Perbandingan Aplikasi Menggunakan Metode Camellia 128 Bit Key dan 256 Bit Key

by Gregorius Satia Budhi

Submission date: 27-Nov-2019 02:37PM (UTC+0700)
Submission ID: 1222721098
File name: kasi_Menggunakan_Metode_Camellia_128_Bit_Key_dan_256_Bit_Key.pdf (576.08K)
Word count: 3546
Character count: 19787

Jurnal Informatika, Vol. 12, No. 2, November 2014, 109-116
ISSN 1411-0105
DOI: 10.9744/informatika.12.2.109-116

PERBANDINGAN APLIKASI MENGGUNAKAN METODE CAMELLIA 128 BIT *KEY* DAN 256 BIT *KEY*

Lanny Sutanto[1], Gregorius Satia Budhi[1*], Leo Willyanto Santoso[1]
[1]Program Studi Teknik Informatika Fakultas Teknologi Industri Universitas Kristen Petra
Jl. Siwalankerto 121–131 Surabaya 60236
Telp. (031) – 2983455, Fax. (031) – 8417658
E-mail: hipophirl91@gmail.com, greg@petra.ac.id, leow@petra.ac.id
*Korespondensi penulis

Abstrak: Perkembangan Internet yang pesat saat ini menyebabkan tingginya tingkat resiko dalam pembajakan data. Salah satu cara mengamankan data dengan menggunakan kriptografi camellia. Camellia dikenal sebagai metode yang memiliki waktu enkripsi dan dekripsi yang cepat. Metode *Camellia* memiliki 3 macam besaran *key* yaitu 128-bit, 192-bit, dan 256-bit. Aplikasi ini dibuat dengan menggunakan bahasa pemrograman c++ dan menggunakan GUI visual studio 2010. Penelitian ini membandingkan besaran kunci terkecil dan terbesar yang digunakan pada file berekstensi .txt, .doc, .docx, .jpg, .mp4, .mkv, dan .flv. Aplikasi ini dibuat untuk mengetahui perbandingan waktu dan tingkat keamanan pada penggunaan kunci 128 bit dan 256 bit. Perbandingan keamanan dilakukan dengan membandingkan hasil nilai *avalanche effect* untuk kunci 128 bit dan 256 bit.

Kata kunci: Kriptografi, *camellia, avalanche effect.*

Abstract: *The rapid development of the Internet today to easily exchange data. This leads to high levels of risk in the data piracy. One of the ways to secure data is using cryptography camellia. Camellia is known as a method that has the encryption and decryption time is fast. Camellia method has three kinds of scale key is 128 bit, 192 bit, and 256 bit.This application is created using the C++ programming language and using visual studio 2010 GUI. This research compare the smallest and largest key size used on the file extension .Txt, .Doc, .Docx, .Jpg, .Mp4, .Mkv and .Flv. This application is made to comparing time and level of security in the use of 128-bit key and 256 bits. The comparison is done by comparing the results of the security value of avalanche effect 128 bit key and 256 bit key.*

Keywords*: Cryptography, camellia, avalanche effect.*

PENDAHULUAN

Internet mempunyai pengaruh yang besar atas ilmu dan pandangan dunia. Dengan hanya terpandukan mesin pencari, pengguna di seluruh dunia mempunyai akses internet yang mudah atas bermacam-macam informasi dan data. Salah satu masalah penting yang terjadi saat ini adalah masalah keamanan data yaitu mengamankan data dari orang yang tidak berhak.

Cara untuk mengamankan suatu data salah satunya dengan menggunakan kriptografi *Camellia.* Metode ini dikenal dengan metode yang memiliki waktu enkripsi dan dekripsi yang cepat. Umumnya metode ini digunakan pada pengamanan jaringan karena pada jaringan, besar data yang dikirimkan tidak besar dan memerlukan proses enkripsi dan dekripsi yang cepat.

Metode Camellia memiliki keistimewaan dibandingkan dengan metode kriptografi yang lainnya. Metode *Camellia* memiliki 3 macam besaran key yang dapat digunakan yaitu 128-bit sebagai besaran key terkecil, 192-bit, dan 256-bit sebagai besaran key terbesar. Oleh karena itu, perbandingan applikasi menggunakan metode Camellia kunci 128-bit dan 256-bit ini bertujuan untuk mengetahui perbandingan tingkat keamanan dan lama waktu enkripsi dan dekripsi data pada 128 bit dan 256 bit.

TINJAUAN PUSTAKA

Kriptografi

Kriptografi berasal dari Bahasa Yunani: "cryptós" artinya "secret" (rahasia), sedangkan "gráphein" artinya "writing" (tulisan). Jadi, kriptografi berarti "secret writing".

Kriptografi adalah ilmu dan seni untuk menjaga keamanan pesan. Kata seni pada definisi di atas berasal dari fakta sejarah bahwa pada masa-masa awal sejarah kriptografi, setiap orang mungkin mempunyai cara yang unik untuk merahasiakan pesan [1].

Camellia

Algoritma Camellia dikembangkan secara bersama oleh NTT dan Mitsubishi Electric Corporation pada tahun 2000. Algoritma ini mengadopsi algoritma kriptografi E2 (dikembangkan oleh NTT) dan algoritma kriptografi MISTY (dikembangkan oleh Mitsubishi).

Pada algoritma Camellia, sebuah blok memiliki ukuran 128 bit. Panjang kunci bervariasi antara 128 bit, 192 bit atau 256 bit. Algoritma Camellia merupakan modifikasi Feistel Cipher sebanyak 18 putaran (ketika panjang kunci 128 bit) atau 24 putaran (ketika panjang kunci 192 atau 256 bit).[2]

Algoritma Camellia merupakan algoritma yang dipatenkan namun memiliki *royalty free licences* yang menjadikan pengguna algoritma Camellia tidak membayar dalam penggunaan algoritma ini. *Royalty free licences* ini berlaku jika tidak merubah algoritma yang telah ditetapkan. [3]

Prosedur Enkripsi 128 bit

Enkripsi menggunakan kunci 128 bit menggunakan struktur Feistel Cipher[4] yang terdiri atas 18 putaran dengan 2 lapisan fungsi FL/FL-1 setelah tahap keenam dan tahap keduabelas. Proses enkripsi dapat dilihat pada Gambar 1.

Pertama-tama blok pertama plainteks dipecah menjadi dua, kemudian masing-masing dixor-kan dengan kw1 dan kw2. Hasilnya ialah L0 dan R0. Pada putaran pertama, L0 dimasukkan ke fungsi F beserta k1. Hasilnya kemudian dixor-kan dengan R0. Pada putaran kedua, L0 akan menjadi R1, sedangkan hasil xor tadi menjadi L1. L1 ini kemudian dimasukan ke fungsi F lagi beserta k2. Demikian seterusnya hingga putaran keenam.

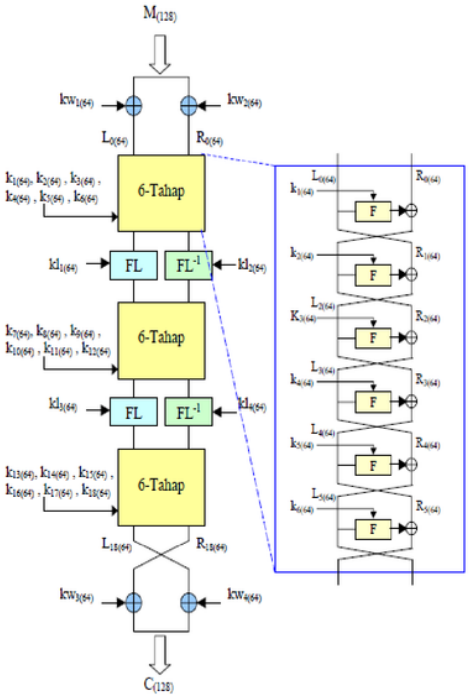
Sebelum putaran ketujuh, L6 akan memasuki fungsi FL bersama kl1, dan R6 akan memasuki fungsi FL-1 bersama kl2. Hasil masing-masing akan kembali dienkripsi dengan cara seperti tadi sebanyak 6 putaran lagi, dan sebelum putaran ketigabelas akan menggunakan fungsi FL/FL-1 lagi. Kemudian dienkripsi dengan fungsi F lagi sebanyak 6 putaran. Sehingga total putaran adalah 18. Terakhir L18 akan dixor-kan dengan kw4 dan R18 akan dixor-kan dengan kw3. Hasilnya kemudian digabung, itulah cipherteks dari blok pertama plainteks.

Prosedur Enkripsi 192 / 256 bit

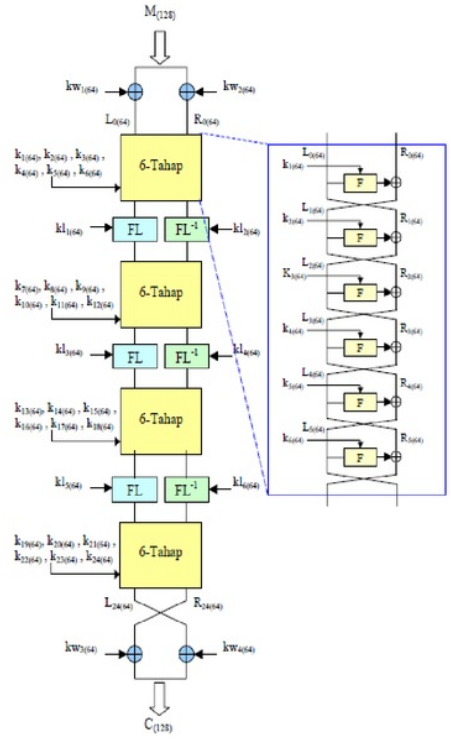
Enkripsi dengan menggunakan kunci 192 bit dan 256 bit memiliki mekanisme yang sedikit berbeda dengan prosedur enkripsi 128 bit. Pada proses ini ditambahkan 6 putaran feistel cipher dan 1 layer yang menggunakan fungsi FL dan FL-1. Fungsi ini digunakan pada sebelum putaran ketujuh, ketigabelas, dan kedelapanbelas. Tetapi pada putaran lainnya secara umum prosedur enkripsinya sama dengan enkripsi menggunakan kunci 128 bit.

Proses Dekripsi 128 bit

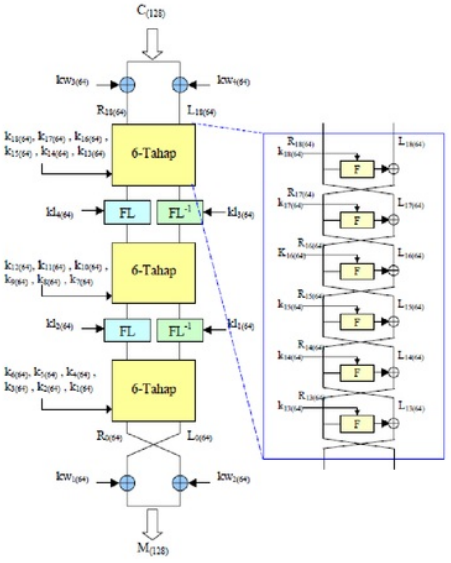
Pada proses dekripsi dengan kunci 128 bit, pertama-tama blok cipherteks akan dibagi menjadi dua kemudian masing-masing dixor-kan dengan kw3 dan kw4. Setelah itu subblok kiri maupun kanan akan didekripsi dengan cara yang sama seperti pada proses enkripsi, namun perbedaannya ialah penggunaan subkuncinya digunakan dari subkunci terakhir. Kemudian sebelum putaran berikutnya akan memasuki fungsi FL dan FL-1 menggunakan subkunci kl4 dan kl3. Setelah itu kembali akan memasuki Feistel Cipher sebanyak 6 putaran, masing-masing dengan menggunakan subkunci k12 hingga k7. Kemudian masuk ke fungsi FL dan FL-1 menggunakan subkunci kl2 dan kl1. Setelah itu didekripsi lagi dengan Feistel Cipher 6 putaran, masing-masing dengan menggunakan subkunci k6 hingga k1. Pada tahap terakhir, subblok kiri dixor-kan dengan kw1 dan subblok kanan akan dixor-kan dengan kw2. Hasil penggabungan dua subblok ini merupakan plainteks blok pertama dari cipherteks.



Gambar 1. Proses Enkripsi 128 bit



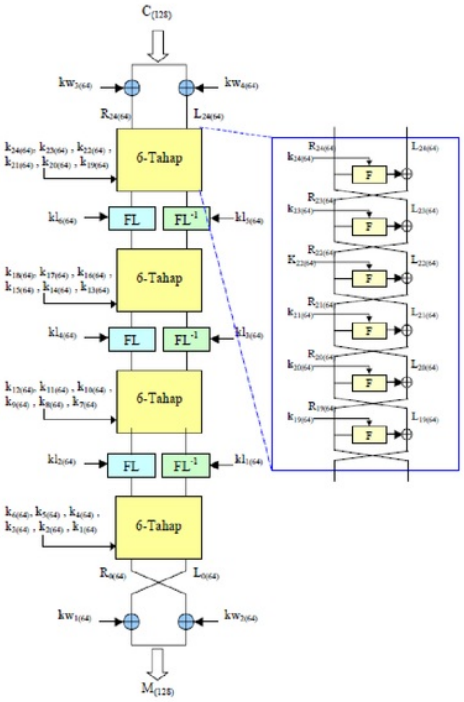
Gambar 2. Proses Enkripsi 192 / 256 bit



Gambar 3. Proses Dekripsi 128 bit

Proses Dekripsi 192 / 256 bit

Proses dekripsi menggunakan kunci 192 bit / 256 bit memiliki kesamaan dengan proses dekripsi menggunakan 128 bit. Proses ini terdiri atas 24 ronde dan cara kerjanya sama seperti proses dekripsi menggunakan kunci 128 bit tapi 6 putaran feistel cipher dan 1 layer menggunakan fungsi FL dan FL-1. Subkunci yang digunakan dimulai dari Subkunci terakhir.



Gambar 4. Proses Dekripsi 192 / 256 bit

Penjadwal Kunci

	subkey	value
Prewhitening	$kw_{1(64)}$	$(K_L <<<_0)_{L(64)}$
	$kw_{2(64)}$	$(K_L <<<_0)_{R(64)}$
F (Round1)	$k_{1(64)}$	$(K_A <<<_0)_{L(64)}$
F (Round2)	$k_{2(64)}$	$(K_A <<<_0)_{R(64)}$
F (Round3)	$k_{3(64)}$	$(K_L <<<_{15})_{L(64)}$
F (Round4)	$k_{4(64)}$	$(K_L <<<_{15})_{R(64)}$
F (Round5)	$k_{5(64)}$	$(K_A <<<_{15})_{L(64)}$
F (Round6)	$k_{6(64)}$	$(K_A <<<_{15})_{R(64)}$
FL	$kl_{1(64)}$	$(K_A <<<_{30})_{L(64)}$
FL^{-1}	$kl_{2(64)}$	$(K_A <<<_{30})_{R(64)}$
F (Round7)	$k_{7(64)}$	$(K_L <<<_{45})_{L(64)}$
F (Round8)	$k_{8(64)}$	$(K_L <<<_{45})_{R(64)}$
F (Round9)	$k_{9(64)}$	$(K_A <<<_{45})_{L(64)}$
F (Round10)	$k_{10(64)}$	$(K_A <<<_{60})_{R(64)}$
F (Round11)	$k_{11(64)}$	$(K_A <<<_{60})_{L(64)}$
F (Round12)	$k_{12(64)}$	$(K_A <<<_{60})_{R(64)}$
FL	$kl_{3(64)}$	$(K_L <<<_{77})_{L(64)}$
FL^{-1}	$kl_{4(64)}$	$(K_L <<<_{77})_{R(64)}$
F (Round13)	$k_{13(64)}$	$(K_L <<<_{94})_{L(64)}$
F (Round14)	$k_{14(64)}$	$(K_L <<<_{94})_{R(64)}$
F (Round15)	$k_{15(64)}$	$(K_A <<<_{94})_{L(64)}$
F (Round16)	$k_{16(64)}$	$(K_A <<<_{94})_{R(64)}$
F (Round17)	$k_{17(64)}$	$(K_A <<<_{111})_{L(64)}$
F (Round18)	$k_{18(64)}$	$(K_A <<<_{111})_{R(64)}$
Postwhitening	$kw_{3(64)}$	$(K_A <<<_{111})_{L(64)}$
	$kw_{4(64)}$	$(K_A <<<_{111})_{R(64)}$

	subkey	value
Prewhitening	$kw_{1(64)}$	$(K_L <<<_0)_{L(64)}$
	$kw_{2(64)}$	$(K_L <<<_0)_{R(64)}$
F (Round1)	$k_{1(64)}$	$(K_R <<<_0)_{L(64)}$
F (Round2)	$k_{2(64)}$	$(K_R <<<_0)_{R(64)}$
F (Round3)	$k_{3(64)}$	$(K_L <<<_{15})_{L(64)}$
F (Round4)	$k_{4(64)}$	$(K_R <<<_{15})_{R(64)}$
F (Round5)	$k_{5(64)}$	$(K_A <<<_{15})_{L(64)}$
F (Round6)	$k_{6(64)}$	$(K_A <<<_{15})_{R(64)}$
FL	$kl_{1(64)}$	$(K_R <<<_{30})_{L(64)}$
FL^{-1}	$kl_{2(64)}$	$(K_R <<<_{30})_{R(64)}$
F (Round7)	$k_{7(64)}$	$(K_R <<<_{45})_{L(64)}$
F (Round8)	$k_{8(64)}$	$(K_R <<<_{45})_{R(64)}$
F (Round9)	$k_{9(64)}$	$(K_A <<<_{45})_{L(64)}$
F (Round10)	$k_{10(64)}$	$(K_A <<<_{45})_{R(64)}$
F (Round11)	$k_{11(64)}$	$(K_A <<<_{15})_{L(64)}$
F (Round12)	$k_{12(64)}$	$(K_A <<<_{15})_{R(64)}$
FL	$kl_{3(64)}$	$(K_A <<<_{60})_{L(64)}$
FL^{-1}	$kl_{4(64)}$	$(K_A <<<_{60})_{R(64)}$
F (Round13)	$k_{13(64)}$	$(K_R <<<_{60})_{L(64)}$
F (Round14)	$k_{14(64)}$	$(K_R <<<_{60})_{R(64)}$
F (Round15)	$k_{15(64)}$	$(K_R <<<_{90})_{L(64)}$
F (Round16)	$k_{16(64)}$	$(K_R <<<_{90})_{R(64)}$
F (Round17)	$k_{17(64)}$	$(K_A <<<_{77})_{L(64)}$
F (Round18)	$k_{18(64)}$	$(K_A <<<_{77})_{R(64)}$
FL	$kl_{5(64)}$	$(K_A <<<_{77})_{L(64)}$
FL^{-1}	$kl_{6(64)}$	$(K_A <<<_{77})_{R(64)}$
F (Round19)	$k_{19(64)}$	$(K_R <<<_{94})_{L(64)}$
F (Round20)	$k_{20(64)}$	$(K_R <<<_{94})_{R(64)}$
F (Round21)	$k_{21(64)}$	$(K_A <<<_{94})_{L(64)}$
F (Round22)	$k_{22(64)}$	$(K_A <<<_{94})_{R(64)}$
F (Round23)	$k_{23(64)}$	$(K_A <<<_{111})_{L(64)}$
F (Round24)	$k_{24(64)}$	$(K_A <<<_{111})_{R(64)}$
Postwhitening	$kw_{3(64)}$	$(K_R <<<_{111})_{L(64)}$
	$kw_{4(64)}$	$(K_R <<<_{111})_{R(64)}$

Gambar 5. Tabel Subkunci 128bit (kiri) dan 256bit (kanan)

Penjadwal kunci atau *Key Schedule*[5] adalah proses untuk memproses kunci masukan *user* menjadi sebuah atau dua buah data yang akan digunakan untuk membuat subkunci. Subkunci adalah kunci yang digunakan untuk proses enkripsi dan dekripsi. Subkunci didapatkan dari menggeser nilai sebanyak bit secara melingkar ke kiri dan diambil 64 bit sebelah kiri atau kanan. Subkunci k, kw dan kl dihasilkan dari nilai hasil rotasi penggeseran KL, KR, KA, dan KB. KL dan KR merupakan kunci masukan user. Jika besar kunci yang digunakan user adalah 128 maka KL berisi kunci masukan user dan nilai KR diisi angka 0, jika kunci masukan user 256 maka 128 bit pertama kunci akan disimpan pada variabel KL dan sisanya disimpan pada variabel KR. Nilai isi masing-masing subkunci dapat dilihat pada Gambar 5.

Avalanche Effect

Salah satu karakteristik untuk menentukan baik atau tidaknya suatu algoritma kriptografi adalah dengan melihat *avalanche effect*-nya. Perubahan yang kecil pada plainteks maupun kunci akan menyebabkan perubahan yang signifikan terhadap cipherteks yang dihasilkan [6]. Dengan kata lain, perubahan satu bit pada plainteks maupun kunci akan menghasilkan perubahan banyak bit pada cipherteks. Suatu *avalanche effect* dikatakan baik jika perubahan bit yang dihasilkan berkisar antara 45-60% (50 % adalah hasil yang sangat baik). Hal ini dikarenakan perubahan tersebut berarti membuat perbedaan yang cukup sulit untuk kriptanalis melakukan serangan. Nilai *avalanche effect* dirumuskan dengan [7]:

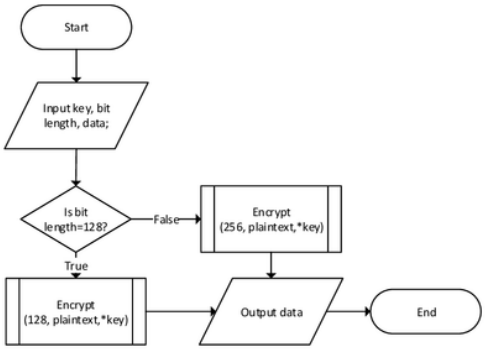
Avalanche Effect = Jumlah bit berubah / jumlah bit total * 100%

ANALISA DAN DESAIN SISTEM

Proses Enkripsi

Proses enkripsi adalah proses mengubah plaintext menjadi ciphertext. Plaintext adalah data awal dan chipertext adalah data hasil yang telah di enkripsi.

Dimulai dengan pembacaan nilai kunci, besaran bit dan data dari masukkan user. Kemudian lakukan proses pengecekan besaran bit. Jika besaran bit adalah 128 maka proses selanjutnya yang dilakukan adalah fungsi *Encrypt* dengan nilai kunci 128bit, jika tidak maka proses yang dilakukan adalah fungsi *Encrypt* dengan nilai kunci 256bit. Fungsi *Encrypt* menggunakan 2 parameter yaitu plainteks dan *key*. Plaintext adalah data dan *key* adalah kunci masukkan user. Proses enkripsi dilakukan setiap 16 byte data hingga data yang akan dienkripsi habis. Setelah fungsi dijalankan maka dihasilkan cipherteks. Variabel plaintext dan *key* menggunakan tipe data *unsigned char**. Alur proses enkripsi dapat dilihat pada Gambar 6.

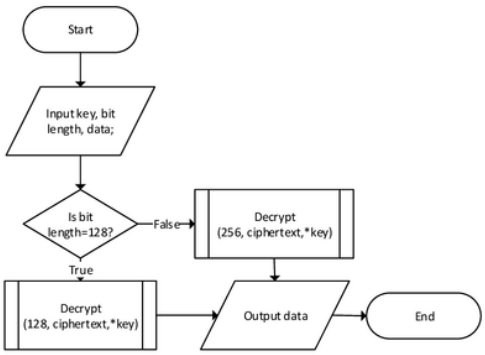


Gambar 6. Desain Sistem Enkripsi

Proses Dekripsi

Proses dekripsi adalah proses mengubah ciphertext menjadi plaintext. Sama halnya dengan proses enkripsi, proses dekripsi memiliki 2 macam ukuran kunci yaitu 128 bit dan 256 bit.

Pertama dilakukan pembacaan terhadap kunci yang digunakan, besar bit yang digunakan dan data. Kemudian lakukan pengecekan apakah besar bit yang digunakan adalah 128, jika benar maka lakukan pemanggilan fungsi decrypt dengan parameter 128, cipherteks dan kunci yang akan digunakan untuk mendekripsi cipherteks. Alur proses dekripsi dapat dilihat pada Gambar 7.



Gambar 7. Desain Sistem Dekripsi

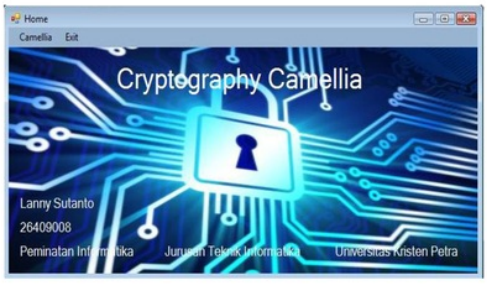
HASIL

Hasil dari enkripsi dan dekripsi aplikasi menggunakan metode *camellia* dengan kunci 128 bit dan 256 bit dibatasi pada file berekstensi .txt, .doc, .docx,

.jpg, .mp4, .mkv dan .flv saja. Terdapat 4 ukuran yang digunakan pada pengujian ini. Masing-masing ukuran menggunakan 5 file yang berbeda dengan jarak 0,1 hingga 1,5 MB. Nilai waktu enkripsi dan dekripsi yang didapat menggunakan fungsi dari *windows.h* yaitu fungsi *Clock()*[8] dalam hitungan detik. Pengujian aplikasi dilakukan dengan menggunakan sistem operasi windows vista, prosesor intel core 2 duo p8600 @2.40GHz, dan 3 GB SD-RAM DDR2.

Halaman Utama

Halaman utama ini adalah halaman yang pertama kali ditampilkan ketika aplikasi ini diakses.



Gambar 8. Tampilan Halaman Utama

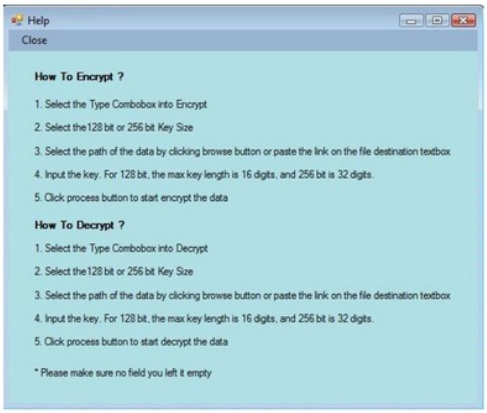
Halaman Home atau halaman utama ini menampilkan data diri penulis skripsi. Pada bagian atas halaman terdapat *menubar*, menu Camellia untuk membuka halaman CamelliaForm dan menu *Exit* untuk mengakhiri aplikasi. Tampilan halaman utama dapat dilihat pada Gambar 8.

Halaman CamelliaForm

Halaman ini terdapat *menubar*, menu *Help* yang berisi panduan penggunaan program ini dan menu *Exit* untuk menutup halaman CamelliaForm. Pada halaman ini pertama kali *user* harus memilih enkripsi atau dekripsi melalui *combobox Type*. Selanjutnya *user* dapat memilih ingin menggunakan *key* besaran 128 bit atau 256 bit.Setelah proses pemilihan besaran *key*, *user* dapat memasukkan data *key* yang diinginkan. Pada kunci besaran 128 bit, panjang kunci maksimum yang dapat dimasukan adalah 16 *digit* dan pada kunci besaran 256 bit, panjang kunci maksimum yang dapat digunakan adalah 32 *digit*. Setelah itu *user* dapat memasukkan alamat file yang diinginkan untuk proses enkripsi atau dekripsi tersebut dengan cara memilih file dari tombol *browse* atau dengan mem-*paste*kan alamatnya langsung pada *textbox file location*. Untuk dapat menjalankan proses enkripsi atau dekripsi tersebut, *user* harus menekan tombol *Process*. Pembacaan kunci dari tipe data String^ akan di*convert* menjadi char* menggunakan *library marshal*.[9]

Gambar 8. Tampilan Halaman CamelliaForm

Halaman Help



Gambar 9. Tampilan Halaman *Help*

Halaman ini berisi tentang langkah-langkah untuk melakukan proses enkripsi dan dekripsi pada halaman camelliaForm. Halaman ini dapat ditutup dengan menggunakan menu close atau menekan tombol silang pada ujung kanan atas aplikasi. Tampilan halaman web dapat dilihat pada Gambar 9.

Perhitungan Tingkat Keamanan Data Berdasarkan Hasil Aplikasi

Kunci 128 bit yang memiliki panjang 16 *digit* memiliki total kemungkinan kunci sebanyak 3,4 x 10^{38}, dan kunci 256 yang memiliki panjang 32 *digit* memiliki total kemungkinan kunci sebanyak 1,2 x 10^{77}. Dari pengujian aplikasi yang telah dilakukan, waktu yang didapatkan untuk melakukan proses enkripsi dan dekripsi untuk 1 kunci akan dikalikan dengan total seluruh jumlah kemungkinan kunci untuk mendapatkan total waktu yang dibutuhkan untuk dapat mendekripsi data hasil enkripsi menjadi data asli dengan menggunakan kunci yang tepat.

Waktu rata-rata enkripsi dan dekripsi berisi hasil rata-rata enkripsi dan dekripsi 128bit dan 256 bit pada masing-masing ukuran yang telah diujikan. Waktu rata-rata enkripsi dan dekripsi file berekstensi txt dapat dilihat pada Tabel 1, file berekstensi .doc dapat dilihat pada Tabel 2, file berekstensi .docx dapat dilihat pada Tabel 3 file berekstensi .jpg dapat dilihat pada Tabel 4, file berekstensi .mp4 dapat dilihat pada Tabel 5, file berekstensi .mkv dapat dilihat pada Tabel 6, file berekstensi .flv dapat dilihat pada Tabel 7.

Tabel 1. Waktu Rata-rata Enkripsi Dan Dekripsi File Txt

Ukuran File (MB)	Waktu Rata-rata Enkripsi 128 bit (detik)	Waktu Rata-rata Dekripsi 128 bit (detik)	Waktu Rata-rata Enkripsi 256 bit (detik)	Waktu Rata-rata Dekripsi 256 bit (detik)
0,6-1,5	3,19	3,164	4,004	4,013
9,9-10,8	5,8118	5,884	7,3856	7,372
19,9-20,5	11,3306	11,4268	14,3652	14,2596
39,8-41	23,4214	23,0142	28,6122	28,4364

Tabel 2. Tabel Waktu Rata-rata Enkripsi Dan Dekripsi File Doc

Ukuran File (MB)	Waktu Rata-rata Enkripsi 128 bit (detik)	Waktu Rata-rata Dekripsi 128 bit (detik)	Waktu Rata-rata Enkripsi 256 bit (detik)	Waktu Rata-rata Dekripsi 256 bit (detik)
5,1-5,5	3,0472	3,067	3,817	3,7856
20,2-20,8	11,5276	11,751	14,7812	14,6464
30-30,8	16,8874	16,9384	21,2284	21,1036
45,2-45,6	25,504	26,0072	32,3432	31,8544

Tabel 3. Tabel Waktu Rata-rata Enkripsi Dan Dekripsi File Docx

Ukuran File (MB)	Waktu Rata-rata Enkripsi 128 bit (detik)	Waktu Rata-rata Dekripsi 128 bit (detik)	Waktu Rata-rata Enkripsi 256 bit (detik)	Waktu Rata-rata Dekripsi 256 bit (detik)
5,2-6,3	3,3238	3,354	4,1734	4,1586
15-16	8,7482	8,776	10,9848	10,878
25,6-26,8	15,096	14,899	19,0132	18,5164
45,2-46,3	25,7448	25,7692	32,7814	32,3512

Tabel 4. Tabel Waktu Rata-rata Enkripsi Dan Dekripsi File Jpg

Ukuran File (MB)	Waktu Rata-rata Enkripsi 128 bit (detik)	Waktu Rata-rata Dekripsi 128 bit (detik)	Waktu Rata-rata Enkripsi 256 bit (detik)	Waktu Rata-rata Dekripsi 256 bit (detik)
4,8-5,9	2,4616	3,0682	3,935	3,838
10,2-10,6	5,8268	5,9068	7,5298	7,313
19,4-20,4	11,2016	11,385	14,4214	14,1086
40,9-42,1	23,1936	23,3958	29,184	29,5902

Dengan hasil perbandingan total waktu rata-rata enkripsi dan dekripsi pada 128 bit dan 256 bit dikalikan dengan jumlah semua kemungkinan kunci dan rata-rata waktu enkripsi dan dekripsi pada 128 bit lebih cepat dibandingkan pada 256 bit, maka dapat disarankan untuk menggunakan metode camellia 128 bit.

Tabel 5. Tabel Waktu Rata-rata Enkripsi Dan Dekripsi File Mp4

Ukuran File (MB)	Waktu Rata-rata Enkripsi 128 bit (detik)	Waktu Rata-rata Enkripsi 256 bit (detik)	Waktu Rata-rata Dekripsi 128 bit (detik)	Waktu Rata-rata Dekripsi 256 bit (detik)
8,1-9	4,8236	6,0934	4,839	6,0496
18,1-19,1	10,5764	13,6456	10,924	13,9398
26,6-27,8	15,088	19,353	15,4708	19,513
48,8-49,8	28,4756	35,4078	28,1338	35,6102

Tabel 6. Tabel Waktu Rata-rata Enkripsi Dan Dekripsi File Mkv

Ukuran File (MB)	Waktu Rata-rata Enkripsi 128 bit (detik)	Waktu Rata-rata Dekripsi 128 bit (detik)	Waktu Rata-rata Enkripsi 256 bit (detik)	Waktu Rata-rata Dekripsi 256 bit (detik)
15,3-16,3	9,3794	9,2512	11,5188	11,487
26,5-27,5	15,111	15,2888	19,0838	19,4302
37,9-38,8	21,2748	21,7738	27,2096	27,1924
48-48.9	28,5604	28,6776	33,3028	32,9428

Tabel 7. Tabel Waktu Rata-rata Enkripsi Dan Dekripsi File Flv

Ukuran File (MB)	Waktu Rata-rata Enkripsi 128 bit (detik)	Waktu Rata-rata Dekripsi 128 bit (detik)	Waktu Rata-rata Enkripsi 256 bit (detik)	Waktu Rata-rata Dekripsi 256 bit (detik)
15,3-16,3	8,9476	9,0446	11,353	11,2348
26,6-27,5	15,1132	15,1632	18,9978	19,007
37,4-38,3	20,9538	21,082	26,473	26,4294
48-48,9	27,505	28,6776	33,3028	32,9428

Perbandingan Tingkat Keamanan Penggunaan Kunci 128 bit dengan 256 bit pada Subkunci yang Digunakan

Penentuan perbandingan tingkat keamanan pada enkripsi data dapat dilihat pada kunci yang digunakan. Kunci dengan panjang 128 bit memiliki panjang 16 byte atau 16 digit, kunci dengan panjang 256 bit memiliki panjang 32 byte atau 32 digit. Panjang kunci 16 digit dan 32 digit ditentukan oleh panjang bit. 8 bit adalah 1 byte dan 1 byte adalah 1 digit.

Kunci dengan panjang 128 bit menggunakan subkunci 128. Proses ini menggunakan 3 macam subkunci 128 bit yang diambil dari nilai KL dan KA yang dirotasi sebanyak bit yang telah ditentukan. Nilai KA didapatkan dengan menjalankan fungsi KeySchedule128. Sedangkan pada kunci dengan

panjang 256 menggunakan subkunci 256. Subkunci 256 menggunakan 3 subkunci 256 bit yang diambil dari nilai KL, KR, KA, dan KB. KL adalah 128bit pertama dari key masukan user dan KR adalah 128 bit terakhir dari key masukan user. KA dan KB adalah nilai yang didapatkan dari fungsi Key Schedule256. Pada fungsi ini terdapat perbedaan pada ronde yang dilakukan. Pada KeySchedule128 hanya menggunakan 4 ronde dengan fungsi F dan 1 kali xor dengan nilai KL, pada KeySchedule256 digunakan 6 ronde dengan 2 kali xor dengan nilai KL dan KR.

Hasil Perhitungan Nilai Avalanche Effect

Perhitungan nilai *avalanche effect* dilakukan pada 3 macam percobaan. File sama kunci beda, file beda kunci sama, dan kunci dikurangkan 1 setiap percobaan. Masing-masing percobaan menggunakan 10 data tes.

Tabel 8. Tabel Hasil Perhitungan *Avalanche Effect*

Mode	Nilai *avalanche effect*
Kunci sama file berbeda menggunakan 128 bit	100%
Kunci sama file berbeda menggunakan 256 bit	100%
Kunci berbeda file sama menggunakan 128 bit	40,74%
Kunci berbeda file sama menggunakan 256 bit	33,33%
Kunci berkurang file sama 128 bit	100%
Kunci berkurang file sama 256 bit	100%

Hasil dari penggunaan kunci yang dikurangi pada file yang sama dengan menggunakan camellia 128 bit atau dengan menggunakan camellia 256 menghasilkan hasil yang memiliki 0 kesamaan sehingga nilai *avalanche effect*nya adalah 100%. Nilai avalanche effect yang sudah didapatkan dihitung rata-ratanya. Untuk penggunaan kunci 128 bit, didapatkan hasil (100% + 40,74% + 100%) = 80,25%, dan untuk penggunaan kunci 256 bit, didapatkan hasil (100% + 33,33% + 100%) = 77,78%.

Kesimpulan dari nilai rata-rata avalanche effect yang didapatkan adalah penggunaan kunci 128 bit dan kunci 256 memiliki nilai avalance effect diatas 50% yang berarti metode kriptografi tersebut terbukti keamanannya dan lebih disarankan menggunakan kunci 128 bit karena memiliki persentase yang lebih tinggi dibandingkan dengan 256 bit.

KESIMPULAN

Kesimpulan yang dapat diambil dari penelitian ini adalah sebagai berikut:

- Dengan aplikasi ini *user* dapat terbantu untuk mengamankan data dari orang yang tidak berhak.
- Melalui aplikasi yang dibuat, dihasilkan perbandingan waktu dari proses enkripsi dan dekripsi dengan menggunakan kunci 128 bit dan 256 bit.
- Tingkat kesulitan dalam mendapatkan kunci yang tepat pada 256 bit lebih susah dari pada 128 bit.
- Waktu yang digunakan pada penggunaan 128 bit lebih cepat dibandingkan dengan waktu yang digunakan pada penggunaan 256 bit.
- Berdasarkan nilai *avalanche effect* camellia 128 bit lebih aman dibandingkan dengan camellia 256 bit.

DAFTAR PUSTAKA

[1] Munir, Rinaldi, *Kriptografi*. Bandung: Informatika, 2006.

[2] Aoki, K., Ichikawa, T., Kanda, M., Matsui, M., Moriai, S., Nakajima, J., & Tokita, T., *Specification of Camellia-a 128-bit Block Cipher*. Japan: Nippon Telegraph and Telephone Corporation and Mitsubishi Electric Corporation, 2000.

[3] Nippon Telegraph and Telephone Corporation, *Announcement of Royalty-free Licenses for Essential Patents of NTT Encryption and Digital Signature Algorithms*. http://www.ntt.co.jp/news/news01e/0104/010417.html, 2001.

[4] Biryukov, A., Nikolic, I., *Complementing Feistel Cipher*. Retrieved July 8 2014, from spms.ntu.edu.sg: http://www1.spms.ntu.edu.sg/~syllab/m/images/6/63/Complementing_Feistel_Ciphers.pdf, 2013.

[5] Biryukov, A., Nikolic, I., *Security Analysis of the Block Cipher Camellia*. Retrieved July 8 2014, from cryptrec.go.jp: www.cryptrec.go.jp/estimation/techrep_id2202-1.pdf, 2012.

[6] Rahardjo, Budi, *Keamanan Sistem Informasi Berbasis Internet*. Bandung: PT Insan Indonesia, Jakarta: PT INDOCISC, 2005.

[7] Ramanujam, S., & Karuppiah, M., *Designing an algorithm with high avalanche effect*. India: International Journal of Computer Science and Network Security,11(1), 2011.

[8] msdn.microsoft.com, *Clock*. Retrieved Mei 26, 2014, from msdn.microsoft.com: http://msdn.microsoft.com/en-us/library/4e2ess30.aspx, 2014.

[9] msdn.microsoft.com, How to: Marshal ANSI Strings Using C++ Interop. Retrieved Mei 12, 2014, from msdn.microsoft.com: http://msdn.microsoft.com/en-us/library/22e4dash.aspx, 2014.

Perbandingan Aplikasi Menggunakan Metode Camellia 128 Bit Key dan 256 Bit Key

ORIGINALITY REPORT

20%	19%	1%	7%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	webmail.informatika.org Internet Source	10%
2	nadyove.wordpress.com Internet Source	2%
3	ejurnal.stmik-budidarma.ac.id Internet Source	2%
4	Submitted to Universitas Brawijaya Student Paper	2%
5	studentjournal.petra.ac.id Internet Source	1%
6	www.scribd.com Internet Source	1%
7	lists.dlitz.net Internet Source	1%
8	en.wikipedia.org Internet Source	1%
9	link.springer.com	

Internet Source

1%

10 Submitted to North West University

Student Paper

1%

Exclude quotes	On	Exclude matches	< 1%
Exclude bibliography	On		