

**Tugas II IF3058 Kriptografi**  
**Sem. II Tahun 2008/2009**

**Perancangan, Implementasi, dan Aplikasi**  
**Algoritma Enkripsi *Block Cipher***

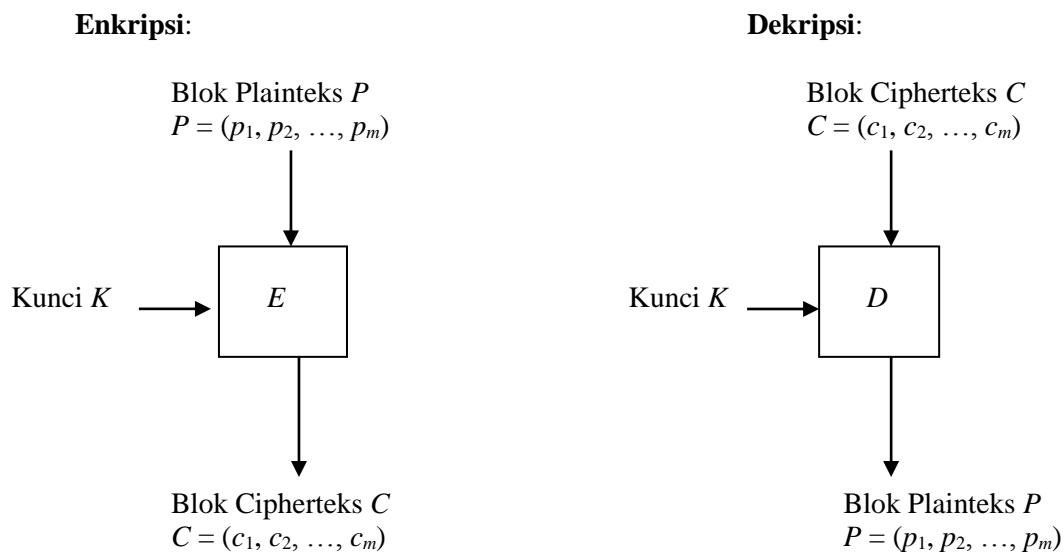
**Tujuan:**

1. Merancang algoritma enkripsi *block cipher*
2. Mengimplementasikan algoritma yang dihasilkan pada point 1 sebagai aplikasi *desktop* dengan mode *ECB (Electronic Code Book)*, *CBC (Cipher Block Chaining)*, dan *CFB (Cipher Feedback)*  $n$ -bit untuk blok data  $n$ -bit.
3. Mengaplikasikan algoritma *block cipher* sebagai program *plug-in* pada program aplikasi lain.

**Deskripsi tugas:**

**1. Bagian I : Aplikasi *Desktop* (Nilai maks: 85)**

Pada Tugas ke-2 ini anda berlaku sebagai seorang kriptografer, yaitu orang yang merancang sebuah algoritma enkripsi "baru" seperti *block cipher* yang sudah dipublikasikan (DES, RC5, Rijndael, GOST, Blowfish, dll). Skema algoritma blok *cipher* adalah Gambar 1.

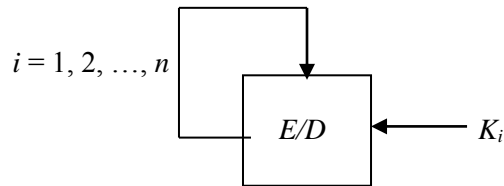


**Gambar 1** Skema enkripsi dan dekripsi pada *cipher* blok

Anda harus merancang fungsi  $E$  dan  $D$  yang sekompleks mungkin sehingga algoritma enkripsi menjadi sangat sukar dipecahkan (menggunakan prinsip *diffusion* dan *confusion* dari Shannon). Fungsi  $E$  dan  $D$  (keduanya sebaiknya identik) harus melibatkan:

1. Operasi substitusi dan transposisi (keduanya beroperasi dalam bit atau dalam hexadesimal). Aturan substitusi dan transposisi diserahkan kepada anda untuk mendefinisikannya (dapat menggunakan tabel substitusi dan tabel permutasi). Rancangan fungsi  $E$  dan  $D$  harus dijelaskan di dalam laporan tugas
2. Untuk menambah kerumitan, maka gunakan struktur Jaringan Feistel.
3. Untuk memperkuat *cipher*, terapkan *cipher* berulang, yaitu untuk setiap blok bit, fungsi  $E$  atau  $D$  dikerjakan sejumlah kali (*round*), seperti pada Gambar 2. Algoritma blok

*cipher* anda yang “baru” harus dapat dioperasikan dalam mode *ECB*, *CBC*, dan *CFB*  $n$ -bit untuk blok data  $n$ -bit (misalnya, untuk *CFB* 64-bit, panjang blok juga 64 bit). Jaringan Feistel digunakan di dalam pengulangan ini.



**Gambar 2** Skema *cipher* berulang untuk setiap blok bit yang dienkripsi/dekripsi

Hal-hal lain yang perlu diperhatikan adalah sebagai berikut:

1. Algoritma kriptografi simetri *block cipher* yang diimplementasikan dapat melakukan proses enkripsi/dekripsi terhadap blok-blok data. Ukuran blok data minimal 64-bit (setara dengan 8 karakter). Ukuran blok adalah kelipatan 8-bit. Panjang blok otomatis diketahui dari panjang kunci yang diberikan oleh pengguna program.
2. Panjang kunci ( $K$ ) harus sama dengan panjang blok yang dispesifikasikan.
3. Khusus untuk mode *CBC*, *initialiazation vector* ( $IV$ ) dibangkitkan secara acak oleh program (pengguna tidak perlu memasukkan  $IV$ , pengguna cukup memasukkan mode blok *cipher* dan kunci saja).
4. Bahasa pemrograman yang digunakan diharapkan menekankan antarmuka yang memudahkan pengguna (*user friendly*) sehingga diharapkan memilih perangkat (*tools*) pemrograman yang mendukung grafis. Lingkungan pemrograman dapat berada pada lingkungan *windows* atau *linux*. Kakas pemrograman yang digunakan bebas.
5. Program yang dibuat mampu menangani:
  - a. Proses enkripsi menerima nama arsip plainteks, kunci ( $K$ ). Kunci  $K$  merupakan *string* alfanumerik yang dibaca dari papan ketik.
  - b. Proses dekripsi menerima nama arsip cipherteks dan kunci ( $K$ ). Ukuran blok, mode, dan  $IV$  seharusnya tidak perlu menjadi masukan untuk proses dekripsi (dengan kata lain, ukuran blok, mode, dan  $IV$  sebaiknya disimpan di dalam *header* arsip cipherteks. **Jangan menyimpan kunci di dalam arsip cipherteks!**).
  - c. Arsip yang dienkripsi adalah sembarang arsip dengan format apa pun (arsip *text*, arsip *word*, arsip *spread sheet*, arsip gambar, arsip *database*, *executable file*, dan sebagainya).
  - d. Menampilkan dan menyimpan arsip hasil enkripsi dan hasil dekripsi. Jadi, anda harus juga membuat *editor* sederhana yang hanya berfungsi menampilkan karakter-karakter hasil enkripsi/dekripsi (tidak dapat melakukan *editing*). Perhatikan bahwa jika arsip yang dienkripsi bukan arsip *text*, maka hasil enkripsinya tidak dapat dibuka oleh program aplikasi yang bersesuaian karena *header file* juga ikut terenkripsi. Namun karakter-karakter hasil enkripsinya masih dapat ditampilkan ke editor sederhana di atas. Khusus arsip *text* (tanpa format), hasil enkripsi maupun dekripsinya dapat dibuka oleh editor sederhana ini tanpa masalah. Contoh program dari angkatan sebelumnya dapat dijadikan acuan (*download* berkas *exe*-nya di <http://mail.informatika.org/~rinaldi>).

6. Berkas *executable* yang didekripsikan harus dapat di-*run* kembali, berkas gambar (*image*) hasil dekripsi harus dapat dibuka kembali oleh aplikasi gambar, berkas musik/video hasil dekripsi harus dapat dimainkan kembali oleh *media player*.

## 2. Bagian II: Program *Add-in* (Nilai maks: 20)

Aplikasikan *block cipher* anda (dengan salah satu mode yang anda tetapkan -- ECB, CBC, atau CFB) menjadi sebuah aplikasi *add-in* (atau *plug-in*) pada *salah satu* dari program aplikasi berikut:

- a. *Notepad*
- b. *Microsoft Word*
- c. *Yahoo! Messenger*

Contoh yang sudah pernah dikembangkan oleh mahasiswa IF (Agus Hilman Majid, IF 2000) adalah program *add-in* pada aplikasi *Microsoft Word*:



Dengan meng-klik ikon *Encryp* atau *Decrypt*, maka dokumen Word yang sedang dibuka dapat dienkripsi dan disimpan. Sebaliknya, jika meng-klik ikon *Decrypt*, maka dikumen yang terenkripsi dapat didekripsi kembali.

Berikut ini dikutip dari laporan Tugas Akhir Agus Hilman Majid:

Untuk implementasi *add-in*, anda dapat menggunakan *Visual Studio Tools for Office (VISTO)*. *VISTO* adalah *framework* aplikasi berbasis *.NET* untuk membangun aplikasi di atas *Office*, dalam hal ini menjadi *add-i* aplikasi di dalam program pengolah kata. Untuk bisa menggunakan *VISTO* harus diinstalasi lebih dahulu kemudian akan ada tipe *project* baru pada *Visual Studio* yaitu jenis *Office Project*.

Untuk ElGamal *Add-in* ini, yang termasuk kategori *Word Add-in Project*, *VSTO* meng-*generate* kelas *ThisAddin* yang berisi dua *method* utama, yaitu:

1. *ThisAddIn\_Startup*

*Method* yang dijalankan ketika aplikasi *Microsoft Word* dibuka.

2. *ThisAddIn\_Shutdown*

*Method* yang dijalankan ketika aplikasi *Microsoft Word* ditutup.

Berikut ini adalah contoh isi kelas *ThisAddin default* yang di-*generate* oleh *VSTO* :

```
using System;
using System.Windows.Forms;
using Microsoft.VisualStudio.Tools.Applications.Runtime;
using Word = Microsoft.Office.Interop.Word;
using Office = Microsoft.Office.Core;

namespace WordAddIn2
{
    public partial class ThisAddIn
    {
        private void ThisAddIn_Startup(object sender, System.EventArgs e)
        {
            // isi disini kode yang akan dijalankan ketika MS Word dibuka
        }

        private void ThisAddIn_Shutdown(object sender, System.EventArgs e)
        {
            // isi disini kode yang akan dijalankan ketika MS Word ditutup
        }

        #region VSTO generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InternalStartup()
        {
            this.Startup += new System.EventHandler(ThisAddIn_Startup);
            this.Shutdown += new System.EventHandler(ThisAddIn_Shutdown);
        }

        #endregion
    }
}
```

Untuk implementasi ElGamal Word Add-in ini, pada *method ThisAddIn\_Startup* dilakukan pembuatan *toolbar* baru yang ‘ditempelkan’ pada *toolbar Microsoft Word* dengan menggunakan *method CreateToolBar*. *Method CreateToolBar* ini juga melakukan *assign* terhadap masing-masing *button* yang ada pada *toolbar* tersebut untuk melakukan *method* sesuai dengan yang telah ditentukan. Untuk jelasnya bisa dilihat di lampiran B.

## Prosedur Pengerjaan

1. Tugas dikerjakan secara berkelompok (1 kelompok @ 3 orang).
2. Waktu pengumpulan tugas: paling lambat 20 Maret 2009 sebelum pukul 17.00 di Lab IRK). Terlambat menyerahkan tugas, nilai = 0.
3. Yang diserahkan pada saat pengumpulan antara lain:
  - a. Disket atau CD yang berisi program sumber (*source code*), arsip siap eksekusi (*executable file*) (termasuk semua *.dll* jika ada), dan arsip-arsip contoh untuk enkripsi/dekripsi.
  - b. Laporan yang memiliki sistematika sebagai berikut :
    - i. Teori singkat (kriptografi, terutama blok cipher, mode *ECB*, *CBC*, dan *CFB*).
    - ii. Perancangan dan Implementasi, termasuk : rancangan fungsi *E* dan *D* yang anda usulkan, modularisasi program, program *add-in*, struktur data, keterangan tentang *header* arsip cipherteks, antarmuka, lingkungan pengembangan, dll. Cantumkan juga pembagian tugas antar anggota kelompok dalam bab ini.
    - iii. Pengujian program dan analisis hasil. Uji program dengan bermacam-macam arsip. Lakukan juga pengujian untuk mengukur tingkat keamanan algoritma (misal: pengubahan 1 bit plainteks/cipherteks, penambahan blok cipherteks semu, penghilangan satu/lebih blok cipherteks, dsb. Anda boleh menggunakan aplikasi lain untuk melakukan pengubahan tersebut, seperti Edit Plus, Ultra Edit, Norton Utilities, dsb)
    - iv. Kesimpulan dari hasil implementasi.

Laporan dikumpulkan dalam bentuk *hard copy* dan *soft copy* dengan format \*.pdf .

4. Penilaian tugas dilakukan pada saat demo.