**MAKALAH PROJECT AKHIR KRIPTOGRAFI**

**TRIPLE DES**



**Disusun Oleh :**

|  |  |  |
| --- | --- | --- |
| **No** | **Nama** | **Nim** |
| **1** | **Destian Gilang Satrianto** | **201581090** |
| **2** | **Indri Yanti** | **201581109** |
| **3** | **Susan** | **201581018** |
| **4** | **Lia Nuriyah** | **201581089** |
| **5** | **Ratna Dewi Pratiwi** | **201581036** |
| **6** | **Nur Alif** | **201581044** |
| **7** | **Wahyu Adjie Prasetyo** | **201581025** |
| **8** | **Windo Yofena** | **201581010** |
| **9** | **Bambang Muhammad Syafi’i** | **201581052** |

**FAKULTAS ILMU KOMPUTER PROGRAM STUDI TEKNIK INFORMATIKA**

**UNIVERSITAS ESA UNGGUL**

**JAKARTA 2018**

**KATA PENGANTAR**

Puji syukur kami panjatkan kehadirat Allah SWT karena atas bimbingan dan petunjuk serta kemudahan yang diberikan-nya, kami dapat menyelesaikan project akhir yang Tema “**Algoritma Kriptografi Triple DES**.” Ini dengan baik dan tepat pada waktunya.

Kami menyadari bahwa dalam penyusunan Makalah Project Akhir ini tidak akan tuntas tanpa adanya bimbingan serta bantuan dari berbagai pihak. Oleh karena itu, pada kesempatan ini kami ucapkan terima kasih yang sebesar-besarnya, khususnya kepada :

1. Bapak Tri Ismardiko Widyawan selaku dosen mata kuliah Kriptografi.
2. Teman-teman kelompok selaku kami dan pembuat Makalah Project Akhir ini. Dan untuk teman-teman yang lain yang tergabung dalam kelas Kriptografi “ sesi 01”.

Kami menyadari dan mengakui bahwa dalam makalah project akhir ini masih banyak terdapat kekurangan dan kesalahan baik dari segi rangkaian kalimat maupun penyajian materinya. Oleh karena itu, untuk perbaikan makalah project akhir ini, kami mengharapkan saran dan kritikan yang membangun dari semua pihak untuk dijadikan pedoman dalam penulisan kearah yang lebih baik lagi. Akhir kata, semoga Makalah Project Akhir ini dapat berguna dan bermanfaat bagi kita semua.

Jakarta, 14 Januari 2018

Penulis

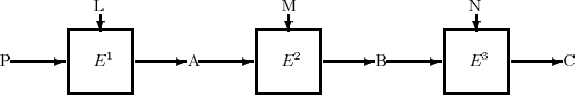
**SEJARAH DES**

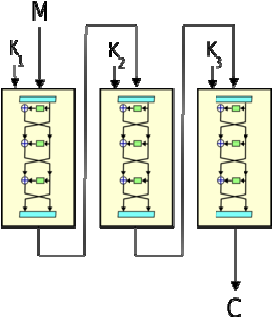
Algoritma enkripsi yang paling banyak digunakan di dunia adalah Data Encryption Standard (DES) yang diadopsi oleh NIST (National Institute of Standards and Technology) sebagai standar pengolah informasi Federal AS. Data dienkrip dalam blok-blok 64 bit menggunakan kunci 56 bit. DES mentransformasikan input 64 bit dalam beberapa tahap enkripsi ke dalam output 64 bit. Dengan demikian, DES termasuk keluarga block cipher. Dengan tahapan dan kunci yang sama, DES digunakan untuk membalik enkripsi.DES sangat banyak digunakan untuk melindungi data dalam dunia elektronika khususnya di bidang perbankan, finansial, dan e-commerce. Sayangnya, DES juga mempunyai kontroversi tentang keamanannya. Untuk mengetahuinya, kita lihat sejarah pembuatannya.Pada akhir tahun 60-an, IBM memulai riset di bidang kriptografi yang kemudian disebut dengan nama LUCIFER dan dijual ke sebuah perusahaan di London pada tahun 1971. LUCIFER merupakan cipher blok yang beroperasi pada blok masukan 64 bit, menggunakan kunci 128 bit. Kemudian dikembangkan versi LUCIFER yang lebih kebal terhadap analisis cipher, namun dengan pengurangan ukuran kunci, sehingga menjadi 56 bit, agar dapat masuk pada satu chip.Sementara itu, biro standar Amerika memerlukan cipher standar nasional. IBM mendaftarkan ciphernya yang akhirnya dijadikan sebagai DES (Data Encryption Standard) pada tahun 1977. Terdapat dua masalah dalam kasus ini. Kunci yang semula 128 bit dikurangi 72 bit hingga menjadi hanya 56 bit. Suatu pengurangan yang terlalu besar, sehingga rawan terhadap serangan brute force. Masalah kedua adalah desain struktur internal DES, bagian substitusinya (S-box), masih dirahasiakan. S-box ini diubah mengikuti saran NSA. Akibatnya, kita tidak bisa yakin bahwa struktur internal DES bebas dari titik-titik lemah yang sengaja disembunyikan, yang membuat NSA dapat membuka cipher tanpa harus mengetahui kuncinya. Menurut penelitian para ahli kriptografi, DES didesain dengan sangat cermat, sehingga bila anda mengubah-ubah kotak-S ini secara acak, sangat mungkin sekali DES yang anda hasilkan justru menjadi lebih mudah dibobol.

**TRIPLE DES**

Triple DES dikenal juga dengan nama TDES, atau lebih standar lagi adalah TDEA ( Triple Data Encryption Algorithm). TDES mengaplikasikan DES sebanyak 3 kali, dengan 3 buah kunci.

Berikut adalah skema global dari TDES:





1. Analisis Keamanan
   1. Kriptanalisis pada DES

Berdasarkan penelitian yang dilakukan oleh Biham dan Shamir pada tahun 1990, DES dapat dengan mudah dihancurkan keamanannya dengan menggunakan differential kriptoanalisis. Sebuah paper dari Matsui pada tahun 1993 juga meneliti bahwa DES juga dapat dengan mudah dipecahkan dengan menggunakan cryptoanalisis linear. Berikut akan dibahas metode linear dan differential cryptanalisis yang dapat memecahkan DES.

* + 1. Kriptoanalisis *Differential*

Serangan ini merupakan serangan pertama yang berhasil menembus keamanan 16 ronde DES dengan kompleksitas lebih kecil daripada

**255** yang merupakan kompleksitas dari

*exhaustive search*. *Exhaustive search* adalah pencarian terhadap semua kemungkinan solusi.

Tahap analisis data merupakan tahap untuk mencari kunci dengan cara menganalisis sekitar **236** teks cipher dalam waktu **237**. 236 teks cipher ini diperoleh dari kumpulan data yang lebih besar yang terdiri dari **247** teks biasa yang dipilih berdasarkan kriteria *simple bit repetition*. Dengan menggunakan kriteria ini, 99,9% dari teks cipher yang dihasilkan tidak akan digunakan.

Versi awal dari serangan *differential* menggunakan tabel pencacah yang besar, sementara versi yang baru menggunakan sedikit memori dan dapat diproses oleh sistem parallel yang maksimum terdiri dari **233** prosesor. Setiap tambahan prosesor meningkatkan kecepatan analisis secara linear. Sebagai tambahan, versi baru tetap dapat dilakukan, walaupun teks cipher yang dianalisis berasal dari **233** kunci yang berbeda. Misalnya karena perubahan kunci yang sering dilakukan saat pengumpulan data dilakukan.

Serangan dapat dilakukan tanpa tergantung dari jumlah teks cipher yang dapat dianalisis. Namun tentu saja tingkat kesuksesan serangan akan meningkat sesuai dengan jumlah teks cipher yang disediakan. Sebagai contoh, ketika **239** teks cipher dihasilkan dari **240** kumpulan teks biasa, waktu analisis akan menurun menjadi **230** dan peluang kesuksesan akan turun menjadi 1%.

Untuk menjelaskan kriptoanalisis differential pada DES, kita mengabaikan permutasi awal IP dan permutasi akhir IP-1. Hal ini dilakukan mengingat permutasi awal dan akhir tidak berpengaruh terhadap analisis yang dilakukan.

Versi lama dari serangan terhadap 15 ronde DES dilakukan berdasarkan sifat pengulangan yang terdapat pada dua ronde berikut ini :

## Ωp=(δ,0)



gangguan dari nilai acak yang diperoleh dari pasangan yang salah. Hal ini terjadi karena nilai yang dihasilkan pasangan yang benar akan menjadi nilai yang paling sering muncul (*most frequently value*).

Algoritma yang digunakan adalah menggunakan pencacah terpisah untuk setiap kali suatu nilai dihasilkan dan menandai indeks dari pencacah yang memiliki nilai tertinggi. Pendekatan ini membutuhkan kapasitas memori yang besar dan memiliki kemungkinan berhasil yang kecil jika jumlah pasangan yang dianalisis kurang dari ambang batas untuk menghilangkan gangguan dari pasangan yang salah.

 A’= 0 F

 B’= 0 F

## a’= 0

b’= δ

selalu

1 234

Pada versi yang baru, kita berusaha menganalisis tiap pasangan dengan lebih baik dan berusaha menghasilkan 56-bit kunci (versi lama hanya menghasilkan subset dari kunci). Hasil ini memungkinkan kita untuk melakukan uji coba langsung terhadap kunci yang dihasilkan. Uji coba dapat dilakukan secara parallel dan hanya membutuhkan kapasitas memori serta kekuatan prosesor yang kecil. Algoritma ini menjamin didapatnya kunci jika

## ΩT=(0,δ)

Karakteristik 13 ronde diperoleh dengan menganalisis karakteristik dua ronde di atas dengan pengulangan sebanyak enam setengah kali. Kemungkinan suksesnya serangan adalah 2-47,2. Serangan akan menggunakan karakteristik ini untuk ronde 1 sampai 13, diikuti dengan serangan 2R untuk ronde 14 dan

15. Setiap pasang teks biasa yang menunjukan karakteristik ini disebut sebagai „ *right pair*“.

Kriptoanalis akan berusaha untuk menghilangkan semua pasangan teks yang masukan dan keluarannya tidak memenuhi kriteria. Namun kriptoanalis tidak dapat menentukan nilai antara, sehingga proses eliminasi tidak sempurna dan menyisakan campuran dari pasangan yang benar dan salah.

Pada versi terdahulu dari kriptoanalisis differential, tiap pasangan yang belum tereliminasi akan memberikan petunjuk kemungkinan nilai bit tertentu dari kunci. Pasangan yang benar akan selalu memberikan petunjuk yang benar untuk beberapa bit (dan memberikan petunjuk yang salah untuk bit lainnya). Pasangan yang salah akan memberikan nilai acak. Jika jumlah pasangan yang benar cukup banyak, analisis dari pasangan yang benar dapat menghilangkan

ditemukan pasangan yang benar.

Karena analisis pasangan yang berbeda dapat dilakukan secara parallel, seringnya perubahan kunci tidak mencegah ditemukannya kunci pada saat kunci tersebut masih berlaku.

Kunci kesuksesan serangan terletak pada tingginya kemungkinan karakteristik yang memungkinkan untuk mempertimbangkan bahwa sedikit pasangan yang salah sebelum munculnya pasangan yang benar.

Algoritma ini digunakan untuk menyerang 15 ronde DES. Dengan menggunakan serangan ini, kemungkinan didapatnya suatu karakteristik untuk menyerang 15 ronde DES adalah sebesar:

.

Untuk menyerang 16 ronde DES, cara yang mungkin ditempuh adalah dengan melakukan pengulangan karakteristik di atas satu kali lagi. Namun, hal ini mengurangi peluang ditemukannya karakteristik dari 2-47.2 menjadi 2-55.1. Angka ini lebih buruk dari kemungkinan yang didapat dengan menggunakan *exhaustive search*. Perlu ditemukan serangan baru untuk menambah ronde tanpa mengurangi peluang ditemukannya kunci.

Tujuan kita saat ini adalah tanpa mengurangi peluang ditemukannya kunci, dapat menghasilkan pasangan teks biasa yang hasil operasi XOR pertamanya dapat menjadi masukan yang sesuai untuk karakterisktik 13 ronde (dari ronde 2 sampai 14).

Asumsi, P adalah sebuah 64 bit teks biasa. v0,

…, v4095 merupakan 212 nilai 32-bit. Nilai 32- bit ini merupakan seluruh kemungkinan nilai dari 12 bit posisi yang akan dikenakan operasi XOR dengan 12 bit keluaran dari S1, S2 dan S3 setelah ronde pertama dan dengan 0 pada tempat lainnya.

Sekarang kita akan merancang struktur yang terdiri dari 213 teks biasa :



Pasangan teks biasa yang akan kita gunakan adalah semua pasangan Pi, Pj di mana 0 < i,j <

212. Akan ada sekitar 224 pasangan teks biasa

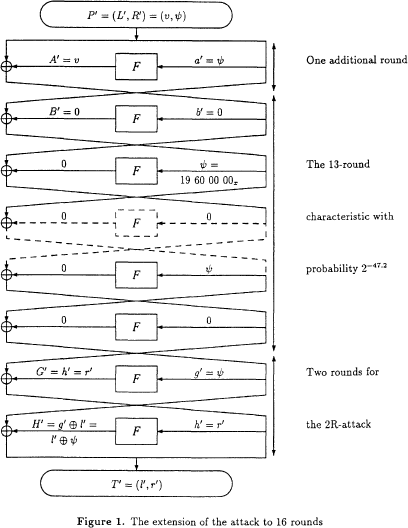
yang memenuhi criteria tersebut. Hasil operasi XOR dari pasangan-pasangan ini akan selalu memiliki bentuk (vk , δ). Setiap vk akan muncul tepat 212 kali.

Pada pemrosesan yang sesungguhnya, setengah bagian kiri dari P akan dikenakan operasi XOR dengan $ pada ronde pertama dengan menggunakan kunci yang sesungguhnya. Proses ini akan menghasilkan suatu nilai yang tidak nol hanya pada keluaran S1, S2 dan S3. Nilai ini merupakan salah satu dari vk, sehingga jika kita menggunakan tepat 212 pasangan teks biasa, hasil XOR dari fungsi F pertama akan menjadi nol. Hasil ini merupakan hasil yang kita harapkan. Setelah menukar posisi kanan dan kiri dari keluaran, kita akan mendapatkan nilai (δ, 0). Hasil ini akan dimasukkan ke dalam struktur yang pada akhirnya memiliki peluang sebesar 212 x 2-47.2

= 2-35.2 untuk menghasilkan pasangan yang benar.

Masalah dengan menggunakan pendekatan ini adalah kita tidak mengetahui nilai vk yang dapat menjadikan nilai keluaran fungsi F pertama menjadi nol, dan karenanya, kita juga tidak mengetahui pasangan teks biasa yang perlu kita proses lebih lanjut. Mencoba seluruh 224 kemungkinan akan memakan waktu yang terlalu lama, oleh karena itu kita mencoba menggunakan struktur perkalian silang untuk mendapatkan pasangan yang benar dalam tempo 212.

Dengan menggunakan pasangan yang benar, hasil keluaran XOR setelah 16 ronde adalah nol untuk keluaran dari lima buat S-Box, misal S4, S5, ..., S8 pada 20 posisi bit. Selanjutnya kita dapat mengurutkan dua buah grup dari 212 teks biasa Ti dan Tj dengan menggunakan 20 posisi bit tersebut dan mendeteksi kemunculan berulang dari nilai-nilai dalam pasangan 224 teks cipher dalam waktu 212.



Semua pasangan teks biasa yang gagal memenuhi tes ini akan menghasilkan teks cipher yang tidak nol pada 20 posisi bit, dan karenanya tidak dapat menjadi pasangan yang benar.

Sebanyak 224 pasangan dapat melewati tes ini dengan peluang 2-20. Kita berharap akan tersisa sekitar 24 pasangan yang berhasil melewati tes ini.

Dengan menguji tambahan S-Box pada ronde pertama, limabelas dan enambelas, dan mengeliminasi semua pasangan yang nilai XOR nya tidak mungkin digunakan. Dengan

menggunakan cara ini, kita dapat membuang sekitar 92,55% dari pasangan yang masih ada, menyisakan sekitar 16 x 0,0745 = 1,19 pasangan per struktur. Hasil ini sesuai dengan perkiraan yang didapat saat tahap pengumpulan data.

Tes tambahan ini dapat dilakukan dengan mengamati tabel beberapa kali dan menyimpan hasilnya dalam tabel yang kecil. Waktu yang dibutuhkan untuk melakukan hal ini jauh lebih kecil daripada waktu yang dibutuhkan untuk melakukan satu kali percobaan enkripsi pada *exhaustive search*. (Perhatikan bahwa tes tambahan ini menghilangkan pasangan yang salah, namun tidak semua. Karena itu masukan dari tahap data analisis masih terdiri atas campuran pasangan benar dan salah).

Tahap analisis data dari versi kriptoanalis *differential* terdahulu memerlukan ukuran tabel yang sangat besar, yaitu sekitar 242 cacahan. Untuk itu algoritma yang baru berusaha mengatasi hal ini dengan cara langsung mencoba setiap nilai yang mungkin menjadi kunci. Sebuah nilai diperhitungkan, jika nilai tersebut dapat menghasilkan keluaran XOR yang diinginkan pada ronde terakhir, dan juga pada ronde pertama dan ronde ke lima belas.

Masukan untuk S3, S5, … , SS akan selalu nol. Sebagai dampak dari algoritma penjadwalan kunci, semua bit genap dari bagian kiri kunci akan digunakan untuk S-Box S1, S2 dan S3 untuk ronde pertama dan ronde ke lima belas. Pada ronde ke enam belas, bit-bit ini akan menjadi masukan untuk S-Box S1, S2, S3 dan S4.

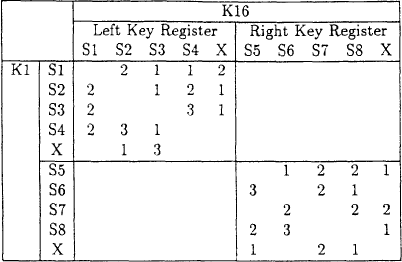
Hanya 24 bit dari kunci kanan yang akan digunakan pada ronde ke-enam belas. Karena itu pada S-Box ini 28 + 24 = 52 bit digunakan

232

Nilai dari 52 bit kunci, dan masing-masing berkaitan dengan 16 angka yang mungkin untuk 56 bit kunci. Berdasarkan ini, setiap struktur akan menghasilkan 1.19 – 0.84 x 16 =

16 pilihan kunci. Dengan menyelidiki lebih lanjut menggunakan dua buah ronde tambahan, kita dapat memperoleh kunci hanya dengan melakukan seperempat enkripsi DES.

Usaha yang harus dikeluarkan untuk menyaring pasangan ini dapat dilihat pada tabel di bawah ini:



**Tabel 1.** Jumlah bits normal yang memasuki S-box pada ronde pertama dan pada ronde ke enam belas.

 Operasi DES yang serupa. Tiap pilihan yang tersisa dari 56 bit kunci dicocokan melalui percobaan enkripsi terhadap salah satu teks normal dan membandingkan hasilnya dengan pasangan teks ciphernya. Jika tes tersebut sukses, kemungkinan besar, kunci ini adalah kunci yang sesungguhnya.

Data analisis ini dapat dijalankan secara efisien dengan memilih urutan secara hati-hati. Dalam

sebagai masukan.

0.88

dari 52 bit akan tetap

setiap urutan kita melakukan pengujian terhadap berbagai variasi bit kunci. Pertama-

dengan membandingkan keluaran XOR ronde

sebelumnya dengan nilai harapan dan membuang nilai-nilai yang tidak mungkin.

212

tama kita mengambil semua nilai yang mungkin dari enam bit kunci S, dan menghilangkan semua nilai yang tidak memberikan hasil XOR yang diharapkan dari

14 . 13 . 15

sisanya didapat dari

empat bit keluaran S-Box ini. Tindakan ini

16 16 16

membandingkan nilai keluaran XOR dari tiga S-Box dari ronde pertama dengan nilai harapannya. Jumlah yang sama juga akan didapat dengan menganalisis tiga buah S-Box pada ronde ke lima belas. Setiap pasangan yang dianalisis akan menghasilkan sekitar :

232 212 212

# . .  0.84

0.8 . . . .

8

14 13 15 14 13 15

16 16 16 16 16 16

menyaring empat dari 64 kemungkinan yang ada. Tabel 1 menunjukkan jumlah bit umum yang memasuki S-Box pada ronde pertama dan pada ronde ke enam belas. Notasi X menandakan bit-bit yang tidak digunakan pada subkey tertentu.

Kita melihat tiga bit S4Kh sama dengan tiga bit dari S3Ka. Kita akan berusaha memenuhi tiga bit yang hilang dari S3Ka dengan berbagai cara

dan mengurangi rata-rata kemungkinan menjadi dua. Dua bit dari SIKh sama dengan dua bit dari S3Ka. Dengan melengkapi empat bit yang hilang dari SIKh dan dua bit yang hilang dari S2Ka, kita dapat mengurangi jumlah kemungkinan hingga setengahnya.

Setelah menyelesaikan 13 sisa bit dari bagian kiri kunci dengan menggunakan cara yang sama, rata-rata jumlah angka yang dihasilkan untuk setengan bagian kunci ini adalah satu.

Untuk menghitung bit-bit dari bagian kanan kunci, pertama-tama kita mengambil bit-bit S box yang sebenarnya dari nilai-nilai asumsi sebelumnya yang dihasilkan dari operasi XOR. Pada ronde ke lima belas, kita mengetahui masukan dan keluaran operasi XOR dari S1, S2, dan S3. Selanjutnya kita dapat menghasilkan 4-5 kandidat masukan untuk tiap S-Box.

Dengan cara yang sama, kita dapat menghitung keluaran S-box S1, S2, S3 dan S4 pada ronde ke enam belas. XOR bit-bit dari H dengan menggunakan bit-bit yang telah diketahui dari bagian sebelah kiri teks cipher I dan dapatkan

16 bit dari g. Dari 16 bit ini, dua bit akan memasuki S1, dua bit akan memasuki S2 dan tiga bit akan memasuki S3 pada ronde ke lima belas.

Dengan membandingkan nilai-nilai bit untuk setiap kandidat masukan S-Box, kita akan mendapatkan satu kandidat masukan untuk S1, satu untuk S2 dan sekitar setengah dari seluruh percobaan kan menghasilkan kandidat untuk S3. Sekarang kita bisa mengurangi seluruh bit dari g yang akan memasuki tiga S-box dan mengurangi bit-bit yang berkaitan dari H dengan melakukan operasi H = g XOR I. Dua dari bit-bit ini adalah keluaran dari S5, dua bit merupakan keluaran dari S6, tiga bit adalah keluran untuk S7 dan 1 merupakan keluaran untuk S8. Untuk masing-masing dari 4 S-Box, kita mengetahui masukan XOR dan keluaran XOR, dan dapat memperoleh 4-5 kemungkinan masukan. Karena kita juga mengetahui bit keluaran yang seharusnya, jumlah kemungkinan masukan yang mungkin dapat dikurangi hingga nyaris sebuah untuk S5 dan S6, dua bit untuk S8, namun hanya setengah dari percobaan akan menghasilkan kandidat untuk S7. Kita dapat mengurangi 24 dari 28 bit kunci bagian kanan dengan menggunakan operasi XOR pada 24 bit masukan dari empat S-box yang nilainya telah dikembangkan dengan menggunakan setengah bagian kanan dari teks cipher.

Kita saat ini dapat menarik kesimpulan mengenai kinerja serangan baru ini seperti berikut : Setiap struktur merupakan pasangan yang benar dengan peluang 2-35.2. Tahap pengumpulan data mengenkripsikan sebanyak 235 kumpulan struktur yang terdiri atas 235-213

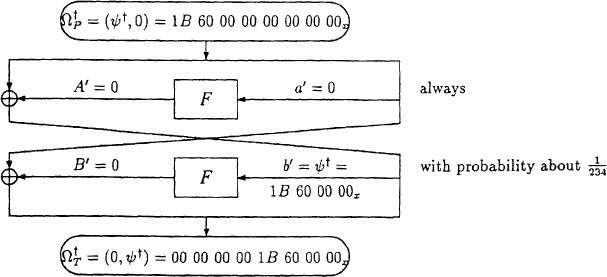
= 248 teks normal pilihan. Dari teks normal pilihan ini kita dapat memperoleh 235 x 1.19 =

235.25 pasangan (236.25 teks cipher) yang dapat

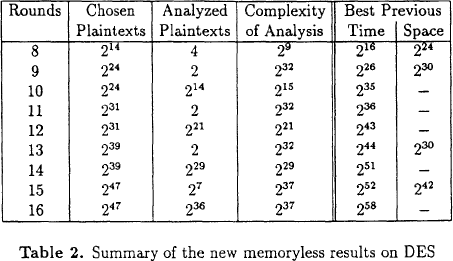
menjadi kandidat masukan dari tahap data analisis.

Kemungkinan didapatnya pasangan yang benar dari salah satu masukan yang diberikan adalah sekitar 58%. Analisis dari pasangan yang benar pasti akan menghasilkan kunci yang benar. Kompleksitas dari tahap analisis data adalah sekitar 235 – 4 = 237 operasi DES.

Untuk mengurangi jumlah teks biasa terpilih lebih jauh lagi, kita dapat menggunakan metode quartet dari [I]. Karena koleksi dasar dari teks biasa pada serangan yang baru adalah struktur (bukan pasangan), kita dapat menciptakan meta-struktur yang memuat 214 teks biasa terpilih. Teks biasa ini didapatkan dari dua struktur yang berkaitan dengan karakteristik pengulangan standard dan dari dua struktur yang berkaitan dengan karakteristik pengulangan berikut :



Karakteristik ini memiliki peluang yang sama dengan yang sebelumnya. Dengan menggunakan meta-struktur ini, kita dapat memperoleh empat kali lebih banyak pasangan dari teks biasa yang jumlahnya dua kali lebih banyak. Hal ini akan mengurangi jumlah teks biasa terpilih yang terenkripsi, dari 248 menjadi 247.



Bentuk umum dari serangan yang baru dapat diringkas seperti berikut :

Diberikan sebuah karakteristik dengan probabilitas p dan perbandingan signal dan gangguan S/N untuk sistem kripto yang menggunakan 6 bit kunci. Kita dapat menerapkan serangan tanpa memori yang

dua. Variasi dengan menggunakan bilangan genap untuk ronde n memiliki karakteristik dengan peluang p



, membutuhkan p-1 teks biasa terpilih dan menganalisa p-1x 2-10.11 dalam tempo p-1x2-10.75. Variasi yang menggunakan teks biasa yang sudah diketahui membutuhkan sekitar 2315 x p-

0.5 teks biasa (menggunakan simetri dari sistem kriptografi yang memungkinkan untuk

menggandakan jumlah enkripsi yang diketahui dan teks cipher yang bersesuaian).

Variasi dengan menggunakan angka ganjil pada ronde n akan memiliki karakteristik dengan peluang



Variasi ini membutuhkan p-1 teks biasa terpilih dan menganalisa p-1 x 2-40.2 teks biasa

dalam waktu p-1 x 2-10. Untuk nilai ganjil n,

mengenkripsi

2

teks biasa terpilih pada tahap

*p*

jika p>2-0.1, jumlah teks biasa yang dianalisis adalah dua dan kompleksitas dari tahap

pengumpulan data dan memiliki kompleksitas

2*k*

analisis data adalah 232. Namun dengan menggunakan sekitar empat kali lebih banyak

sebesar

*S* / *N*

percobaan enkripsi pada saat

teks biasa terpilih, kita dapat menggunakan algoritma clique dan mengurangi kompleksitas

tahap data analisis. Jumlah teks biasa terpilih dapat dikurangi hingga 1 dengan

*p*

menggunakan meta-struktur yang sesuai, dan dengan menggunakan efektif kompleksitas waktu yang dapat dikurangi sebesar f \* 51 jika sebuah kunci yang telah diuji dapat dibuang, dengan hanya melakukan sebagian ronde.

Berdasarkan hal-hal di atas, serangan tanpa memori dapat dilakukan kapanpun selama p > 21-k dan S/N > 1. Serangan tanpa memori memerlukan jumlah yang lebih kecil untuk teks biasa terpilih dibandingkan dengan skema menghitung. Tapi jika S/N rasio terlalu rendah, atau jumlah kunci yang kita hitung kecil, Kompleksita dari analisis data dapat lebih besar dari skema menghitung.

Pada serangan yang telah dideskripsikan sebelumnya (p=2-472, k=56, f=1/4 dan S/N=216) jumlah teks biasa yang diperlukan adalah 2/p=248.2. Jumlah ini dapat dikurangi lagi menjadi 1/p=247.2 dengan menggunakan meta- struktur. Kompleksitas dari tahap analisis data adalah setara dengan 237.2 operasi DES.

Kinerja dari serangan baru untuk bermacam- macam jumlah ronde dapat dilihat pada table

waktu dari tahap analisis data hingga kurang dari satu detik dengan menggunakan komputer biasa.

Serangan dengan menggunakan teks biasa yang sudah diketahui, memerlukan sekitar 232 x p-0.5 teks biasa (Pada kasus ini karakteristik simetri dari sistem kripto tidak membantu). Penerapan serangan dengan menggunakan teks biasa yang sudah diketahui pada 8 ronde sistem membutuhkan sekitar 238.5 teks biasa. Penerapan terhadap 12 ronde membutuhkan

247.2 teks biasa. Untuk 15 ronde dibutuhkan sekitar 255.6 teks biasa, dan untuk 16 ronde DES dibutuhkan 255 teks biasa yang telah

diketahui.

Hasil ini sedikit lebih jelek daripada exhaustive search yang membutuhkan kompleksitas 255 (Dalam kasus, menggunakan teks biasa yang sudah diketahui, hanya membutuhkan 233 teks biasa untuk menghasilkan pasangan yang sesuai dengan menggunakan paradigma ulang tahun).

Serangan ini tidak dapat diterapkan untuk teks biasa yang hanya terdiri dari karakter ASCII. Hal ini dikarenakan karakter ASCII tidak memberikan perbedaan XOR yang diharapkan.

Dengan menggunakan karakteristik pengulangan yang lainnya, kita dapat menyerang 16 ronde DES dengan menggunakan 249 teks ASCII yang sudah dipilih (dari total 256 kemungkinan teks ASCII).

* + 1. Linear cryptanalisis

Linear cryptanalisis, yang dapat disebut juga *known-plaintext attack*, dapat memecahkan DES 8 putaran (round) hanya dengan 221 known-plaintext, dan DES 16 putaran hanya dengan 247 known-plaintext. Terlebih lagi, metode ini juga dapat memecahkan DES dengan only-chipertext attack pada beberapa kondisi tertentu. Contohnya, jika plainteks terdiri dari bahasa Inggris yang menggunakan ASCII code, DES dengan 8 putaran dapat dipecahkan hanya dengan 229 chiperteks saja.

Kriptoanalisis linear merupakan sistem kedua yang berhasil mengurangi kompleksitas pertahanan DES hingga 243 teks biasa yang diketahui.

Hal-hal yang perlu diperhatikan untuk serangan ini adalah :

* + - 1. Semua operasi kecuali pada bagian S- box adalah linear.
      2. Penggunaan kunci di semua ronde adalah linear, yaitu dengan menggunakan operasi XOR.
      3. Hubungan linear antara bit-bit masukan dan bit-bit keluaran dari S- Box dapat diturunkan
      4. Persamaan linear dari teks biasa, teks cipher dan bit kunci dapat dituliskan sebagai berikut : (PΩP) XOR (CΩT) XOR (KΩK) = 0 dengan peluang ≠ 0,5

Pendekatan linear dari S-Box:

Asumsi X adalah masukan dari S-box dan Y adalah keluaran dari S Box, maka Y=S(X).

Notasi yang digunakan :

Bit-bit dari X : X1X2X3X4X5X6 Bit-bit dari Y : Y1Y2Y3Y4

Bit-bit dari X’: X’1X’2X’3X’4X’5X’6

Bit-bit dari Y’: Y’1Y’2Y’3Y’4

X’ dan Y’ merepresentasikan sub himpunan dari bit X dan Y.



XX’ adalah perkalian skalar dari vektor X dan vektor X’ (Sebagai contoh, bit-bit X dipilih oleh X’). Demikian pula dengan YY’.

Dengan melihat susunan XX’ XOR YY’ untuk semua nilai yang mungkin, X, Y memiliki pilihan yang X’ dan Y’ yang tetap. Tabel perkiraan linear dapat diturunkan dengan menggunakan asumsi ini.

Sebagai contoh, kita akan menurunkan tabel perkiraan untuk S5.

* Pada baris pertama X’=0: untuk semua sub himpunan Y’=0, maka susunan dikatakan seimbang. Dan karenanya seluruh masukan adalah nol, dan untuk Y’=0, semua susunan adalah 0. Masukan akan memiliki nilai 32
* Pada baris sisanya, seluruh nilai adalah genap pada lingkup -20 sampai -14 (S- Box yang lain memiliki lingkup -20 sampai -16).

Masukan dengan nilai -20 berarti hanya 12 masukan untuk susunan XX’ XOR YY’ = 0, dan untuk 52 masukan XX’ XOR YY’=1. Masukan dengan nilai 0 berarti susunan seimbang.

Kriptoanalisis linear menggunakan masukan yang tidak berimbang.

Tabel Perkiraan Linear dari S5

½ + p menjadi peluang dari masukan acak blok P, dan enkripsi satu rondenya kita namakan C. Kita mengambil kunci K, di mana K memenuhi persamaan (P x P) XOR (C x T) XOR (K x K) = 0. P adalah himpuan bagian dari bit-bit data sebelum ronde dimulai. T adalah himpunan bagian dari bit-bit data yang didapat setelah ronde berakhir. K adalah himpunan bagian dari bit-bit dari kunci yang akan kita perkirakan.

Kita mendefinisikan karakteristik n-ronde sebagai berikut :

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Input |  |  |  |  |  |  | O | utput | Su | bse | t |  |  |  |  |  |
| Subset | 0x | 1x | 2x | 3x | 4x | 5x | 6x | 7x | 8x | 9x | Ax | Bx | Cx | Dx | Ex | Fx |
| 0x | 32 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1x | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2x | 0 | 4 | -2 | 2 | -2 | 2 | -4 | 0 | 4 | 0 | 2 | -2 | 2 | -2 | 0 | -4 |
| 3x | 0 | 0 | -2 | 6 | -2 | -2 | 4 | -4 | 0 | 0 | -2 | 6 | -2 | -2 | 4 | -4 |
| 4x | 0 | 2 | -2 | 0 | 0 | 2 | -2 | 0 | 0 | 2 | 2 | 4 | -4 | -2 | -2 | 0 |
| 5x | 0 | 2 | 2 | -4 | 0 | 10 | -6 | -4 | 0 | 2 | -10 | 0 | 4 | -2 | 2 | 4 |
| 6x | 0 | -2 | -4 | -6 | -2 | -4 | 2 | 0 | 0 | -2 | 0 | -2 | -6 | -8 | 2 | 0 |
| 7x | 0 | 2 | 0 | 2 | -2 | 8 | 6 | 0 | -4 | 6 | 0 | -6 | -2 | 0 | -6 | -4 |
| 8x | 0 | 0 | 2 | 6 | 0 | 0 | -2 | -6 | -2 | 2 | 4 | -12 | 2 | 6 | -4 | 4 |
| 9x | 0 | -4 | 6 | -2 | 0 | -4 | -6 | -6 | 6 | -2 | 0 | -4 | 2 | -6 | -8 | -4 |
| Ax | 0 | 4 | 0 | 0 | -2 | -6 | 2 | 2 | 2 | 2 | -2 | 2 | 4 | -4 | -4 | 0 |
| Bx | 0 | 4 | 4 | 4 | 6 | 2 | -2 | -2 | -2 | -2 | -2 | 2 | 0 | -8 | -4 | 0 |
| Cx | 0 | 2 | 0 | -2 | 0 | 2 | 4 | 10 | -2 | 4 | -2 | -8 | -2 | 8 | -6 | 0 |
| Dx | 0 | 6 | 0 | 2 | 0 | -2 | 4 | -10 | -2 | 0 | -2 | 4 | -2 | 8 | -6 | 0 |
| Ex | 0 | -2 | -2 | 0 | -2 | 4 | 0 | 2 | -2 | 0 | 4 | 2 | -4 | 6 | -2 | -4 |
| Fx | 0 | -2 | -2 | 8 | 6 | 4 | 0 | 2 | 2 | 4 | 8 | -2 | 8 | -6 | 2 | 0 |
| 10x | 0 | 2 | -2 | 0 | 0 | -2 | -6 | -8 | 0 | -2 | -2 | -4 | 0 | 2 | 10 | -20 |
| 11x | 0 | 2 | -2 | 0 | 4 | 2 | -2 | -4 | 4 | 2 | 2 | 0 | -8 | -6 | 2 | 4 |
| 12x | 0 | -2 | 0 | -2 | 2 | -4 | -2 | -8 | 4 | 6 | 4 | 6 | -2 | 4 | -6 | 0 |
| 13x | 0 | -6 | 0 | 2 | -2 | 4 | 2 | 0 | 4 | -6 | 4 | 2 | -6 | 4 | -2 | 0 |
| 14x | 0 | 4 | -4 | 0 | 0 | 0 | 0 | 0 | -4 | -4 | 4 | 4 | 0 | 4 | -4 | 0 |
| 15x | 0 | 4 | 0 | -4 | -4 | 4 | -8 | -8 | 0 | 0 | -4 | 4 | 8 | 4 | 0 | 4 |
| 16x | 0 | 0 | 6 | 6 | 2 | -2 | 4 | 0 | 4 | 0 | 6 | 2 | 2 | 2 | 0 | 0 |
| . |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| . |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| . |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3Fx | 0 | -4 | -4 | -4 | 14 | 6 | -6 | -2 | 2 | -2 | 6 | -2 | 0 | 0 | -4 | 0 |

Ω1 = (ΩP1, ΩT1, ΩK1, ½ + p1)

Karakteristik ini dapat digabungkan dengan karakteristik m-ronde :

Ω2 = (Ω 2, Ω 2, Ω 2, ½ + p )

P T K 2

Jika Ω 1 sama dengan nilai Ω 2 sebelum

T P

Tabel Perkiraan Linear – Probabilitas

Definisi : Peluang dari suatu masukan [X’, Y’] pada tabel perkiraan linear adalah bagian dari masukan X yang menghasilkan XX’ XOR YY’= 0.

Hasil : Peluang dari masukan [X’, Y’] adalah

½ ditambah nilai masukan, kemudian dibagi dengan jumlah masukan yang mungkin dari S- Box (64 pada DES).

Contoh : Peluang dari masukan dengan nilai 8 adalah ½ + 8/64 = ½ + 1/8, dan peluang dari masukan [10x, Fx] pada S5 adalah ½ - 20/64 = 12/64.

Karakteristik Linear

Pada kriptoanalisis linear, kita mempelajari sifat himpunan bagian dari bit-bit saat enkripsi. Kita berharap untuk menemukan informasi statistik dari kesamaan himpunan bagian bit-bit teks cipher dan bit-bit kunci dengan menggunakan kesamaan bit-bit himpunan bagian teks biasa.

Seperti pada kriptoanalisis *differential*, kita mendefinisikan karakteristik. Karakteristik linear tidak menggambarkan perbedaan, mereka menggambarkan himpunan bagian dari bit-bit yang digunakan untuk menganalisa kesamaan dalam bit-bit. Sebagai catatan, karakteristik linear menunjukan hubungan linear matsui dengan notasi yang berbeda.

Dengan mendefinisikan sebuah ronde adalah (P, T, K, ½ + p), di mana (P)L=(T)L=A, kita dapat menuliskan (P)R XOR (T)R = a.

mengalami proses penukaran (Setengan bagian kiri dari ΩP2 menjadi bagian kanan ΩP2, dan sebaliknya), Gabungan karakteristik Ω1 dan Ω2 (Jika Ω1 dan Ω2 dapat digabungkan) adalah karakteristik (n+m)-ronde:

Ω = (ΩP1, Ω 2, Ω 1  Ω 2, ½ + 2 x p x p )

T K K 1 2

Peluang ½ + 2 x p1 x p2 dapat terpenuhi sebab harapan nilai nol dapat dicapai jika kedua nilai adalah nol atau nilai awal keduanya adalah satu, dan

(½ + p1)(½ + p2) + (½ – p1)(½ – p2)

= ¼ + ½ p1 + ½ p2 + p1p2 + ¼ - ½ p1 – ½ p2

+ p1p2

= ½ + 2p1p2

Jika kita menggabungkan sebuah karakteristik (yang dapat digabungkan), peluang karakteristik gabugan tersebut adalah

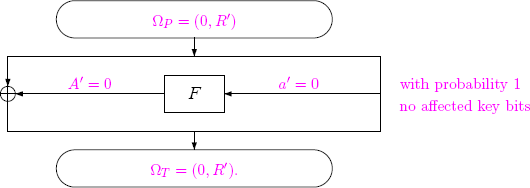
*l*

½ + p = ½ + 2l-1. *pi*

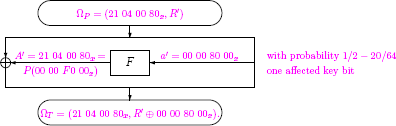
*i*1

Contoh karakteristik linear:

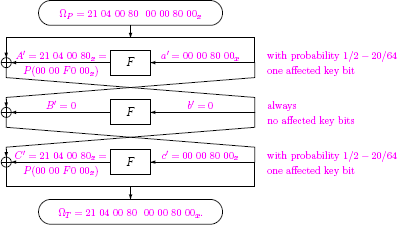
Karakteristik satu ronde terbaik adalah ronde yang tidak memiliki S-Box, memiliki probabilitas 1:



Karakteristik linear yang terbaik lainnya adalah yang menggunakan masukan -20 untuk S5:

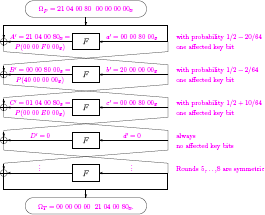


Karakteristik linear tiga ronde yang terbaik adalah karakteristik tiga ronde yang merupakan gabungan dari tiga buah karakteristik satu ronde yang telah diberikan pada contoh sebelumnya:



Karakteristik ini memiliki peluang

# ⎛  20 ⎞2



Delapan ronde karakteristik memiliki peluang sekitar ½ + 2-13.7. Jika diulang hingga 16 ronde, karakteristik ini akan memiliki peluang sebesar ½ + 2-26.4.

Matsui memiliki karakteristik 16 ronde dengan peluang sebesar 2-24. Hal ini dapat dilakukan dengan cara mengganti bilangan pertama dan terakhir dengan bilangan yang lebih baik.

Properti-Properti dari Karakteristik Linear:

Pada kriptoanalisis differential, ketika data digandakan saat enkripsi, nilai perbedaannya juga digandakan. Ketika suatu data dikenakan operasi XOR, nilai perbedaannya juga terkena operasi XOR.

Pada kriptoanalisis linear, peranan penggandaan dan XOR dapat dipertukarkan. Ketika sebuah data digandakan, Himpunan bagiannya dikenakan operasi XOR. Dan ketika kita melakukan operasi XOR terhadap suatu

data, himpunan bagiannya akan digandakan.

½ + 2 ⎜

# ⎝

⎟

# 64 ⎠

= ½ + 25/128

Fenomena ini terjadi karena himpunan bagian bukanlah nilai yang bebas. Mereka merupakan representasi dari bit-bit yang merupakan

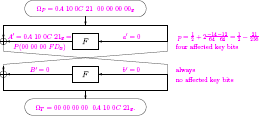
Karakteristik Matsui yang terbaik :

bagian dari kumpulan bit yang dianalisis. Persamaan sebuah himpunan bagian dari bit- bit, yang merupakan hasil dari operasi XOR dua nilai, dapat diketahui hanya jika kedua nilai dalam persamaan tersebut diketahui pada himpunan bagian yang sama.

Sebagai contoh, kita menggunakan persamaan Z = X XOR Y, dan X merupakan bagian dari persamaan. Kita dapat mengganti X dengan Y XOR Z. Jika X XOR Y ikut berperan, mereka dapat diganti dengan variabel Z.

Dalam penggandaan, kita harus menjaga nilai- nilai yang ada sehingga nilai-nilai tersebut tidak berubah. Misalnya jika kita memiliki X = Y = Z, maka kita dapat mengganti variabel X dengan Y atau Z. Atau jika lebih dari satu variabel di atas ikut dalam kumpulan nilai, kita dapat mendapatkan persamaan X XOR Y = 0, hal ini berarti kedua variabel dapat dihilangkan tanpa perlu melakukan substitusi.

Karakteristik pengulangan dua ronde:



Karakteristik ini menggunakan pengembangan dua bit menjadi S-Box S7 dan S8. Masukan untuk himpunan bagian a’= 0 – himpunan bagian yang kosong. Karakteristik ini mengasumsikan beberapa bit digandakan untuk mendapatkan bit-bit masukan yang merupakan himpuan bagian dari S-Box.

Karena mereka ditambahkan sebanyak dua kali

himpunan bagian kunci kemungkinan besar adalah satu.

Serupa dengan kriptoanalisis differential, penyerang dapat menggunakan karakteristik yang lebih pendek dan mencoba semua nilai bit-bit kunci yang mungkin dengan cara memasukan nilai-nilai tersebut ke dalam S- Box. Penyerang akan mendapatkan nilai bit-bit kunci yang merupakan himpunan bagian kunci dan nilai sebenarnya dari bit-bit kunci yang telah dicoba.

Diberikan sebuah ekspresi linear yang dihasilkan oleh karakteristik:

PΩP  CΩT = KΩK

Dengan probabilitas ½ + p, dengan beberapa nilai p ≠ 0.

Algoritma 1:

1. Diberikan N pasangan teks biasa – teks cipher. Untuk ukuran N yang cukup besar, didapatkan M yang merupakan jumlah teks biasa yang bagian kiri dari persamaanya sama dengan nol :

PΩP  CΩT = 0

1. Algoritma akan menebak nilai bit-bit kunci KK dengan cara sebagai berikut:

(M > N/2)  (1/2 + p > ½)

Contoh :

(Satu kali untuk setiap S-Box), mereka

membatalkan satu sama lain dengan kondisi a’=0. Kita akan mendapatkan perbedaan masukan yang kosong dengan perbedaan keluaran yang tidak kosong. Sifat ini sangat mirip dengan sifat pengulangan dari kriptoanalisis *differential*, namun dengan nilai nol dan tidak nol berbeda posisi.

Serangan Linear

Serangan linear menggunakan teks biasa yang sudah diketahui. Teks ini akan dihitung karakteristik linearnya.

Sebuah serangan terhadap cipher n-ronde dengan karakteristik n-ronde menghitung semua kemungkinan nilai dari persamaan bit- bit pada himpunan bagian teks biasa dan teks cipher. Nilai-nilai ini saling berkaitan dengan nilai lainnya yang terdapat dalam himpunan bagian bit-bit kunci (bit-bit kunci bernilai tetap untuk semua data). Jika potongan dari nilai nol sama dengan peluang dari karakteristik, Nilai dari himpunan bagian kunci diperkirakan sama dengan nol. Jika potongan nilai satu sama dengan peluang dari karakteristik, nilai

1/2 + p > 1/2 1/2 + p < 1/2 M > N/2 0 1

M < N/2 1 0

Karena jumlah N yang cukup besar, dan karena jarak peluang dari setengah |p| lebih besar, tingkat kesuksesan dari algoritma ini juga lebih besar.

Algoritma ini menemukan hanya satu bit nilai dari kunci untuk setiap karakteristik.

Algoritma 2:

Algoritma ini memiliki dua keunggulan dibandingkan algoritma sebelumnya:

1. Algoritma ini menggunakan karakteristik yang lebih pendek, dan karenanya algoritma ini memiliki |p| yang lebih besar.
2. Algoritma ini menemukan beberapa bit kunci dan nilai KΩK.

Berikut ini adalah penjelasan mengenai algoritma 2:

1. Kita mencoba semua kemungkinan nilai dari bit kunci yang memasuki S-Box aktif

pada ronde terakhir atau pertama. Untuk tiap percobaan:

* 1. Dekripsi ronde terakhir atau ronde pertama dengan menggunakan prediksi bit-bit kunci.
  2. Hitung untuk setiap percobaan nilai i. Jumlah i untuk pasangan teks biasa- teks cipher memenuhi persamaan: PΩP  C iΩT = 0.

Di mana P dan T adalah himpunan bagian yang telah diprediksi dengan menggunakan karakteristik yang lebih pendek. C’i adalah data yang digunakan sebelumnya pada ronde terakhir ketika nilai kunci i dicoba.

1. Algoritma mencoba nilai i yang menghasilkan nilai maksimum untuk [Mi

– N/2]. Algoritma kemudian akan menebak nilai dari bit-bit kunci yang dicobakan.

Algoritma menebak nilai bit-bit kunci KΩK adalah (Mi > N/2)  (½ + p > ½ )

* 1. Meet in the middle attack (kriptanalisis pada Double DES)

Meet in the middle attack dalam hal ini digunakan untuk serangan terhadap Double DES.

Meet in the middle attack adalah salah satu serangan terhadap kriptografi, seperti birthday attack, yang menggunakan space-time tradeoff. Ketika birthday attack ingin menemukan dua nilai dalam domain dari sebuah fungsi yang memetakan nilai yang sama dalam sebuah range, meet in the middle attack mencari nilai dalam setiap range dan domain dari komposisi dari dua fungsi sehingga forward mapping dari satu dari fungsi pertama sama dengan image inverse dari yang lainnya ke fungsi kedua.

Meet in the middle attack pertama kali diciptakan sebagai ekspansi dari block cipher oleh Diffie dan Hellman pada tahun 1977. Ketika ingin meningkatkan keamanan dari sebuah block cipher, salah seorang dari mereka memiliki ide untuk menggunakan dua kunci yang independen satu sama lainnya untuk mengenkripsi data dua kali. Salah seorang tersebut berpikir bahwa dengan naif berpikir bahwa cara ini akan meningkatkan keamanan skema enkripsi tersebut menjadi dua kali lipat. Tentu saja exhaustive search dari seluruh kombinasi yang mungkin akan memakan waktu 22n percobaan jika tiap kunci panjangnya n bit, dibandingkan dengan 2n

percobaan yang dibutuhkan untuk kunci tunggal.

Ternyata serangan pada metoda enkripsi tersebut jauh lebih mudah dibandingkan skema yang hanya menggunakan sekali enkripsi. Serangan tersebut bekerja dengan mengenkripsi dari salah satu ujung, dan dekripsinya dari ujung yang lainnya, sehingga bertemu di tengah-tengah ( meeting in the middle ).

Asumsi penyerang mengetahui sebuah pasangan plainteks dan cipherteks: P dan C, sehingga

 ,

Dimana K2 dan K1 adalah dua kunci yang berbeda.

Penyerang dapat menghitung *EK*(*P*) untuk seluruh kemungkinan kunci K dan menyimpan hasilnya di memory. Setelah itu, penyerang dapat menghitung *DK*(*C*) untuk setiap kunci K dan membandingkannya dengan tabel yang ada di memory tadi. Jika dia mendapatkan yang cocok, dapat dipastikan dia menemukan dua kuncinya dan juga dapat memverifikasinya dengan set kedua dari plainteks dan cipherteks. Jika ukuran kunci n, serangan tersebut hanya membutuhkan 2*n* + 1 enkripsi (dan *O*(2*n*) space). Kontras sekali dengan serangan biasa, yang membutuhkan 22*n* enkripsi (tetapi hanya *O*(1) space).

* 1. Kriptanalisis pada TDES

Kriptanalisis pada Triple DES dapat menggunakan key-search attacks dan exploit known (atau chosen) pairs of plainteks dan cipherteks. Pengukuran keberhasilan dan kompleksitas dari serangan kriptanalisis adalah sebagai berikut:

* + 1. Jumlah pasangan known plainteks- cipherteks.
    2. Storage space yang dibutuhkan untuk serangan.
    3. Jumlah single encription



* + 1. Jumlah keseluruhan operasi / step untuk serangan.

Cara yang paling ampuh untuk serangan Triple DES adalah dengan menggunakan MITM. Jika

pasangan plainteks dan cipherteks (p,c) diberikan, dapat dilakukan proses sebagai berikut:

1. Hitung semua nilai bN = D3 ( c), N Є

N

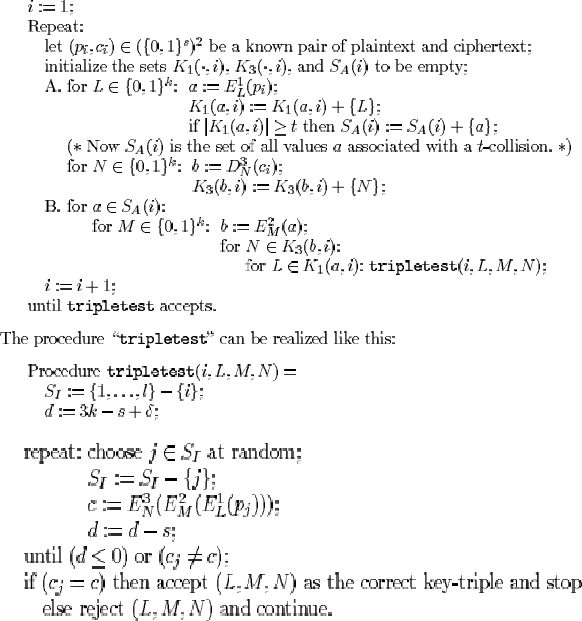
{0,1}k, dan simpan pasangan (bN,N) dalam tabel, dengan indeks bN.

1. Hitung semua nilai bL,M = E2 (E1L(p)) dengan L,M Є {0,1}k, dan cari (bL,M,N) di tabel pasangan (bN,N) yang telah dihitung sebelumnya.

M

1. Tes seluruh triple (L,M,N) dengan bL,M = bN sampai hanya tinggal satu dari triple yang ada.

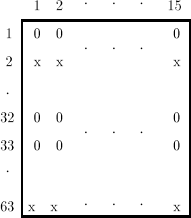
Berikut adalah code untuk kriptanalisis pada TDES:

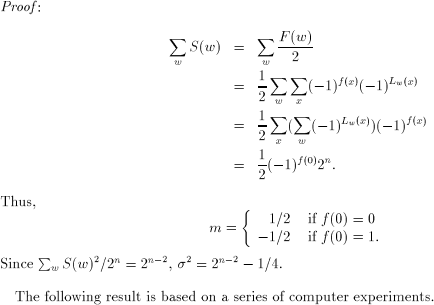


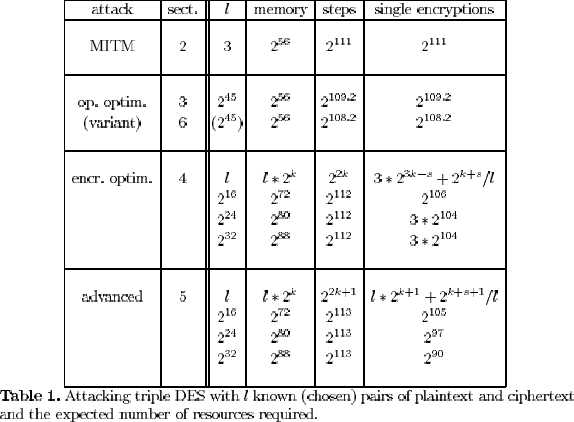
* 1. Keamanan DES dari kriptanalisis

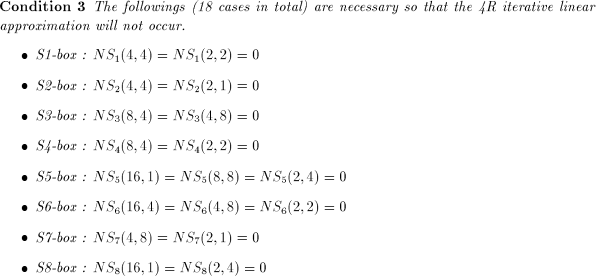
Menurut penelitian dari Kwang-jo Kim, dkk dari Etri, Korea, keamanan DES dapat ditingkatkan jika S-box pada DES dirancang ulang. Terdapat beberapa cara yang dibutuhkan untuk mempertahankan S-box DES dari serangan linear kriptanalisis. Beberapa cara tersebut adalah:

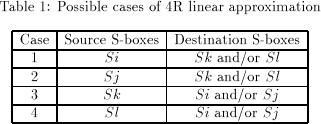
* + 1. uniformity of a linear distribution table





* + 1. iterative linear approximation (3R, 4R, 5R, nR)





Keamanan pada Triple DES

TDES dengan 3 kunci yang berbeda memiliki total panjang kunci secara keseluruhan adalah 168 bit: tiga 56-bit kunci DES (dengan parity bit TDES memiliki total penyimpanan 192 bit), tetapi karena adanya metode meet in the middle attack, keamanan yang efektif ternyata hanya membutuhkan 112 bit. Variasi lainnya, Double DES, menggunakan k1=k3, yang mengurangi ukuran kunci dari 112 bit dan panjang penyimpanan menjadi 128 bit. Walau bagaimanapun, mode ini rentan terhadap chosen-plainteks atau known-plainteks attacks walaupun telah didesain untuk hanya memiliki 80 bit security.

Pada tahun 2005, serangan yang terbaik pada TDES membutuhkan 232 known plaintexts, 2113 steps, 290 single DES encryptions, dan 288 memory (tetapi ada tradeoff antara waktu dan memory). Tentu saja hal ini tidak praktis. Jika penyerang mencari satu dari banyak kunci kriptografi, ada sebuah serangan yang mengefisienkan penggunaan memory yang akan mendapatkan satu dari 228 kunci, jika diberikan chosen plaintexts per key dan 284 operasi enkripsi yang berguna. Serangan ini dapat diparalelisasikan jauh dari praktek, dan dibutuhkan milyaran dolar budget dan tahun untuk mencoba serangan tersebut, dan juga kegunaannya terbatas.

CODINGAN TRIPLE DES DENGAN PHP

Namam file nya controller.php

<?php

/\*

inspirasi itu datang ketika kepepet .... hahahaha

\*/

require\_once 'kriptoopenssl.php';

require\_once'kripto.php';

if (!empty($\_POST["hash"])) {

$hash = $\_POST["hash"];

echo "kunci = <b>{$hash}</b>";

$kripto = new kripto($hash);

$kriptoopenssl = new kriptoopenssl($hash);

if (!empty($\_POST["data"])) {

$data = $\_POST["data"];

echo "<br/>plaintext = <b>{$data}</b>";

echo "<br/>pilihan= <b>{$\_POST["option"]}</b><br/>";

if ($\_POST["option"] == "encrypt") {

//menggunakan mcrypt

$start\_time = microtime(true);

$result = $kripto->Encrypt($data);

$stop\_time = microtime(true);

$time\_mcrypt = $stop\_time - $start\_time;

//menggunakan openssl

$start\_time = microtime(true);

$result = $kriptoopenssl->Encrypt($data);

$stop\_time = microtime(true);

$time\_openssl = $stop\_time - $start\_time;

CheckResult($result);

echo "<br/>hasil enkripsi= <b> {$result} </b>";

ShotTime($time\_mcrypt,$time\_openssl);

}

if ($\_POST["option"] == "decrypt") {

//menggunakan mcrypt

$start\_time = microtime(true);

$result = $kripto->Decrypt($data);

$stop\_time = microtime(true);

$time\_mcrypt = $stop\_time - $start\_time;

//menggunakan openssl

$result = $kriptoopenssl->Decrypt($data);

$stop\_time = microtime(true);

$time\_openssl = $stop\_time - $start\_time;

CheckResult($result);

$start\_time = microtime(true);

echo "<br/>hasil dekrip = <b> {$result} </b>";

ShotTime($time\_mcrypt,$time\_openssl);

}

} else {

throw new Exception("{{ plain text kosong }}");

}

} else {

throw new Exception("{{ kunci kosong bro }}");

}

function CheckResult($result) {

if ($result == "" || $result == false) {

throw new Exception("{{ plaintext tidak valid}}");

}

}

function ShotTime($time\_mcrypt, $time\_openssl)

{

echo "<br/>Time (mcrypt) :{$time\_mcrypt} ms";

echo "<br/>Time (openssl) :{$time\_openssl} ms";

}

Lanjutan

Dengan Nama file kripto.php

<?php

/\*\*

inspirasi itu datang ketika kepepet .... hahahaha

\*/

class kripto{

private $hash;

function \_\_construct($hash){

$key = md5($hash,TRUE);

$key .= substr($key,0,8);

$this->hash = $key;

}

private function pkcs5\_pad($data){

//Pad for PKCS5

$blockSize = mcrypt\_get\_block\_size('tripledes', 'ecb');

$len = strlen($data);

$pad = $blockSize - ($len % $blockSize);

$data .= str\_repeat(chr($pad), $pad);

return $data;

}

private function pkcs5\_unpad($data){

//Unpad for PKCS5

$pad = ord($data{strlen($data)-1});

if ($pad > strlen($data)) return false;

if (strspn($data, chr($pad), strlen($data) - $pad) != $pad) return false;

return substr($data, 0, -1 \* $pad);

}

/\*\*

\* @param $data

\* @return string

\*/

public function Encrypt($data){

$data = $this->pkcs5\_pad($data);

$encData = mcrypt\_encrypt('tripledes', $this->hash , $data, 'ecb');

return base64\_encode($encData);

}

/\*\*

\* @param $data

\* @return bool|string

\*/

public function Decrypt($data){

$data = base64\_decode($data);

$data = mcrypt\_decrypt('tripledes', $this->hash, $data, 'ecb');

return $this->pkcs5\_unpad($data);

}

}

Lanjutan

Dengan Nama file kriptoopenssl.php

<?php

/\*\*

inspirasi itu datang ketika kepepet .... hahahaha

\*/

class kriptoopenssl{

private $hash;

function \_\_construct($hash){

$key = md5($hash,TRUE);

$key .= substr($key,0,8);

$this->hash = $key;

}

/\*\*

\* @param $data

\* @return string

\*/

public function Encrypt($data){

$encData = openssl\_encrypt($data, 'DES-EDE3', $this->hash, OPENSSL\_RAW\_DATA);

$encData = base64\_encode($encData);

return $encData;

}

/\*\*

\* @param $data

\* @return string

\*/

public function Decrypt($data){

$data = base64\_decode($data);

$decData = openssl\_decrypt($data, 'DES-EDE3', $this->hash, OPENSSL\_RAW\_DATA);

return $decData;

}

}

**DAFTAR PUSAKA**

1. Kurniawan, Yusuf. Kriptografi Keamanan Internet dan Jaringan Komunikasi. Informatika Bandung. 2004.
2. M. Matsui, “Linear Cryptanalisis Method for DES Cipher”, Eurocrypt’93 Extended Abstracts, 1993.
3. Kim. Kwang-jo, dkk. “DES can be Immune to Linear Cryptanalisis”. Etri, Korea. 1993.