BAB II

LANDASAN TEORI

II.1. Perancangan

Untuk membuat tampilan yang menarik memang tidak mudah dilakukan. Seorang perancang tampilan selain harus mempunyai jiwa seni yang memadai, seorang perancang tampilan juga harus mengerti selera pengguna secara umum. Hal lain yang perlu disadari oleh seorang perancang tampilan bahwa seorang perancang tampilan harus bisa menyakinkan pemrogramnya bahwa apa yang seorang perancang tampilan bayangkan dapat diwujudkan dengan piranti bantu yang tersedia. (Sandro Willy Hutabarat ; 2014 : 6)

Perancangan merupakan proses pengolahan hasil analisis perangkat lunak menjadi rencana pengembangan perangkat lunak dan batasan-batasan perangkat lunak atau masalah yang mungkin dihadapi dalam pengembangan perangkat lunak. Perancangan yang dilakukan meliputi perancangan arsitektur, perancangan modul, dan perancangan antarmuka. (Sandro Willy Hutabarat ; 2014 : 6)

II.1.1. Cara Pendekatan

Program aplikasi pada dasarnya dapat dikelompokkan dalam dua kategori, yaitu program aplikasi untuk keperluan khusus dengan pengguna yang khusus pula dan program aplikasi yang akan digunakan oleh pengguna umum, yang sering dikenal dengan sebutan *public software*. Karena perbedaan pada calon

pengguna, maka perancang program antarmuka perlu memperhatikan hal ini. (Sandro Willy Hutabarat ; 2014 :7)

Pada kelompok pertama, yakni pada program aplikasi untuk keperluan khusus, misalnya program aplikasi untuk inventori gudang, pengelolaan data akademis mahasiswa, pelayanan reservasi horel, dan program-program aplikasi yang serupa. Kelompok calon pengguna yang akan memanfaatkan program aplikasi tersebut dapat dengan mudah diperkirakan, baik dalam hal keahlian pengguna maupun ragam antarmuka yang digunakan. Untuk kelompok ini ada satu pendekatan yang dapat dilakukan, yakni pendekatan yang disebut dengan pendekatan perancangan berpusat ke pengguna (*over centered design approach*). (Sandro Willy Hutabarat ; 2014 : 7)

Pendekatan perancangan berpusat ke pengguna adalah perancangan antarmuka yang melibatkan pengguna. Keterlibatan pengguna disini tidak diartikan bahwa pengguna harus ikut memikirkan bagaimana implementasinya nanti, tetapi pengguna diajak untuk aktif berpendapat ketika perancangan antarmuka sedang menggambar wajah antarmuka yang mereka inginkan. Dengan kata lain, perancangan dan pengguna bersama-sama untuk merancang wajah antarmuka yang diinginkan pengguna. Pengguna menyampaikan keinginannya, sementara perancangan menggambar keinginan pengguna tersebut serta menjelaskan keuntungan dan kerugian wajah antarmuka yang diinginkan oleh pengguna, seolah-olah sudah mempunyai gambaran nyata tentang antarmuka yang nanti akan digunakan. (Sandro Willy Hutabarat ; 2014 : 7-8)

Pada perancangan oleh pengguna, pengguna sendirilah yang merancang wajah antarmuka yang diinginkan. Di satu sisi, cara ini akan mempercepat proses pengimplementasian modul antarmuka. Tetapi disisi lain, hal ini justru sangat memberatkan pemrogram karena apa yang diinginkan pengguna belum tentu dapat diimplementasikan dengan mudah atau bahkan tidak dapat dikerjakan dengan menggunakan piranti bantu yang ada.

Perancangan program aplikasi yang dimasukkan dalam kelompok kedua atau *public software* perlu menganggap bahwa program aplikasi tersebut akan digunakan oleh pengguna dengan berbagai tingkat kepandaian dan karakteristik yang sangat beragam. Di satu sisi, keadaan ini dapat digunakan untuk memaksa pengguna menggunakan antarmuka yang dibuat, tetapi pada sisi lain pemaksaan itu akan berakibat bahwa program aplikasinya menjadi tidak banyak penggunanya. Satu kunci penting dalam pembuatan modul antarmuka untuk program-program aplikasi pada kelompok ini adalah dengan melakukan *customization*. Dengan *customization*, pengguna dapat menggunakan program aplikasi dengan wajah antarmuka yang sesuai dengan selera masing-masing pengguna. (Sandro Willy Hutabarat ; 2014 : 8)

II.1.2. Prinsip dan Petunjuk Perancangan

Antarmuka pengguna secara alamiah terbagi menjadi 4 (empat) komponen, yaitu model pengguna, bahasa perintah, umpan balik, dan penampilan informasi. Model pengguna merupakan dasar dari tiga komponen yang lain. (Sandro Willy Hutabarat ; 2014 : 9)

Model pengguna merupakan model konseptual yang dimiliki oleh pengguna ketika pengguna menggunakan sebuah sistem atau program aplikasi. Model ini memungkinkan seorang pengguna untuk mengembangkan pemahaman mendasar tentang bagian yang dikerjakan oleh program, bahkan oleh pengguna yang sama sekali tidak mengetahui komputer. Dengan pertolongan model itu, pengguna dapat mengantisipasi pengaruh suatu tindakan yang dilakukan dan dapat memilih strategi yang cocok untuk mengoperasikan program tersebut. Model pengguna dapat berupa suatu simulasi tentang keadaan yang sebenarnya dalam dunia nyata, sehingga pengguna tidak perlu mengembangkannya sendiri dari awal.

Setelah pengguna mengetahui dan memahami model yang diinginkan, pengguna memerlukan piranti untuk memanipulasi model itu. Piranti pemanipulasian model ini sering disebut dengan bahasa perintah (*command language*), yang sekaligus merupakan komponen kedua dari antarmuka pengguna. Idealnya program komputer mempunyai bahasa perintah yang alami, sehingga model pengguna dengan cepat dapat dioperasionalkan. (Sandro Willy Hutabarat ; 2014 : 10)

Komponen ketiga adalah umpan balik. Umpan balik disini diartikan sebagai kemampuan sebuah program yang membantu pengguna untuk mengoperasikan program itu sendiri. Umpan balik dapat berbentuk pesan penjelasan, pesan penerimaan perintah, indikasi adanya obyek terpilih dan penampilan karakter yang diketikkan lewat papan ketik. Beberapa bentuk umpan balik terutama ditujukan kepada pengguna yang belum berpengalaman dalam menjalankan program sebuah aplikasi. Umpan balik dapat digunakan untuk memberi keyakinan

bahwa program telah menerima perintah pengguna dan dapat memahami maksud perintah tersebut.

Komponen keempat adalah tampilan informasi. Komponen ini digunakan untuk menunjukkan status informasi atau program ketika pengguna melakukan suatu tindakan. Pada bagian ini, perancang harus menampilkan pesan-pesan tersebut seefektif mungkin sehingga mudah dipahami oleh pengguna. Setelah memahami beberapa prinsip dalam perancangan antarmuka pengguna. Pada bagian berikut ini akan diberikan petunjuk singkat tentang perancangan antarmuka yang akan dilakukan sebagai seorang perancang tampilan.

II.1.3. Urutan Perancangan

Perancangan dialog, seperti halnya perancangan sistem yang lain, harus dikerjakan secara atas ke bawah. Proses perancangannya dapat dikerjakan secara bertahap sampai rancangan yang diinginkan terbentuk, yaitu sebagai berikut:

1. Pemilihan Ragam Dialog

Untuk suatu tugas tertentu, pilihlah ragam dialog yang menurut perkiraan cocok untuk tugas tersebut. Pemilihan ragam dialog dipengaruhi oleh karakteristik populasi pengguna, tipe dialog yang diperlukan, dan kendala teknologi yang ada untuk mengimplementasikan ragam dialog tersebut. Ragam dialog yang terpilih dapat berupa sebuah ragam tunggal atau sekumpulan ragam dialog yang satu sama lainnya saling mendukung.

2. Perancangan Struktur Dialog

Tahap kedua adalah melakukan analisis tugas dan menentukan model pengguna dari tugas tersebut untuk membentuk struktur dialog yang sesuai. Dalam tahap ini, pengguna sebaiknya banyak dilibatkan, sehingga pengguna langsung mendapatkan umpan balik dari diskusi yang terjadi. Pada tahap ini, suatu purwarupa dialog seringkali dibuat untuk memberikan gambaran yang lebih jelas kepada calon pengguna.

3. Perancangan Format Pesan

Pada tahap ini, tata letak tampilan dan keterangan tekstual secara terinci harus mendapat perhatian lebih. Selain itu, kebutuhan data masukkan yang mengharuskan pengguna untuk memasukkan data ke dalam komputer juga harus dipertimbangkan dari segi efisiensinya. Salah satu contohnya adalah dengan mengurangi pengetikan yang tidak perlu dengan cara mengefektifkan pengguna tombol.

II.2. Aplikasi

Aplikasi berasal dari kata *application* yang artinya penerapan, lamaran, penggunaan. Secara istilah aplikasi adalah program yang siap pakai yang direka untuk melaksanakan suatu fungsi bagi pengguna atau aplikasi yang lain dan dapat digunakan oleh sasaran yang dituju. Aplikasi merupakan perangkat lunak yang dimasukkan atau terdapat dalam komputer dan memiliki fungsi-fungsi khusus. (Iko Ari Guna ; 2014 : 10)

II.3. Keamanan Data

Keamanan data merupakan hal yang sangat penting dalam bidang bisnis komersial (perusahaan) dan tradisional saat ini, contohnya penggunaan media pengiriman data elektronik melalui e-mail dan media lainnya yang sering digunakan dalam dunia bisnis. Data yang dikirim melalui email memiliki variasi, ada yang bersifat umum dan rahasia. Berdasarkan kondisi dan kenyataan tersebut, para ahli dan peneliti dibidang IT mengupayakan berbagai cara yaitu dengan membuat sistem keamanan data. Salah satu metode keamanan data adalah metode kriptografi yang dapat digunakan untuk mengamankan data yang bersifat rahasia agar data tersebut tidak diketahui oleh orang lain yang tidak berkepentingan. (Budi Triandi ; Vol. 2 No. 2, Oktober 2013 : 124)

Masalah keamanan merupakan salah satu aspek terpenting dari sebuah sistem informasi. Informasi menentukan hampir setiap elemen dari kehidupan manusia. Informasi sangat penting artinya bagi kehidupan karena tanpa informasi, maka hampir semua tidak dapat dilakukan dengan baik. Contohnya, jika membeli tiket penerbangan dan membayarnya dengan menggunakan kartu kredit. Informasi mengenai diri nantinya akan disimpan dan dikumpulkan serta digunakan oleh bank dan penerbangan. Demikian juga halnya saat membeli obat di apotik. Harus mendapat resep dari dokter dan memberikan resep tersebut ke pelayan apotik. Resep itu merupakan satu informasi yang disampaikan dokter ke pihak apotik tentang obat yang dibutuhkan.

Kemajuan sistem informasi memberikan banyak keuntungan bagi kehidupan manusia. Meski begitu, aspek negatifnya juga banyak, seperti kejahatan komputer

yang mencakup pencurian, penipuan, pemerasan, kompetisi, dan banyak lainnya. Jatuhnya informasi ke pihak lain, misalnya lawan bisnis, dapat menimbulkan kerugian bagi pemilik informasi. Sebagai contoh, banyak informasi milik perusahaan yang hanya boleh diketahui oleh orang-orang tertentu di perusahaan tersebut, seperti misalnya informasi tentang produk yang sedang dalam pengembangan. Algoritma dan teknik yang digunakan untuk menghasilkan produk tersebut. Untuk itu keamanan dari sistem informasi yang digunakan harus terjamin dalam batas tertentu.

II.3.1. Ancaman Keamanan

Terjadi banyak pertukaran informasi setiap detiknya di internet. Juga banyak terjadi pencurian atas informasi oleh pihak-pihak yang tidak bertanggung jawab. Ancaman keamanan yang terjadi terhadap informasi adalah:

1. *Interruption*, merupakan ancaman terhadap *availability* informasi, data yang ada dalam sistem komputer dirusak atau dihapus sehingga jika data atau informasi tersebut dibutuhkan maka pemiliknya akan mengalami kesulitan untuk mengaksesnya, bahkan mungkin informasi itu hilang.
2. *Interception*, merupakan ancaman terhadap kerahasiaan *(secrecy)*. Informasi disadap sehingga orang yang tidak berhak dapat mengakses komputer dimana informasi tersebut disimpan.
3. *Modification*, merupakan ancaman terhadap integritas. Orang yang tidak berhak berhasil menyadap lalu-lintas informasi yang sedang dikirim dan kemudian mengubahnya sesuai keinginan orang tersebut.

4. *Fabrication*, merupakan ancaman terhadap integritas. Orang yang tidak berhak berhasil meniru atau memalsukan informasi sehingga orang yang menerima informasi tersebut menyangka bahwa informasi tersebut berasal dari orang yang dikehendaki oleh si penerima informasi.

Tabel II.1 Ancaman Terhadap Keamanan Data

System	***Avability***	***Secrecy***	***Integrity***
Hardware	Dicuri atau dirusak		
Software	Program dihapus	*Software* di *copy*	Program dimodifikasi
Data	*File* dihapus atau dirusak	Dicuri, disadap	*File* dimodifikasi
Line Komunikasi	Kabel diputus	Informasi disadap	Informasi dimodifikasi

(Sumber : Dony Ariyus ; 2008 : 24)

II.3.2. Aspek-Aspek Keamanan Komputer

Keamanan komputer meliputi empat aspek, antara lain:

1. *Authentication*, agar penerima informasi dapat memastikan keaslian pesan, bahwa pesan itu datang dari orang yang dimintai informasi. Dengan kata lain, informasi itu benar-benar datang dari orang yang dikehendaki.
2. *Integrity*, keaslian pesan yang dikirm melalui jaringan dan dapat dipastikan bahwa informasi yang dikirim tidak dapat dimodifikasi oleh orang yang tidak berhak.
3. *Non-repudiation*, merupakan hal yang berhubungan dengan si pengirim. Pengirim tidak dapat mengelak bahwa dialah yang mengirim informasi tersebut.

4. *Authority*, informasi yang berada pada sistem jaringan tidak dapat dimodifikasi oleh pihak yang tidak berhak untuk mengaksesnya.
5. *Confidentiality*, merupakan usaha untuk menjaga informasi dari orang yang tidak berhak mengakses. Kerahasiaan ini biasanya berhubungan dengan informasi yang diberikan oleh pihak lain.
6. *Privacy*, lebih ke arah data-data yang bersifat pribadi.
7. *Availability*, aspek availabilitas berhubungan dengan ketersediaan informasi ketika dibutuhkan. Sistem informasi yang diserang atau dijebol dapat menghambat atau meniadakan akses ke informasi.
8. *Access Control*, aspek ini berhubungan dengan cara pengaturan akses ke informasi. Hal ini biasanya berhubungan dengan masalah otentikasi dan privasi. Kontrol akses seringkali dilakukan dengan menggunakan kombinasi *user id* dan *password* ataupun dengan mekanisme lain.

II.3.3. Strategi Keamanan Data

Keamanan data pada lalu-lintas jaringan adalah suatu hal yang diinginkan semua orang untuk menjaga privasi, agar data yang dikirim aman dari gangguan orang yang tidak bertanggung jawab, yang disembunyikan menggunakan algoritma kriptografi. (Dony Ariyus, 2008 : 10)

II.4. *Folder*

Sebuah *folder*, yang nama lainnya adalah direktori, merupakan suatu kontainer yang dapat digunakan untuk menyimpan *file*. Selain *file*, suatu *folder*

juga dapat menampung *folder* lain yang disebut dengan *subfolder*. Akibat adanya *folder* dengan *subfolder*-nya ini, dapat terbentuk suatu *tree* dengan seluruh *parent*-nya merupakan *folder*. Pada sistem operasi Linux dan turunan Unix lainnya, segala sesuatu dalam sistem dianggap sebagai *file*, termasuk direktori. Direktori menjadi suatu *file* khusus yang mengandung daftar dari nama-nama *file* beserta *node* masing-masing. Direktori memiliki peran penting dalam sistem *file* yang hierarkis pada sistem operasi komputer modern dengan menginzinkan pengelompokan direktori dan *file* untuk mengatur sistem *file* dalam hierarki yang modular. (Rinaldi Munir, Bernardino Madaharsa ; 2011)

II.5. Kriptografi

Kriptografi berasal dari bahasa Yunani, *crypto* dan *graphia*. Crypto berarti *secret* (rahasia) dan graphia berarti *writing* (tulisan). Menurut terminologinya, kriptografi adalah ilmu dan seni untuk menjaga keamanan pesan ketika pesan dikirim dari suatu tempat ke tempat lain. (Dony Ariyus, 2008 : 13)

II.5.1. Sejarah Kriptografi

Kriptografi sudah digunakan 400 tahun yang lalu, diperkenalkan oleh orang-orang Mesir lewat *hieroglyph*. Jenis tulisan ini bukanlah bentuk standar untuk menulis pesan.

Pada zaman romawi kuno, saat Julius Caesar ingin mengirimkan pesan rahasia kepada seorang jendral di medan perang. Pesan tersebut harus dikirmkan melalui seorang kurir. Karena pesan tersebut mengandung rahasia, Julius Caesar tidak ingin pesan rahasia tersebut sampai terbuka di jalan. Julius Caesar kemudian

memikirkan bagaimana mengatasinya. Julius Caesar pun mengacak pesan tersebut hingga menjadi suatu pesan yang tidak dapat dipahami oleh siapapun terkecuali oleh jendralnya saja. Tentu sang jendral telah diberi tahu sebelumnya bagaimana cara membaca pesan teracak tersebut. Yang dilakukan Julius Caesar adalah mengganti semua susunan alfabet dari a, b, c, yaitu a menjadi d, b menjadi e, c menjadi f dan seterusnya.

Dari ilustrasi tersebut, beberapa istilah kriptografi dipergunakan untuk menandai aktivitas-aktivitas rahasia dalam mengirim pesan. Apa yang dilakukan Julius Caesar yang mengacak pesan, disebut sebagai enkripsi. Pada saat sang Jendral merapikan pesan yang teracak itu, proses itu disebut dekripsi. Pesan awal yang belum diacak dan pesan yang telah dirapikan, disebut *plaintext*, sedangkan pesan yang telah diacak disebut *ciphertext*.

Pada zaman romawi juga telah ada alat pembuat pesan rahasia yang disebut *scytale* yang digunakan oleh tentara Sparta. *Scytale* merupakan suatu alat uang memiliki pita panjang dari daun papirus dan ditambah dengan sebatang silinder. Mula-mula pengirim pesan menuliskan pesannya di atas pita papirus yang digulung pada batang silinder. Setelah itu, pita dilepaskan dan dikirim. Misalkan batang silinder cukup lebar untuk menulis 6 huruf dan bisa memuat 3 huruf secara melingkar. Jika pengirim ingin mengirimkan pesan:

TOLONG SAYA DISERANG

Maka ia menulis di atas batang silinder:

T O L O N G

S A Y A D I

S E R A N G

Jika pitanya dilepaskan dari batang silinder, maka tulisan yang muncul di atas pita adalah TSSOAELYROAANDNGIG.

Untuk membaca pesan yang dikirim, penerima pesan harus melilitkan kembali pita tersebut pada batang silinder yang memiliki diameter yang sama. Yang menjadi kunci dalam penyandian scytale adalah diameter batang atau jumlah huruf yang dapat ditulis secara melingkar (dalam hal ini 3 huruf). Penyandian dengan scytale sangat mudah dipecahkan karena kriptanalis hanya perlu menerka jumlah huruf yang dapat ditulis secara melingkar pada batang silinder yang digunakan, apalagi karena jumlah huruf yang dapat ditulis secara melingkar pada suatu batang silinder relatif sedikit (maksimum adalah setengah dari jumlah huruf yang tertulis pada pita). Kunci hasil dekripsi:

1. TSSOAELYROAANDNGIG
2. TSALRANNISOEYOADGG
3. TOLONGSAYADISERANG

Karena dekripsi dengan *k=3* menghasilkan pesan yang bermakna, maka bisa disimpulkan bahwa pesan yang dikirim adalah TOLONG SAYA DISERANG. (Dony Ariyus, 2008 : 13)

II.5.2. Algoritma Kriptografi

Kata algoritma muncul di dalam kamus Webster sampai akhir 1957. Kata *algorism* mempunyai arti proses perhitungan dalam bahasa Arab. Algoritma berasal dari nama penulis buku Arab yang terkenal, yaitu Abu Ja'far Muhammad Ibnu Musa al-Khuwarizmi (al-Khuwarizmi dibaca oleh orang barat sebagai *algorism*). Kata *algorism* lambat laun berubah menjadi *algorithm*. (Dony Ariyus, 2008 : 43)

Definisi algoritma adalah urutan langkah-langkah logis untuk menyelesaikan masalah yang disusun secara sistematis. Algoritma kriptografi merupakan langkah-langkah logis bagaimana menyembunyikan pesan dari orang-orang yang tidak berhak atas pesan tersebut.

Algoritma kriptografi terdiri dari tiga fugsi dasar, yaitu:

1. *Enkripsi* merupakan hal yang sangat penting dalam kriptografi, merupakan pengamanan data yang dikirimkan agar terjaga kerahasiaannya. Pesan asli disebut *plaintext*, yang diubah menjadi kode-kode yang tidak dimengerti. Enkripsi bisa diartikan dengan *cipher* atau kode. Sama halnya dengan kita tidak mengerti akan sebuah kata maka kita akan melihatnya di dalam kamus atau daftar istilah. Beda halnya degan enkripsi, untuk mengubah teks-asli ke bentuk teks-kode kita menggunakan algoritma yang dapat mengkodekan data yang diinginkan.
2. *Dekripsi* merupakan kebalikan dari enkripsi. Pesan yang telah dienkripsi dikembalikan ke bentuk asalnya (teks-asli), disebut dengan dekripsi pesan.

Algoritma yang digunakan untuk dekripsi tentu berbeda dengan algoritma yang digunakan untuk enkripsi.

3. *Kunci*, yang dimaksud disini adalah kunci yang dipakai untuk melakukan enkripsi dan dekripsi. Kunci terbagi menjadi dua bagian kunci rahasia *(private key)* dan kunci umum *(public key)*.

Keamanan dari algoritma kriptografi tergantung pada bagaimana algoritma itu bekerja. Oleh sebab itu, algoritma semacam ini disebut dengan algoritma terbatas. Algoritma terbatas merupakan algoritma yang dipakai sekelompok orang untuk merahasiakan pesan yang mereka kirim. Jika salah satu dari anggota kelompok keluar dari kelompoknya maka algoritma yang dipakai diganti dengan yang baru. Jika tidak, maka hal itu bisa menjadi masalah dikemudian hari. (Dony Ariyus, 2008 : 44)

Keamanan dari kriptografi modern didapat dengan merahasiakan kunci yang dimiliki dari orang lain, tanpa harus merahasiakan algoritma itu sendiri. Kunci memiliki fungsi yang sama dengan *password*. Jika keseluruhan dari keamanan algoritma tergantung pada kunci yang dipakai maka algoritma ini bisa dipublikasikan dan dianalisis oleh orang lain. Jika algoritma yang telah dipublikasikan bisa dipecahkan dalam waktu singkat oleh orang lain maka berarti agloritma tersebut tidaklah aman untuk digunakan.

II.5.3. Macam-Macam Algoritma Kriptografi

Algoritma kriptografi dibagi menjadi tiga bagian berdasarkan kunci yang dipakainya:

1. Algoritma Simetri

   Algortima simetri ini sering disebut dengan algoritma klasik karena memakai kunci yang sama untuk kegiatan enkripsi dan dekripsi. Algoritma ini sudah ada sejak lebih dari 4000 tahun yang lalu. Bila mengirim pesan dengan menggunakan algoritma ini, orang yang menerima pesan harus diberitahu kunci dari pesan tersebut agar bisa mendekripsikan pesan yang dikirim. Keamanan dari pesan yang menggunakan algortima ini tergantung pada kunci. Jika kunci tersebut diketahui oleh orang lain, maka orang tersebut akan dapat melakukan enkripsi dan dekripsi terhadap pesan. Algoritma yang memakai kunci simetri diantaranya adalah:

   a. *Data Encryption Standard (DES)*
   b. *RC2, RC4, RC5, RC6*
   c. *International Data Encryption Algorithm (IDEA)*
   d. *One Time Pad (OTP)*
   e. *A5* dan lain sebagainya

2. Algoritma Asimetri

   Algoritma asimetri sering juga disebut dengan algoritma kunci publik, dengan arti kata kunci yang digunakan untuk melakukan enkripsi dan dekripsi berbeda. (Dony Ariyus, 2008 : 45)

3. Hash

   Fungsi hash sering disebut dengan fungsi Hash satu arah *(one-way funtion)*, *message digest, fingerprint*, fungsi kompresi dan *message authentication code* (MAC), merupakan suatu fungsi matematika yang mengambil

masukan panjang variabel dan mengubahnya ke dalam urutan biner dengan panjang yang tetap.

II.6. Algoritma *Serpent*

Algoritma *serpent* Eli Biham, dan Lars Knudsen. Algoritma ini merupakan *runner-up* dalam kompetisi Advanced Encryption Standart (AES) yang dimenangkan oleh algoritma *Rijandael*. Para perancangnya mengklaim meskipun *Rijandael* lebih cepat karena memiliki putaran lebih sedikit, *serpent* lebih aman. Meskipun lebih lambat daripada algoritma AES (Rijandael), algoritma ini masih lebih cepat daripada algoritma DES (Data Encryption Algorithm) yang telah banyak digunakan sebelumnya. Pada mesin Pentium 200MHz, algoritma *serpent* dapat berjalan dengan kecepatan 45 Mbit/s. Ini lebih cepat daripada DES yang memliki kecepatan 15 Mbit/s pada mesin yang sama.

Serpent merupakan operasi SP-*network* (*substitution permutation network*) 32 putaran pada 4 word berukuran 32 bit (blok berukuran 128 bit). Pada komputasi internal, semua nilai direpresentasikan *little-endian* dengan word pertama adalah least significant word dan word terakhir adalah most significant word dengan bit 0 merupakan least significant bit dari word pertama. Panjang kunci yang dapat digunakan berukuran 128, 192, atau 256 bit. Jika panjang kurang dari 256 bit, kunci tersebut ditambahkan satu bit ’1’ yang diikuti ‘0’ hingga panjangnya 256 bit.

Dalam enkripsi serpent terdapat tiga tahap :

1. Permutasi awal (Initial Permutation /IP).

2. Proses 32 putaran, yang masing-masing terdiri atas operasi pencampuran kunci (Ki), proses nilai S-boxes (Si), dan transformasi linear (L). Pada putaran terakhir, transformasi linear diganti dengan operasi pencampuran kunci.
3. Permutasi akhir (Finali Permutation /FP)

(Rinaldi Munir, Bernardino Madaharsa ; 2011)

II.7. UML *(Unified Modelling Language)*

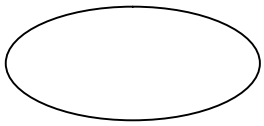
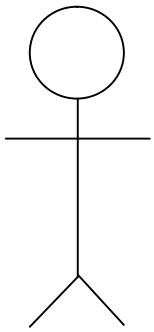
Penjadwalan telah lama diteliti, contohnya dalam penghasilan tenaga oleh *Windu Gata* dan *Grace Gata* (2013), UML *(Unified Modelling Language)* adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membangun perangkat lunak. UML *(Unified Modelling Language)* merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem.

II.7.1 *Use Case Diagram*

Use Case Diagram merupakan pemodelan untuk kelakuan *(behavior)* sistem informasi yang akan dibuat. Mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Dapat dikatakan *usecase* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut.

Simbol-simbol yang digunakan dalam *Use Case Diagram*, yaitu:

Tabel II.2 *Diagram Use Case*

Gambar	Keterangan
	*Use Case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukar pesan antar unit dengan aktor, biasanya dinyatakan dengan menggunakan kata kerja diawal nama *Use Case*.
	*Actor* atau Aktor adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan aktor, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. Orang atau sistem bisa muncul dalam beberapa peran. Perlu dicatat bahwa aktor berinteraksi dengan *use case*, tetapi tidak memiliki kontrol terhadap *use case*.
	Asosiasi antara aktor dan *use case*, digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengindikasikan aliran data.
	Asosiasi antara aktor dan *use case* yang menggunakan panah terbuka untuk mengindikasikan bila aktor berinteraksi secara pasif dengan sistem.

(Sumber : Windu Gata dan Grace Gata ; 2013)

II.7.2 *Class Diagram* (Diagram Kelas)

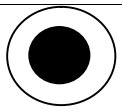
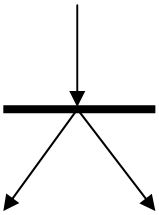
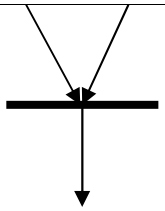
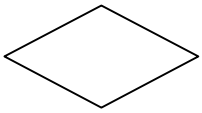
Class Diagram merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan

aturan-aturan dan tanggung jawab entitas yang menentukan perilaku sistem. *Class Diagram* juga menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan *constraint* yang berhubungan dengan objek yang dikoneksikan.

II.7.3 *Activity Diagram* (Diagram Aktivitas)

Activity Diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Simbol-simbol yang digunakan dalam *activity diagram*, yaitu:

Tabel II.3 *Diagram Aktivitas*

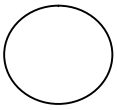
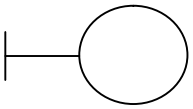
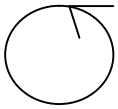
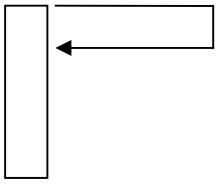
Gambar	Keterangan
	*Start Point* diletakkan pada pojok kiri atas dan merupakan awal aktifitas.
	*End Point*, akhir aktifitas.
	*Activities*, menggambarkan suatu proses/kegiatan bisnis.
	*Fork* (percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara paralel atau untuk menggabungkan dua kegiatan paralel menjadi satu.
	*Join* (penggabungan) atau *Rake*, digunakan untuk menunjukkan adanya dekomposisi.
	*Decision Points*, menggambarkan pilihan untuk pengambilan keputusan, *true* atau *false*.

(Sumber : Windu Gata dan Grace Gata ; 2013)

II.7.4 *Sequence Diagram* (Diagram Urutan)

Sequence Diagram menggambarkan kelakuan objek pada *usecase* dengan mendekripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. Simbol-simbol yang digunakan dalam *sequence diagram*, yaitu:

Tabel II.4 *Diagram Urutan*

Gambar	Keterangan
	*Entity Class*, merupakan bagian dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran awal sistem dan menjadi landasan untuk menyusun basis data.
	*Boundary Class*, berisi kumpulan kelas yang menjadi *interface* atau interaksi antara satu atau lebih aktor dengan sistem, seperti tampilan *formentry* dan *form* cetak.
	*Control Class*, suatu objek yang berisi logika aplikasi yang tidak memiliki tanggung jawab kepada entitas, contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai objek. Control *object* mengkoordinir pesan antara *boundary* dengan entitas.
	Message, simbol mengirim pesan antar *class*.
	*Recursive*, menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri.

	*Activation*, mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivitas sebuah operasi.
	*Lifeline*, garis titik-titik yang terhubung dengan objek, sepanjang *lifeline* terdapat *activation*.

(Sumber : Windu Gata dan Grace Gata ; 2013)

II.8. *Microsoft Visual Basic 2010*

*Visual Ba*sic 2010 adalah reinkarnasi dari bahasa Visual Basic yang sangat populer dan telah dilengkapi dengan fitur serta fungsi yang setara dengan bahasa tingkat tinggi lainnya seperti C++.

Anda dapat menggunakan *Visual Basic* 2010 untuk membuat aplikasi *Windows, mobile, Web, dan Office* yang kompleks dengan menggunakan kode yang anda tulis, atau kode yang telah ditulis oleh orang lain dan kemudian dimasukkan ke dalam program anda.

Visual Basic menyediakan beberapa *tools* dan fitur canggih yang memungkinkan anda untuk menulis kode, menguji dan mesnjalankan program tunggal atau terkadang serangkaian program yang terkait dengan satu aplikasi. (Lee, C, 2014 : 1)

Microsoft Visual Basic 2010 merupakan salah satu bagian dari produk pemrograman yang dikeluarkan oleh *Microsoft*, yaitu *microsoft visual studio* 2010. *Microsoft visual studio* 2010 merupakan produk lingkungan terintegrasi atau IDE yang dikeluarkan oleh *microsoft*.

Visual Basic 2010 merupakan versi perbaikan dan pengembangan dari versi pendahulunya, yaitu *visual basic* 2008. Bahasa *Visual Basic* 2010 awalnya berasal dari bahasa pemrograman yang sangat populer dikalangan *programmer*, yaitu bahasa *Basic.* (Rivai Hidayat ; Vol. 1 No. 2)