STUDI MENGENAI SERANGAN *DIFFERENT CRYPTANALYSIS* PADA ALGORITMA *SUBSTITUTION PERMUATION NETWORK*

M Gilang Kautzar H Wiraatmadja – NIM : 13505101

Program Studi Teknik Informatika, Institut Teknologi Bandung
Jl. Ganesha 10, Bandung
E-mail : if15101@students.if.itb.ac.id

Abstrak

Kriptanalisis merupakan suatu usaha untuk memecahkan kerahasiaan suatu pesan yang terenkripsi. Salah satu metode kriptanalisis yang paling umum digunakan adalah serangan *differential cryptanalysis*. Metode ini mempelajari bagaimana suatu *input* yang berbeda akan menghasilkan suatu *output* yang berbeda pula. Kemudian dari pasangan *input* dan *output* tersebut dipelajari transformasi pesan yang terjadi, dicari keberadaan pola yang tidak acak (*non-random behaviour*), dan dieksploitasi agar properti tersebut dapat digunakan untuk menemukan kunci enkripsi.

Substitution-Permuation Network (SPN) adalah sejumlah operasi matematika yang berhubungan yang digunakan pada beberapa algoritma enkripsi blok, misalnya AES. Jaringan (*network*) tersebut terdiri dari kotak-S dan kotak-P yang melakukan tranformasi blok-blok bit *input* menjadi bit *output*. Transformasi yang biasanya digunakan adalah operasi yang bersifat efisien, misalnya operasi XOR atau *bitwise rotation*.

Dalam makalah ini akan digambarkan bagaimana sebuah serangan *differential cryptanalysis* dilakukan terhadap algoritma SPN. Metode *differential cryptanalysis* dilakukan sengaja dipilih dengan alasan metode ini merupakan salah satu jenis serangan yang paling banyak digunakan. Sementara algoritma SPN dipilih karena algoritma ini digunakan sebagai arsitektur dasar berbagai algoritma enkripsi, termasuk AES. Struktur dan operasi dasar pada algoritma ini juga mirip dengan yang digunakan pada DES dan algoritma enkripsi modern lainnya.

***Kata kunci**: Differential cryptanalysis*, *Substitution-permuation network*

1. Pendahuluan

Differential cryptanalysis merupakan satu serangan kriptanalisis yang paling kuat terhadap algoritma enkripsi kunci simetri. *Differential cryptanalysis* diperkenalkan oleh Bilham dan Shamir pada CRYPTO ’90 untuk menyerang DES di mana detail penyerangannya ditulis dalam sebuah buku. Walaupun serangan tersebut pertama kali ditargetkan terhadap DES, namun luasnya jangkauan pengaplikasiannya menyebabkan pentingnya teknik serangan tersebut masuk ke dalam pertimbangan keamanan seluruh enkripsi blok. Sebagai contoh, banyak dari seluruh kandidat yang dikirimkan untuk proses *Advanced Encryption Standard* yang diadakan oleh *National Institute of Standards* didesain menggunakan teknik-teknik yang secara spesifik ditargetkan oleh *differential cryptanalysis*. Hal ini terlihat jelas, misalnya, pada algoritma Rijndael, algoritma enkripsi yang kemudian dijadikan sebagai standard baru.

2. *Substitution-Permuation Network*

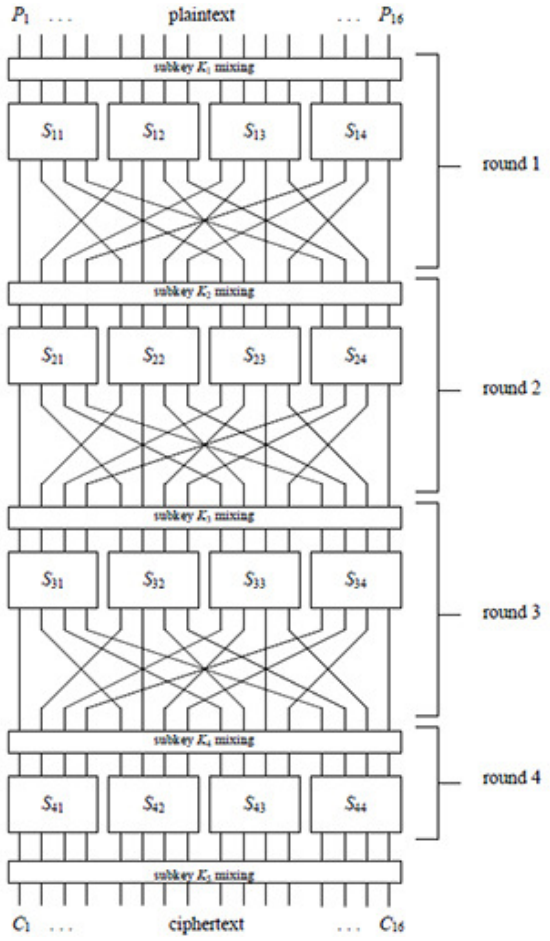
Algoritma SPN yang akan digunakan adalah algoritma dasar yang menerima masukan 16 bit dan memroses setiap blok dengan sebuah operasi dasar sebanyak 4 putaran. Setiap putaran terdiri dari operasi (1) substitusi, (2) transposisi bit, dan (3) *key-mixing*. Ilustrasi terlihat pada gambar 1.

Struktur dasar ini diperkenalkan oleh Feistel di tahun 1973. Operasi-operasi dasar tersebut mirip dengan yang ditemukan pada DES dan berbagai *cipher* modern lainnya. Sehingga walaupun algoritma SPN yang digunakan cenderung simpel, namun dapat memberikan gambaran terhadap keamanan *cipher* dengan konstruksi yang lebih besar.

2.1. Substitusi

Dalam SPN ini, 16 bit blok data akan dibagi-bagi menjadi 4 buah upa-blok berukuran 4 bit. Setiap upa-blok memebuntuk masukan terhadap kotak-S berukuran 4x4 (substitusi dengan masukan 4 bit dan keluaran 4 bit) yang dapat dengan mudah diimplementasikan dengan sebuah tabel *look-up* dari 16 buah nilai 4-bit dan diberi indeks dengan integer yang direpresentasikan oleh 4-bit tersebut. Properti paling dasar dari sebuah kotak-S adalah kotak-S merupakan *non-linear mapping*, di mana setiap bit keluaran tidak dapat direpresentasikan sebagai sebuah operasi linear terhadap bit masukan.

Algoritma SPN yang akan digunakan mengandung pemetaan *non-linear* yang sama untuk semua kotak-S (pada DES setiap putaran memiliki kotak-S yang berbeda). Serangan *differential cryptanalysis* berakibat sama baik menggunakan pemetaan kotak-S yang terus sama ataupun berganti-ganti. Pemetaan yang akan digunakan diberikan pada tabel 1. Tabel ini diambil dari DES (baris paling pertama). Pada tabel, bit paling signifikan dari notasi heksadesmila merepresentasikan bit paling kiri pada kotak-S di gambar 1 di bawah ini.



Gambar 1 Ilustrasi algoritma SPN yang digunakan

2.3. *Key-mixing*

Untuk melakukan proses *key-mixing*, kita akan menggunakan operasi *bit-wise exclusive-OR* antara bit kunci dalam setiap putaran (upa-kunci) dan blok data masukan. Upa-kunci juga diaplikasikan terhadap putaran terakhir untuk menjamin bahwa lapisan terakhir pada substitusi tidak dapat diabaikan oleh kriptanalis yang bekerja mundur dari putaran terakhir substitusi. Normalnya, dalam sebuah *cipher*, upa-kunci untuk suatu putaran diturunkan dari kunci utama melalui proses yang disebut *key scheduling*. Namun dalam algoritma SPN ini diasumsikan seluruh upa-kunci yang dilahirkan adalah independen dan tidak saling berhubungan.

2.4. Proses dekripsi

Untuk melakukan dekripsi, data secara esensial cukup diproses mundur dalam jaringan. Oleh karena itu, ilustrasinya juga mirip dengan yang ada pada gambar 1. Karena pemorsesan dilakukan mundur, maka urutan kotak-S yang digunakan adalah invers dari pemetaan pada jaringan enkripsi (*input* menjadi *ouput*, dan sebaliknya). Hal ini berarti bahwa agar SPN dapat melakukan proses dekripsi, seluruh kotak-S harus bersifat bijektif, pemetaan satu ke satu dengan jumlah bit *input* dan *output* yang sama. Juga, agar proses dekripsi berjalan benar, upa-kunci yang dipakai digunakan dalam urutan terbalik dan setiap bit upa-kunci harus dipindahkan sesuai dengan permutasinya. Tidak adanya permutasi setelah putaran terakhir menjamin bahwa proses dekripsi dan enkripsi memiliki struktur jaringan yang sama. Jika terdapat permutasi setelah substitusi yang terakhir, maka pada saat proses dekripsi deperlukan pula permutasi sebelum substitusi pertama dilakukan.

3. *Differential cryptanalysis*

Differential cryptanalysis mengeksploitasi tingginya kemungkinan beberapa kejadian (*occurrence*) dalam perbedaan *plaintext* dan perbedaan dalam putaran terakhir pada *cipher*.

Misalnya, sebuah system dengan masukan $X = [X_1 X_2 \dots X_n]$ dan keluaran $Y = [Y_1 Y_2 \dots Y_n]$. Anggap kedua masukan pada sistem adalah X' dan X'' dengan keluaran Y' dan Y''. Perbedaan yang dihasilkan adalah $\Delta X = X' \oplus X''$ di mana "$\oplus$" merepresentasikan operasi *exclusive-OR* dari bit vektor. Maka

$$\Delta X = [\Delta X_1 \Delta X_2 \dots \Delta X_n]$$

di mana $\Delta X_i = X_i' \oplus X_i''$ dengan X_i' dan X_i'' merupakan bit ke-i dari X' dan X''. Dan

$$\Delta Y = [\Delta Y_1 \Delta Y_2 \dots \Delta Y_n]$$

di mana $\Delta Y_i = Y_i' \oplus Y_i''$.

Dalam sebuah *cipher* acak yang ideal, kemungkinan bahwa perbedaan keluaran (*output difference*) ΔY keluar dengan suatu masukan tertentu ΔX adalah $1/2^n$ di mana n adalah jumlah bit pada X. *Differential cryptanalysis* mencoba untuk mengeksploitasi skenario di mana suatu ΔY muncul dengan diberikan suatu masukan ΔX dengan probabilitas p_D (jauh lebih besar dari $1/2^n$). Pasangan ΔX dan ΔY dikatakan sebagai suatu perbedaan (*differential*).

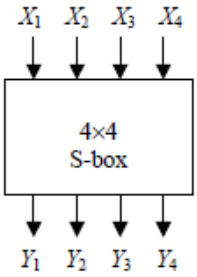
Serangan *differential cryptanalysis* adalah serangan *chosen plaintext* yang berarti bahwa penyerang dapat menentukan masukan dan memeriksa keluaran untuk memecahkan kunci. Untuk serangan *differential cryptanalysis*, penyerang akan memilih beberapa pasangan masukan X' dan X" untuk menentukan ΔX, di mana untuk ΔX tersebut sebuah ΔY muncul dengan probabilitas yang tinggi.

Kita akan melakukan investigasi terhadap pembangunan perbedaan (ΔX, ΔY) dengan *plaintext bits* seperti yang dirpresentasikan oleh X dan input terhadap putaran terakhir pada *cipher* seperti yang direpresentasikan oleh Y. Hal ini dilakukan dengan memeriksa karakteristik *differential* yang tinggi (*high-likely differential*) di mana karakteristik *differential* adalah sebuah sekuens perbedaan masukan dan keluaran pada suatu putaran sedemikian rupa sehingga perbedaan keluaran dari suatu putaran berkorespondensi dengan perbedaan masukan untuk putaran berikutnya. Dengan menggunakan *high-likely differential* tersebut maka akan memberikan kesempatan untuk mengeksploitasi informasi yang masuk ke putaran terakhir dari *cipher* untuk menurunkan bit dari lapisan upa-kunci terakhir.

Untuk membangun karakteristik *high-likely differential*, kita harus memeriksa properti dari setiap kotak-S dan menggunakanannya untuk menentukan karakteristik-karakteristik tadi. Secara spesifik kita menimbang perbedaan masukan dan keluaran dari S-Box untuk menentukan pasangan dengan probabilitas perbedaan yang tinggi. Dengan mengombinasikan pasangan perbedaan kotak-S dari setiap putaran sehingga setiap perbedaan bit *non-zero* pada keluaran di suatu putaran berkorespondensi dengan perbedaan bit *non-zero* pada masukan putaran berikutnya, maka akan mempermudah kita dalam mencari probabilitas yang tinggi.

4.2. Analisa komponen-komponen *cipher*

Pada kotak-S 4x4 (gambar 2) terlihat masukan X = $[X_1 X_2 X_3 X_4]$ dan keluaran Y = $[Y_1 Y_2 Y_3 Y_4]$.



Gambar 2 Pemetaan kotak-S

Setiap perbedaan sebuah kotak-S (ΔX, ΔY) dapat diperiksa dan probabilitas ΔY setelah diberikan ΔX dapat diturunkan dengan menganalisa pasangan masukan (X', X") di mana X' ⊕ X" = ΔX. Karena setiap urutan pasangan tidak penting, untuk kotak-S 4x4 kita cukup mempertimbangkan 16 nilai untuk X' dan kemudian nilai ΔX akan menentukan X" dengan X" = X' ⊕ ΔX.

Dengan mempertimbangkan kotak-S dari *cipher* SPN yang telah ditentukan sebelumnya (di bagian 2), kita dapat menurunkan nilai dari ΔY untuk setiap pasangan masukan (X', X" = X' ⊕ ΔX). Misalnya, nilai biner dari X, Y, dan ΔY untuk pasangan masukan (X, X ⊕ ΔX) diberikan pada tabel 1 untuk nilai ΔX 1011 (hex B), 1000 (hex 8), dan 0100 (hex 4). 3 kolom terakhir dari tabel merepresentasikan nilai ΔY untuk nilai X (seperti pada baris) dan nilai ΔX (kolom).

Dari tabel dapat terlihat bahwa angka kemunculan ΔY = 0010 untuk ΔX = 1011 adalah 8 dari 16; kemunculan ΔY = 1011 untuk ΔX = 1000 adalah 4 dari 16; dan kemunculan ΔX = 1010 untuk ΔX = 0100 adalah 0 dari 16. Jika kotak-S bersifat ideal, maka kemunculan nilai pasangan perbedaan akan menjadi 1 yang berarti nilai kemunculan 1/16 dari suatu ΔY oleh suatu ΔX. Namun hal ideal ini secara matematis tidak memungkinkan.

Data kotak-S dapat disimpan dalam suatu tabel distribusi perbedaan (*difference distribution*) di mana baris merpresentasikan ΔX (heksadesimal) dan kolom merepresentasikan ΔY..

Tabel distribusi perbedaan diberikan pada tabel 2. Setiap elemen tabel merepresentasikan jumlah kemunculan dari ΔY setelah diberikan ΔX. Di samping kasus khusus ΔX = 0 dan ΔY = 0, nilai terbesar pada tabel adalah 8, ΔX = B dan ΔY = 2. Maka kemungkinan ΔY = 2 setelah diberikan nilai pasangan masukan acak yang memuhi ΔX = B adalah 8/16. Yang terkecil pada tabel adalah 0 dan muncul untuk banyak pasangan yang berbeda. Pada kasus ini, probabilitas nilai ΔY diberikan ΔX adalah 0.

X	Y	ΔY		
		$\Delta X = 1011$	$\Delta X = 1000$	$\Delta X = 0100$
0000	1110	0010	1101	1100
0001	0100	0010	1110	1011
0010	1101	0111	0101	0110
0011	0001	0010	1011	1001
0100	0010	0101	0111	1100
0101	1111	1111	0110	1011
0110	1011	0010	1011	0110
0111	1000	1101	1111	1001
1000	0011	0010	1101	0110
1001	1010	0111	1110	0011
1010	0110	0010	0101	0110
1011	1100	0010	1011	1011
1100	0101	1101	0111	0110
1101	1001	0010	0110	0011
1110	0000	1111	1011	0110
1111	0111	0101	1111	1011

Tabel 1 Contoh pasangan perbedaan dari kotak-S

		Output Difference															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Input Difference	0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0	0	2	0	0	0	2	0	2	4	0	4	2	0	0
	2	0	0	0	2	0	6	2	2	0	2	0	0	0	0	2	0
	3	0	0	2	0	2	0	0	0	0	4	2	0	2	0	0	4
	4	0	0	0	2	0	0	6	0	0	2	0	4	2	0	0	0
	5	0	4	0	0	0	2	2	0	0	0	4	0	2	0	0	2
	6	0	0	0	4	0	4	0	0	0	0	0	0	2	2	2	2
	7	0	0	2	2	2	0	2	0	0	2	2	0	0	0	0	4
	8	0	0	0	0	0	0	2	2	0	0	0	4	0	4	2	2
	9	0	2	0	0	2	0	0	4	2	0	2	2	2	0	0	0
	A	0	2	2	0	0	0	0	0	6	0	0	2	0	0	4	0
	B	0	0	8	0	0	2	0	2	0	0	0	0	0	2	0	2
	C	0	2	0	0	2	2	2	0	0	0	0	2	0	6	0	0
	D	0	4	0	0	0	0	0	4	2	0	2	0	2	0	2	0
	E	0	0	2	4	2	0	0	0	6	0	0	0	0	0	2	0
	F	0	2	0	0	6	0	0	0	0	4	0	2	0	0	2	0

Tabel 2 Tabel distribusi perbedaan

Ada beberapa properti umum dari tabel distribusi perbedaan yang harus disebutkan. Pertama, jumlah seluruh elemen pada suatu baris harus 2n=16; jumlah dari suatu kolom juga harus 2n=16. Seluruh elemen harus bernilai genap: karena setiap pasangan nilai masukan (atau keluaran) yang direpresentasikan (X', X") memiliki nilai ΔX yang sama seperti pasangan (X', X") karena $\Delta X = X' \oplus X'' = X'' \oplus X'$. Perbedaan masukan ΔX = 0 harus menghasilkan perbedaan keluaran ΔY = 0 untuk pemetaan satu ke satu pada S-box. Maka, sudut kiri atas tabel memiliki nilai 2n=16 dan seluruh nilai elemen di baris pertama dan terakhir harus 0. Terakhir, apabila kita ingin membangun suatu S-box yang ideal, yang tidak memberikan informasi perbedaan (*differential*) sama sekali pada keluaran terhadap suatu masukan, maka S-box memiliki seluruh elemen pada tabel bernilai 1 dan probabilitas kemunculan dari suatu ΔY terhadap suatu ΔX adalah 1/2n=1/16. Namun seperti yang disebutkan sebelumnya, keadaan seperti ini tentunya tidak mungkin dicapai.

Sebelum kita dapat berbicara tentang pengombinasian pasangan perbedaan kotak-S (*S-Box difference pairs*) untuk menurunkan karakteristik *differential* dan perhitungan sebuah *differential* yang baik untuk digunakan menyerang, kita harus terlebih dahulu memperkirakan pengaruh kunci terhadap *differential* kotak-S. Pada gambar 3, masukan untuk kotak-S "tanpa kunci" adalah X dan keluaran Y. Namun, pada struktur *cipher* kita harus mempertimbangkan kunci yang diaplikasikan terhadap masukan pada tiap kotak-S. Pada kasus ini, jika kita tentukan masukan kotak-S "dengan kunci" adalah W = [W_1 W_2 W_3 W_4], kita mendapatkan perbedaan masukan terhadap kotak-S berkunci tersebut

$$\Delta W = [W_1' \oplus W_1''\ W_2' \oplus W_2'' \ldots W_n' \oplus W_n'']$$

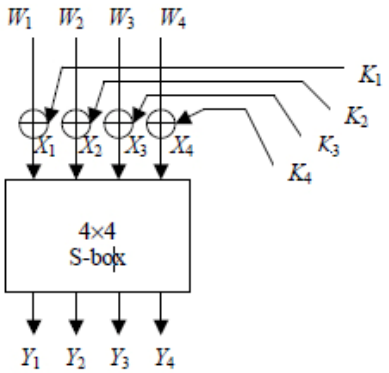
di mana W' = [W_1' W_2' W_3' W_4'] dan W'' = [W_1'' W_2'' W_3'' W_4''] adalah kedua nilai masukan.

Karena kunci tetap sama untuk W' maupun W'',

$$\Delta W_i = \Delta W_i' \oplus \Delta W_i'' = (X_i' \oplus K_i) \oplus (X_i'' \oplus K_i) = X_i' \oplus X_i'' = \Delta X_i$$

karena $K_i \oplus K_i = 0$.

Maka kunci bit tidak memiliki pengaruh terhadap nilai perbedaan masukan, sehingga dapat diabaikan. Dengan kata lain, S-box berkunci memiliki perbedaan yang sama dengan tabel distribusi perbedaan.



Gambar 3 Kotak-S berkunci

4.3. Membangun karakteristik *differential*

Ketika informasi *differential* telah dikompilasikan untuk kotak-S pada sebuah SPN, maka kita memiliki data untuk menentukan karakteristik *differential* dari seluruh *cipher*. Ini dapat dilakukan dengan mengonkatenasi pasangan perbedaan kotak-S yang tepat. Dengan membangun karakteristik *differential* dari pasangan perbedaan kotak-S tertentu di setiap putaran, sehingga suatu *differential* mengandung bit *plaintext* dan data masukan putaran terakhir pada kotak-S, maka serangan terhadap *cipher* dapat dilakukan dengan melakukan *recovery* terhadap *subset* dari bit upa-kunci pada putaran selanjutnya. Contohnya adalah sebagai berikut.

Anggap karakteristik *differential* mengandung S_{12}, S_{23}, S_{32}, dan S_{33}. Gambar 4 merupakan visualisasi karakteristik *differential* dalam bentuk diagram. Diagram tersebut mengilustrasikan pengaruh perbedaan *non-zero* pada bit pada jaringan. Ini akan mengembangkan karakteristik *differential* untuk 3 putaran yang pertama pada *cipher*, tidak seluruh 4 putaran.

Kita menggunakan pasangan perbedaan kotak-S berikut:

S_{12}: $\Delta X = B \rightarrow \Delta Y = 2$ dengan kemungkinan 8/16

S_{23}: $\Delta X = 4 \rightarrow \Delta Y = 6$ dengan kemungkinan 6/16
S_{32}: $\Delta X = 2 \rightarrow \Delta Y = 5$ dengan kemungkinan 6/16
S_{33}: $\Delta X = 2 \rightarrow \Delta Y = 5$ dengan kemungkinan 6/16

Seluruh kotak-S lainnya akan memiliki perbedaan masukan *zero* yang berarti perbedaan keluaran *zero* juga.

Perbedaan masukan pada *cipher* ekivalen dengan perbedaan masukan pada putaran pertama, yaitu

$$\Delta P = \Delta U_1 = [0000\ 1011\ 0000\ 0000]$$

di mana digunakan U_i untuk merepresentasikan masukan putaran kotak-S ke-i dan V_i untuk merepresentasikan keluaran putaran kotak-S ke-i. Maka ΔU_i dan ΔV_i merepresentasikan perbedaannnya. Maka,

$$\Delta V_1 = [0000\ 0010\ 0000\ 0000]$$

dengan memperhitungkan perbedaan pasangan perbedaan S_{12} sebelumnya dan 1 putaran permutasi.

$$\Delta U_2 = [0000\ 0000\ 0100\ 0000]$$
$$\Delta P = [0000\ 1011\ 0000\ 0000]$$

dengan probabilitas 8/16, diberikan perbedaan *plaintext* adalah ΔP.

Pada putaran kedua dengan pasangan perbedaan S_{23} menghasilkan

$$\Delta V_2 = [0000\ 0000\ 0110\ 0000]$$

dan permutasi pada putaran ke-2 menghasilkan

$$\Delta U_3 = [0000\ 0010\ 0010\ 0000]$$

dengan probabilitas 6/16 diberikan ΔU_2, dan probabilitas 8/16 x 6/16 = 3/16 diberikan ΔP. Dalam menentukan probabilitas perbedaan *plaintext* ΔP, kita telah mengasumsikan bahwa *differential* pada putaran pertama independen terhadap *differential* pada putaran ke-2, maka, probabilitas keduanya muncul ditentukan dengan hasil produk dari keduanya. Kita dapat menggunakan perbedaan kotak-S pada putaran ke 3, S_{32} dan S_{32}, dan permutasi putaran ke-3 untuk mendapatkan

$$\Delta V_3 = [0000\ 0101\ 0101\ 0000]$$

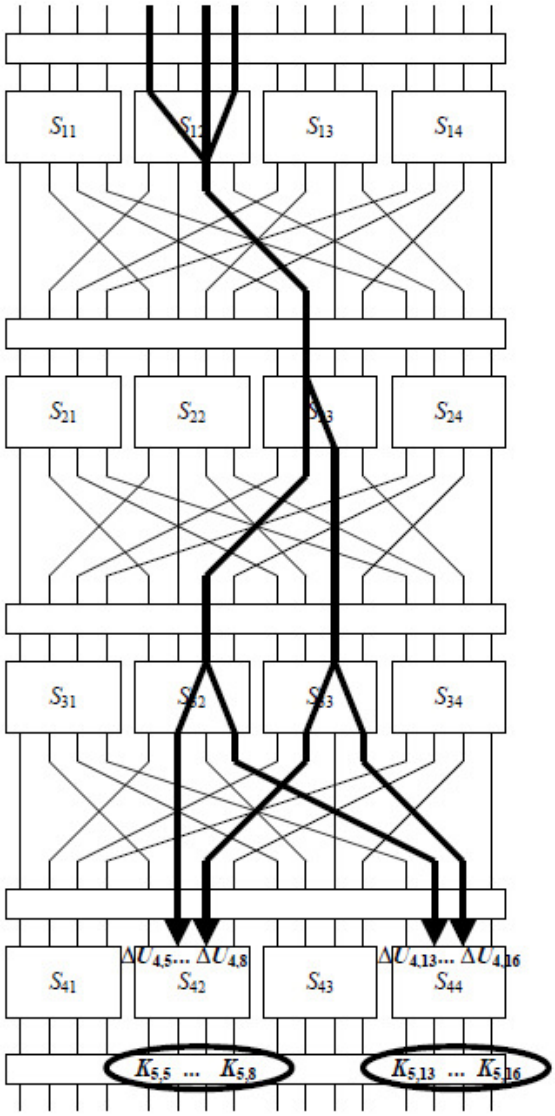
dan

$$\Delta V_4 = [0000\ 0110\ 0000\ 0110] \qquad (1)$$

dengan probabilitas $(6/16)^2$ diberikan ΔU_3 dan probabilitas 8/16 x 6/16 x $(6/16)^2$ = 27/1024 diberikan perbedaan *plaintext* ΔP, yang telah

diasumsikan independen antar pasangan perbedaan pada kotak-S di setiap putaran.

Pada proses kriptanalisis, banyak pasangan *plaintexts* di mana ΔP = [0000 1011 0000 0000] akan dienkripsikan. Dengan probabilitas yang tinggi, 27/1024, karakteristik *differential* yang telah diilustrasikan akan muncul. Pasangan-pasangan ΔP seperti itu disebut *the right pairs*. Pasangan perbedaan *plaintext* yang tidak mengandung karakteristik tersebut disebut *the wrong pairs*.



Gambar 4 Contoh Karakteristik *Differential*

4.4. Ekstraksi bit-kunci

Dari contoh *cipher* sebelumnya, karakter *differential* mempengaruhi masukan kotak-S_{42} dan S_{44} pada putaran terakhir. Untuk setiap pasangan *ciphertext*, kita akan mencuba seluruh 256 nilai [$K_{5,5}$ ... $K_{5,8}$, $K_{5,13}$ ... $K_{5,16}$]. Untuk setiap nilai upa-kunci parsial, kita akan melakukan *increment* setiap kali perbedaan masukan pada putaran akhir yang ditentukan oleh dekripsi parsial sama dengan (6), di mana kita menentukan nilai dari [$\Delta U_{4,5}$ ... $\Delta U_{4,8}$, $\Delta U_{4,13}$ ... $\Delta U_{4,16}$] dengan menjalankan data mundur sepanjang upa-kunci parsial dan kotak-S S_{24} dan S_{44}. Untuk setiap nilai kunci, total *increment* menunjukkan jumlah kemunculan perbedaan yang konsisten dengan *right pairs* (asumsi upa-kunci parsial bernilai benar). Total *increment* terbesar diambil sebagai nilai yang benar karena diasumsikan kita mengobservasi kemunculan probabilitas tinggi pada *right pair*. Perlu dicatat bahwa, pada setiap pasangan *ciphertext* tidak diperlukan eksekusi dekripsi parsial. Karena perbedaan masukan pada putaran terakhir hanya mempengaruhi 2 kotak-S, maka ketika karakteristiknya muncul, perbedaan bit *ciphertext* S-Box S_{41} S_{43} harus nol (*zero*). Dengan begitu kita dapat menyaring berbagai *wrong pairs* dengan membuang pasangan *ciphertext* yang tidak mengandung nol pada upa-blok yang tepat. Oleh sebab itu, karena pasangan *ciphertext* tidak dapat berkorespondensi ke *right pair*, maka tidak diperlukan memeriksa [$\Delta U_{4,5}$ … $\Delta U_{4,8}$, $\Delta U_{4,13}$ … $\Delta U_{4,16}$].

Kita telah mensimulasikan serangan pada *cipher* menggunakan upa-kunci acak dengan menbangkitkan 5000 pasangan *chosen plaintext* dan *ciphertext* (missal, 10000 enkripsi dengan pasangan *plaintext* yang memenuhi ΔP = [0000 1011 0000 0000]). Target nilai upa-kunci parsial yang benar adalah [$K_{5,5}$ … $K_{5,8}$, $K_{5,13}$ … $K_{5,16}$] = [0010,0100] = $[2,4]_{hex}$. Seperti yang diperkirakan, *count* tertinggi terlihat pada nilai upa-kunci parsial $[2,4]_{hex}$, menjamin bahwa serangan berhasil menurunkan bit-bit upa-kunci. Pada tabel 3terdapat ringkasan parsial dari data yang diturunkan dari *counts* upa-kunci. Nilai-nilai di tabel ini mengindikasikan perkiraan probabilitas dari kemuculan *right pairs* untuk setiap kandidat upa-kunci parsial yang diturunkan dari

$$\text{prob} = \text{count} / 5000$$

di mana *count* adalah *count* yang berkorespondensi dengan suatu nilai upa-kunci parsial. Seperti yang dapat terlihat dari sampel hasil pada tabel, probabilitas terbesar muncul untuk nilai upa-kunci parsial [$K_{5,5}$ … $K_{5,8}$, $K_{5,13}$ … $K_{5,16}$] = $[2,4]_{hex}$. Pengamatan ini ternyata benar untuk seluruh set nilai upa-kunci parsial.

Pada contoh yang kita miliki, kita akan mengharapkan probabilitas kemunculan dari *right pair* sebesar p_D = 27/1024 = 0,0264 dan kita menemukan secara eksperimental probabilitas untuk upa-kunci benar bernilai [2,4] memberikan p_D = 0,0244. Hal ini perlu dicatat karena terkadang nilai *count* besar lainnya dapat muncul pada target upa-kunci parsial yang salah.

partial subkey $[K_{5,5}...K_{5,8}, K_{5,13}...K_{5,16}]$	prob	*partial subkey* $[K_{5,5}...K_{5,8}, K_{5,13}...K_{5,16}]$	prob
1 C	0.0000	2 A	0.0032
1 D	0.0000	2 B	0.0022
1 E	0.0000	2 C	0.0000
1 F	0.0000	2 D	0.0000
2 0	0.0000	2 E	0.0000
2 1	0.0136	2 F	0.0000
2 2	0.0068	3 0	0.0004
2 3	0.0068	3 1	0.0000
2 4	**0.0244**	3 2	0.0004
2 5	0.0000	3 3	0.0004
2 6	0.0068	3 4	0.0000
2 7	0.0068	3 5	0.0004
2 8	0.0030	3 6	0.0000
2 9	0.0024	3 7	0.0008

Gambar 5 Hasil Eksperimen *Differential Attacks*

Hal ini berarti pemeriksaan terhadap target upa-kunci parsial yang salah tidak secara tepat ekivalen dengan perbedaan acak (*random differences*) pada nilai *differential* yang diharapkan. Ada beberapa faktor yang mempengaruhi *count* menjadi berbeda dengan yang secara teoritis diharapkan, termasuk property kotak-S mempengaruhi dekripsi parsial untuk upa-kunci parsial yang berbeda, ketidaktepatan pada asumsi independen yang diperlukan untuk menentukan probabilitas karakteristik, dan konsep *differential* yang tersusun dari berbagai karakteristik *differential* yang berbeda.

5. Kesimpulan

Pada makalah ini telah disajikan konsep fundamental serangan *differential cryptanalysis* yang diaplikasikan pada sebuah algoritma enkripsi dasar, yaitu SPN. Walaupun algoritma yang dipakai tergolong tidak rumit, namun struktur ini berguna untuk membangun penjelasan mengenai jenis serangan *differential cryptanalysis* tersebut.

Berbagai modifikasi dan ekstensi terhadap metode kriptanalisis *differential attacks* telah dikembangkan dan dianalisa hingga sat ini. Pada makalah ini tidak dibahas mengenai konsep dalam dan mendetil, namun dapat mempermudah memberikan konsep untuk penelitian lebih lanjut. Pada makalah ini juga tidak dibahas mengenai metode untuk menentukan karakteristik *differential* yang paling baik.

DAFTAR PUSTAKA

[1] E. Biham and A. Shamir, "Differential Cryptanalysis of DES-like Cryptosystems", *Journal of Cryptology*, vol. 4, no. 1, pp. 3-72, 1991.

[2] M. Matsui, 'The First Experimental Cryptanalysis of the Data Encryption Standard", *Advances in Cryptology – CRYPTO '94 (Lecture Notes in Computer Science no. 839)*, Springer-Verlag, pp. 1-11, 1994.

[3] Munir, Rinaldi, "*Bahan Kuliah IF5054 Kriptografi*", Departemen Teknik Informatika, Institut Teknologi Bandung, 2004.

[4] H.M. Heys and S.E. Tavares, "Substitution-Permutation Networks Resistant to Differential and Linear Cryptanalysis", *Journal of Cryptology*, vol. 9, no.1, pp. 1-19, 1996.

[5] Howard M. Heys, "*A Tutorial on Linear and Differential Cryptanalysis*", University of Newfoundland.