setiap pihak yang tidak berwenang untuk mengaksesnya. Dengan demikian informasi hanya akan dapat diakses oleh pihak-pihak yang berhak saja.

2. Integritas data (*data integrity*)

   Integritas data merupakan layanan yang bertujuan untuk mencegah terjadinya pengubahan informasi oleh pihak-pihak yang tidak berwenang. Untuk meyakinkan integritas data ini harus dipastikan agar sistem informasi mampu mendeteksi terjadinya manipulasi data. Manipulasi data yang dimaksud meliputi penyisipan, penghapusan, maupun penggantian data.

3. Keaslian (*authentication*)

   Keaslian merupakan layanan yang terkait dengan pembuktian identifikasi terhadap pihak-pihak yang ingin mengakses sistem informasi (*entity authentication*) maupun pembuktian data dari sistem informasi itu sendiri (*data origin authentication*).

4. Ketiadaan penyangkalan (*non-repudiation*)

   Ketiadaan penyangkalan merupakan layanan yang berfungsi untuk mencegah entitas yang berkomunikasi melakukan penyangkalan, yaitu pengiriman pesan menyangkal melakukan pengiriman atau penerima pesan menyangkal telah menerima pesan.

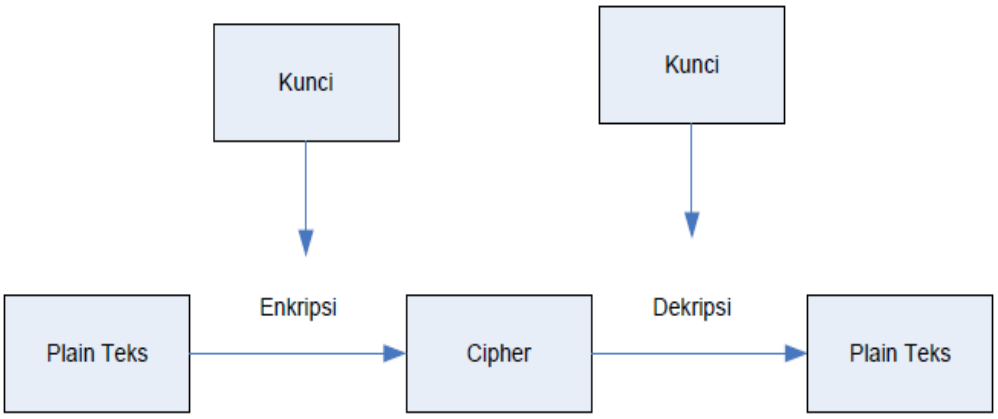
2.1.3 Mekanisme Kriptografi

Suatu sistem kriptografi (kriptosistem) bekerja dengan cara menyandikan suatu pesan menjadi suatu kode rahasia yang dimengerti oleh pelaku sistem informasi saja. Pada dasarnya mekanisme kerja semacam ini telah dikenal sejak zaman dahulu. Dalam era teknologi informasi sekarang ini, mekanisme yang sama masih digunakan tetapi tentunya implementasi sistemnya berbeda. Berikut ini beberapa istilah yang umum digunakan dalam pembahasan kriptografi (Munir, 2011).

1. *Plaintext* (*message*) merupakan pesan asli yang ingin dikirimkan dan dijaga keamanannya.
2. *Chipertext* merupakan pesan yang telah dikodekan atau disandikan sehingga siap untuk dikirimkan.
3. *Cipher* merupakan algoritma matematis yang digunakan untuk proses penyandian *plaintext* menjadi *ciphertext*.
4. Enkripsi (*encryption*) merupakan proses yang dilakukan untuk menyandikan *plaintext* sehingga menjadi *chipertext.*
5. Dekripsi (*decryption*) merupakan proses yang dilakukan untuk memperoleh kembali *plaintext* dari *ciphertext*.
6. Kriptosistem merupakan sistem yang dirancang untuk mengamankan suatu sistem informasi dengan memanfaatkan kriptografi.

Prosesnya pada dasarnya sangat sederhana. Sebuah *plaintext* (m) akan dilewatkan pada proses enkripsi (E) sehingga menghasilkan suatu *ciphertext* (c). Kemudian untuk memperoleh kembali *plaintext*, maka *ciphertext* (c) melalui proses dekripsi

(D) yang akan menghasilkan kembali *plaintext* (m). Kriptografi modern selain memanfaatkan algoritma juga menggunakan kunci (*key*) untuk memecahkan masalah tersebut dengan (e)= kunci enkripsi dan (d) = kunci dekripsi. Mekanisme kriptografi seperti ini dinamakan kriptografi berbasis kunci. Dengan demikian kriptosistemnya akan terdiri atas algoritma dan kunci, beserta segala *plaintext* dan *ciphertext*-nya. Proses enkripsi dan dekripsi dilakukan dengan menggunakan kunci ini yang digambarkan seperti pada Gambar 2.1 (Mollin, 2007).



Gambar 2.1 Kriptografi berbasis kunci (Mollin, 2007)

Persamaan matematisnya seperti berikut:

$$Ee(m) = c \quad \text{..... (1)}$$

$$Dd(c) = m \quad \text{..... (2)}$$

$$Dd(Ee(m)) = m \quad \text{..... (3)}$$

2.1.4 Kriptografi Klasik dan Modern

Kriptografi berdasarkan sejarahnya dibagi atas dua yaitu kriptografi klasik dan modern (Mollin, 2007).

1. Kriptografi Klasik

   Algoritma kriptografi yang digunakan pada zaman sebelum komputer ada disebut algoritma klasik yang berbasiskan karakter. Proses persandian dilakukan pada setiap karakter pesan. Semua algoritma klasik termasuk ke dalam sistem kriptografi simetris dan digunakan jauh sebelum sistem dan digunakan jauh sebelum sistem kriptografi publik ditemukan (Mollin, 2007).

   Kriptografi klasik dibagi menjadi dua yaitu *Substitution Ciphers* (*Cipher* Substitusi) dan *Transposition Ciphers* (Transposisi *cipher*) (Sadikin, 2012).

   a. *Substitution Ciphers (Cipher Substitusi)*

      Sistem kriptografi yang menggunakan operasi substitusi disebut dengan sistem substitusi. Prinsip utama *cipher* substitusi yaitu mengganti munculnya sebual simbol dengan simbol lain. Sistem kriptografi yang berbasis substitusi diantaranya adalah *Shift Cipher* (*Caisar Cipher*), *Vigenère Cipher* dan *Hill Cipher* (Sadikin, 2012).

   b. *Transposition Ciphers (Cipher Transposisi)*.

      Algoritma ini melakukan *transpose* terhadap rangkaian karakter di dalam teks. Pada *Cipher* transposisi huruf-huruf pada *plaintext* tetap sama hanya urutannya yang diubah. *Cipher* transposisi dikenal dengan metode permutasi atau pengacakan (*scrambling*) karena *transpose* setiap karakter di dalam teks sama dengan mempermutasikan karakter-karakter tersebut. *Cipher* Transposisi dapat dikelompokkan ke dalam dua jenis yaitu sandi transposisi *columnar* dan sandi permutasi (Sadikin, 2012).

2. Kriptografi Modern

Algoritma kriptografi modern umumnya beroperasi dalam mode *bit* ketimbang mode karakter (seperti yang dilakukan pada *cipher* substitusi atau *cipher* transposisi dari algoritma kriptografi klasik). Algoritma kriptografi modern dibuat sedmikian kompleks sehingga kripnatalis sangat sulit memecahkan *ciphertext* tanpa mengetahui kunci (Munir, 2006).

Perkembangan algoritma kriptografi modern berbasis *bit* didorong oleh penggunaan komputer digital yang merepresentasikan data dalam bentuk biner. Algoritma kriptografi yang beroperasi dalam mode *bit* dapat dikelompokkan menjadi dua kategori (Munir, 2006).

a. *Cipher* aliran (*stream cipher*)

Algoritma kriptografi beroperasi pada *plaintext* dan *ciphertext* dalam bentuk *bit* tunggal, yang dalam hal ini rangkaian *bit* dienkripsikan atau didekripsikan *bit* per *bit*.

b. *Cipher* blok (*block cipher*)

Algoritma kriptografi beroperasi pada *plaintext* dan *ciphertext* dalam bentuk blok *bit*, yang dalam hal ini rangkaian *bit* dibagi menjadi blok-blok *bit* yang panjangnya sudah ditentukan sebelumnya. Misalnya panjang blok adalah 64 *bit*, maka itu berarti algoritma enkripsi memperlakukan 8 karakter setiap kali penyandian (1 karakter = 8 *bit* dalam pengkodean ASCII).

2.1.5 Kriptografi Kunci Simetris dan Asimetris

Selain berdasarkan sejarah yang membagi kriptografi menjadi kriptografi klasik dan kriptografi modern, maka berdasarkan kunci yang digunakan untuk enkripsi

dan dekripsi, kriptografi dapat dibedakan lagi menjadi Kriptografi Kunci Simetris (*Symmetric-key Cryptography*) dan Kriptografi Kunci Asimetris (*Asymmetric-key Cryptography*) (Kristanto, 2003).

1. Kriptografi Kunci Simetris

   Algoritma ini mengasumsikan pengirim dan penerima pesan sudah berbagi kunci yang sama sebelum bertukar pesan. Algoritma simetris (*symmetric algorithm*) adalah suatu algoritma dimana kunci enkripsi yang digunakan sama dengan kunci dekripsi, sehingga algoritma ini disebut juga sebagai *single-key algorithm*. Keamanan dari pesan yang menggunakan algoritma ini tergantung pada kunci. Jika kunci tersebut diketahui oleh orang lain maka orang tersebut akan dapat melakukan enkripsi dan dekripsi terhadap pesan. Berikut ini adalah ilustrasi penggunaan algoritma simetris yang digambarkan pada Gambar 2.2 (Kristanto, 2003).

Gambar 2.2 Proses algoritma simetris (Kristanto, 2003)

   Algoritma yang memakai kunci simetris diantaranya adalah *Data Encryption Standard* (DES), RC2, RC4, RC5, RC6, *International Data Encrytion Algorithm* (IDEA), *Advanced Encryption Standard* (AES) ,*One Time Pad* (OTP) dan lain lain.

Sebelum melakukan pengiriman pesan, pengirim dan penerima harus memilih suatu kunci tertentu yang sama untuk dipakai bersama, dan kunci ini haruslah rahasia bagi pihak yang tidak berkepentingan sehingga algoritma ini disebut juga algoritma kunci rahasia (*secret-key algorithm*).

Kelebihan algoritma simetris yaitu:

a. Kecepatan operasi lebih tinggi bila dibandingkan algoritma asimetris,
b. Kecepatannya yang cukup tinggi, dapat digunakan pada sistem *real-time.*

Kelemahan algoritma simetris yaitu:

a. Untuk tiap pengiriman pesan dengan pengguna yang berbeda dibutuhkan kunci yang berbeda juga, sehingga akan terjadi kesulitan dalam manajemen kunci tersebut,
b. Permasalahan dalam pengiriman kunci itu sendiri yang disebut "*key distribution problem*".

2. Kriptografi Asimetris

Kriptografi Asimetris juga disebut dengan kriptografi kunci-publik. Dengan arti kata kunci yang digunakan untuk melakukan enkripsi dan dekripsi adalah berbeda. Pada kriptografi jenis ini, setiap orang yang berkomunikasi mempunyai sepasang kunci yaitu kunci umum dan rahasia.

a. Kunci umum (*publik key*)

   Kunci umum adalah kunci yang boleh semua orang tahu. Pengirim mengenkripsi pesan dengan menggunakan kunci publik si penerima pesan.

b. Kunci rahasia (*private key*)

Kunci rahasia adalah kunci yang dirahasiakan atau diketahui oleh satu orang saja. Hanya penerima pesan yang dapat mendekripsi pesan, karena hanya penerima pesan yang mengetahui kunci *private*-nya sendiri. Contoh algoritma kriptografi kunci-publik diantaranya RSA, Elgamal, DSA, dll. Berikut ilustrasi penggunaan algoritma asimetris yang digambarkan pada Gambar 2.3.

Gambar 2.3 Proses algoritma asimetris (Kristanto, 2003)

Kelebihan algoritma asimetris yaitu:

a. Keamanan pada distribusi kunci dapat lebih baik,
b. Manajemen kunci yang lebih baik karena jumlah kunci yang lebih sedikit.

Kelemahan algoritma asimetris yaitu:

a. Kecepatan yang lebih rendah bila dibandingkan dengan algoritma simetris,
b. Untuk tingkat keamanan sama, kunci yang digunakan lebih panjang dibandingkan dengan algoritma simetris.

2.2 *Database* dan DBMS

Database atau basis data adalah kumpulan informasi yang disimpan di dalam komputer secara sistematis sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi dari basis data tersebut. Perangkat lunak yang digunakan untuk mengelola dan memanggil *query* basis data disebut Sistem Manajemen Basis Data (*Database Management System* atau DBMS) (Fathansyah, 2007).

Ada beberapa definisi umum yang digunakan dalam basis data, antara lain sebagai berikut (Riyanto, 2011).

1. Entitas

   Entitas adalah orang, tempat, kejadian atau konsep yang informasinya direkam.

2. Atribut

   Atribut biasa juga disebut juga data elemen, *data field*, atau *data item* yang digunakan untuk menerangkan suatu entitas dan mempunyai harga tertentu.

3. Nilai data (*data value*)

   Nilai data adalah data aktual atau informasi yang disimpan pada tiap data, elemen, atau atribut.

4. Tabel (*file)*

   Tabel adalah kumpulan *record* sejenis yang mempunyai panjang elemen yang sama, atribut yang sama, namun berbeda nilai datanya.

5. *Record (tuple)*

   Kumpulan elemen yang saling berkaitan menginformasikan tentang suatu entitas secara lengkap.

Terdapat dua macam basis data antara lain (Simarmata, 2007).

1. Basis Data Analitis

   Basis data analitis (atau yang dikenal dengan OLAP-*On Line Analytical Processing*), terutama sekali statis, hanya membaca basis data dan menyimpan data historis, yakni arsip yang digunakan untuk analisis.

   Sebagai contoh, suatu perusahaan mungkin menyimpan arsip penjualan lebih dari sepuluh tahun yang lalu pada basis data analitis dan menggunakan basis data untuk meneliti strategi pemasaran saat berhubungan dengan demografis.

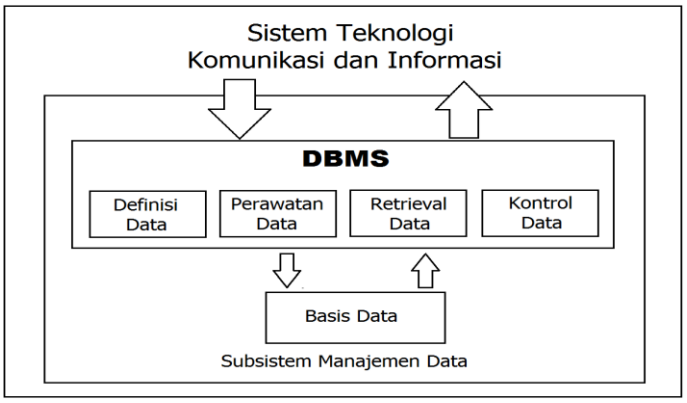
2. Basis Data Operasional

   Basis data operasional (OLTP - *On line Transaction Processing*) di sisi lain digunakan untuk mengatur *bit* data yang dinamis. Basis data tersebut merupakan jenis basis data yang mengizinkan untuk melakukan lebih dari sekedar pandangan data yang diarsip. Basis data operasional mengizinkan untuk memodifikasi data (menambahkan, mengubah, atau menghapus data).

DBMS merupakan suatu sistem perangkat lunak kompleks yang mengatur permintaan dan penyimpanan data ke dan dari *disk*. DBMS menyediakan keamanan (*security*), privasi (*privacy*), integritas (*integrity*) dan kontrol konkurensi (*concurrency control*). DBMS dimasukkan ke dalam 4 kelompok utama fungsi DBMS seperti yang terlihat pada Gambar 2.4 (Simarmata, 2007).

1. *Data Definiton* – penjelasan struktur data baru untuk suatu basis data, pemindahan struktur data dari basis data, serta pemodifikasian struktur dari data yang ada.

2. *Data Maintenance* – memasukkan data baru ke dalam struktur data yang ada, memperbarui data di dalam struktur data yang ada dan menghapus data dari struktur data yang ada .
3. *Data Retrieval* – peng-*query*-an data yang ada oleh pengguna akhir dan pengekstrakan data sebagai penggunaan oleh program aplikasi.
4. *Data Control* – menciptakan dan mengawasi pengguna basis data, pembatasan akses untuk data di dalam basis data, dan pengawasan kinerja basis data.



Gambar 2.4 Fungsi-fungsi DBMS (Simarmata, 2007)

2.3 Algoritma RC4

RC4 merupakan jenis aliran kode yang berarti operasi enkripsinya dilakukan per karakter 1 *byte* untuk sekali operasi. Algoritma kriptografi *Rivest Code* 4 (RC4) merupakan salah satu algoritma kunci simetris dibuat oleh RSA *Data Security Inc* (RSADSI) yang berbentuk *stream cipher*. Algoritma ini ditemukan pada tahun 1987 oleh Ronald Rivest dan menjadi simbol keamanan RSA (merupakan singkatan dari tiga nama penemu yaitu Rivest, Shamir, dan Adleman) (Ariyus, 2008).

Secara garis besar algoritma dari metode RC4 *Stream Cipher* ini terbagi menjadi dua bagian, yaitu *Key Setup* atau *Key Schedulling Algorithm* (KSA) dan *Stream*

Generation atau *Pseudo Random Generation Algorithm* (PRGA) dan proses XOR dengan *stream* data.

Berikut ini proses bagian dari metode RC4 menurut Piansyah (2008).

1. *Key Setup* atau *Key Schedulling Algorithm* (KSA)

   Pada bagian ini terdapat tiga tahapan proses di dalamnya yaitu:

   a. Inisialisasi S-Box

      Pada tahapan ini S-Box akan diisi dengan nilai sesuai indeksnya untuk mendapatkan S-Box awal. Algoritmanya adalah sebagai berikut :

- untuk i = 0 hingga i = 255 lakukan,
- isikan s dengan nilai i,
- tambahkan i dengan 1, kembali ke langkah 2.

Dari algoritma tersebut akan didapat urutan nilai S-Box yang direpresentasikan dalam Gambar 2.5.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255

Gambar 2.5 Contoh Hasil inisialisasi SBox dengan memasukkan data urut pada setiap blok (Piansyah, 2008)

b. Menyimpan kunci dalam *Key Byte Array*

Pada tahapan ini, kunci (*key*) yang akan digunakan untuk mengenkripsi atau dekripsi akan dimasukkan ke dalam *array* berukuran 256 secara berulang sampai seluruh *array* terisi. Algoritmanya adalah sebagai berikut:

- isi j dengan 1,
- untuk i = 0 hingga i = 255 lakukan,
- jika j > panjang kunci maka,
- j diisi dengan nilai 1,
- akhir jika,
- isi k ke i dengan nilai ascii karakter kunci ke j,
- nilai j dinaikkan 1,
- tambahkan i dengan 1, kembali ke 2.

Dari algoritma tersebut akan didapatkan urutan *array key* misalkan sebagai berikut untuk kunci dengan panjang 8 karakter dengan urutan karakter dalam ASCII “109 97 104 98 98 97 104”.

109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104

Gambar 2.6 Contoh Hasil penyimpanan kunci pada *key byte array* (Piansyah, 2008)

c. Permutasi pada S-Box

Pada tahapan ini akan dibangkitkan sebuah nilai yang akan dijadikan aturan untuk permutasi pada S-Box. Algoritmanya adalah sebagai berikut:

- isi nilai j dengan 0,
- untuk i = 0 hingga i = 255 lakukan,
- isi nilai j dengan hasil operasi (j + s (i) + k (i)) mod 256,
- tukar nilai s (i) dan s (j),
- tambahkan i dengan 1, kembali ke 2.

Dari algoritma tersebut akan diperoleh nilai S-Box yang telah mengalami proses tranposisi sehingga urutannya diacak misalkan untuk kunci pada contoh tersebut sebagai berikut.

109	26	57	76	157	245	52	49	181	31	45	103	84	240	73	192
61	235	28	98	3	205	160	128	130	59	22	38	92	122	137	30
247	121	4	135	156	148	231	170	228	16	68	208	159	53	210	20
176	168	220	214	136	19	87	42	120	219	27	169	183	116	200	96
204	110	64	216	89	222	44	126	105	213	164	188	23	47	194	33
209	65	162	142	107	195	119	80	166	238	237	72	246	102	243	241
35	40	77	215	141	100	74	115	146	56	189	163	113	55	67	229
236	185	83	46	18	140	118	50	233	248	99	8	172	203	54	132
13	124	152	252	151	153	147	139	179	190	82	154	224	161	86	0
34	149	29	225	78	14	48	158	25	104	150	184	218	187	95	193
79	43	155	223	143	232	177	251	66	114	202	10	9	88	6	226
239	250	178	71	106	51	11	60	125	242	17	90	197	91	249	173
94	15	201	101	108	234	227	41	75	117	165	174	230	138	207	62
70	244	129	145	182	131	112	7	171	253	111	198	134	180	167	211
39	255	2	1	32	85	254	21	63	217	212	175	12	93	123	191
36	186	37	133	144	24	69	81	206	5	196	199	97	221	58	127

Gambar 2.7 Contoh Hasil pengacakan S-Box berdasarkan *key* yang digunakan (Piansyah, 2008)

2. *Stream Generation* / *Pseudo Random Generation Algorith*m (PRGA)

Pada tahapan ini akan dihasilkan nilai *pesuodorandom* yang akan dikenakan operasi XOR untuk menghasilkan *ciphertext* ataupun sebaliknya yaitu untuk menghasilkan *plaintext*. Algoritmanya adalah sebagai berikut:

- isi indeks i dan j dengan nilai 0,
- untuk i=0 hingga i=panjang *plaintext,*
- isi nilai i dengan hasil operasi (i+1) mod 256,
- isi nilai j dengan hasil operasi (j+s(i)) mod 256,
- tukar nilai s(i) dan s(j),
- isi nilai t dengan hasil operasi (s(i)+(s(j) mod 256))mod 256,
- isi nilai y dengan nilai s(t),
- nilai y dikenakan operasi XOR terhadap *plaintext*,
- tambahkan i dengan 1, kembali ke 2.

Dengan demikian akan dihasilkan *cipher*text dengan hasil XOR antar *sream key* dari S-Box dan *plaintext* secara berurutan.

2.4 PHP

PHP singkatan dari *Hypertext PreProcesso*r merupakan bahasa pemograman *scripting* yang bersifat *open source*. Program ini bersifat *server side*, artinya tanpa adanya *server* yang berjalan disisinya *script* program PHP tidak dapat dijalankan. Dengan menggunakan program ini dapat dibuat sebuah tampilan *web* yang terlihat dinamis dan fleksibel serta dapat dijalankan pada berbagai *platform* termasuk Linux dan Windows. Pada umumnya program PHP selalu berjalan menggunakan program apache sebagai *web server*-nya. Kedua program tersebut sama-sama memiliki lisensi GNU/GPL (*General Public License*) dan dapat diperoleh secara gratis pada alamat http://www.php.net (Nugroho, 2005).

PHP adalah bahasa *script* yang ditanam di sisi *server* memiliki kemampuan untuk memisahkan kode dari HTML. Beberapa kelebihan dari PHP sebagai berikut (Prasetyo, 2012).

1. Kesederhanaan, mudah pelajari karna banyak referensi serta bisa membuat *website* dinamis.
2. PHP bersifat *open source*, oleh karena itu PHP mudah didapatkan dan tersedia secara versi-versi baru dalam jangka waktu yang cepat.
3. Stabilitas dan Kompatibilitas, PHP stabil di berbagai sistem operasi seperti Linux dan Macs selain itu PHP juga terintergrasi secara baik dengan berbagai macam *webserver* termasuk 2 yang paling popular yaitu IIS dan Apache.
4. Kemampuan proses cepat dalam menampilkan halaman *web* serta mampu berintraksi banyak *database*.

Dalam penulisan *syntax* kode PHP diawali dengan tanda "<?php" (atau cukup singkat bisa ditulis seperti ini "<?" dan ditutup dengan dengan menambahkan tanda tutup kurung seperti ini "?>" di akhir blok kodenya. Pada setiap baris instruksi program di akhir dengan tanda titik koma (;). Selain itu banyak pengunaan tanda kurung dalam penulisan *syntax* kode PHP yang sering dilakukan adalah untuk memanggil fungsi. Secara sederhana, setiap fungsi PHP akan berbentuk seperti ini print ( ); (Prasetyo, 2012).

Sebuah file PHP biasanya berisi *tag* html, seperti sebuah *file* html, dan beberapa kode PHP *scripting*. Di bawah ini, terdapat contoh sederhana *script* PHP yang mengirim teks "*Hello World*" ke *browser* :

```
<html>
<body>
<?php
echo "Hello World";
?>
</body>
</html>
```

Setiap baris kode PHP harus diakhiri dengan titik koma (;). Titik koma adalah pemisah dan digunakan untuk membedakan satu set instruksi dari yang lain. Ada dua pernyataan dasar untuk teks *output* dengan PHP yaitu *echo* dan *print*. Pada contoh tersebut menggunakan pernyataan *echo* untuk output teks "*Hello W*or*ld*". *File* harus memiliki ekstensi PHP. Jika *file* tersebut memiliki ekstensi HTML, kode PHP tidak akan dieksekusi (Prasetyo, 2012).

PHP mengenal adanya variabel dan *operator* sama seperti bahasa pemrograman lainnya. Variabel diperlukan untuk mendeklarasikan dan menganalisis suatu nilai. Adapun *operator* diperlukan untuk melakukan perhitungan-perhitungan matematis, memberi nilai ke dalam variabel tertentu dan membandingkan dua nilai variabel (Kustiyahngningsih dan Devie, 2011).

1. Variabel

   Variabel digunakan untuk menyimpan data sementara dan nilainya bisa berubah-ubah setiap kali program dijalankan. Data yang disimpan dalam variabel akan hilang setelah program selesai dieksekusi. Variabel dimulai dengan tanda $ dan diikuti dengan nama variabel tanpa memandang apakah data tersebut *integer*, *real* maupun *string*. PHP akan secara otomatis mengkoversi data menurut tipenya.

   Nama variabel dapat berupa kombinasi antara huruf alphabet, angka dengan panjang maksimal 32 karakter. Nama variabel bersifat *case-sensitive* atau mengenal perbedaan huruf besar dan huruf kecil. Nama variabel hanya bisa diawali dengan huruf atau garis bawah, karakter selanjutnya dapat huruf, angka

maupun garis bawah. Untuk dapat menggunakan variabel, perlu dilakukan dua hal yaitu sebagai berikut:

a. Deklarasi, yaitu memperkenalkan variabel ke program. Dalam penulisan *script* PHP, deklarasi string digabung dengan inisialisasi.
b. Inisialisasi, yaitu memberi suatu nilai untuk pertama kalinya kepada suatu variabel.

2. Tipe Data

Variabel PHP mempunyai beberapa tipe data, yaitu sebagai berikut.

a. *Integer* (bilangan bulat)

Meliputi semua bilangan bulat yang berada pada *range* 2,147,483,684 sampai +2,147,483,647. Jika suatu nilai berada di luar *range* tersebut maka PHP akan secara otomatis mengonversi menjadi *floating point*. *Integer* dapat ditulis dalam bentuk:

1. Bilangan desimal, contoh: $a=1234;
2. Bilangan desimal negatif, contoh: $a=-123;
3. Bilangan heksadesimal, contoh: $a=0x1A; sama dengan bilangan desimal 26;
4. Bilangan oktal, contoh: $a=0123, sama dengan bilangan desimal 83.

b. *Floating Point*

Nilai maksimal sebuah bilangan *floating point* adalah ~1.8e308 dengan ketelitian mencapai 14 digit desimal. Bilangan *floating point* dapat dideklarasikan menggunakan sintaks berikut.

```
$a=1.234;
```

c. *String*

Setiap tipe data *string* selalu diapit oleh tanda petik tunggal (‘’) maupun ganda (“”).

Contoh: `$string1='Belajar PHP';`

Perbedaan antara petik tunggal dan ganda adalah “jika pada petik tunggal maka pada string tidak dapat dimasukkan suatu variabel dan “*escape sequence handling*”. Karakter \ digunakan untuk menentukan karakter khusus seperti pada Tabel 2.1.

Tabel 2.1 *Escape Sequence*

Sequence	**Keterangan**
\n	Membuat baris baru
\r	*Carriage*
\t	Tab Horizontal
\‘	Petik tunggal
\“	Petik ganda
\$	Tanda dolar
\\	*Backslash*

3. Operasi

Operator adalah suatu simbol yang dipakai untuk memanipulasi nilai suatu variabel. Variabel yang nilainya dimodifikasi oleh *moderator* disebut *operand*. Misalnya 3-2, 3 dan 2 adalah *operand* dan – adalah operator.

Dalam mengeksekusi suatu perintah ada urutan operator yang akan dilakukan oleh PHP. Urutan operator akan menentukan hasil akhir dari sebuah eksekusi. Contoh: 5+2*3 akan menghasilkan nilai 11 (2*3 akan dihitung terlebih dahulu kemudian ditambahkan dengan 5). Hal ini karena operator * memiliki prioritas

lebih tinggi daripada operator +. Urutan operator dalam PHP dari urutan tertinggi sampai terendah dapat dilihat pada Tabel 2.2.

Tabel 2.2 Operator PHP

Prioritas	Operator
Tertinggi	() {}
.	‘ ! + + - $ &
.	* / %
.	+ -
.	< > <=> =
.	= =! =
.	&
.	^
.	!
.	&&
.	| |
.	= += -= *= /= &= |= ^= .=
.	AND (&&)
.	XOR (| |)
Terendah	OR

Macam-macam operator yaitu:

a. Operator Aritmatika

Operator yang digunakan untuk melakukan perhitungan matematika. Contoh:

$nilai = 5 + 3; Tipe-tipe operator aritmatika, diantaranya adalah:

+ : penjumlahan

- : pengurangan

* : perkalian

/ : pembagian

% : module (sisa pembagian)

b. Operator *Comparison*

Operator ini digunakan untuk membandingkan nilai dari dua *operand*. Hasil perbandingan dinyatakan dalam nilai *Boolean*. Beberapa jenis relasional operator dapat dilihat pada Tabel 2.3.

Tabel 2.3 Operator *Comparison*

Operator	Keterangan
==	Memeriksa apakah operand kanan bernilai sama dengan operand kiri
>	Memeriksa apakah operand kiri bernilai lebih besar daripada operand kanan
<	Memeriksa apakah operand kiri bernilai lebih kecil daripada operand kanan
>=	Memeriksa apakah operand kiri bernilai lebih besar atau sama dengan daripada operand kanan
<=	Memeriksa apakah operand kiri bernilai lebih kecil atau sama dengan daripada operand kanan
!=	Memeriksa apakah operand kiri bernilai tidak sama dengan operand kanan

c. Operator Logika

Operator ini digunakan untuk membandingkan dua nilai variabel yang bertipe *Boolean*. Hasil yang didapat penggunaan logical operator adalah *Boolean*. Beberapa jenis operator logika dapat dilihat Tabel 2.4.

Tabel 2.4 Operator Logika

Operand 1	Operand 2	AND (&&)	OR (| |)	XOR
Salah	Salah	Salah	Salah	Salah
Salah	Benar	Salah	Benar	Benar
Benar	Salah	Salah	Benar	Benar
Benar	Benar	Benar	Benar	Salah

d. Operator *Assigment*

Operator *assignment* digunakan untuk memberikan atau mengisi nilai ke dalam variabel tertentu. Operator yang digunakan adalah "=" yang berarti *operand* kiri diberi nilai sama seperti *operand* kanan.

2.5 XAMPP

XAMPP adalah perangkat lunak bebas, yang mendukung banyak sistem operasi, merupakan kompilasi dari beberapa program. Fungsinya adalah sebagai *server* yang berdiri sendiri (*localhost*), yang terdiri atas program Apache HTTP *Server*, MySQL *database*, dan penerjemah bahasa yang ditulis dengan bahasa pemrograman PHP dan Perl. Nama XAMPP merupakan singkatan X (empat sistem operasi apapun), Apache, MySQL, PHP, dan Perl. Program ini tersedia dalam GNU *General Public License* dan bebas, merupakan *web server* yang mudah digunakan yang dapat melayani tampilan halaman *web* yang dinamis. Untuk mendapatkannya dapat men-*download* langsung dari *web* resminya. Mengenal bagian XAMPP yang biasa digunakan pada umumnya (Riyanto, 2011).

1. *htdocs* adalah folder tempat meletakkan berkas-berkas yang akan dijalankan, seperti berkas PHP, HTML, dan skrip lain.
2. PHPMyAdmin merupakan bagian untuk mengelola basis data MySQL yang ada dikomputer. Untuk membukanya, buka *browser* lalu ketikkan alamat http://localhost/PhpMyAdmin, maka akan muncul halaman PHPMyAdmin.
3. *Control Panel*, yang berfungsi untuk mengelola layanan (*service*) XAMPP seperti menghentikan (*stop*) layanan, ataupun memulai (*start*).

2.6 MySQL

Database MySQL merupakan sistem manajemen basis data SQL yang sangat terkenal dan bersifat *Open Source*. MySQL dibangun, didistribusikan dan didukung oleh MySQL AB. MySQL AB merupakan perusahaan komersial yang dibiayai oleh pengembang MySQL. Software MySQL mempunyai dua macam lisensi. Lisensi pertama bersifat open source dengan menggunakan GNU *General Public License* dan lisensi kedua berupa lisensi komersial standar (*Standar Commercial License*) yang dapat dibeli dari MySQL AB (Kustiyahningsih dan Devie, 2011).

MySQL merupakan *software* yang tergolong sebagai DBMS (*Database Management System*) yang bersifat *open source*. *Open source* menyatakan bahwa *software* ini dilengkapi dengan *source code* (kode yang dipakai untuk membuat MySQL), selain tentu saja bentuk *executable*-nya atau kode yang dapat dijalankan secara langsung dalam sistem operasi, dan bisa diperoleh dengan cara men-*download* (mengunduh) di internet secara gratis (Kadir, 2008).

Sebagai software DBMS, MySQL memiliki sejumah fitur seperti yang dijelaskan di bawah ini (Kadir, 2008).

1. *Multiplatform*

   MySQL tersedia pada beberapa *platform* (Windows, Linux, Unix, dan lain-lain).

2. Andal, cepat, dan mudah digunakan

   MySQL tergolong sebagai *database server* (*server* yang melayani permintaan terhadap *database*) yang andal, dapat menangani *database* yang besar dengan kecepatan tinggi, mendukung banyak sekali fungsi untuk mengakses *database*, dan sekaligus mudah untuk digunakan.

3. Jaminan keamanan akses

   MySQL mendukung pengamanan *database* dengan berbagai kriteria pengaksesan. Sebagai gambaran, dimungkinkan untuk mengatur *user* tertentu agar bisa mengakses data yang bersifat rahasia, sedangkan *user* lain tidak boleh. MySQL juga mendukung konektivitas ke berbagai *software*. MySQL juga bisa diakses melalui aplikasi berbasis *web*, misalnya dengan menggunakan PHP.

4. Dukungan SQL

   Seperti tersirat dalam namanya, MySQL mendukung perintah SQL (*Structured Query Language*). Sebagaimana diketahui, SQL merupakan standar dalam pengaksesan *database* relasional. Pengetahuan akan SQL akan memudahkan siapa pun untuk menggunakan MySQL.

2.6.1 Tipe Data MySQL

Data yang terdapat dalam sebuah tabel berupa *field-field* yang berisi nilai dari data tersebut. Nilai data dalam *field* memiliki tipe sendiri-sendiri. MySQL mengenal beberapa tipe data *field* yaitu sebagai berikut (Kustiyahningsih dan Devie, 2011).

1. Tipe data *numeric*

   Tipe data *numeric* dibedakan dalam dua macam kelompok, yaitu *integer* dan *floating point*. *Integer* digunakan untuk data bilangan bulat sedangkan *floating point* digunakan untuk bilangan *decimal*. Tipe data *numeric* selengkapnya dapat dilihat pada Tabel 2.5.

   Tabel 2.5 Tipe data numerik

Tipe Data	Kisaran Nilai
TINYINT	(-128) s.d 127 atau (0 s.d 255)
SMALLINT	(32768) s.d 32767 atau (0 s.d 65535)
MEDIUMINT	(-3888608) s.d 8388607 atau (0 s.d 16777215)
INT, INTEGER	(-2147683648) s.d (21447683647) atau (0 s.d 4294967295)
FLOAT	(-3.4E+38) s.d (-1.17E-38), 0 dan (1.175E-38 s.d 3.4e+38)
DOUBLE	(-1.79E+308) s.d (-2.225E-308), 0 dan (2.225E-308 s.d 1.79E+308)

2. Tipe data *string*

   String adalah rangkaian karakter. Tipe-tipe data yang termasuk dalam tipe data *string* dapat dilihat pada Tabel 2.6.

   Tabel 2.6. Tipe data string

Tipe Data	Kisaran Nilai
CHAR	1 sampai 255 karakter
VARCHAR	1 sampai 255 karakter
TINYTEXT	1 sampai 255 karakter
TEXT	1 sampai 65535 karakter
MEDIUMTEXT	1 sampai 16777215 karakter
LONGTEXT	1 sampai 424967295 karakter

3. Tipe data *char* dan *varchar*

   Tipe data *char* dan *varchar* pada prinsipnya sama, perbedaannya hanya terletak pada jumlah memori yang dibutuhkan untuk penyimpanannya. Memori yang dibutuhkan untuk tipe data *char* bersifat statis, besarnya bergantung pada berapa jumlah karakter yang ditetapkan pada saat *field* tersebut dideklarasikan. Pada tipe data *varchar* besarnya memori penyimpanan tergantung pada jumlah karakter ditambah 1 *byte.*

4. Tipe data *date*

   Untuk tanggal dan jam, tersedia tipe-tipe data *field* berupa *DATETIME, DATE, TIMESTAMP, TIME dan YEAR*. Masing-masing tipe mempunyai kisaran nilai tertentu. MySQL akan memberikan peringatan kesalahan (*err*or) apabila tanggal atau waktu yang dimasukkan salah. Kisaran nilai dan besar memori penyimpanan yang diperlukan untuk setiap tipe dapat dilihat pada Tabel 2.7.

Tabel 2.7 Tipe data tanggal

Tipe Data	Kisaran Nilai	Memori
DATETIME	1000-01-01 00:00 sampai 9999-12-31 23:59:59	3 byte
DATE	1000-01-01 sampai 9999-12-31	8 byte
TIMESTAMP	1970-01-01 00:00:00 sampai 2037	4 byte
TIME	-839:59:59 sampai 838:59:59	3 byte
YEAR	1901 sampai 2155	1 byte

2.6.2 Operator MySQL

MySQL mendukung penggunaan operator-operator diantaranya sebagai berikut (Kadir, 2008).

1. Operator Aritmatika

   Operator aritmatika mempunyai derajat pengerjaan yang berbeda-beda. Urutan prioritas pengerjaan adalah dari yang tertinggi hingga yang terendah. Sejumlah operator aritmatika yang dapat digunakan untuk melakukan perhitungan sacara numeris yang dapat dilihat pada Tabel 2.8.

Tabel 2.8 Operator aritmatika

Operator	Keterangan
*	Perkalian
/	Pembagian
%	Sisa pembagian
+	Penjumlahan
-	Pengurangan
DIV	Pembagian bulat

2. Operator Relasional

   Yang dimaksud dengan operator relasional adalah operator yang digunakan untuk melakukan pembandingan antara dua buah nilai. Operator relasional dapat dilihat pada Tabel 2.9.

Tabel 2.9 Daftar operator relasional

Operator	Keterangan
=	Sama dengan
>	Lebih dari
<	Kurang dari
>=	Lebih dari atau sama dengan
<=	Kurang dari atau sama dengan
<>	Tidak sama dengan
<=>	*NULL-safe equal*

3. Operator *Boolean*

Operator *Boolean* atau operator logika mencakup operator OR, AND, XOR dan NOT (Kadir, 2008).

a. Operator OR

Operator OR berguna untuk melakukan *query* dengan kondisi majemuk. Bentuk penggunaan OR:

```
kondisi_1 OR kondisi_2 atau kondisi_1 || kondisi_2
```

Hasil ekspresi dengan OR berupa benar (TRUE) kalau terdapat kondisi yang bernilai benar. Khusus kalau kedua operand bernilai *NULL*, hasilnya berupa *NULL*. Hasilnya *NULL* juga kalau salah satu operand bernilai *NULL* dan yang lain bernilai nol.

Tabel 2.10 Operasi dengan OR

Kondisi_1	Kondisi_2	Hasil
Salah	Salah	Salah
Salah	Benar	Benar
Benar	Salah	Benar
Benar	Benar	Benar
NULL	Benar	Benar
Benar	*NULL*	Benar
NULL	Salah	*NULL*
Salah	*NULL*	*NULL*
NULL	*NULL*	*NULL*

b. Operator AND

Kaidah pemakaian operator AND seperti berikut :

```
kondisi_1 AND kondisi_2 atau kondisi_1 && kondisi_2
```

Operator ini memiliki sifat sebagaimana terlihat pada Tabel 2.11.

Tabel 2.11 Operasi dengan AND

Kondisi	**Benar**	**Salah**	***NULL***
Benar	Benar	Salah	*NULL*
Salah	Salah	Salah	Salah
NULL	*NULL*	Salah	Salah

c. Operator XOR

Operator hasilnya benar kalau hanya satu operator bernilai benar dan akan bernilai salah bila kedua operand bernilai benar atau bernilai salah. Sekiranya ada operand bernilai *NULL*, hasilnya selalu bernilai *NULL*. Kaidah pemakaian operator XOR seperti berikut :

```
kondisi_1 XOR kondisi_2
```

Operator ini memiliki sifat sebagaimana terlihat pada Tabel 2.12.

Tabel 2.12 Operasi dengan XOR

Kondisi_1	**Kondisi_2**	**Hasil**
Salah	Salah	Salah
Salah	Benar	Benar
Benar	Salah	Benar
Benar	Benar	Salah
NULL	Benar	*NULL*
Benar	*NULL*	*NULL*
NULL	Salah	*NULL*
Salah	*NULL*	*NULL*
NULL	*NULL*	*NULL*

d. Operator NOT

Operator NOT berguna untuk melakukan pembalikan nilai logika. Kaidah pemakaiannya:

```
NOT kondisi atau ! kondisi
```

Hasil ekspresi ditunjukkan pada Tabel 2.13.

Tabel 2.13 Operasi dengan NOT

Kondisi	Hasil
NOT Benar	Salah
NOT Salah	Benar
NOT *NULL*	*NULL*

2.6.3 Predikat MySQL

Predikat diletakkan setelah klausa *WHERE* untuk pencarian *record database* agar mendapatkan string, karakter atau range tertentu (Kustiyahningsih dan Devie, 2011).

1. *LIKE* dan *NOT LIKE*

   Like digunakan untuk mendapatkan *record* yang memenuhi sebagian kriteria pencarian yaitu mencari data yang menyerupai atau seperti. Perintah *LIKE* sering dikombinasikan dengan tanda 'persen' (%) dan '*underscore'* (_). '%' digunakan di awal atau akhir teks kriteria sedangkan '_' dimanapun diinginkan. Sintaks dasar dari *SELECT* yang melibatkan *LIKE* adalah

   ```
   Select nama FROM mahasiswa WHERE nama LIKE %ahmad%;
   ```

   Not Like merupakan lawan dari *Like*. Semua data yang masuk kriteria *Like* secara otomatis tidak masuk kriteria *Not Like*.

2. *BETWEEN*

   Digunakan untuk menyeleksi nilai-nilai yang berada dalam kisaran (*range*). Sintaks perintah *SELECT* dengen *BETWEEN* adalah

```
Select nama FROM mahasiswa WHERE ipk BETWEEN 3.00 and 3.50;
```

3. *LIMIT*

   Memungkinkan pembatasan jumlah *record* yang diambil dari *database*.

   ```
   Select nama,npm,alamat FROM mahasiswa LIMIT 100;
   ```

4. *INNER JOIN*

   INNER JOIN digunakan bersama *SELECT* untuk mengombinasikan kolom dari satu tabel dengan kolom pada tabel lain. Proses utama yang dilakukan oleh INNER *JOIN* adalah mencocokkan nilai pada *field* kunci pada kedua tabel. *INNER JOIN* mengembalikkan semua baris sebagai hasil yang memenuhi suatu kondisi.

5. *LEFT JOIN*

   Suatu *LEFT JOIN* mengembalikkan semua baris-baris sisi kiri kondisional bahkan jika tidak ada sisi kanan yang memenuhi sekalipun.

6. *RIGHT JOIN*

   Suatu *RIGHT JOIN* akan menampilkan baris-baris sisi kanan kondisional yang memenuhi atau tidak memenuhi kondisi.

7. *UNION*

   Untuk menggabung dua tabel, menempatkan dua *query* terpisah secara bersama membentuk satu tabel. *UNION* akan memberikan hasil terbaik saat menggunakan dua tabel dengan kolom serupa karena setiap kolom harus mempunyai tipe data sama.

8. *UNION ALL*

   Memilih semua baris dari setiap tabel dan mengombinasikan ke dalam satu tabel. Perbedaan antara *UNION* dan *UNION ALL* yaitu *UNION ALL* tidak akan menghapus (mengeliminir) baris-baris yang sama (*duplicate-rows*) hanya

mengambil semua baris dari semua tabel sesuai *query* yang dikirim ke *server database*.

9. *Sub-Query*

   Merupakan *query SELECT* yang ditempatkan di dalam suatu pernyataan SQL lain. Data diletakkan sebagai bagian dari *query* utama : *SELECT*, *INSERT*, *UPDATE, DELETE, SET, DO*.

2.6.4 Fungsi pada MySQL

PHP menyediakan fungsi untuk koneksi ke *database* dengan sejumlah fungsi untuk pengaturan baik menghubungkan maupun memutuskan koneksi dengan *server database* MySQL. Fungsi koneksi ke *server database* menggunakan pola yang sama yaitu *server, p*ort, *user, password* (Kustiyahningsih dan Devie, 2011).

1. MySQL_*connect (host,user,password)*

   Fungsi ini digunakan untuk membuka koneksi dengan *server* MySQL. Parameter yang digunakan sebagai berikut:

   host : nama *server* dengan *server* lokal dapat dengan menggunakan *localhost.*

   user : *user* yang terdaftar di MySQL yang digunakan untuk *login* ke *server*.

   *passwo*rd : *passwo*rd dari *user* yang melakukan koneksi.

   Contoh :

```
$host =”localhost”;
$name =”root”;
$pass =””;
$conn =MySQL connect{$host,$name,$pass};
```

2. MySQL_*select*_db (nama_*database*, nama_koneksi)

   Fungsi ini digunakan untuk memilih *database* yang akan digunakan.

   Contoh:

```
$conn = MySQL_connect(“localhost”,”root”,””);
$sukses = MySQL_select_db(“mahasiswa”,$conn);
```

3. MySQL_*query*(perintah_*query*,nama_koneksi)

   Perintah ini digunakan untuk mengirimkan *query* ke *server database* melalui *link* nama koneksi. Fungsi ini mengembalikan nilai *FALSE* baik *CREATE*, *UPDATE*, *DELETE, DROP*, dan lain lain.

   Contoh:

```
$conn=MySQL_connect("localhost","root", "");
$sql="SELECT nama, npm FROM mahasiswa  WHERE nama='$username'";
$rs = MySQL query($sql,$conn);
```

4. MySQL_*fetch_array*(hasil_*query*)

   Fungsi ini digunakan untuk melakukan pembacaan *record* hasil *query* yang dilakukan. Pembacaan ini dilakukan mulai dari *record* pertama sampai nilai terakhir. Tiap *record* dibaca, dibentuk menjadi *array* dengan indeks numerik dari 0 sampai dengan n-1 dan indeks assosiatif dengan indeks adalah nama *field* dari tabel.

5. MySQL_*fetch_row* (hasil_*query*)

   Fungsi ini mempunyai hampir sama dengan MySQL_*fetch_array* tetapi *array* hasil pembacaan data hanya menggunakan indeks numerik saja dimulai dari 0 untuk kolom pertama sampai n-1 untuk kolom terakhir hasil *query*.

6. MySQL_*result* (hasil_*query*,no_*record*,nama_*field*)

   Fungsi ini berguna untuk mengambil langsung nilai hasil *query* pada suatu baris dan kolom tertentu (satu sel) dengan menyebutkan parameter variabel hasil proses *query*, no_*record*, untuk nomor baris (dimulai dari 0) dan nama *field*.

7. MySQL_*num_field* (hasil_*query*)

   Fungsi ini berguna untuk mendapatkan jumlah *field* dari hasil *query*.

   Contoh : `$jum_kolom = MySQL_num_fields($rs);`

8. MySQL_*num_rows*(hasil_*query*)

   Fungsi ini berguna untuk mendapatkan jumlah baris dari hasil *query*. Menghasilkan suatu *array* yang berisi seluruh kolom dari sebuah baris pada suatu himpunan hasil. Mengambil data secara baris per baris. Data yang diambil dalam bentuk *array* (elemen dari *array* adalah *field-field* dari tabel data).

2.6.5 Perintah MySQL

Query sebenernya berarti permintaan atau perintah. Dengan menggunakan *query*, maka dapat melihat, mengubah dan menganalisis data dengan berbagai titik pandang yang dikehendaki. Selain itu, *query* juga dapat dipakai sebagai data bagi *form*ulir, laporan, dan halaman *web*. Pernyataan SQL dapat digolongkan atas tiga golongan yaitu sebagai berikut (Kustiyahningsih dan Devie, 2011).

1. *Data Definition Language* (DDL) yang mendefinisikan struktur suatu data. Perintah-perintah SQL yang termasuk DDL antara lain :
   a. *CREATE*: untuk membuat,
   b. *ALTER* : untuk mengubah,
   c. *DROP* : untuk menghapus.
2. *Data Manipulatin Language* (DML) yang dapat mencari (*query*) dan mengubah (*modify*) suatu tabel. Perintah-perintah SQL yang tergolong DML diantaranya :
   a. *SELECT* : untuk membaca,
   b. *INSERT* : untuk memasukkan,

c. *UPDATE*: untuk mengubah,

d. *DELETE*: untuk menghapus.

3. *Data Control Language* yang mengatur hak-hak (*privillege*) untuk seorang pemakai *database*. Perintah-perintah SQL yang tergolong DCL yaitu

   a. *GRANT*,

   b. *REVOKE*.

2.7 Apache

Apache merupakan *web server* yang saat ini telah digunakan hampir dari 60% oleh *server* di dunia, banyaknya *server* yang memanfaatkan apache sebagai *webserver*-nya disebabkan karena sifat *sotware* ini sangat fleksibel dan dapat digunakan pada berbagai *platform* seperti Linux dan Windows. Kemudian instalasi dan konfigurasi menyebabkan *web server* apache semakin digemari di kalangan pengelola *web*. *Web server* apache mampu mendukung berbagai bahasa pemrograman yang sifatnya *server side* seperti PHP, Perl, CGI, Java, WML, dan lain sebagainya. Distribusi program apache dapat di-*download* secara gratis pada alamat http://www.apache.org (Nugroho, 2005).

Web server adalah komputer digunakan untuk menyimpan dokumen-dokumen *web*, komputer ini akan melayani permintaan dokumen *web* dari kliennya. *Web browser* seperti *explorer* atau *navigator* berkomunikasi melalui jaringan (termasuk jaringan internet) dengan *web server*, menggunakan HTTP. *Browser* akan mengirimkan *request* ke *server* untuk meminta dokumen tertentu atau layanan lain yang disediakan oleh *server*. *Server* memberikan dokumen atau layanannya jika tersedia juga dengan menggunakan protokol HTTP (Kustiyahningsih dan Devie, 2011).

2.8 PHPMyAdmin

PHPMyAdmin merupakan *tools open source* yang dibuat menggunakan program PHP untuk mengakses *database* MySQL via *web*. Dengan menggunakan program ini dapat dikelola *database* MySQL dengan sangat mudah. Hampir semua operasi terhadap *database* dan perintah yang ada dapat dijalankan pada program tersebut, termasuk diantaranya menggunakan perintah *Database Defenition Language* (*DDL*), *Database Manipulation Language* (*DML*) serta *Data Control Language* (*DCL*). Program PHPMyAdmin didapatkan secara gratis dengan men-*download*-nya dari situs http://www.phpmyadmin.net atau pada http://www.sourceforge.com (Nugroho, 2005).

2.9 HTML

HTML kependekan dari *Hyper Text Markup Language*. Dokumen HTML adalah file *text* murni yang dapat dibuat dengan *edit*or teks sembarang. Dokumen ini dikenal sebagai *web page*. *File-file* HTML ini berisi instruksi-instruksi yang kemudian diterjemahkan oleh *browser* yang ada dikomputer *client* (*user*) sehingga isi informasinya dapat ditampilkan secara visual di komputer pengguna (*user*) (Kustiyahningsih dan Devie, 2011).

Elemen yang dibutuhkan untuk membuat suatu dokumen HTML dinyatakan dengan *tag-tag* sebagai berikut (Kustiyahningsih dan Devie, 2011).

1. HTML

   Setiap dokumen HTML selalu diawali dan ditutup dengan *tag* HTML.

2. HEAD

   Bagian *head* biasanya berisikan *tag TITLE*, meta *tag* dan semua *script java* atau yang lain yang akan dieksekusi di *browser*. Di bagian inilah diberikan *bookmark* untuk keperluan pencarian dengan *keyword.*

3. BODY

Bagian *body* digunakan untuk menampilkan *text,image link*, dan semua yang akan digunakan pada *web page*.

2.9.1 *Tag* Tabel

Tabel banyak digunakan pada pembuatan halaman *web* dan data memisahkan teks menjadi beberapa kolom sehingga penempatan teks lebih mudah pengaturannya. Untuk membuat tabel maka diperlukan *tag* <tabel> dan ditutup dengan *tag* </tabel>, karena tabel melibatkan banyak pengaturan dan pilihan untuk pembuatan bentuk tabel, maka guna mendukung keperluan pembuatan tabel, telah disediakan atribut-atribut yang khusus disediakan untuk keperluan pembuatan tabel, adapun atribut-atribut tersebut seperti pada Tabel 2.14 (Kustiyahningsih dan Devie, 2011).

Tabel 2.14 *Atribut-atribut untuk pembuatan tabel*

Kode	***Value***	**Keterangan**
Border	0-15	Meniadakan atau menampilan ketabalan garis-garis pada table
<tr> </tr>	-	Membuat baris tabel, pengaturan semua teks dapat dilakukan disini
<th> </th>	-	Membuat kolom judul
<td> </td>	-	Membuat kolom tabel isi
Align	LEFT CENTER RIGHT	Mengatur *horizontal alignment*
valign	TOP MIDDLE BOTTOM	Mengatur *vertical alignment*
nowrap	-	Meniadakan pindah baris baru pada saat tabel ditampilkan pada jendela *browser* yang tidak mencukupi
rowspan	N	Menggabungkan baris tabel menjadi satu (*merge cells*)
colspan	N	Menggabungkan kolom tabel menjadi satu (*merge cells*)

2.9.2 Tag Frameset

Sesuai dengan namanya *frame* yang berarti bingkai, adalah berhubungan dengan pengaturan bingkai sebagai pembentuk jendela tampilan pada *browser*. Sehingga dapatlah jendela *browser* dibagi menjadi beberapa *frame*. Ada tiga pengaturan *frame*, yaitu secara *vertical*, *horizontal* dan kombinasi dari *vertical* dan *horizontal*. *Script* HTML yang digunakan untuk pengaturan *frame* maka tidak dapat digunakan untuk membuat halaman tampilan *web*, untuk itu diperlukan file yang berisi *script* HTML lain dan sisipkan ke dalam *script frame* tersebut. Jadi bila ada dua *frame* maka diperlukan dua *file* HTML. Untuk dapat menyisipkan ketiga *file* HTML tesebut ke dalam *frame*, diperlukan nama yang jelas dan benar serta dimana letak folder/direktorinya, karena informasi tersebut digunakan di dalam *tag* <*frame*set> (Kustiyahningsih dan Devie, 2011).

2.9.3 *Form*

Fungsi *form* adalah menerima informasi atau meminta umpan balik dari *user* dan memproses informasi tersebut di *server*. *Tag* <*form*> digunakan untuk membuat *form* dalam dokumen HTML yang dapat dilihat pada Tabel 2.15 (Kustiyahningsih dan Devie, 2011).

Tabel 2.15 Fungsi elemen *tag form*

Atribut	Fungsi
Method	Metode pengiriman file data ke file tujuan (POST atau GET). Get : data akan dikirim dengan menggunakan *query string* pada URL. Protokol_transfer://nama_host/path/nama_file: lokasi informasi pada suatu *web server* Post: data akan dikirim ke *server* sebagai block data ke skrip.

	<*Form* method=”post” action =”simpan_bktamu”.php”>
Action	Aksi yang akan dilakukan jika *user* menekan tombol submit.
Name	Memberikan nama tiap masukan
Value	Memberikan nilai suatu masukan
Type	Tipe *form* yang akan digunakan

2.9.4 HTML Input Elemen

Pada saat membuat *form* bisa diletakkan kontrol-kontrol pada *form* untuk memperbolehkan masukan dari *user*. Semua kontrol biasanya diantara *tag* <*FORM*> </*FORM*> tapi bisa juga diletakkan kontrol di luar *tag* tersebut. Untuk menambahkan kontrol gunakan *tag* <input>. Berikut macam-macam komponen input (Kustiyahningsih dan Devie, 2011).

1. *Textbox*

   Untuk membuat *single line text control*. Atribut *size* digunakan untuk menentukan jumlah karakter yang bisa ditampilkan, sementara *maxlength attribute* digunakan untuk menentukan *maximum character* yang bisa dimasukan. Contoh penulisan *syntax textbox*:

```
<INPUT TYPE=”TEXT” NAME=”textbox” VALUE=”” SIZE=”20”>
```

Tabel 2.16 Atribut dan Fungsi *Textbox*

Atribut	Fungsi
type=[“text”|”password”]	Menentukan jenis *field* masukan taxt, submit, password
Name	Menentukan nama untuk *field* sehingga dapat dirujuk nantinya
Value	Memberi nilai suatu input
Size	Mengatur lebar *field* secara horizontal, berapa huruf maksimalyang dapat ditampilkan
Maxlength	Menentukan jumlah maksimum huruf (karakter) yang dapat dimasukkan

2. *Submit* dan *Reset*

   Tombol *submit* digunakan ketika *user* mengisi *form* dan ingin mengirimkan ke *server*, sedangkan tombol *reset* digunakan ketika *user* ingin menghapus atau mengosongkan semua masukan yang ditulis di *form*.

3. *Checkbox*

   Checkbox digunakan memberi beberapa pilihan kepada *user* yang dapat dilihat pada Tabel 2.17.

   Tabel 2.17 Atribut dan Fungsi *Checkbox*

Atribut	Fungsi
Checked	Untuk memberi default check
Name	Nama dari kontrol
Size	Ukuran control
Type	<input type = “checkbox”>
Value	Untuk memberikan value ke input

4. *Radio Button*

   Fungsi untuk memberi hanya satu pilihan kepada *user*. Setiap *radio button control* harus memiliki nama yang sama sehingga *user* hanya bisa memilih satu option saja. *Radio button* juga harus secara eksplisit memberikan nilai ke *attribute value*.

   Tabel 2.18 Atribut dan fungsi *Radio Button*

Atribut	Fungsi
Checked	Untuk memberi default check
Name	Nama dari kontrol
Size	Ukuran control
Type	<input type = “button”>
Value	Untuk memberikan value ke input

5. *Text Area*

   Fungsi sebagai *field* masukan untuk pengunjung (dapat menerima lebih dari satu baris teks). Biasa disebut sebagai kotak komentar. Untuk membuat *text area* gunakan *tag* <*TEXTAREA*> </*TEXTAREA*>.

   Tabel 2.19 Atribut dan fungsi *Text Area*

Atribut	Fungsi
Rows	Untuk memberi default check
Columns	Nama dari kontrol
Wrap =[“off”| “virtual”|”physical”	Ukuran kontrol

6. *Daftar Drop Down*

   Fungsi memberikan menu pilihan kepada *user* (cara kerjanya seperti *radio button* yang hanya mengizinkan *user* untuk memilih 1 pilihan saja).

2.9.5 HTML *Editor*

Secara umum, ada dua jenis HTML *Editor*, yaitu text editor dan WYSIWYG *editor* (Sampurna, 1996).

1. *Text Editor*

   Text Editor biasa digunakan oleh *user* yang sudah mahir dalam menggunakan bahasa HTML, karena melalui *editor* jenis ini dapat langsung dituliskan kode-kode HTML satu persatu, sesuai prosedur teknis yang berlaku. Untuk editor jenis ini, dapat menggunakan *notepad*.

2. WYSIWYG *Editor*

   WYSIWYG *Editor* adalah solusi bagi *user* yang belum mahir dalam menggunakan bahasa HTML. Pada jenis aplikasi ini, dapat dibangun halaman

web dengan lebih mudah, karena apa yang terlihat di layar akan sama dengan hasil yang didapatkan.

WYSIWYG adalah singkatan dari *What You See Is What You Get*. Untuk *editor* jenis ini, dapat digunakan aplikasi *Microsoft Word, Excel, Access, PowerPoint, Outlook, FrontPage* dan yang cukup popular *Macromedia Dreamweaver*.

2.10 Tabel ASCII

Kode Standar Amerika untuk Pertukaran Informasi atau ASCII (*American Standard Code for Information Interchange*) merupakan suatu standar internasional dalam kode huruf dan simbol seperti Hex dan Unicode tetapi ASCII lebih bersifat universal, contohnya 124 adalah untuk karakter "|". Kode ASCII selalu digunakan oleh komputer dan alat komunikasi lain untuk menunjukkan teks. Kode ASCII sebenarnya memiliki komposisi bilangan biner sebanyak 7 bit. Namun, ASCII disimpan sebagai sandi 8 bit dengan menambakan satu angka 0 sebagai bit signifikan paling tinggi. Berikut adalah daftar tabel ASCII yang dapat dilihat pada Tabel 2.20.

Tabel 2.20 Tabel ASCII

Decimal	Octal	Hex	Binary	Value
000	000	000	00000000	NUL (Null char \0)
001	001	001	00000001	SOH (Start of Header)
002	002	002	00000010	STX (Start of Text)
003	003	003	00000011	ETX (End of Text)
004	004	004	00000100	EOT (End of Transmission)
005	005	005	00000101	ENQ (Enquiry)
006	006	006	00000110	ACK (Acknowledgment)
007	007	007	00000111	BEL (Bell \a)
008	010	008	00001000	BS (Backspace \b)
009	011	009	00001001	HT (Horizontal Tab \t)
010	012	00A	00001010	LF (Line Feed \n)
011	013	00B	00001011	VT (Vertical Tab \v)
012	014	00C	00001100	FF (Form Feed \f)
013	015	00D	00001101	CR (Carriage Return \r)
014	016	00E	00001110	SO (Shift Out)
015	017	00F	00001111	SI (Shift In)
016	020	010	00010000	DLE (Data Link Escape)
017	021	011	00010001	DC1 (XON) (Device Control 1)

018	022	012	00010010	DC2 (Device Control 2)
019	023	013	00010011	DC3 (XOFF)(Device Control 3)
020	024	014	00010100	DC4 (Device Control 4)
021	025	015	00010101	NAK (Negative Acknowledgement)
022	026	016	00010110	SYN (Synchronous Idle)
023	027	017	00010111	ETB (End of Trans. Block)
024	030	018	00011000	CAN (Cancel)
025	031	019	00011001	EM (End of Medium)
026	032	01A	00011010	SUB (Substitute)
027	033	01B	00011011	ESC (Escape)
028	034	01C	00011100	FS (File Separator)
029	035	01D	00011101	GS (Group Separator)
030	036	01E	00011110	RS (Request to Send)
031	037	01F	00011111	US (Unit Separator)
032	040	020	00100000	SP (Space)
033	041	021	00100001	! (exclamation mark)
034	042	022	00100010	" (double quote)
035	043	023	00100011	# (number sign)
036	044	024	00100100	$ (dollar sign)
037	045	025	00100101	% (percent)
038	046	026	00100110	& (ampersand)
039	047	027	00100111	' (single quote)
040	050	028	00101000	((left/opening parenthesis)
041	051	029	00101001	) (right/closing parenthesis)
042	052	02A	00101010	* (asterisk)
043	053	02B	00101011	+ (plus)
044	054	02C	00101100	, (comma)
045	055	02D	00101101	- (minus or dash)
046	056	02E	00101110	. (dot)
047	057	02F	00101111	/ (forward slash)
048	060	030	00110000	0
049	061	031	00110001	1
050	062	032	00110010	2
051	063	033	00110011	3
052	064	034	00110100	4
053	065	035	00110101	5
054	066	036	00110110	6
055	067	037	00110111	7
056	070	038	00111000	8
057	071	039	00111001	9
058	072	03A	00111010	: (colon)
059	073	03B	00111011	; (semi-colon)
060	074	03C	00111100	< (less than)
061	075	03D	00111101	= (equal sign)
062	076	03E	00111110	> (greater than)
063	077	03F	00111111	? (question mark)
064	100	040	01000000	@ (AT symbol)
065	101	041	01000001	A
066	102	042	01000010	B
067	103	043	01000011	C
068	104	044	01000100	D
069	105	045	01000101	E
070	106	046	01000110	F
071	107	047	01000111	G
072	110	048	01001000	H
073	111	049	01001001	I
074	112	04A	01001010	J
075	113	04B	01001011	K
076	114	04C	01001100	L
077	115	04D	01001101	M
078	116	04E	01001110	N
079	117	04F	01001111	O
080	120	050	01010000	P
081	121	051	01010001	Q
082	122	052	01010010	R
083	123	053	01010011	S
084	124	054	01010100	T

085	125	055	01010101	U
086	126	056	01010110	V
087	127	057	01010111	W
088	130	058	01011000	X
089	131	059	01011001	Y
090	132	05A	01011010	Z
091	133	05B	01011011	[(left/opening bracket)
092	134	05C	01011100	\ (back slash)
093	135	05D	01011101	] (right/closing bracket)
094	136	05E	01011110	^ (caret/circumflex)
095	137	05F	01011111	_ (underscore)
096	140	060	01100000	`
097	141	061	01100001	a
098	142	062	01100010	b
099	143	063	01100011	c
100	144	064	01100100	d
101	145	065	01100101	e
102	146	066	01100110	f
103	147	067	01100111	g
104	150	068	01101000	h
105	151	069	01101001	i
106	152	06A	01101010	j
107	153	06B	01101011	k
108	154	06C	01101100	l
109	155	06D	01101101	m
110	156	06E	01101110	n
111	157	06F	01101111	o
112	160	070	01110000	p
113	161	071	01110001	q
114	162	072	01110010	r
115	163	073	01110011	s
116	164	074	01110100	t
117	165	075	01110101	u
118	166	076	01110110	v
119	167	077	01110111	w
120	170	078	01111000	x
121	171	079	01111001	y
122	172	07A	01111010	z
123	173	07B	01111011	{ (left/opening brace)
124	174	07C	01111100	| (vertical bar)
125	175	07D	01111101	} (right/closing brace)
126	176	07E	01111110	~ (tilde)
127	177	07F	01111111	DEL (delete)

III. METODE PENELITIAN

Metode penelitian merupakan suatu cara dan alur yang digunakan untuk melakukan penelitian. Dalam penelitian implementasi enkripsi dan dekripsi dengan metode algoritma RC4 ini meliputi tempat dan waktu penelitian, bahan dan alat penelitian dan tahapan penelitian.

3.1 Tempat dan Waktu Penelitian

Penelitian ini dilakukan di Jurusan Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Lampung yang beralamat di Jalan Sumantri Brojonegoro No. 01, Gedong Meneng, Rajabasa, Kota Bandar Lampung, Lampung. Waktu penelitian dilakukan selama Semester Ganjil dan Genap Tahun Ajaran 2016/2017.

3.2 Bahan dan Alat Penelitian

Bahan yang digunakan dalam penelitian ini adalah data mahasiswa Jurusan Ilmu Komputer Universitas Lampung. Data mahasiswa ini berjumlah 970 yang diinterpretasikan ke dalam sebuah tabel yang bernama tabel_mahasiswa yang terdiri dari 25 atribut. Atribut-atribut ini terdiri dari beberapa tipe data yaitu tipe data *integer*, tipe data *char*, tipe data *varchar*, tipe data *text* dan tipe data *date.* Atribut tabel_mahasiwa dapat dilihat pada Tabel 3.1.

Tabel 3.1 Tabel Mahasiswa

No.	Nama Atribut	Tipe Data
1.	npm	char (10)
2.	nama	varchar (50)
3.	alamat_asal	text
4.	alamat_sekarang	text
5.	tempat_lahir	varchar (50)
6.	tanggal_lahir	varchar (10)
7.	id_jenis_kelamin	int (1)
8.	id_golongan_darah	int (1)
9.	id_agama	int (1)
10.	nama_ayah	varchar (25)
11	nama_ibu	varchar (25)
12	id_pekerjaan_ayah	int (2)
13	id_pekerjaan_ibu	int (2)
14	id_pendidikan_ayah	int (1)
15	id_pendidikan_ibu	int (1)
16	alamat_ayah/ibu	text
17	no_telp_ortu	varchar (15)
18	no_telp/e-mail_mhs	varchar (23)
19	asal_sekolah	varchar (25)
20	kab/kota	varchar (25)
21	id_provinsi	int (2)
22	id_jalur_masuk_unila	int (1)
23	id_program_studi	int (1)
24	pilihan	int (1)
25	id_tahun_masuk	int (2)

Dalam pengerjaan penelitian ini, alat penelitian yang digunakan untuk mendukung penelitian itu sendiri adalah sebagai berikut.

1. Perangkat keras (*hardware*) dengan spesifikasi:
   a. *Processor* Intel ® Celeron ® CPU 1007U
   b. RAM 2GB
   c. *Hardisk* 320GB
2. Perangkat lunak (*software*) yang meliputi:

a. Sistem Operasi: Windows 8.1

b. Bahasa pemograman: PHP

c. DBMS: MySQL

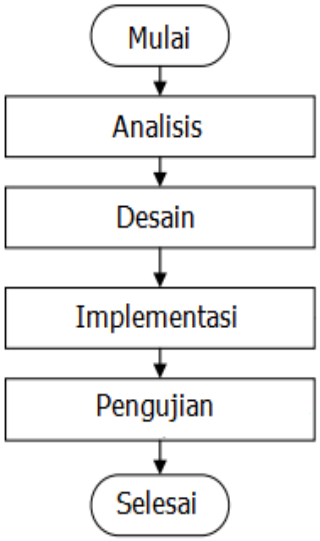
d. Database Server : Apache

e. Aplikasi: PHPMyAdmin

f. Program *editor*: Notepad++

3.3 Tahapan Penelitian

Pada bagian ini dijelaskan mengenai tahapan penelitian yang akan dilakukan. Tahapan-tahapan ini disesuaikan dengan kebutuhan secara berurutan. Metode penelitian ini meliputi analisis, desain, implementasi algoritma RC4 dan pengujian hasil enkripsi dan dekripsi *database*. Diagram alir perancangan pada penelitian ini dapat dilihat pada Gambar 3.1.



Gambar 3.1 Diagram alir penelitian

3.3.1 Analisis

Pada tahapan analisis kebutuhan dimulai dengan mengidentifikasi, mengumpulkan studi literatur mengenai metode-metode tentang kriptografi khususnya metode algoritma RC4. Metode studi literatur digunakan dengan melihat penelitian yang sudah ada dan merujuk pada penelitian yang telah dilakukan.

3.3.2 Desain

Setelah tahap analisis telah dilakukan tahap selanjutnya adalah perancangan desain. Rancangan desain dibuat berdasarkan hasil dari analisis kebutuhan yang telah diperoleh. Dimulai dari bagaimana *input*, proses hingga hasil yang diperoleh. Desain ini meliputi desain *database*, desain tampilan, desain *flowchart*, dan desain sistem.

3.3.2.1 Desain *Database*

Desain *database* pada penelitian ini yaitu sebuah *database* awal yaitu tabel_mahasiswa yang kemudian akan dienkripsi sehingga menghasilkan *database* baru yaitu *database* tabel_mahasiswa_enkripsi. Kemudian untuk mendekripsi, maka data terenkripsi pada *database* tabel_mahasiswa_enkripsi akan dikembalikan pada database tabel_mahasiswa.

Tabel 3.2 Perancangan *database*

No.	tabel_mahasiswa		tabel_mahasiswa_enkripsi	
	Nama atribut	Tipe Data	Nama atribut	Tipe Data
1.	Npm	char (10)	npm	char (10)
2.	Nama	varchar (50)	nama	varchar (50)
3.	alamat_asal	Text	alamat_asal	text
4.	alamat_sekarang	Text	alamat_sekarang	text
5.	tempat_lahir	varchar (50)	tempat_lahir	varchar (50)
6.	tanggal_lahir	varchar (10)	tanggal_lahir	varchar (10)
7.	id_jenis_kelamin	int (1)	id_jenis_kelamin	int (1)
8.	id_golongan_darah	int (1)	id_golongan_darah	int (1)

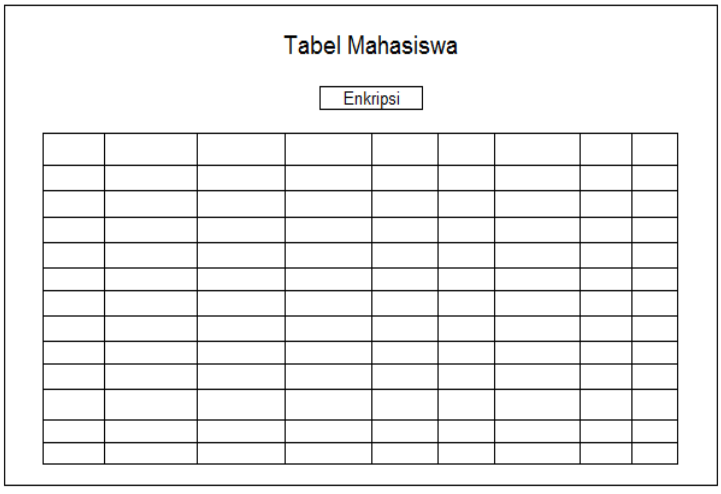
No	tabel_mahasiswa		tabel_mahasiswa_enkripsi	
	Nama Atribut	Tipe Data	Nama Atribut	Tipe Data
9.	id_agama	int (1)	id_agama	int (1)
10.	nama_ayah	varchar (25)	nama_ayah	varchar (25)
11	nama_ibu	varchar (25)	nama_ibu	varchar (25)
12	id_pekerjaan_ayah	int (2)	id_pekerjaan_ayah	int (2)
13	id_pekerjaan_ibu	int (2)	id_pekerjaan_ibu	int (2)
14	id_pendidikan_ayah	int (1)	id_pendidikan_ayah	int (1)
15	id_pendidikan_ibu	int (1)	id_pendidikan_ibu	int (1)
16	alamat_ayah/ibu	Text	alamat_ayah/ibu	text
17	no_telp_ortu	varchar (15)	no_telp_ortu	varchar (15)
18	no_telp/e-mail_mhs	varchar (23)	no_telp/e-mail_mhs	varchar (23)
19	asal_sekolah	varchar (25)	asal_sekolah	varchar (25)
20	kab/kota	varchar (25)	kab/kota	varchar (25)
21	id_provinsi	int (2)	id_provinsi	int (2)
22	id_jalur_masuk_unila	int (1)	id_jalur_masuk_unila	int (1)
23	id_program_studi	int (1)	id_program_studi	int (1)
24	Pilihan	int (1)	pilihan	int (1)
25	id_tahun_masuk	int (2)	id_tahun_masuk	int (2)

3.3.2.2 Desain Tampilan

Desain tampilan yang nantinya akan dibuat terdapat tampilan enkripsi dan tampilan dekripsi.

1. Perancangan Tampilan Enkripsi

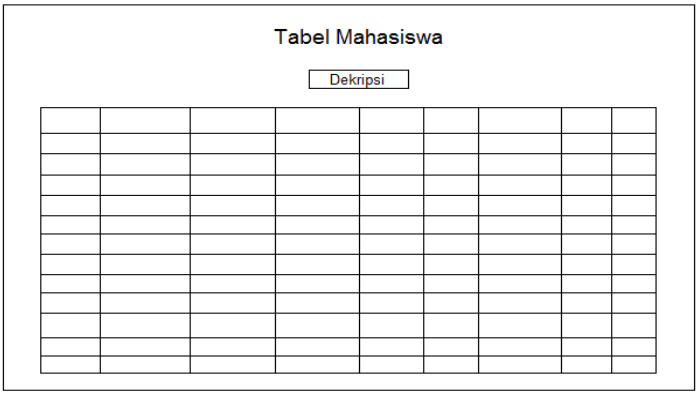
   Perancangan ini akan ditampilkan sebuah tabel dari database tabel_mahasiswa. Di bagian atas halaman, terdapat tombol yang digunakan untuk mengenkripsi data dari tabel tersebut. Rancangan tampilan dapat dilihat pada Gambar 3.2.



Gambar 3.2 Rancangan Tampilan Enkripsi

2. Perancangan Tampilan Dekripsi

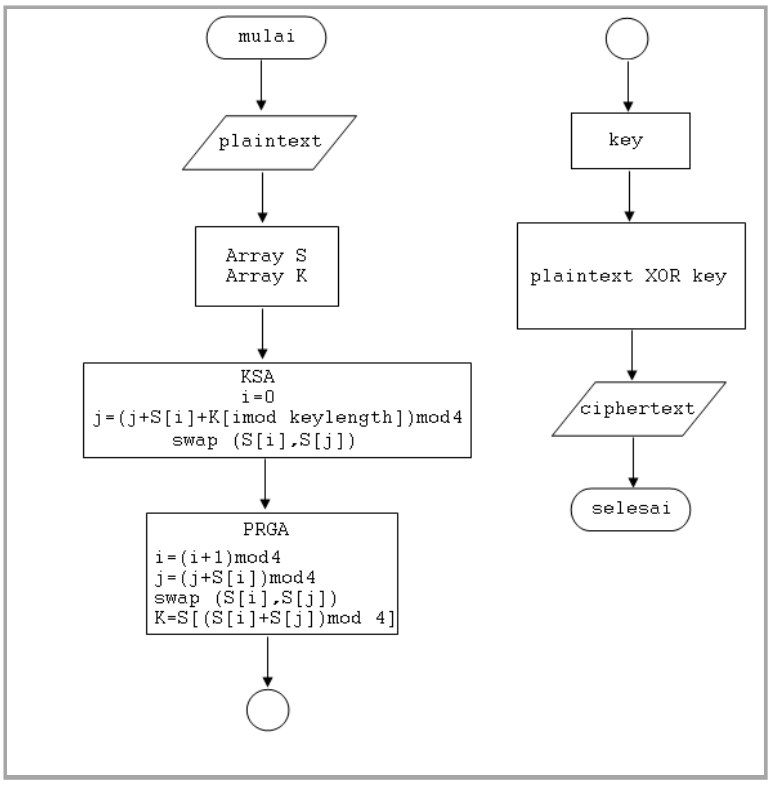
Perancangan ini akan ditampilkan sebuah tabel dari database tabel_mahasiswa_enkripsi. Di bagian atas halaman, terdapat tombol yang digunakan untuk mendekripsi data dari tabel tersebut. Rancangan tampilan dapat dilihat pada Gambar 3.3.



Gambar 3.3 Rancangan Tampilan Dekripsi

3.3.2.3 Perancangan Flowchart

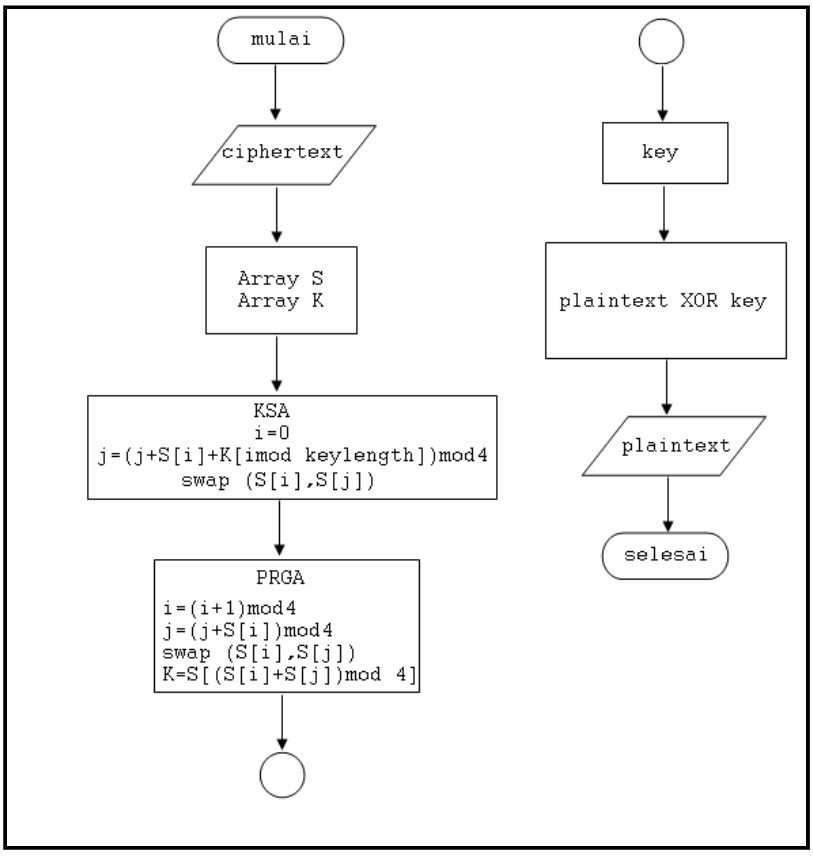
1. Flowchart algoritma RC4 Proses Enkripsi



Gambar 3.4 Flowchart Algoritma RC4 pada Proses Enkripsi

Flowchart pada Gambar 3.4 dimulai dengan masukan yang diinisialisasikan dengan array S dan array K lalu dilakukan proses KSA (*Key Schedulling Algorithm*) dan dilanjutkan dengan proses PRGA (*Pseudo Random Generation Algorithm*). Dari proses tersebut akan dihasilkan sebuah kunci baru (*cipher*). Proses selanjutnya adalah dilakukan proses XOR antara *plaintext* dengan kunci yang didapatkan (*cipher*) sehingga hasil akhirnya berupa *chipertext.*

2. Flowchart algoritma RC4 Proses Dekripsi



Gambar 3.5 Flowchart Algoritma RC4 pada Proses Dekripsi

Flowchart pada Gambar 3.5 sama halnya dengan flowchart enkripsi pada pembentukan kunci. Namun, setelah kunci (*cipher*) terbentuk, maka akan dilakukan proses dekripsi yaitu melakukan operasi XOR

antara *ciphertext* dengan kunci tersebut sehingga hasil akhirnya adalah sebuah *plaintext*.

3.3.3 Implementasi

Merupakan tahapan penulis melakukan analisis terhadap cara kerja dari algoritma RC4 serta mengimplementasikannya ke dalam sebuah program dengan mengguanakan PHP dan MySQL tersebut dalam melakukan proses pengamanan data. Lebih spesifiknya, RC4 beroperasi dengan langkah-langkah sebagai berikut :

1. Melakukan inisialisasi nilai S

   Cara kerja algoritma RC4 yaitu inisialisasi S-Box pertama, S[0],S[1],...,S[3], dengan bilangan 0 sampai 3. Pertama isi secara berurutan S[0] = 0, S[1] =1,...,S[3] = 3. Kemudian inisialisasi *array* lain (S-Box lain), misal *array* K dengan panjang 3. Isi array K dengan kunci yang diulangi sampai seluruh array K[0], K[1],..., K[3] terisi seluruhnya.

2. Pencarian nilai *keystream* (k)

   Pencarian nilai *keystream* dilakukan dengan melakukan pertukaran lagi antar elemen S, tetapi salah satu nilai S kemudian disimpan pada k yang kemudian digunakan sebagai *keystream*. Nilai k inilah yang kemudian digunakan sebagai *keystream*. *Keystream* digunakan untuk melaklukan operasi XOR dengan *plaintext*.

3. Operasi XOR k dengan *plaintext*

   Nilai k yang sudah didapatkan dari langkah di atas kemudian dimasukkan dalam operasi XOR dengan *plaintext* yang ada, dengan sebelumnya pesan dipotong-potong terlebih dahulu menjadi *byte-byte*. Setelah operasi ini dilakukan, langkah awal kembali dilakukan untuk mendapatkan indeks baru dari setiap elemen S.

3.3.4 Pengujian

Tahap ini merupakan pengujian dari keseluruhan tahap-tahap yang telah dilalui dimulai dari analisis kebutuhan hingga tahap implementasi. Pengujian dilakukan terhadap hasil enkripsi (*ciphertext*) pada *database* dan hasil dekripsi. Pengujian terhadap enkripsi dilakukan dengan memproses apakah sebuah data dapat disandikan sedangkan pengujian terhadap dekripsi dilakukan dengan memproses apakah data yang terenkripsi dapat diubah menjadi seperti semula. Pengujian terhadap hasil enkripsi juga melihat apakah hasil enkripsi (*ciphertext*) mempunyai ukuran data yang sama dengan pesan asli atau tidak.

BAB 5

KESIMPULAN

5.1 Kesimpulan

Dari penelitian dan pembahasan implementasi enkripsi dan dekripsi dengan metode RC4 (Rivest Code 4) *stream cipher* pada *database* mahasiswa Ilmu Komputer maka didapat kesimpulan sebagai berikut.

1. Proses enkripsi dan dekripsi yang dilakukan pada *database* mahasiswa berhasil dilakukan. *Database* asli (*plaintext*) dapat dienkripsi menjadi *database* yang disandikan (*chipertext)* dan dapat didekripsi menjadi *database* asli kembali.
2. Jumlah karakter pada *database* asli dengan karakter pada *database* enkripsi dan dekripsi sama ukurannya karena proses metode RC4 dilakukan *byte* per *byte*.
3. Proses enkripsi dan dekripsi dengan algoritma RC4 lebih cepat dilakukan karena berbasis *stream cipher* yang melakukan enkripsi *one byte at a time*.
4. Semakin panjang kunci untuk proses enkripsi maka semakin kuat keamanan enkripsi datanya.

5.2 Saran

Beberapa saran yang perlu diperhatikan untuk pengembangan penelitian selanjutnya yaitu sebagai berikut.

1. Implementasi enkripsi pada *database* dengan metode algoritma RC4 dapat dibandingkan dengan metode lainnya agar dapat membandingkan performa antara keduanya yaitu algoritma RC4 dengan algoritma yang lain.
2. Penelitian selanjutnya dapat dilakukan dengan menggunakan server *database* lainnya yaitu selain server *database* XAMPP MySQL agar keduanya dapat dibandingkan.
3. Penelitian selanjutnya dapat dilakukan penambahan fungsi edit pada tampilan webnya agar lebih sempurna.
4. Proses enkripsi dapat diperbaiki secara lebih baik lagi dengan membuat pilihan tabel (*field*) yang akan dienkripsi sehingga bisa dipilih tabel (*field*) apa saja yang perlu dienkripsi.
5. Proses enkripsi dan dekripsi langsung dilakukan saat ada data yang dimasukkan sehingga proses enkripsi dan dekripsi tidak dilakukan sekaligus dengan data yang lainnya.

DAFTAR PUSTAKA

Arifyanto, A.E, 2013. 'Implementasi Enkripsi Basis Data Berbasis Web Dengan Algoritma Stream Cipher RC4', Dokumen Karya Ilmiah Program Studi Teknik Informatika Fakultas Ilmu Komputer Universitas Dian Nuswantoro Semarang

Ariyus, *Dony.* 2008. *Pengantar Ilmu Kriptografi Teori, Analisis dan Implementasi*. Yogyakarta: Penerbit Andi

Fathansyah. 2007. *Buku Teks Komputer Sistem Basis Data*. Bandung: Informatika

Jumrin, Sutardi, Subardin, 2016. 'Aplikasi Sistem Keamanan Basis Data Dengan Teknik Kriptografi RC4 Stream Cipher', Jurusan Teknik Informatika Universitas Halu Oleo, Vol.2, pp.59-64

Kadir, Abdul. 2008. *Belajar Database Menggunakan MySQL*. Yogyakarta: Andi

Kristanto, Andri. 2003. *Keamanan Data pada Jaringan Komputer.* Yogyakarta: Penerbit Gava Media

Kustiyahningsih,Y dan Devie R.A. 2011. *Pemrograman Basis Data Berbasis Web Menggunakan PHP dan MySQL*. Yogyakarta: Graha Ilmu

Mollin, R. A. 2007. *An Introduction to Cryptography*. 2nd ed. Florida: Chapman & Hall/CRC.

Munir, Rinaldi. 2006. *Diktat Kuliah Kriptografi*. Bandung: Program Studi Tenik Informatika, Institut Teknologi Bandung

Munir, Rinaldi. 2008. *Belajar Ilmu Kriptografi*. Yogyakarta: Penerbit Andi

Munir, Rinaldi. 2011. *Kriptografi Keamanan*. Bandung: Informatika Bandung

Nugroho, Bunafit. 2005. *Administrasi Database MySQL*. Yogyakarta: Penerbit Graha Ilmu

Piansyah, Endang, 2008. 'Implementasi algoritma Dasar RC4 Stream Cipher dan Pengacakan Plaintext dengan Teknik Dynamic Blocking pada Aplikasi Sistem Informasi Kegatan Blocking pada Aplikasi Sistem Informasi Kegiatan Skripsi di Departemen Teknik Elektro'Departemen Tenik Elektro Universitas Indonesia

Prasetyo, A. 2012. *Buku Pintar Pemrograman Web*. Jakarta: Mediakita

Riyanto. 2011. *Membuat Sendiri Aplikasi E-Commerce dengan PHP & MySQL Menggunakan CodeIgneter &JQuery*. Yogyakarta: Pener*bit* Andi

Sadikin, Rifki. 2012. *Kriptografi Untuk Keamanan Jaringan.*Yogyakarta: Penerbit Andi.

Sampurna.*1996. Belajar Sendiri Membuat Homepage dengan HTML*. Jakarta: PT Elex. Media Komputindo

Simarmata, Janner. 2007. *Perancangan Basis Data*. Yogyakarta: Andi