Mengenal Algoritma DES

Rusydi Hasan M
rusdi_hasan@yahoo.com

Semenjak kehadiran internet pada kehidupan manusia, kontrol atas informasi bergerak dengan amat cepat.Termasuk pula informasi-informasi yang harus mendapatkan “perhatian” khusus karena nilai informasi tersebut yang sangat penting semisal informasi intelijen,militer, dan berbagai macam informasi yang sering dilabeli TOP SECRET.

Dengan adanya masalah di atas maka muncul ilmu baru pada dunia informatika yang disebut kriptografi yang merupakan pengembangan dari kriptologi.Berbagai pakar kriptografi telah mengembangkan berbagai macam algoritma enkripsi seperti AES,Lucifer,OTP,IDEA,Triple DES,DES,dsb.Diantara berbagai macam jenis algoritma, DES merupakan algoritma yang paling terkenal dan paling banyak digunakan di internet semisal untuk aplikasi e-commerce,perbankan,dll.Namun sebelum mempelajari bagaimana cara kerja algoritma DES maka akan “sedikit” dibahas terlebih dahulu riwayat hidup sang algoritma.

Kisah Hidup DES (Data Encryption Standard)

Pada sekitar akhir tahun 1960, IBM melakukan riset pada bidang kriptografi yang pada akhirnya disebut Lucifer.Lucifer dijual pada tahun 1971 pada sebuah perusahaan di london.Lucifer merupakan algoritma berjenis Block Cipher yang artinya bahwa input maupun output dari algoritma tersebut merupakan 1 blok yang terdiri dari banyak bit seperti 64 bit atau 128 bit.Lucifer beroperasi pada blok input 64 bit dan menggunakan key sepanjang 128 bit.

Lama kelamaan Lucifer semakin dikembangkan agar bisa lebih kebal terhadap serangan analisis cypher tetapi panjang kuncinya dikurangi menjadi 56 bit dengan masksud supaya dapat masuk pada satu chip.

Di tempat yang lain, biro standar amerika sedang mencari-cari sebuah algoritma enkripsi untuk dijadikan sebagai standar nasional.IBM mencoba mendaftarkan algoritmanya dan di tahun 1977 algoritma tersebut dijadikan sebagai DES (Data Encryption Standard).

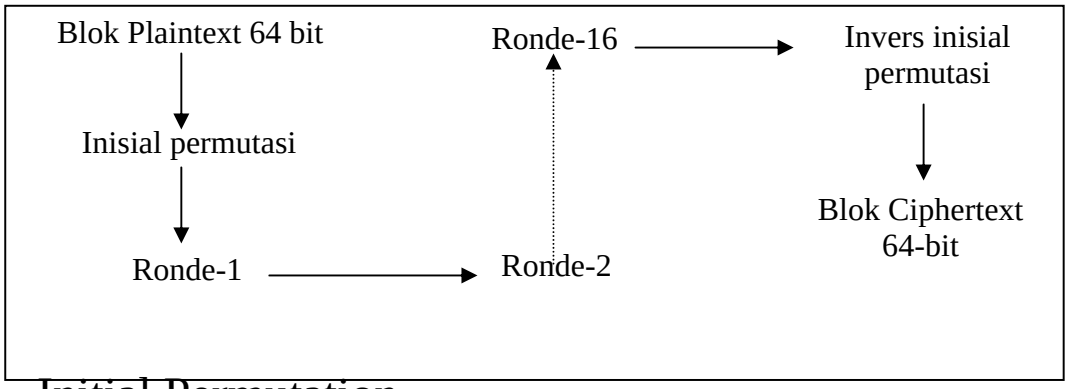
Ternyata timbul masalah setelah DES resmi dijadikan algoritma standar nasional.Masalah pertama adalah panjang kunci DES yang hanya 56-bit sehingga amat

sangat rawan dan riskan serta berbahaya , terhadap *brute-force attack*.Masalah kedua adalah struktur DES pada bagian *substitution-box* (S-box) yang diubah menurut saran dari NSA.Desain *substitution-box* dirahasiakan oleh NSA sehingga kita tidak mengetahui kemungkinan adanya kelemahan-kelemahan pada DES yang sengaja disembunyikan oleh NSA.Dan juga muncul kecurigaan bahwa NSA mampu membongkar cypher tanpa harus memiliki key-nya karena menurut para "pakar" kriptografi, DES sudah didesain secara cermat sehingga kalau S-box ini diubah secara acak maka sangat mungkin DES justru lebih mudah "dijebol" meskipun DES cukup kebal terhadap serangan *differential cryptanalysis* maupun *linier cryptanalysis.*

Seperti kata peribahasa "Karena susu setitik rusak iman sebelanga" ,.Di dunia ini tak ada ciptaan manusia yang sempurna.Pada tahun 1998, 70 ribu komputer di internet berhasil menjebol satu kunci DES dengan waktu sekitar 96 hari.Bahkan pada tahun 1999 berhasil dibobol dalam waktu kurang dari 22 hari.Pada tanggal 16 juni 1998 ada sebuah kelompok yang menamakan dirinya Electronic Frontier Foundation (EFF) telah berhasil memecahkan DES dalam waktu 4-5 hari menggunakan komputer yang dilengkapi dengan Integrated Circuit Chip DES Cracker.Di akhir tragedi ini, DES dianggap sudah tak aman lagi sehingga ia dicampakkan begitu saja dan digantikan oleh AES (Anvanced Encryption Standard).

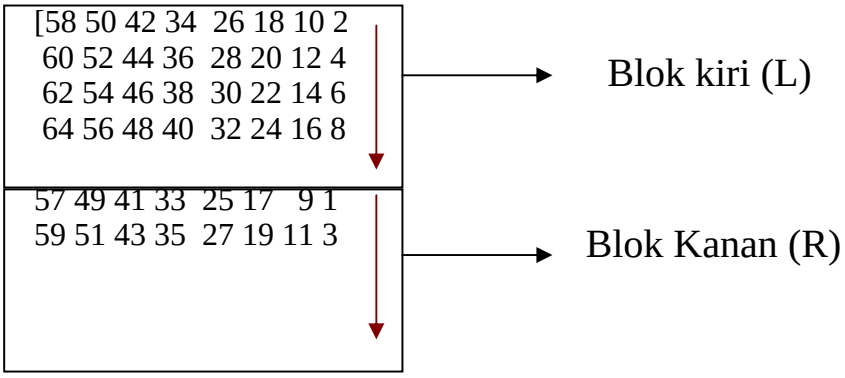
Proses Kerja DES

Cara kerja DES secara sederhana dapat digambarkan sebagai berikut :



Initial Permutation

Plaintet sebesar 64-bit akan dipecah menjadi 2 bagian yaitu Left (L) dan Right.Bit-bit dari plaintext akan mengalami permutasi sehingga susunannya akan berubah sebagai berikut :

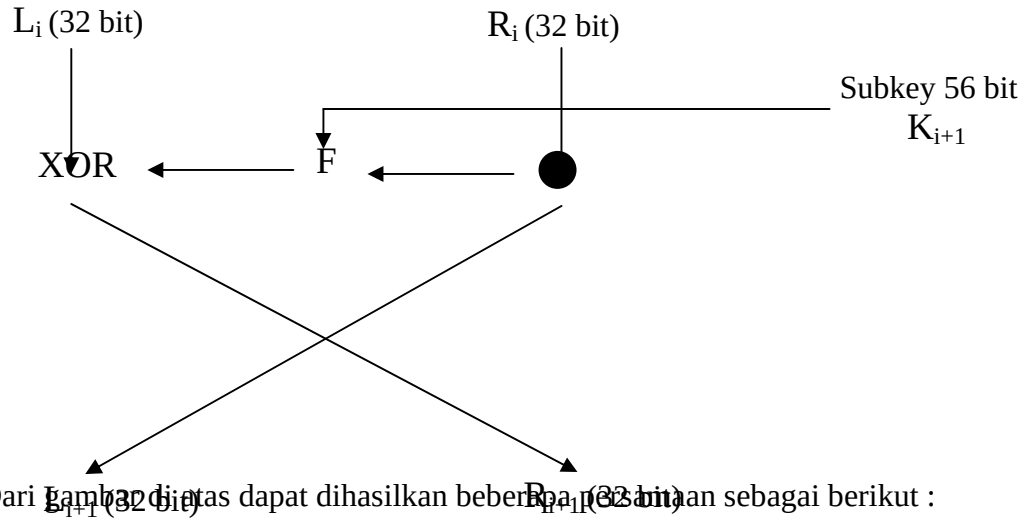


61 53 45 37 29 21 13 5
63 55 47 39 31 23 15 7]

Maksud daripada keterangan di atas adalah, bit yang pada plaintext terletak pada urutan ke 58, setelah di Inisial permutasi posisinya berubah yang tadinya berada di urutan ke 58 menjadi urutan ke-1 atau yang pertama.Jika diperhatikan, bit-bit bernomor genap setelah dipecah berada di blok L (kiri) dan bit yang bernomor ganjil terletak di blok R (kanan).Permutasi diatas menggunakan sebuah urutan yang ditunjukkan oleh arah panah berwarna merah.Dimulai pada kolom paling kanan dan bit-bit yang ada akan bergerak secara urut dengan vertikal ke arah bawah.Sebagai contoh pada blok L, bilangan genap asli terkecil adalah 2 sehingga dimulai dengan angka 2 lalu 4,6,8 lalu menuju kolom di sebelah kirinya 10,12,dst.

16 Ronde pada DES

Algoritma DES mengalami 16 ronde untuk membentuk sebuah cipher.Pada setiap ronde, blok R (kanan) tidak akan mengalami perubahan apapun karena hanya akan dipindah menjadi blok L pada ronde selanjutnya.Namun blok R akan digunakan bersamaan dengan subkey 56-bit untuk diolah pada fungsi F dan akan di XOR-kan dengan blok L (kiri).Jika bingung dengan penjelasan di atas, coba perhatikan dan amati skema pada masing-masing ronde DES di halaman selanjutnya.



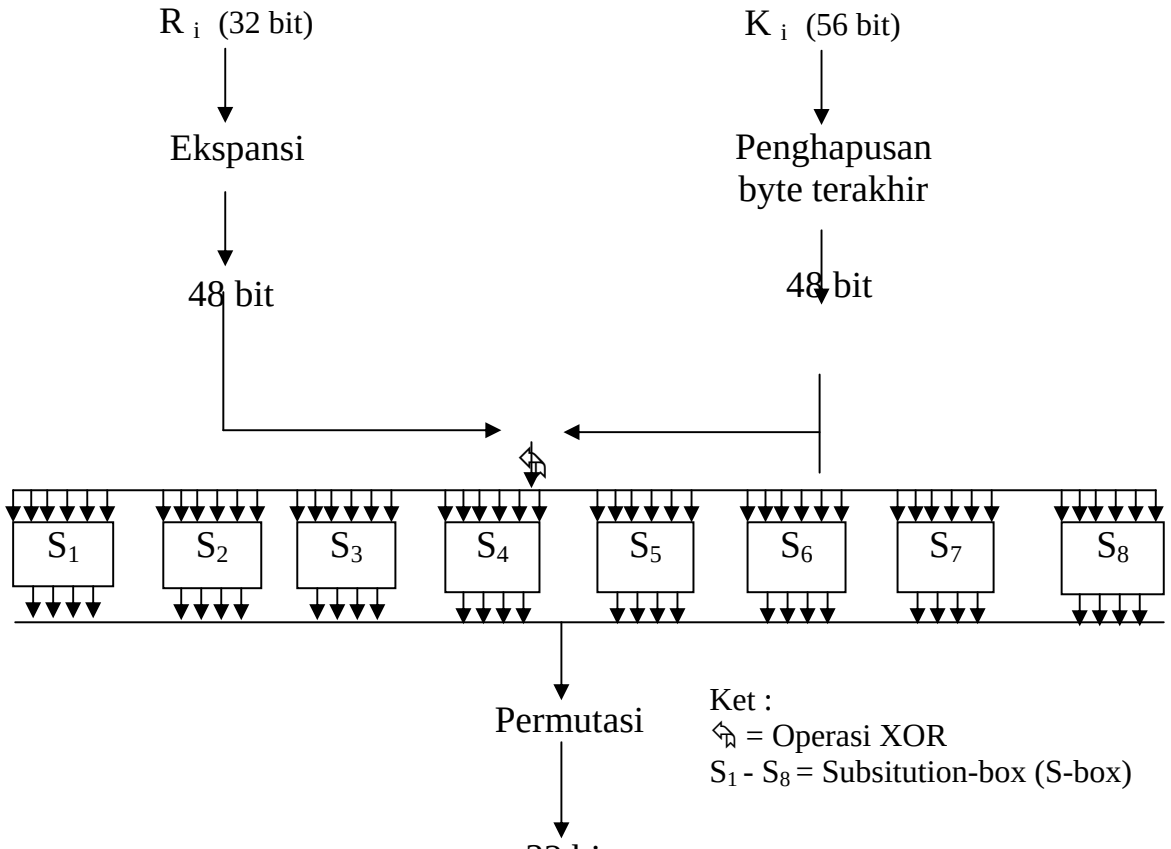
Dari gambar di atas dapat dihasilkan beberapa persamaan sebagai berikut :

1. $L_{i+1} = R_i$
2. $R_{i+1} = L_i \oplus F(K_{i+1}, R_i)$

Ket :
$\oplus$ = Operasi XOR

Nah, semoga kamu sudah paham setelah melihat penjelasan, bagan, dan persamaan yang telah dijelaskan di atas tadi.Fungsi F merupakan satu-satunya fungsi linier yang nilainya berubah-ubah tergantung pada nilai Blok R (kanan) dan sub-key 56-

bit.Saya yakin kalau kamu sudah bertanya-tanya tentang fungsi F.Sekarang, untuk kesekian kalinya, perhatikan lagi sebuah bagan di halaman selanjutnya (soalnya tempatnya nanggung banget nih ,)



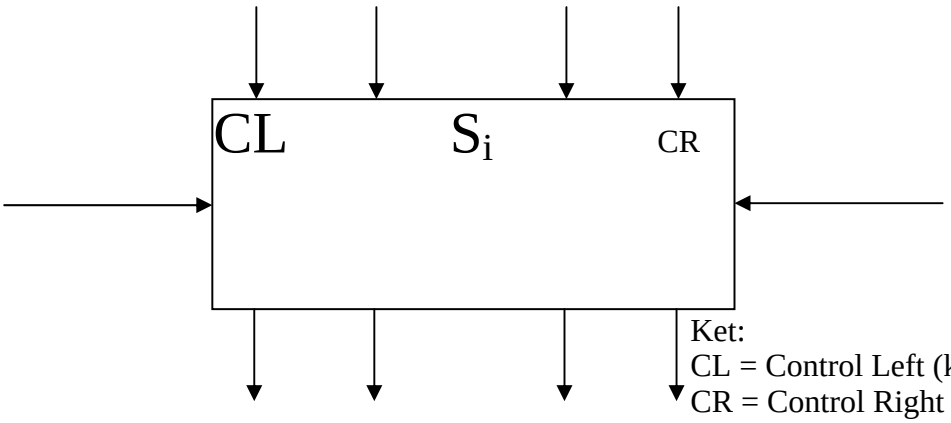
Fungsi F dapat dilihat pada gambar diatas.Blok R (kanan) di-ekspansi sehingga susunannya akan berubah sebagai berikut :

[**32** 1 2 3 4 **5** **4** 5 6 7 8 **9**
8 9 10 11 12 **13** **12** 13 14 15 16 **17**
16 17 18 19 20 **21** **20** 21 22 23 24 **25**
24 25 26 27 28 **29** **28** 29 30 31 32 **1**]

Angka-angka yang dicetak tebal merupakan bit-bit yang akan ditambahkan pada input sebesar 32 bit agar menjadi 48 bit.Angka tersebut akan muncul 2 kali seperti angka 1,4,5, dst.Bilangan yang dicetak tebal juga bertambah dengan aturan (a,a+8,a+16,a+24 dengan a sebagai bilangan minimum pada satu kolom)secara vertikal.Jika angka yang dicetak tebal dihilangkan maka akan muncul bit-bit yang berurutan.Jadi bit pertama yang akan di-outputkan oleh blok ekspansi adalah bit ke 32

pada input ekspansi.

Hasil dari output ekspansi akan di XOR dengan subkey yang telah mengalami penghapusan byte terakhir (1 byte=8 bit) yang diubah ukurannya dari 56 bit menjadi 48 bit.Hasil XOR tadi adalah sebuah bilangan 48 bit yang akan dibagi secara merata kepada 8 Substitution Box sehingga masing-masing box akan di-input sebesar 6 bit (48/8=6).Substitution box yang agak gamblang dapat dilihat pada gambar di bawah ini :



Ket:
CL = Control Left (kontrol kiri)
CR = Control Right (kontrol kanan)

dari 6 bit input pada S-box, bit paling kiri akan dimasukkan pada CL (Control Left) dan bit paling kanan akan dimasukkan pada CR (Control Right) sedangkan 4 bit sisanya akan dianggap sebagai input yang biasa-biasa saja ,.

Output dari masing-masing S-box akan menjadi bilangan sepanjang 4 bit sehingga output total dari 8 S-box menjadi 32 bit (4*8=32).Lantas, bagaimana pengolahan input pada masing-masing S-box ? OK, saya akan memberikan salah satu contoh yaitu pada S-box yang pertama.Amati tabel di bawah ini :

Bit kontrol		***Nilai input 4 bit***															
CL	CR	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
1	0	4	1	14	8	13	6	2	11	15	12	9	7	7	10	5	0
1	1	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

Contoh :
input pada S-box ke adalah 110010.Berapa nilai outputnya ?

Jawab:
Bit paling kiri akan dijadikan CL dan bit paling kanan akan dijadikan CR.

110010
CL=1

CR=0

Bit kontrol		***Nilai input 4 bit***															
CL	CR	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
1	0	4	1	14	8	13	6	2	11	15	12	9	7	7	10	5	0
1	1	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

Bit ke 2 sampai bit ke 5 harus diubah ke bilangan basis 10 atau bilangan desimal.

Dari contoh di atas bit ke 2 sampai ke 5 bernilai 1001.

$1001_{(2)} = 9_{(10)}$

Bit kontrol		***Nilai input 4 bit***															
CL	CR	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
1	0	4	1	14	8	13	6	2	11	15	12	9	7	7	10	5	0
1	1	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

Pertemuan antara baris dan kolom di atas merupakan output dari s-box pertama yang kemudian harus diubah dalam bentuk biner.

$12_{(10)} = 1100_{(2)}$

Jadi jika input dari s-box pertama adalah 110010 maka outputnya bernilai 1100.

Jika seluruh input pada S-box telah didapatkan hasil yang panjangnya 32 bit maka hasil tersebut akan dipermutasi dengan aturan :

```
[16  7   20 21   29 12  28  17
 1  15   23 26    5  18  31  10
 2   8   24 14   32 27   3   9
```

19 13 30 6 22 11 4 25]

Jadi, jika input dari blok permutasi pada bit ke 16 akan berubah menjadi bit pertama dari output blok permutasi.

Menggunakan Algoritma DES pada aplikasi openssl

Openssl merupakan toolkit kriptografi yang merupakan inplementasi dari secure socket layer (SSL) sebagaimana pada manual pages-nya

OpenSSL is a cryptography toolkit implementing the Secure Sockets Layer (SSL v2/v3) and Transport Layer Security (TLS v1) network protocols and related cryptography standards required by them.

Openssl masih berupa tool command line yang bisa digunakan untuk mengenkrip pesan dengan algoritma simetris maupun dengan fungsi hash satu arah.Di sini saya mencoba menjelaskan bagaimana menggunakan openssl untuk mengenkrip plaintext menggunakan algoritma DES.Openssl yang digunakan merupakan OpenSSL 0.9.7e yang dirilis tanggal 25 oktober 2004 pada distro mandriva 2005.

Sebagai contoh saya akan membuat sebuah plaintext untuk dienkrip menggunakan openssl.Misalnya text tersebut berisi :

```
<?
phpinfo();
?>
```

Plaintext tersebut disave dengan nama plaintext.php di direktori home.

```
$ ls
plaintext.php
```

kemudian saya enkrip menggunakan openssl.

```
$ openssl enc -e -des -in plaintext.php -out ciphertext.php
```

maka akan ada perintah untuk memasukkan key.Sebagai contoh key yang saya gunakan adalah hacker.Dan akan ada konfirmasi yang kedua untuk melakukan pengecekan.

```
enter des-cbc encryption password:
Verifying - enter des-cbc encryption password:
```

maka akan muncul sebuah file baru bernama ciphertext.php

```
$ ls
plaintext.php
ciphertext.php
```

isi dari ciphertext.php dapat dilihat menggunakan text editor.

```
Salted__¢ $Î§# ú8#YÑÏP³# {,ä ¯#ç^i S#0Ò
```

Lalu untuk mendekripsinya dapat digunakan perintah

$ openssl enc -d -des -in ciphertext.php -out plaintext2.php

Maka akan ada konfirmasi key.Anda hanya perlu memasukkan satu kali tanpa verifikasi.jika key yang dimasukkan salah maka akan muncul pesan error sbb :

bad decrypt
32416:error:06065064:digital envelope routines:EVP_DecryptFinal:bad decrypt:evp_enc.c:450:

Jika key-yang dimasukkan benar maka akan muncul file beru bernama plaintext2.php yang isinya sama dengan plaintext.php.

Mengenai openssl anda dapat mempelajarinya sendiri lewat man pages-nya atau dari howtos-howtos yang ada di internet karena saya tidak bisa membahas secara rinci di sini.Openssl juga mendukung algoritma-algoritma lain seperti blowfish, AES, IDEA, rc2, rc4, rc5 dan juga fungsi hash seperti md5 dan sha1.

Penutup

Algoritma DES dalam melakukan proses enkripsi dan dekripsi menggunakan teknik yang disebut feistel yang muncul ketika awal tahun 70-an.Fungsi pada feistel dijamin dapat didekripsi :

$$L_i \circlearrowleft f(R_i , K_{i+1}) \circlearrowleft f(R_i , K_{i+1}) = L_i$$

Fungsi di atas dijamin dapat didekripsi selama input f dalam setiap tahap dapat dikembalikan juga.Tidak perduli macam f (meskipun fungsi f tidak dapat dibalik sekalipun) kita dapat mendesain serumit apapun tanpa perlu susah-susah untuk membuat 2 algoritma untuk enkripsi dan dekripsi.Teknik ini digunakan pada banyak algoritma seperti DES, Lucifer, FEAL, Blowfish, dll.

Sepeti sudah disampaikan di awal bahwa panjang kunci DES yang hanya 56 bit sangat rawan di *brute force* sehingga saat ini digunakan 3 buah DES secara berurutan untuk mengenkripsi sebuah paintext yang disebut Triple DES.Panjang kunci Triple DES juga diperpanjang 3 kali menjadi 168 bit (56*3 = 168).

Terakhir, sebagai ucapan penutup saya mengucapkan banyak terima kasih kepada :

1. Allah SWT and my belove parents
2. staff echo (y3dips, cc lirva32, the_day, m0by, de el el)
3. para member di forum echo, linux.or.id dan jambihackerlink
4. member newbie_hacker dan jasakom-perjuangan
5. Konco-konco neng SMAN1C Cilacap

bagi yang hendak mengirimkan saran, kritik, koreksi, atau apapun silahkan diungkapkan lewat e-mail ke rusdi_hasan@yahoo.com.

Source :

1. Stalling, William. *Crypthography and Network Security*. Prentice-hall, 2003
2. Kurniawan, Yusuf. *Kriptografi Keamanan Internet dan Jaringan Komunikasi*. Penerbit Informatika, 2004
3. Ariyus, Doni. *Kamus Hacker*. Penerbit Andi, 2005
4. Anonim. *Memahami model enkripsi dan Security Data*. Penerbit Andi, 2003
5. Stiawan, Deris. *Sistem Keamanan Komputer*. Elex Media Komputindo, 2005

Biografi

Rusydi Hasan M, lahir di cilacap pada tanggal 7 juni 1989.Saat ini sedang menempuh pendidikan di bangku SMA 1 Cilacap.Sedang berusaha untuk menulis beberapa artikel yang bermula dari iseng dan coba-coba.Beberapa artikel dimuat di situs www.linux.or.id.

Sekarang lagi rajin-rajinnya ngoprek linux dan segala yang berbau opensorce.Berkeinginan untuk mendirikan KPLI di cilacap namun sampai tulisan ini dibuat, keinginan tersebut masih belum terealisir.Ingin mendalami bidang security, network, cryptography dan ....., pokoknya security lah.Bagi yang ingin menghubungi penulis bisa melalui e-mail atau yahoo-ID atau berhubungan di forum echo atau linux.or.id atau di forum jambihackerlink.

e-mail : rusdi_hasan@yahoo.com
yahoo-id : rusdi_hasan