Aplikasi Teknik Enkripsi Dan Dekripsi File Dengan Algoritma Blowfish Pada Perangkat Mobile Berbasis Android

Siswo Wardoyo, Rian Fahrizal , Zaldi Imanullah[1]
Jurusan Teknik Elektro, Universitas Sultan Ageng Tirtayasa Cilegon, Indonesia

Abstrak *- Kriptografi merupakan salah satu cara yang digunakan untuk mengamankan data dalam bentuk file dengan mengenkripsi file sehingga orang lain tidak berhak mengetahui file yang sifatnya pribadi dan rahasia. Salah satu metode kriptografi adalah algoritma Blowfish yang merupakan algoritma yang menggunakan kunci simetris untuk melakukan enkripsi dan dekripsi. Aplikasi yang dibangun ini dapat melakukan enkripsi file berbentuk gambar, video, dan dokumen. Aplikasi ini dapat berjalan pada ponsel yang minimal memiliki sistem operasi Android versi 2.3. Perangkat lunak yang digunakan untuk membangun aplikasi ini adalah Eclipse. Hasil dari penelitian ini menunjukkan bahwa aplikasi yang dibangun mampu melakukan enkripsi dan dekripsi dengan baik. Hasil enkripsi file membuat file menjadi tidak diketahui lagi maknanya. Dengan menggunakan kunci berjumlah 72 bit atau 9 karakter dibutuhkan waktu* $1{,}49x10^{8}$ *tahun untuk membongkarnya dengan kecepatan komputasinya adalah* 10^{6} *key/sec. Panjang kunci dan kerahasiaan kunci mempengaruhi dari tingkat keamanan file.*

Kata kunci*: kriptografi, algoritma Blowfish, kunci simteris, Android, enkripsi, dekripsi.*

Abstract *- Cryptography is one of the means used to secure the data in the file to encrypt the file so that others are not entitled to know the file that is private and confidential. One method of cryptography is the Blowfish algorithm is an algorithm that uses a symmetric key for encryption and decryption. Applications built can perform file encryption in the form of pictures, videos, and documents. These applications can run on any phone that has a minimum of the Android operating system version 2.3. The software used to build this application is Eclipse. The results of this study indicate that the application is built is able to perform encryption and decryption with well. The results create a file encryption file becomes no longer known meaning. By using the numbered keys 72 bit or 9 characters 1,49x108 takes years to dismantle the computational speed is 106 keys / sec. Key length and key secrecy influence of the security level of the file.*

Keywords: *cryptography, Blowfish algorithm, key simteris, Android, encryption, decryption.*

I. PENDAHULUAN

Pada saat ini, penggunaan perangkat *mobile* sudah menjadi *trend* di masyarakat dunia. Perangkat *mobile* yang beredar saat ini sangat menakjubkan. Teknologi *mobile* berkembang sangat pesat sehingga mempunyai dampak dalam meningkatkan efektifitas dan efisiensi dalam melakukan setiap pekerjaan. Sistem operasi untuk perangkat *mobile* semakin berkembang. Android merupakan salah satu sistem operasi *mobile* buatan Google yang kini sangat populer dan banyak digunakan orang-orang. Android juga merupakan sistem operasi yang berbasis perangkat lunak yang dapat dikembangkan secara terbuka (*open source*) sehingga banyak pengembang yang kini turut serta ikut mengembangkan aplikasi untuk Android.

Perangkat *mobile* yang dijalankan oleh Android tidak hanya menjadi alat komunikasi saja, melainkan dapat menjadi *self-assistant*, dapat digunakan untuk gaming, *browsing*, pemutar musik dan video, memotret gambar dan merekam video, media penyimpanan, modem, bahkan sampai *internet banking*. Perangkat *mobile* sekarang memiliki memori eksternal yang memiliki kapasitas cukup besar dan akan terus meningkat kapasitasnya. Dalam suatu media penyimpanan, terdapat
suatu data penting atau rahasia yang tidak semua orang boleh mengetahuinya. Data-data penting yang hanya boleh diketahui oleh pemiliknya saja antara lain dokumen, video, foto, akun email, akun jejaring sosial, akun kartu kredit, akun internet banking. Apalagi saat proses pengiriman file melalui media internet maupun saat perangkat *mobile* itu hilang, membuat pemiliknya sangat riskan kehilangan data-data pentingnya.

Oleh karena itu, dalam penelitian ini akan coba dibuat sebuah aplikasi pengamanan data berupa dokumen, gambar, dan video dengan menggunakan metode algoritma Blowfish untuk mengenkripsi data yang berjalan pada sistem operasi Android sehingga pemilik merasa aman untuk menyimpan datanya. Aplikasi ini sekaligus sebagai aplikasi alternatif

keamanan yang sebelumnya sudah terdapat di perangkat Android.

II. DASAR TEORI

A. Kriptografi

Kriptografi (*cryptography*) berasal dari Bahasa Yunani: "*cryptos*" artinya "*secret*" (rahasia), sedangkan "*graphein*" artinya "*writing*" (tulisan). Jadi kriptografi adalah ilmu dan seni untuk menjaga kerahasiaan pesan dengan cara menyandikannya ke dalam bentuk yang tidak dapat dimengerti lagi maknanya.

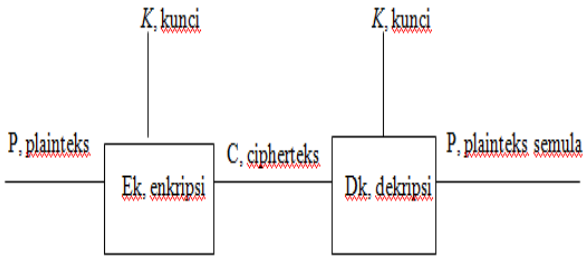
Dalam menjaga kerahasiaan data dengan kriptografi, data sederhana yang dikirim (*plaintext*) diubah ke dalam bentuk data sandi (*ciphertext*), kemudian data sandi tersebut hanya dapat dikembalikan kebentuk data sebenarnya hanya dengan menggunakan kunci (*key*) tertentu yang dimiliki oleh pihak yang sah saja. Tentunya hal ini menyebabkan pihak lain yang tidak memiliki kunci tersebut tidak akan dapat membaca data yang sebenarnya sehingga dengan kata lain data akan tetap terjaga kerahasiannya.

B. Plaintext dan Ciphertext

Pesan (*message*) adalah data atau informasi yang dapat dibaca dan dimengerti maknanya. Nama lain untuk pesan adalah plainteks (*plaintext*) atau teks jelas (*cleartext*). Pesan dapat berupa data atau informasi yang dikirim (melalui kurir, saluran telekomunikasi) atau yang disimpan di dalam media perekaman (kertas, *storage*). Pesan yang tersimpan tidak hanya berupa teks, tetapi juga dapat berbentuk citra (*image*), suara/bunyi (*audio*), *archieve*, dan *video*. Agar pesan tidak dapat dimengerti maknanya oleh pihak lain, maka pesan perlu disandikan kebentuk lain yang tidak dapat dipahami. Bentuk pesan yang tersandi disebut cipherteks (*ciphertext*) atau kriptogram (*cryptogram*). Cipherteks harus dapat ditransformasikan kembali menjadi plainteks semula agar pesan yang diterima bisa dibaca.

C. Enkripsi dan dekripsi

Proses menyandikan plainteks menjadi cipherteks disebut enkripsi (*encryption*) atau *enciphering* (standar nama menurut ISO 7498-2). Sedangkan proses mengembalikan cipherteks menjadi plainteks semula dinamakan dekripsi (*decryption*) atau *deciphering* (standar nama menurut ISO 7498-2). Enkripsi dan dekripsi dapat diterapkan baik pada pesan yang dikirim maupun pada pesan tersimpan. Istilah *encryption of data in motion* mengacu pada enkripsi pesan yang ditransmisikan melalui saluran komunikasi, sedangkan istilah *encryption of data at-rest* mengacu pada enkripsi dokumen yang disimpan di dalam *storage*. Contoh *encryption of data in motion* adalah pengiriman nomor PIN dari mesin ATM ke komputer *server* di kantor bank pusat. Contoh *encryption of data at-rest* adalah enkripsi file basis data di dalam *harddisk*. Gambar 1 menunjukkan skema enkripsi dan dekripsi.



Gambar 1. Skema enkripsi dan dekripsi

D. Tujuan Kriptografi

Ada empat tujuan mendasar dari ilmu kriptografi ini yang juga merupakan aspek keamanan informasi yaitu:

1. Kerahasiaan adalah layanan yang digunakan untuk menjaga isi dari informasi dari siapapun kecuali yang memiliki otoritas atau kunci rahasia untuk membuka/mengupas informasi yang telah disandi.
2. Integritas data adalah berrhubungan dengan penjagaan dari perubahan data secara tidak sah. Untuk menjaga integritas data, sistem harus memiliki kemampuan untuk mendeteksi manipulasi data oleh pihak-pihak yang tidak berhak, antara lain penyisipan, penghapusan, dan pensubsitusian data lain kedalam data yang sebenarnya.
3. Autentifikasi adalah berhubungan dengan identifikasi atau pengenalan, baik secara kesatuan sistem maupun informasi itu sendiri. Dua pihak yang saling berkomunikasi harus saling memperkenalkan diri. Informasi yang dikirimkan melalui kanal harus diautentikasi keaslian, isi datanya, dan waktu pengiriman.
4. Nirpenyangkalan adalah usaha untuk mencegah terjadinya penyangkalan terhadap pengiriman atau terciptanya suatu informasi oleh yang orang mengirimkan atau membuat.

E. Algoritma Blowfish

Blowfish alias "*OpenPGP.Cipher.4*" merupakan enkripsi yang termasuk dalam golongan *Symmetric Cryptosystem* , metode enkripsinya mirip dengan DES (DES *like* Cipher) diciptakan oleh seorang Cryptanalyst bernama Bruce Schneier Presiden perusahaan Counterpane Internet Security, Inc (Perusahaan konsultan tentang kriptografi dan keamanan komputer) dan dipublikasikan tahun 1994. Dibuat untuk digunakan pada komputer yang mempunyai mikroprosesor besar (32-*bit* keatas dengan *cache* data yang besar). Blowfish dikembangkan untuk memenuhi kriteria desain yang cepat dalam implementasinya dimana pada keadaan optimal dapat mencapai 26 *clock cycle* per *Byte*, kompak dimana dapat berjalan pada memori kurang dari 5 KB, sederhana dalam algoritmanya sehingga mudah diketahui kesalahannya, dan keamanan yang variabel dimana panjang kunci bervariasi (minimum 32 *bit*, maksimum 448 *bit*, *multiple* 8 *bit*, *default* 128 *bit*).

Blowfish dioptimasikan untuk berbagai aplikasi dimana kunci tidak sering berubah, seperti pada jaringan komunikasi atau enkripsi file secara otomatis. Dalam pengimplementasiannya dalam computer bermikroprosesor 32-*bit* dengan *cache* data yang besar (Pentium dan Power PC) Blowfish terbukti jauh lebih cepat dari DES. Tetapi Blowfish tidak cocok dengan aplikasi dengan perubahan kunci yang sering atau sebagai fungsi *hash* satu arah seperti pada aplikasi *packet switching*. Blowfish pun tidak dapat digunakan pada aplikasi kartu pintar (*smart card*) karena memerlukan memori yang besar. Algoritma Blowfish terdiri atas dua bagian: *key expansion* dan enkripsi data.

1. *Key Expansion*

Berfungsi merubah kunci (minimum 32-*bit*, maksimum 448-*bit*) menjadi beberapa *array* subkunci (*subkey*) dengan total 4168 *Byte* (18x32-*bit* untuk P-*array* dan 4x256x32-*bit* untuk S-*box* sehingga totalnya 33344 *bit* atau 4168 *Byte*). Kunci disimpan dalam K-*array*:

K1, K2, ... Kj $1 \leq j \leq 14$

Kunci-kunci ini yang dibangkitkan (*generate*) dengan menggunakan subkunci yang harus dihitung terlebih dahulu sebelum enkripsi atau dekripsi data. Sub-sub kunci yang digunakan terdiri dari:

P-*array* yang terdiri dari 18 buah 32-*bit* subkunci,

P1, P2, ..., P18

S-*box* yang terdiri dari 4 buah 32-*bit*, masing-masing memiliki 256 entri:

$S_{1,0}, S_{1,1}, S_{1,2}, S_{1,3}, \ldots\ldots\ldots\ldots S_{1,255}$
$S_{2,0}, S_{2,1}, S_{2,2}, S_{2,3}, \ldots\ldots\ldots\ldots S_{2,255}$
$S_{3,0}, S_{3,1}, S_{3,2}, S_{3,3}, \ldots\ldots\ldots\ldots S_{3,255}$
$S_{4,0}, S_{4,1}, S_{4,2}, S_{4,3}, \ldots\ldots\ldots\ldots S_{4,255}$

Langkah-langkah perhitungan atau pembangkitan subkunci tersebut adalah sebagai berikut:

a. Inisialisasi P-*array* yang pertama dan juga empat S-*box*, berurutan, dengan *string* yang telah pasti. *String* tersebut terdiri dari digit-digit heksadesimal dari phi, tidak termasuk angka tiga di awal.
b. XOR-kan P1 dengan 32-*bit* awal kunci, XOR-kan P2 dengan 32-*bit* berikutnya dari kunci, dan seterusnya untuk semua *bit* kunci. Ulangi siklus seluruh *bit* kunci secara berurutan sampai seluruh P-*array* ter-XOR-kan dengan *bit-bit* kunci. Atau jika disimbolkan : P1 = P1 ⊕ K1, P2 = P2 ⊕ K2, P3 = P3 ⊕ K3, . . . P14 = P14 ⊕ K14, P15 = P15 ⊕ K1, . . . P18 = P18 ⊕ K4. Keterangan : ⊕ adalah simbol untuk XOR.
c. Enkripsikan *string* yang seluruhnya nol dengan algoritma Blowfish, menggunakan subkunci yang telah dideskripsikan pada langkah 1 dan 2.
d. Gantikan P1 dan P2 dengan keluaran dari langkah 3.
e. Enkripsikan keluaran langkah 3 menggunakan algoritma Blowfish dengan subkunci yang telah dimodifikasi.
f. Gantikan P3 dan P4 dengan keluaran dari langkah 5.
g. Lanjutkan langkah-langkah di atas, gantikan seluruh elemen P-*array* dan kemudian keempat S-*box* secara berurutan, dengan hasil keluaran algoritma Blowfish yang terus-menerus berubah.

Total keseluruhan, terdapat 521 iterasi untuk menghasilkan subkunci-subkunci dan membutuhkan memori sebesar 4KB.

2. Enkripsi Data

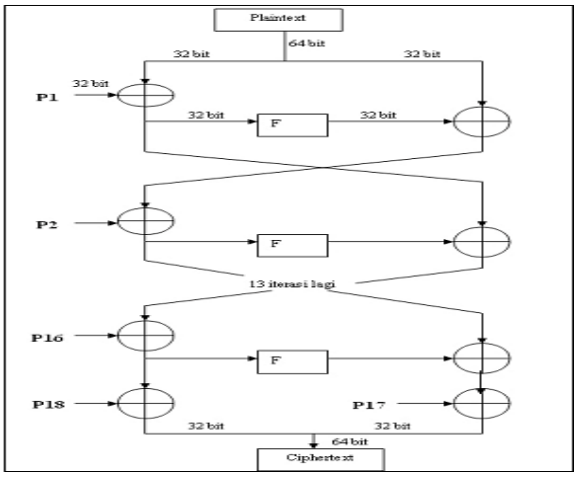
Terdiri dari iterasi fungsi sederhana (*Feistel Network*) sebanyak 16 kali putaran (iterasi), masukannya adalah 64 *bit* elemen data X. Setiap putaran terdiri dari permutasi kunci *dependent* dan substitusi kunci dan data *dependent*. Semua operasi adalah penambahan (*addition*) dan XOR pada variabel 32-*bit*. Operasi tambahan lainnya hanyalah empat penelusuran tabel *array* berindeks untuk setiap putaran. Langkahnya adalah seperti berikut:

a. Bagi X menjadi dua bagian yang masing-masing terdiri dari 32-*bit*: XL, XR.
b. Lakukan langkah berikut
   For i = 1 to 16:
   XL = XL ⊕ Pi
   XR = F(XL) ⊕ XR
   Tukar XL dan XR
c. Setelah iterasi ke-16, tukar XL dan XR lagi untuk melakukan membatalkan pertukaran terakhir.
d. Lalu lakukan
   XR = XR ⊕ P17
   XL = XL ⊕ P18
e. Terakhir, gabungkan kembali XL dan XR untuk mendapatkan cipherteks.

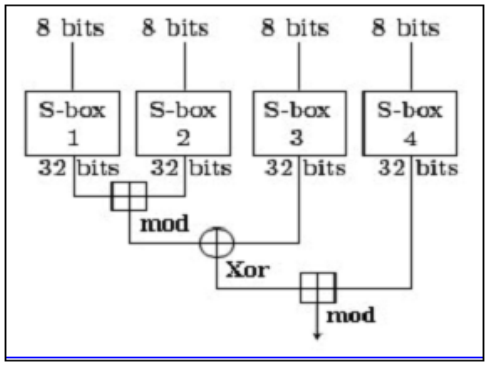
Untuk lebih jelasnya blok diagram enkripsi algoritma Blowfish dapat dilihat pada Gambar 2.

Fungsi F adalah sebagai berikut: bagi x_L dalam empat kuarter 8-*bit* yaitu a, b, c dan d seperti Gambar 3 maka:

$F(x_L) = ((S_{1,a} + S_{2,b} \bmod 2^{32}) \oplus S_{3,c}) + S_{4,d} \bmod 2^{32}$



Gambar 2. Blok diagram algoritma enkripsi Blowfish

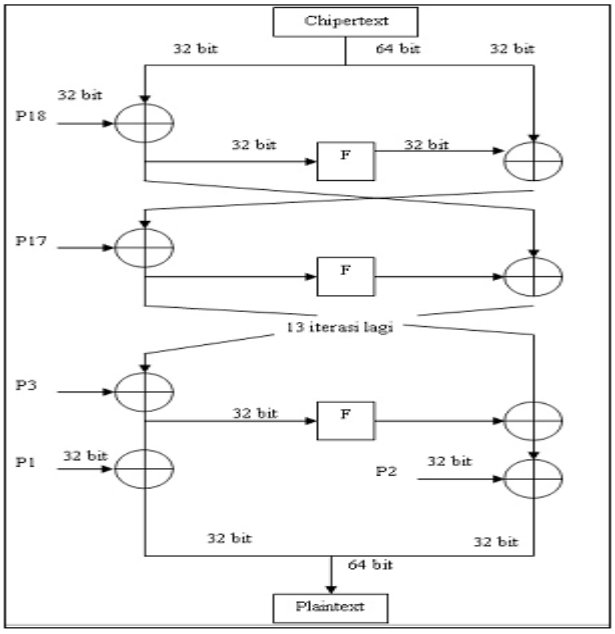


Gambar 3. Fungsi F

Dekripsi sama persis dengan enkripsi, kecuali bahwa P1, P2,…, P18 digunakan pada urutan yang berbalik (*reverse*). Algoritmanya dapat dinyatakan sebagai berikut:

for i = 1 to 16 do
XRi = XLi-1 ⊕ P19-i;
XLi = F[XRi] ⊕ XRi-1;
XL17 = XR16 ⊕ P1;
XR17 = XL16 ⊕ P2;

Untuk lebih jelasnya blok diagram dekripsi algoritma Blowfish dapat dilihat pada Gambar 4.



Gambar 4. Blok diagram dekripsi Blowfish

f. Android

Android adalah kumpulan perangkat lunak yang ditujukan bagi perangkat bergerak mencakup sistem operasi, *middleware*, dan aplikasi kunci. *Android Standart Development Kid* (SDK) menyediakan perlengkapan dan *Application Programming Interface* (API) yang diperlukan untuk mengembangkan aplikasi pada *platform* Android menggunakan bahasa pemrograman Java.

Android dikembangkan oleh Google bersama Open Handset Allience (OHA) yaitu aliansi perangkat selular terbuka yang terdiri dari 47 perusahaan *hardware*, *software* dan perusahaan telekomunikasi ditujukan untuk mengembangkan standar terbuka bagi perangkat selular.

Pada mulanya terdapat berbagai macam sistem operasi pada perangkat selular, diantaranya sistem operasi Symbian, Microsoft Windsos Mobile, Mobile Linux, iPhone, dan sistem operasi lainnya. Namun diantara sistem operasi yang ada belum mendukung standar dan pener*bit*an API yang dapat dimanfaatkan secara keseluruhan dan dengan biaya yang murah. Kemudian Google ikut berkecimpung di dalamnya dengan *platform* Android, yang menjanjikan keterbukaan, keterjangkauan, *open source*, dan *framework* berkualitas.

Pada tahun 2005, Google mengakuisisi perusahaan Android Inc. untuk memulai pengembangan *platform* Android. Dimana terlibat dalam pengembangan ini Andy Rubin, Rich Miner, Nick Sears, dan Chris White. Pada pertengahan 2007 sekelompok pemimpin industri bersama-sama membentuk aliansi perangkat selular terbuka, Open Handset Alliance (OHA). Bagian dari tujuan aliansi ini adalah berinovasi dengan cepat dan menanggapi kebutuhan konsumen dengan lebih baik, dengan produk awalnya adalah *platform* Android.

Dimana Android dirancang untuk melayani kebutuhan operator telekomunikasi, manufaktur *handset*, dan pengembang aplikasi. Android pertama kali diluncurkan pada 5 November 2007, dan *smartphone* pertama yang menggunakan sistem operasi Android adalah HTC Dream yang dirilis pada 22 Oktober 2008. Hingga saat ini Android telah merilis beberapa versi Android untuk menyempurnakan versi sebelumnya. Selain berdasarkan penomoran, pada setiap versi Android terdapat kode nama berdasarkan nama-nama kue.

Berikut ini adalah macam-macam versi dari Android:

1. Android versi 1.1
2. Android versi 1.5 (Cupcake)
3. Android versi 1.6 (Donut)
4. Android versi 2.0/2.1 (Eclair)
5. Android versi 2.2 (Froyo: Frozen Yoghurt)
6. Android versi 2.3 (Gingerbread)
7. Android versi 3.0/3.1 (Honeycomb)
8. Android versi 4.0 (ICS: Ice Cream Sandwich)
9. Android versi 4.1 (Jelly Bean)
10. Android versi 4.2 (Jelly Bean)
11. Android versi 4.3 (Jelly Bean)
12. Android versi 4.4 (KitKat)

III. METODOLOGI PENELITIAN

A. Perancangan Antarmuka (Interface)

Perancangan *interface* adalah bagian yang penting dalam aplikasi karena yang pertama kali dilihat ketika aplikasi dijalankan adalah *interface* aplikasi. Perancangan antarmuka sendiri terdiri dari perancangan antarmuka menu utama, perancangan antarmuka *about* dan perancangan antarmuka *help*.

Perancangan antarmuka sendiri menggunakan bahasa XML.

1. Perancangan Antarmuka Menu Utama

Perancangan antarmuka menu utama adalah tampilan yang terdiri dari bagian *input file*, *browse*, *password*, *encrypt*, *decrypt*, *home*, *about*, *help*.

2. Perancangan Antarmuka About

Perancangan antarmuka *about* adalah tampilan yang berisi mengenai penjelasan tentang aplikasi yang telah dibuat.

3. Perancangan Antarmuka Help

Perancangan antarmuka *help* adalah tampilan yang berisi mengenai penjelasan tentang cara melakukan enkripsi dan dekripsi file.

B. Pembuatan Kelas Encryption,java

Kelas *encryption* ini adalah kelas yang digunakan untuk melakukan proses enkripsi dan dekripsi file. Pada pembuatan program ini, digunakan *software* Eclipse dengan menggunakan bahasa Java. Tabel 1 memperlihatkan pembagian modul-modul program beserta keterangan dari modul-modul yang telah dibuat. Tampilan pada aplikasi di OS Android diatur oleh file XML yang terdapat dalam folder res/layout. Tabel 2 memperlihatkan daftar tampilan *layout* yang telah dibuat.

Tabel 1. Modul-Modul Program dari Aplikasi Enkripsi dan Dekripsi File

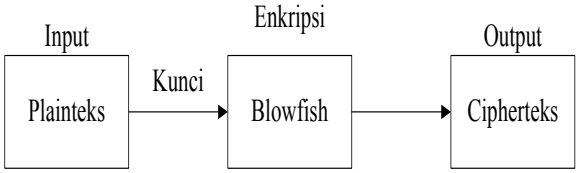
Nama Modul	Keterangan
AboutActivity.java	*Activity* yang menampilkan menu *about*
DashBoardActivity.java	*Activity* yang mengatur perpindahan antar tampilan dan perintah *exit*
FileChooser.java & File*Array*Adapter.java	Mengatur menu *browse* file serta memilih file pada memori *handphone*
HelpActivity.java	*Activity* yang menampilkan menu *help*
MainActivity.java	*Activity* yang menampilkan menu di halaman utama atau awal
Item.java	Menyimpan variabel dan fungsi-fungsi Item
Encryption.java	Fungsi enkripsi dan dekripsi berdasarkan algoritma Blowfish

Tabel 2. Nama File *Layout* Beserta Fungsinya

Nama File	Fungsi
about.xml	Mengatur tampilan pada halaman *about*
file_view.xml	Mengatur tampilan *browse* file
footer.xml	Mengatur tampilan menu di bawah
header.xml	Mengatur tampilan menu di atas
help.xml	Mengatur tampilan pada halaman *help*
main.xml	Mengatur tampilan pada halaman awal

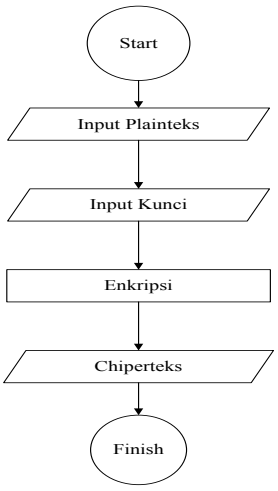
C. Proses Enkripsi File

Pada tahap ini merancang program untuk mengenkripsi file menggunakan algoritma Blowfish. Diagram blok sistem untuk proses enkripsi file diperlihatkan pada Gambar 5.



Gambar 5. Diagram Blok Proses Enkripsi File

Gambar 6 memperlihatkan *flowchart* proses enkripsi file secara keseluruhan. Untuk melakukan proses enkripsi file hal pertama yang dilakukan adalah input plainteks berupa file yang telah ditentukan ukuran dan format filenya.

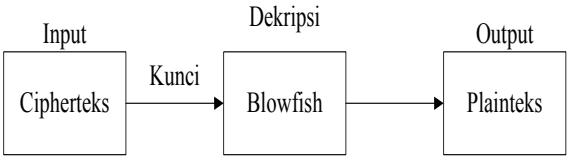


Gambar 6. *Flowchart* Proses Enkripsi File

Kunci yang digunakan untuk proses enkripsi file bisa berupa gabungan angka, huruf dan karakter khusus sesuai keinginan dari penggunanya. Blowfish sendiri menggunakan kunci simetris dimana kunci untuk enkripsi dan dekripsi sama. Setelah proses enkripsi file berhasil maka hasil outputnya berupa cipherteks yang sudah tidak dapat dimengerti maknanya.

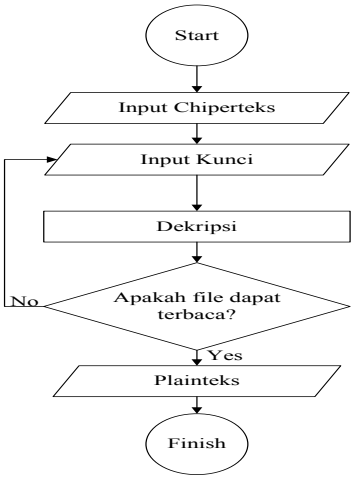
D. Proses Dekripsi File

Pada tahap ini merancang program untuk mendekripsi file menggunakan algoritma Blowfish. Diagram blok sistem untuk proses dekripsi file diperlihatkan pada Gambar 7.



Gambar 7. Diagram Blok Proses Dekripsi File

Gambar 8 memperlihatkan *flowchart* proses enkripsi file secara keseluruhan. Untuk melakukan proses dekripsi file hal pertama yang dilakukan adalah input cipherteks berupa file yang telah ditentukan ukuran dan format filenya.



Gambar 8. *Flowchart* Proses Dekripsi File

Kunci yang digunakan untuk proses enkripsi file bisa berupa gabungan angka, huruf dan karakter khusus sesuai keinginan dari penggunanya. Blowfish sendiri menggunakan kunci simetris dimana kunci untuk enkripsi dan dekripsi sama. Setelah proses dekripsi file berhasil maka hasil outputnya berupa plainteks yang bisa dimengerti maknanya.

IV. HASIL DAN PEMBAHASAN

A. Pengujian Antarmuka Aplikasi

Pengujian perancangan antarmuka aplikasi dilakukan dengan tujuan untuk melihat hasil antarmuka aplikasi enkripsi dan dekripsi file pada *handphone* Android. Pada pengujian antarmuka ini terdiri dari antarmuka menu utama, antarmuka *about*, antarmuka *help*. Untuk antarmuka aplikasi menggunakan bahasa pemrograman XML dan bahasa pemrograman Java untuk membuat *source code* enkripsinya.

1. Antarmuka Menu Utama

Saat awal aplikasi dijalankan pada *handphone* berbasis Android seperti pada Gambar 9. Pada tampilan menu utama terdapat bagian-bagian dari inti aplikasi yang telah dibuat. Fungsi dari bagian-bagian tersebut meliputi:

a. *Input File* : pilihan ini untuk memasukan file apa saja yang akan dienkripsi yang terdapat dalam *SD Card*.
b. *Password* : bagian ini untuk memasukan sandi untuk mengenkripsi dan dekripsi file.
c. *Encrypt* : pilihan ini untuk memproses perintah enkripsi yang telah diberikan untuk dieksekusi sehingga file tidak dapat terbaca.
d. *Decrypt* : pilihan ini untuk membalikan file seperti semula sehingga dapat terbaca kembali.
e. *Home* : pilihan ini untuk ke menu utama aplikasi.
f. *About* : pilihan ini untuk ke menu *about*.
g. *Help* : pilihan ini untuk ke menu *help*.
h. *Exit* : pilihan ini untuk keluar aplikasi.

Gambar 9. Antarmuka Menu Utama

Untuk memilih file yang akan dienkripsi, pengguna harus menekan tombol *browse file*. Gambar 10 menunjukan antarmuka untuk memilih file di *SD Card* pada *handphone* berbasis Android. File-file yang akan diujicobakan ditempatkan ke dalam folder "File Ujicoba".

Gambar 10. Antarmuka memilih file

Untuk melakukan proses enkripsi dan dekripsi file pada aplikasi yang telah dibuat, langkah-langkahnya adalah sebagai berikut:

a. Masukkan file yang akan dienkripsi dengan menekan *button browse*.
b. Masukkan *password* dengan panjang *password* yang diinginkan.
c. Tekan tombol *encrypt* untuk mengeksekusinya.

Jika proses enkripsi berhasil akan muncul tulisan *file encrypt* seperti tampilan pada Gambar 11. Langkah-langkah untuk enkripsi file sama dengan dekripsi file. Tekan tombol *decrypt* untuk mengembalikan file seperti semula sehingga bisa terbaca kembali.

Gambar 11. Peringatan jika proses enkripsi berhasil

2. Antarmuka About

Pada saat tombol *About* diklik maka tampilan awal berpindah ke tampilan *About* seperti terlihat pada Gambar 12. Tampilan *About* ini berisi mengenai penjelasan tentang aplikasi yang telah dibuat, fitur-fitur yang terdapat dalam aplikasi, ucapan terimakasih kepada pihak-pihak yang terlibat dalam pembuatan aplikasi dan informasi mengenai pengembang.

Gambar 12. Antarmuka *About*

3. Antarmuka Help

Pada saat tombol *Help* diklik maka akan berpindah ke tampilan *Help* seperti terlihat pada Gambar 13. Tampilan *Help* ini berisi mengenai penjelasan tentang cara penggunaan enkripsi dan dekripsi file.

Gambar 13. Antarmuka *Help*

B. Pengujian Program Aplikasi

Pada bagian ini dilakukan pengujian aplikasi untuk mengenkripsi file dan setelah proses enkripsi file selesai dilakukan akan dilihat hasilnya kemudian dilakukan pengujian dekripsi file untuk mengembalikan file seperti semula. Pengujian dilakukan dengan format file yang berbeda-beda yang umum terdapat pada perangkat Android. Untuk tingkat keamanannya dilakukan terhadap serangan *brute force*.

1. Pengujian Terhadap Ukuran File

Hasil pengujian terhadap file-file yang telah ditentukan sebelumnya dari berbagai macam format file dapat dilihat pada Tabel 3 yang menunjukkan hasil ukuran file setelah dilakukan proses enkripsi file dan Tabel 4 yang menunjukkan hasil ukuran file setelah dilakukan proses dekripsi file.

Tabel 3. Hasil Proses Enkripsi beberapa format file yang berbeda

No	Nama file plainteks	Ukuran file plainteks (*Byte*)	Ukuran file cipherteks (*Byte*)
1	Algoritma dan Flowchart.pptx	1528716	1528720
2	Berapa Lama Waktu Bongkar Password.doc	117248	117256
3	Cara Merubah Background Pas Foto.mp4	4910554	4910560
4	Fellaini.jpg	334713	334720
5	LIST SILABUS CCNA EXPLORATION.txt	1114	1120
6	MAC Random Access 10.pptx	442168	442176
7	Matematika Teknik 2 exel 97-2003.xls	151552	151560

8	Motorola Charm.jpg	201285	201288
9	Paper Aes.pdf	673435	673440
10	Sejarah Adobe Photoshop.pdf	1130348	1130352

Dari hasil pengujian pada Tabel 3 dan Tabel 4 tersebut dapat diketahui bahwa ukuran file setelah dilakukan proses enkripsi akan berbeda hasilnya dibandingkan sebelumnya. Setelah file didekripsi kembali ukuran file akan sama seperti semula. Hal ini menunjukan aplikasi ini dapat bekerja dengan baik. Ukuran file setelah dienkripsi lebih besar dari ukuran asli disebabkan karena adanya proses *padding* yaitu untuk memperoleh blok terakhir sebesar 64 *bit*. Panjang *password*/*key* untuk melakukan pengujian ini adalah 9 karakter yang terdiri dari kombinasi huruf dan angka. *Key* yang digunakan untuk ujicoba adalah qwerty123.

Tabel 4. Hasil Proses Dekripsi beberapa format file yang berbeda

No	Nama file plainteks	Ukuran file cipherteks (*Byte*)	Ukuran file plainteks (*Byte*)
1	Algoritma dan Flowchart.pptx	1528720	1528716
2	Berapa Lama Waktu Bongkar Password.doc	117256	117248
3	Cara Merubah Background Pas Foto.mp4	4910560	4910554
4	Fellaini.jpg	334720	334713
5	LIST SILABUS CCNA EXPLORATION.txt	1120	1114
6	MAC Random Access 10.pptx	442176	442168
7	Matematika Teknik 2 exel 97-2003.xls	151560	151552
8	Motorola Charm.jpg	201288	201285
9	Paper Aes.pdf	673440	673435
10	Sejarah Adobe Photoshop.pdf	1130352	1130348

2. Pengujian Enkripsi Terhadap Beberapa Tipe File

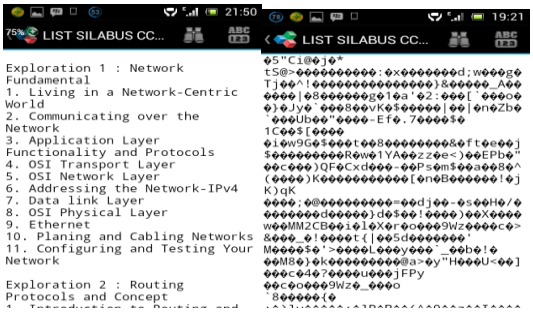
Ujicoba enkripsi dan dekripsi yang telah dilakukan pada file yang berformat (.pptx), (.doc), (.mp4), (.jpg), (.txt), (.xls) dan (.pdf). Di bawah ini merupakan hasil enkripsi dan dekripsi dari format file yang telah ditentukan tersebut.

a. File .txt

Untuk pengujian pada file (.txt) dengan nama file LIST SILABUS CCNA EXPLORATION.txt dengan menggunakan *password* "qwerty123" untuk enkripsi dan dekripsinya. Plainteksnya ditunjukkan pada Gambar 14 (a) dengan ukuran file sebesar 1114 *Byte*. Hasil enkripsinya ditunjukkan pada Gambar 14 (b) dengan ukuran file sebesar 1120 *Byte* atau lebih besar 6 *Byte* dari ukuran aslinya. Hasil enkripsinya berupa simbol-simbol acak yang tidak bisa dibaca kembali.

b. File .jpg

Untuk pengujian pada file (.jpg) dengan nama file Motorola Charm.jpg dengan menggunakan *password* "qwerty123" untuk enkripsi dan dekripsinya. Plainteksnya ditunjukkan pada Gambar 15 (a) dengan ukuran file sebesar 201285 *Byte*. Hasil enkripsinya ditunjukkan pada Gambar 15 (b) dengan ukuran file sebesar 201288 *Byte* atau lebih besar 3 *Byte* dari ukuran aslinya. Hasil enkripsinya berupa gambar yang tidak bisa dilihat kembali.



(a) (b)

Gambar 14. Plainteks (a) dan Cipherteks (b) dari file berformat .txt

(a) (b)

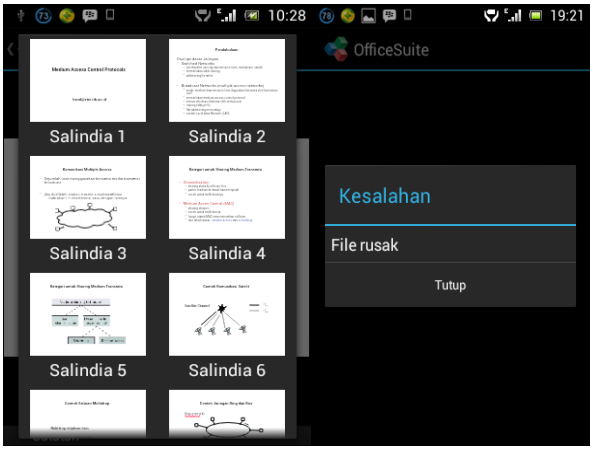
Gambar 15. Plainteks (a) dan Cipherteks (b) dari file berformat .jpg

c. File .pptx

Untuk pengujian pada file (.pptx) dengan nama file MAC Random Access 10.pptx dengan menggunakan *password* "qwerty123" untuk enkripsi dan dekripsinya. Plainteksnya ditunjukkan pada Gambar 16 (a) dengan ukuran file sebesar 442168 *Byte*. Hasil enkripsinya ditunjukkan pada Gambar 16 (b) dengan ukuran file sebesar 442176 *Byte* atau lebih besar 8 *Byte* dari ukuran aslinya. Hasil enkripsinya berupa file yang tidak bisa dibaca kembali.

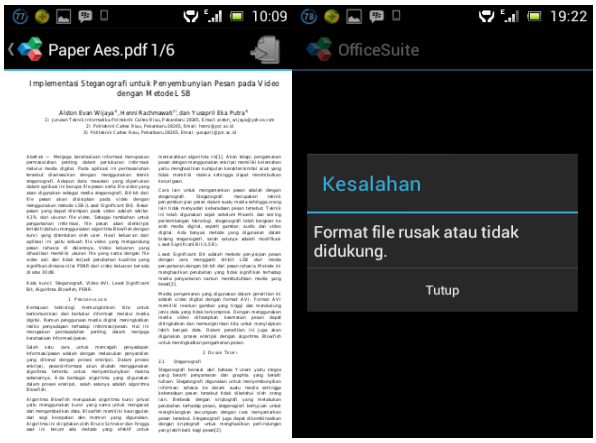
d. File .pdf

Untuk pengujian pada file (.pdf) dengan nama file Paper Aes.pdf dengan menggunakan *password* "qwerty123" untuk enkripsi dan dekripsinya. Plainteksnya ditunjukkan pada Gambar 17 (a) dengan ukuran file sebesar 673435 *Byte*. Hasil enkripsinya ditunjukkan pada Gambar 17 (b) dengan ukuran file sebesar 673440 *Byte* atau lebih besar 5 *Byte* dari ukuran aslinya. Hasil enkripsinya berupa file yang tidak bisa dibaca kembali.



(a) (b)

Gambar 16. Plainteks (a) dan Cipherteks (b) dari file berformat .pptx

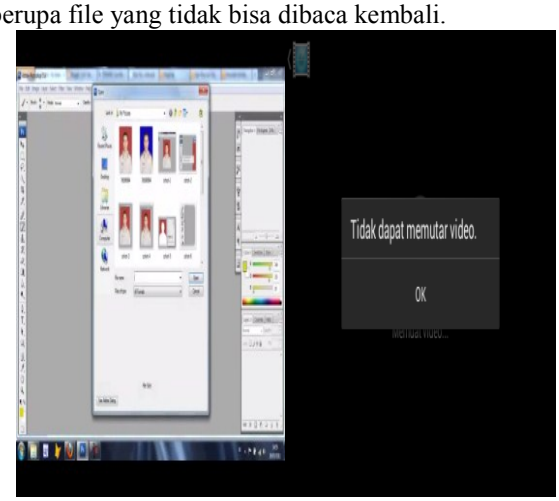


(a) (b)

Gambar 17. Plainteks (a) dan Cipherteks (b) dari file berformat .pdf

e. File .mp4

Untuk pengujian pada file (.mp4) dengan nama file Cara Merubah Background Pas Foto.mp4 dengan menggunakan *password* "qwerty123" untuk enkripsi dan dekripsinya. Plainteksnya ditunjukkan pada Gambar 18 (a) dengan ukuran file sebesar 4910554 *Byte*. Hasil enkripsinya ditunjukkan pada Gambar 18 (b) dengan ukuran file sebesar 4910560 *Byte* atau lebih besar 6 *Byte* dari ukuran aslinya. Hasil enkripsinya berupa file yang tidak bisa dibaca kembali.

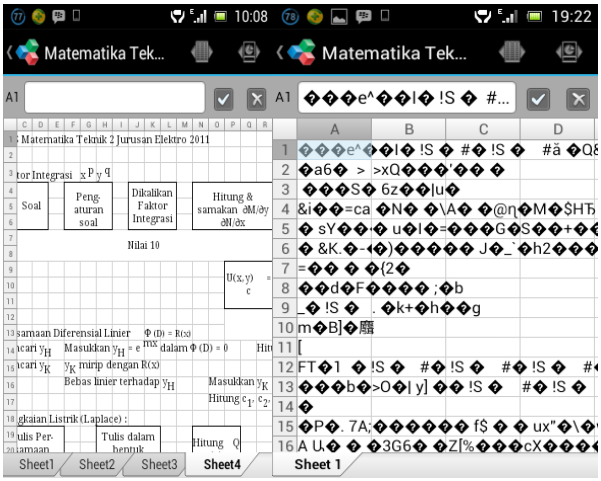


(a) (b)

Gambar 18. Plainteks (a) dan Cipherteks (b) dari file berformat .mp4

f. File .xls

Untuk pengujian pada file (.xls) dengan nama file Matematika Teknik 2 excel 97-2003.xls dengan menggunakan *password* "qwerty123" untuk enkripsi dan dekripsinya. Plainteksnya ditunjukkan pada Gambar 19 (a) dengan ukuran file sebesar 151552 *Byte*. Hasil enkripsinya ditunjukkan pada Gambar 19 (b) dengan ukuran file sebesar 151560 *Byte* atau lebih besar 8 *Byte* dari ukuran aslinya. Hasil enkripsinya berupa simbol-simbol yang tidak bisa dibaca kembali.



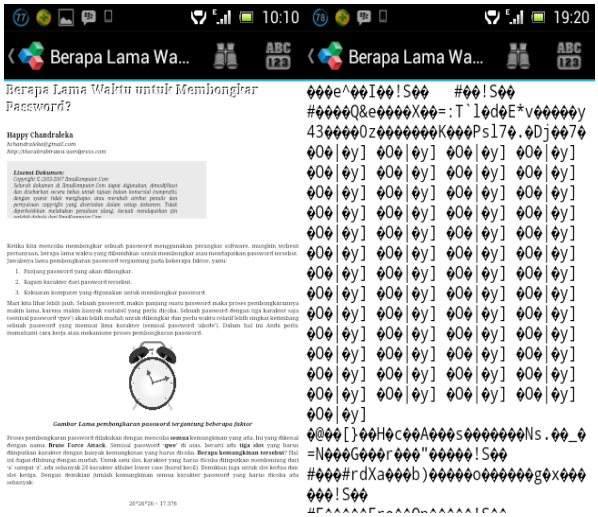
(a) (b)

Gambar 19. Plainteks (a) dan Cipherteks (b) dari file berformat .xls

g. File .doc

Untuk pengujian pada file (.doc) dengan nama file Berapa Lama Waktu Bongkar *Password*.doc dengan menggunakan *password* "qwerty123" untuk enkripsi

dan dekripsinya. Plainteksnya ditunjukkan pada Gambar 20 (a) dengan ukuran file sebesar 117248 *Byte*. Hasil enkripsinya ditunjukkan pada Gambar 20 (b) dengan ukuran file sebesar 117256 *Byte* atau lebih besar 8 *Byte* dari ukuran aslinya. Hasil enkripsinya berupa simbol-simbol yang tidak bisa dibaca kembali.



(a) (b)

Gambar 20. Plainteks (a) dan Cipherteks (b) dari file berformat .doc

Dari hasil pengujian aplikasi terhadap file-file yang telah ditentukan format filenya, rancangan proses enkripsi pada file tersebut yang telah dirancang dapat diimplementasikan dan berhasil dijalankan dengan baik. Hal ini dibuktikan dengan aplikasi yang bisa diinstal ke dalam *handphone* Android dan dapat digunakan untuk melakukan proses enkripsi dan dekripsi file. *Password/key* digunakan harus sama karena algoritma Blowfish menggunakan kunci simetris untuk melakukan proses enkripsi dan dekripsi file. Proses dekripsi file dengan *password/key* yang salah tidak dapat membuka file kembali seperti keadaan semula. Untuk itu pengguna aplikasi ini disarankan untuk mengingat dengan baik *password/key* yang digunakan. Jika pengguna lupa *password/key* maka file tidak dapat didekripsi lagi. Panjang *password/key* dan kerahasiaan *password/key* mempengaruhi dari tingkat keamanan file itu sendiri.

Hasil pengujian enkripsi file yang dilakukan terhadap file dokumen, gambar, dan video membuat file tersebut tidak dapat dimengerti maknanya kembali. Karena pada saat proses enkripsi terdapat operasi penambahan dan XOR sehingga karakter yang sebelumnya bisa dibaca menjadi tidak terbaca. Gambar 14 sampai dengan Gambar 20 menunjukkan perbedaan tampilan file sebelum dan setelah dilakukan proses enkripsi. Perbedaan ukuran file sebelum dan setelah proses enkripsi dikarenakan adanya proses *padding* yaitu untuk memperoleh blok terakhir sebesar 64 *bit*. Hal ini karena algoritma Blowfish adalah algoritma yang menggunakan *cipher* blok 64 *bit* untuk membagi ukuran file sebelum dilakukan proses enkripsinya.

Untuk tingkat keamanan dilakukan pengujian terhadap serangan *brute force*. Serangan bertipe *brute force* adalah dengan menerapkan percobaan setiap kemungkinan kunci yang ada satu per satu sampai diperoleh plainteks yang diharapkan. Waktu yang diperlukan untuk memperoleh kunci yang diharapkan, selalu berbanding lurus dengan panjang *bit* kunci yang dimiliki oleh algoritma kriptografi, dan makin cepat prosesor yang dipakai, makin cepat waktu yang pula dibutuhkan untuk dapat memperoleh kunci tersebut. Analisa *brute force attack* pada sistem ini adalah dengan melakukan perhitungan matematis dari masing-masing kunci untuk mendapatkan nilai durasi proses memecahkan kemungkinan kunci. Panjang kunci yang digunakan dalam penelitian ini adalah 72 *bit*. Kemungkinan kunci yang dapat dihasilkan dengan menggunakan panjang 72 *bit* adalah sejumlah 4,7 x 10^{21} kunci. Asumsi kecepatan komputasi adalah 10^6 *key/sec* maka :

$$\frac{4{,}7\times10^{21}}{10^{6}}=\frac{4{,}7\times10^{15}\,\text{sec}}{3{,}1536\times10^{7}}=1{,}49\times10^{8}$$

menghasilkan durasi sejumlah 1,49x10^8 tahun untuk memecahkan seluruh kemungkinan kunci.

Berdasarkan dari hasil pengujian tersebut, maka aplikasi enkripsi dan dekripsi file ini cukup aman untuk melindungi file-file yang sifatnya pribadi dan rahasia yang orang lain tidak berhak untuk mengetahuinya.

V. PENUTUP

A. Kesimpulan

Berdasarkan pengujian dan analisis yang telah dilakukan dalam penelitian ini dapat diambil kesimpulan sebagai berikut:

1. Aplikasi enkripsi dan dekripsi file dapat berfungsi dengan baik pada *handphone* Android dengan OS Android 2.3, 4.0, dan 4.1.
2. Aplikasi yang dibuat dapat diimplementasikan dengan baik untuk melakukan enkripsi dan dekripsi file karena file yang telah dienkripsi menjadi tidak bisa dimengerti dan pahami lagi isinya.
3. Tingkat keamanan dari aplikasi yang dibuat cukup aman karena algoritma Blowfish memiliki panjang kunci yang besar. Dengan menggunakan kunci berjumlah 72 *bit* atau 9 karakter dibutuhkan waktu 1,49x10^8 tahun untuk membongkarnya dengan kecepatan komputasinya adalah 10^6 *key/sec*.

B. Saran

Aplikasi enkripsi dan dekripsi file ini masih banyak kekurangannya sehingga diharapkan kedepannya sistem aplikasi ini bisa dikembangkan menjadi lebih baik lagi. Terdapat beberapa saran diantaranya:

1. Menambahkan beberapa fitur diantaranya penambahan pilihan algoritma lain dan kemampuan untuk mengenkripsi folder.
2. Pembuatan fungsi waktu sehingga bisa diketahui seberapa cepat enkripsi dan dekripsi filenya.

.

DAFTAR PUSTAKA

[1] Rahman, Abdul. (2012). *Implementasi Algoritma Serpent Untuk Enkripsi Dan Dekripsi Data File Pada Ponsel Berbasis Android*. Jurnal Jurusan Teknik Informatika STMIK GI MDP.

[2] Defni, Indri Rahmayun. (2014). *Enkripsi SMS (Short Message Service) Pada Telepon Selular Berbasis Android Dengan Metode RC6*. Jurnal Momentum Vol.16 No.1. Februari 2014 Jurusan Teknologi Informasi Politeknik Negeri Padang.

[3] Munir, Rinaldi. (2006). *Kriptografi*. Bandung: Pener*bit* Informatika.

[4] Munir, Rinaldi. (2004). *Diktat Kuliah IF5054 Kriptografi*. Bandung. Teknik Informatika ITB.

[5] Syafari, Anjar. Sekilas Tentang Enkripsi Blowfish. Tersedia dari: ilmukomputer.org/wp.../anjar-enkripsi-blowfish.doc. [URL dikunjungi pada tanggal 2 September 2013]

[6] Juwairiah. (2010). *Aplikasi Kriptografi File Menggunakan Algoritma Blowfish*. Seminar Nasional Informatika 2010 (semnasIF 2010) UPN "Veteran" Yogyakarta, 22 Mei 2010.

[7] Tri Massandy, Danang. (2011). *Studi dan Implementasi Cryptography Package pada Sistem Operasi Android*. Makalah Jurusan Teknik Informatika-STEI Institut Teknologi Bandung.

[8] Satvika Aswari, Ni Made. (2011). *Eksplorasi Java Cryptography Architecture (JCA) dan Implementasinya Pada Perangkat Android*. Makalah Jurusan Teknik Informatika-STEI Institut Teknologi Bandung.

[9] Amiral, Muhammad. (2010). *Aplikasi Pengingat Shalat dan Arah Kiblat Menggunakan Global Positioning System (GPS) Berbasis Android 1.6*. Tugas Akhir Jurusan Teknik Informatika Institut Teknologi Indonesia.

[10] Inggiantowi, Hafid. (2011). *Studi Implementasi Algoritma Block Cipher pada Platform Android*. Makalah Jurusan Teknik Informatika-STEI Institut Teknologi Bandung.

[11] Pratama, Widianto. *Pengenalan Android Part 1*. Tersedia dari: ml.scribd.com/doc/190153286/Pengenalan-Android. [URL dikunjungi pada tanggal 20 Juni 2013]

[12] E Pratiwi, Apriyanti. (2011). *Implementasi Enkripsi Data Dengan Algoritma Blowfish Menggunakan Java Pada Aplikasi Email*. Jurnal Jurusan Teknik Komputer Politeknik Telkom Bandung.

[13] Erikawaty Aryani Tambunan, Shanty. (2010). *Implementasi Algoritma Kriptografi Blowfish Untuk Keamanan Dokumen Pada Microsoft Office*. Jurnal Jurusan Teknik Informatika STMIK Amikom Yogyakarta.

[14] Pambudi Nusa, Tetuko. (2013). *Rancang Bangun Aplikasi Enkripsi Database MYSQL Dengan Algoritma Blowfish*. Jurnal Jurusan Manajemen Informatika Universitas Negeri Surabaya.

[15] Alim Sutanto,Candra. (2010). *Penggunaan Algoritma Blowfish Dalam Kriptografi,Bandung*. Makalah Jurusan Teknik Informatika-STEI Institut Teknologi Bandung.