Algoritma Block Cipher ASRCI

Ade Surya Ramadhani, 13514049, Teknik Informatika/Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung, Bandung, Indonesia, adesurya559@gmail.com

Cendhika Imantoro, 13514037, Teknik Informatika/Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung, Bandung, Indonesia, 13514037@std.stei.itb.ac.id

***Abstract* — Algoritma Block Cipher umumnya adalah agoritma kriptografi dengan membagi plainteks blok-blok bit dengan panjang yang sama. Enkripsi pada block cipher dilakukan terhadap block bit plainteks menggunakan bit - bit kunci secara simetris. Block cipher dapat menggunakan beberapa mode dalam operasinya. Namun dalam implementasinya block cipher dapat dimodifikasi dengan berbagai operasi dasar seperti subtitusi, transposisi dan subtitusi sehingga algoritma ini mengikuti prinsip diffusion dan confussion yang menjadikan algoritma kriptografi ini terjamin kuat.**

***Keywords* — block,bit,feistel,subtitusi,transposisi,shannon**

I. Pendahuluan

Seiring dengan perkembangan teknologi jaringan, layanan yang memanfaatkan teknologi jaringan sebagai media komunikasi pun semakin banyak. Di antara layanan-layanan tersebut, ada beberapa yang membutuhkan pengiriman informasi sensitif seperti password. Ketika sebuah informasi sedang bergerak dalam jaringan, beberapa jenis serangan dapat dilakukan terhadap informasi tersebut oleh pihak ketiga. Salah satu bentuk serangan yang paling sederhana adalah penyadapan. Penyadap yang berhasil menangkap pesan dalam jaringan dapat memanfaatkan informasi sensitif dalam pesan tersebut untuk memperoleh keuntungan lebih. Teknologi kriptografi merupakan salah satu cara untuk menghindari kerugian dari serangan ini. Pesan yang dienkripsi dengan teknologi kriptografi, isinya tidak akan bisa dibaca meskipun berhasil ditangkap oleh penyadap.

Pada domain kriptografi modern, dari segi satuan yang dieenkripsi, ada istilah bernama *Block Chiper*. *Block Chiper* melakukan enkripsi pesan pada satuan blok yang terdiri dari beberapa *byte*. *Block Chiper* memiliki berbagai mode operasi yang memungkinkan untuk menangani berbagai jenis pesan.

II. Dasar Teori

A. Block Cipher

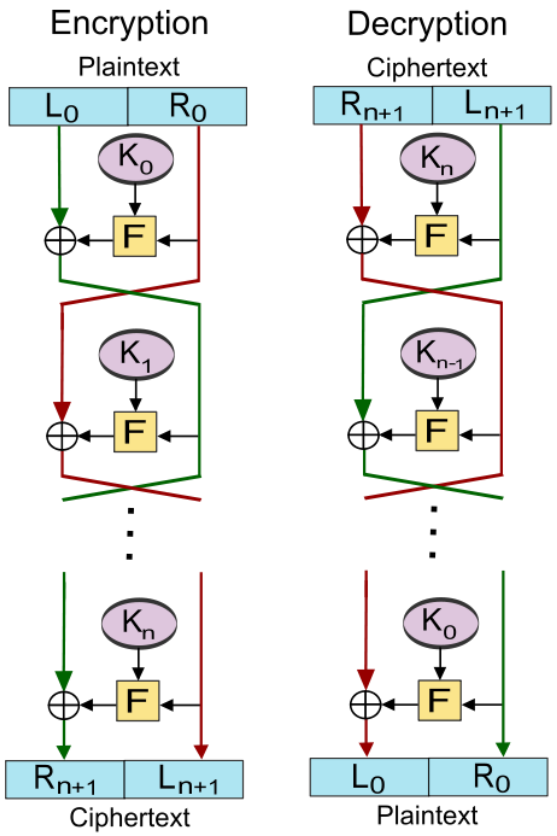
Block Cipher merupakan metode enkripsi plainteks menjadi cipher teks dengan cara membagi plainteks yang akan di enkripsi menjadi sekumpulan blok bit dengan panjang sama. Hasil enkripsi berupa blok cipher teks dengan panjang yang sama pula. Biasanya metode ini dilakukan secara simetris. Kemudian ada beberapa mode operasi yang bisa dilakukan oleh block cipher. Pada mode Electronic Code Book (ECB), setiap block dienkripsi langsung secara indepeden. Pada mode Chiper Block Chaining (CBC), sebelum plainteks dienkripsi, dilakukan operasi XOR terlebih dahulu dengan chiperteks blok sebelumnya. Pada Chiper Feedback (CFB), Yang dienkripsi adalah sebuah antrian, lalu operasi XOR dilakukan antara hasil enkripsi antrian dengan plainteks. Mode Output Feedback (OFB) mirip dengan CFB. Perbedaannya adalah perpanjangan antrian CFB diperoleh dari chiperteks akhir hasil operasi XOR dengan plainteks, sedangkan perpanjangan antrian OFB diperoleh dari hasil enkripsi antrian sebelumnya. Mode terakhir adalah mode Counter. Mode ini menggunakan sebuah counter sebagai elemen tambahan enkripsi. Counter yang digunakan selalu diincrement tiap selesai malakukan enkripsi.

B. Prinsip Diffusion dan Confussion (Shannon)

Ada dua prinsip algoritma kriptografi yang biasa digunakan sebagai acuan untuk mendesain algoritma kriptografi yang baik. Prinsip yang dimaksud adalah prinsip *confusion* dan *diffusion*. Kedu prinsip ini dikemukakan oleh Claude Shannon pada tahun 1945. Pada prinsip *confusion*, hubungan antara *key* dan *chipertext* perlu disamarkan dengan membuat setiap bit dari *chipertext* dependen ke beberapa bagian *key*. Prinsip *diffusion* menyatakan bahwa tiap perubahan bit pada *plaintext* harus menghasilkan perubahan terhadap setidaknya setengah dari *chipertext* dan sebaliknya. Implementasi kedua prinsip ini pada algoritma kriptografi akan menyebabkan kegiatan kriptoanalisis lebih sulit sehingga informasi lebih aman.

C. Struktur Feistel

Agar algoritma kriptografi kuat, transformasi pada pesan sebaiknya dilakukan berulang-kali. Struktur Feistel merupakan salah satu struktur algoritma enkripsi yang melakukan langkah transformasi pesan berulang-ulang.



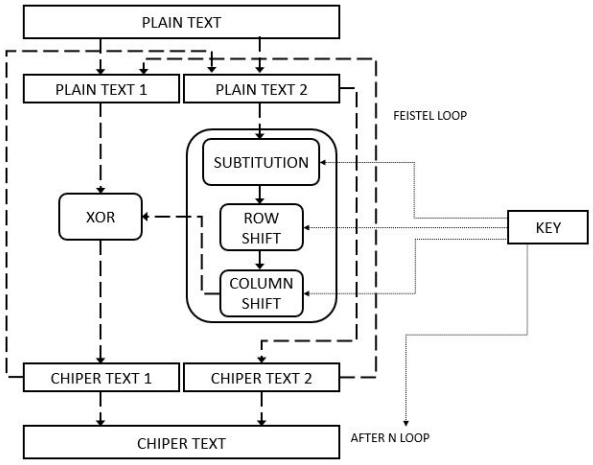
https://en.wikipedia.org/wiki/Feistel_cipher#/media/File:Feistel_cipher_diagram_en.svg

Pada struktur Feistel standar, yang pertama dilakukan adalah pembangunan *round key* sebanyak jumlah pengulangan (*round*) yang akan dilakukan. *Round key* dibangun dari key utama yang digunakan untuk enkripsi. Tiap *round* nantinya akan menggunakan *round key* yang berbeda. Untuk mengenkripsi pesan, pada tiap *round*, input dipecah menjadi dua. Transformasi yang dilakukan terhadap bagian pertama adalah operasi XOR dengan output dari *round function*. Input dari *round function* adalah *round key* dan bagian pesan kedua. Pada bagian kedua tidak dilakukan transformasi apapun, namun tiap antar *round*, posisi bagian pertama dan kedua ditukar, sehingga bagian ini akan ditransformasi pada *round* berikutnya.

Kelebihan utama dari struktur Feistel adalah Untuk melakukan dekripsi, langkah yang diperlukan sama dengan enkripsi. Yang membedakan adalah pada proses dekripsi, urutan *round key* yang digunakan dibalik. Selain itu, pada struktur Feistel, *round function* yang digunakan tidak harus memiliki invers.

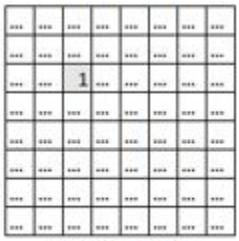
III. Rancangan Block cipher

Pada algoritma yang kami rancang, secara umum struktur yang kami gunakan adalah struktur Feistel. Yang kami definisikan pada algoritma kami adalah *round function* yang digunakan dan jumlah *round* yang dilakukan. Dengan adanya pendefinisian pada dua aspek tersebut, kami juga perlu mendefinisikan algoritma untuk membangun *round key* dari *key* utama.



A. *Round Function*

Round Function yang kami buat menerima input pesan berukuran 64-bit dan *round key* berupa matriks 8x8 dengan isi tiap sel matriks memiliki nilai pada *range* 0-63 (memungkinkan duplikasi nilai). Round Function memiliki dua tahap transformasi, yaitu tahap subtitusi dan tahap transposisi.



Pada tahap subtitusi, pesan diproses sebagai matriks bit 8x8. Tiap sel pesan memiliki index (mulai dari 0 hingga 63). Untuk menentukan nilai pengganti tiap bit, dilakukan pencarian elemen pada *round key* yang posisinya bersesuaian dengan bit pada pesan. Lalu, diambil empat elemen *round key* yang bertetangga. Berikutnya, diambil empat bit dari pesan yang indexnya sama dengan empat elemen Round key terpilih. Keempat nilai bit pesan ini dijumlahkan lalu dimodulo 2. Hasilnya menjadi nilai subtitusi.

ROUND KEY

15	17	7	46	34	2	43	23
35	63	43	53	9	54	0	36
36	0	46	39	7	50	44	39
50	11	1	49	43	20	32	20
22	63	37	32	27	14	28	45
15	48	5	34	18	11	13	44
5	31	4	31	6	56	6	28
14	32	22	58	12	36	61	27

SUM OF EACH ROW
187
293
261
226
268
188
167
262

ROW SHIFT (MOD 8)
3
5
5
2
4
4
7
6

1	2	1	3	1	2	2	1
9	6	6	4	5	4	2	6
2	5	5	2	6	3	7	2

SUM OF EACH COLUMN

0	1	5	6	4	3	3	2

COLUMN SHIFT (MOD 8)

Pada tahap transposisi, pertama dilakukan shift terhadap tiap baris pesan hasil subtitusi. Besar shift yang dilakukan pada tiap baris sesuai dengan jumlah nilai *round key* pada baris yang sama. Setelah shift baris, dilakukan shift pada kolom. Besar shift yang dilakukan pada tiap kolom mirip dengan shift pada baris, yaitu sebesar jumlah nilai pada kolom *round key* bersesuaian.

B. *Jumlah Round*

Jumlah *round* yang digunakan untuk proses enkripsi dan dekripsi ditentukan berdasarkan *key* yang digunakan. Karakter dari key semuanya dijumlahkan, lalu nilai penjumlahannya dimodulo 8. Hasil modulo lalu ditambah 8. Hasil penjumlahan ini menjadi jumlah *round* yang dilakukan.

KEY DALAM BIT	10100010 10101101 10110101 11010110 10100100 ...
SUM	...10110101
SUM % 8	101 (5 DALAM DESIMAL)
JUMLAH ROUND	8 + 5 = 13

C. *Round Key*

Untuk membangun *round key*, pertama dihitung terlebih dahulu banyak nilai yang diperlukan secara keseluruhan. Perhitungan dilakukan berdasarkan banyak *round.* Karena tiap *round key* berukuran 8x8, maka banyak nilai yang diperlukan adalah 8 x 8 x banyak *round*. Lalu, dilakukan pembangunan integer dengan kuantitas sesuai yang dibutuhkan. Integer yang dibangun harus memiliki nilai pada range 0-63. Integer ke-n dibangun dengan menjumlahkan karakter *key* dari karakter pertama hingga karakter ke-n, lalu dimodulo dengan 64. Jika semua karakter *key* sudah digunakan, karakter selanjutnya yang dijumlahkan kembali ke karakter pertama. Setelah semua integer berhasil dibangun, integer dikelompokkan ke dalam kelompok berisi 64 integer. Tiap kelompok lalu diubah ke matriks *round key* 8x8.

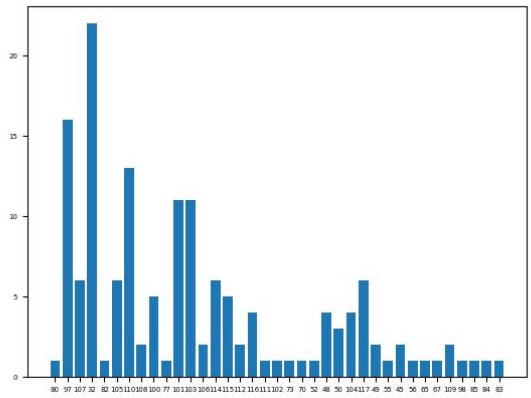
IV. Hasil Percobaan

Kami telah melakukan percobaan di program kripto ASRCI yang telah kami buat dalam bahasa python. Untuk pengujian dari tiap mode eksekusi, parameter pengujian yang digunakan adalah sebagai berikut.

Plainteks : **"**Pak Rinaldi Mengajar kelas kriptografi IF4020 tahun 2017 - 2018 Ade dan Cendhika mengerjakan tugas besar pengganti UTS mereka dengan sungguh - sungguh**"**

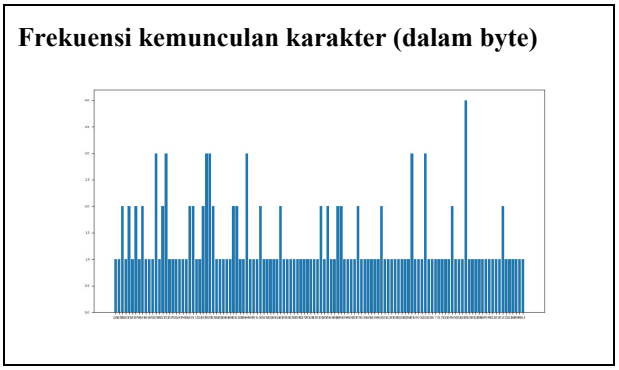
Keyword : **"**Tugas Pengganti UTS Kriptografi**"**

Frekuensi kemunculan karakter (dalam byte)

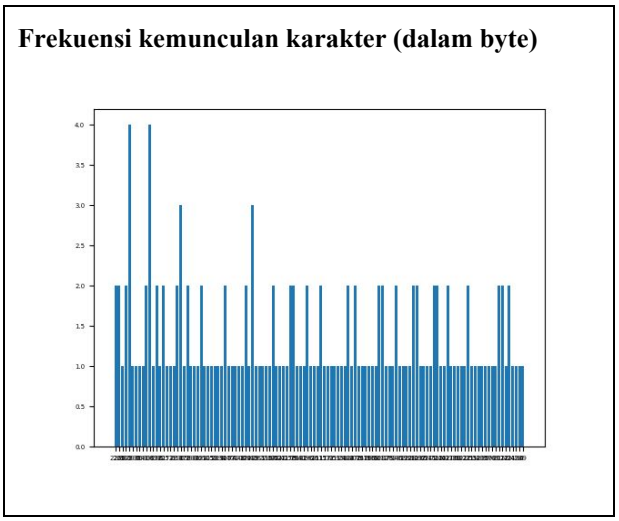


Pada pengujian menggunakan parameter di atas, hasil pengujian divisualisasikan ke dalam bentuk diagram frekuensi kemunculan tiap byte.

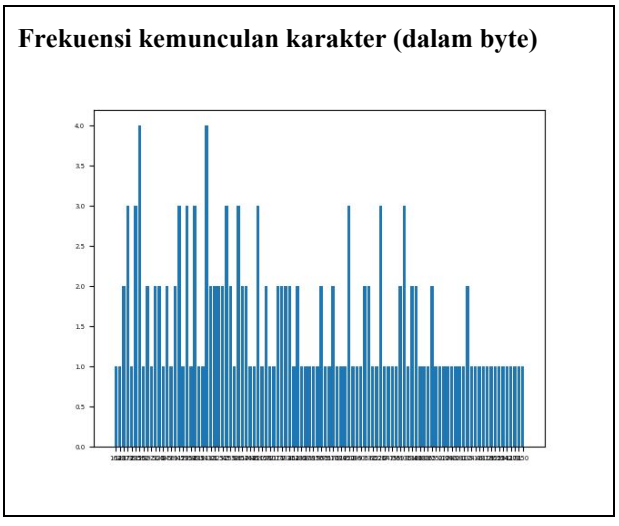
A. Mode Electronic Code Book (ECB)



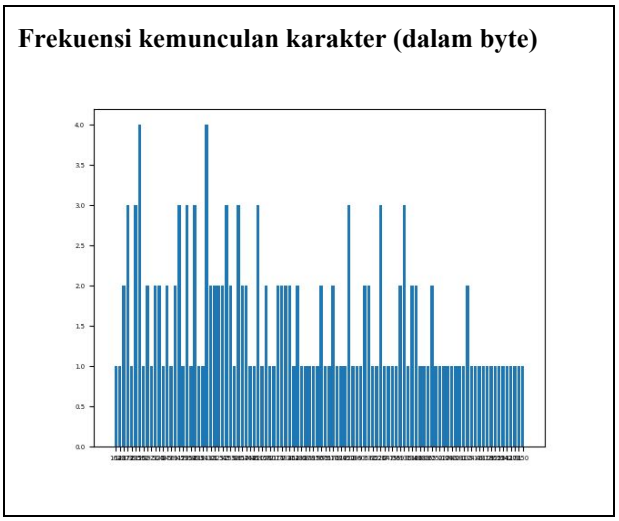
B. Cipher Block Chaining (CBC)



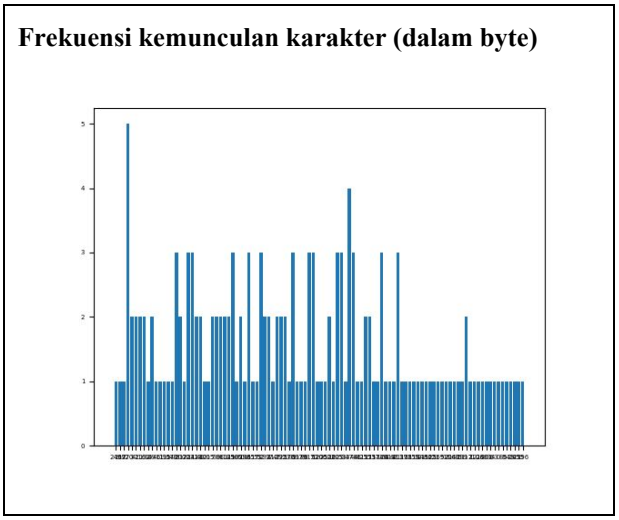
C. Cipher Feedback (CFB)



D. Output Feedback (OFB)

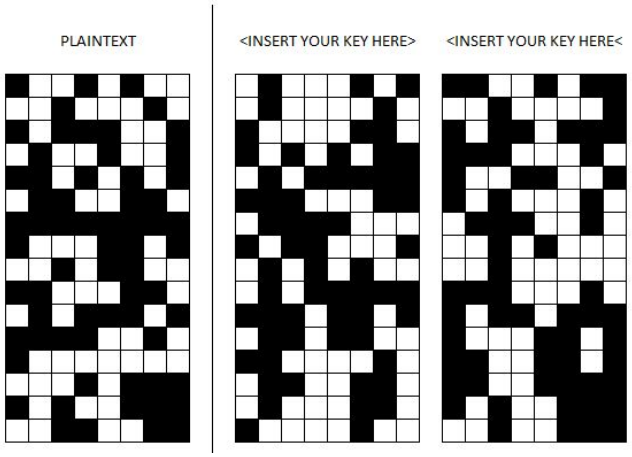


E. Mode Counter



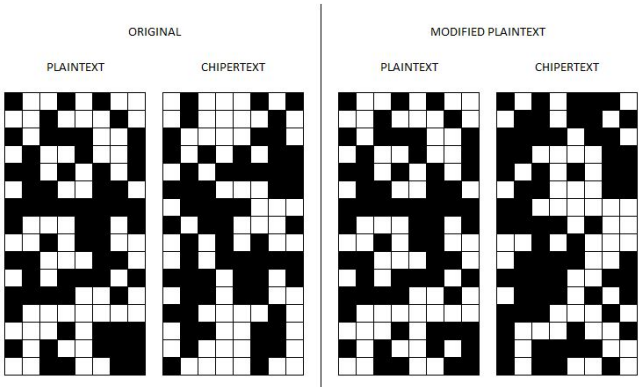
F. Dampak Perubahan Bit Kunci

Pengujuan terhadap aspek ini dilakukan dengan mencoba mengenkripsi dengan key “<INSERT YOUR KEY HERE>”. Setelah itu, dilakukan percobaan enkripsi dengan menggunakan key yang mirip namun karakter terakhirnya berbeda, yaitu “<INSERT YOUR KEY HERE<”. Perubahan bit pada chipertext yang terjadi akibat perubahan key cukup banyak dan acak seperti pada gambar berikut.



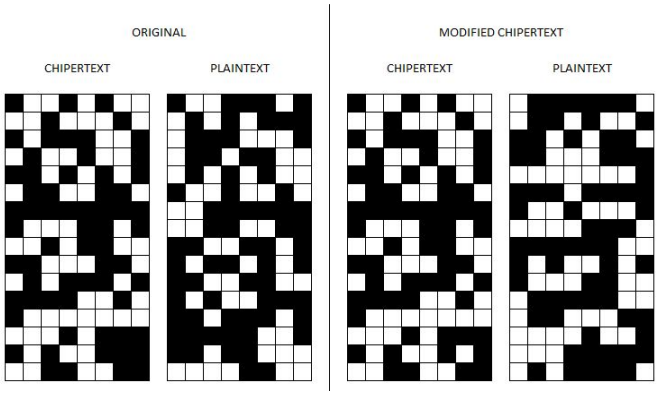
G. *Dampak Perubahan Bit Plainteks*

Pengujian terhadap aspek ini dilakukan terhadap plainteks 16 Byte yang cukup acak. Bit ke-7 dari byte plaintext ke-15 diubah dari angka 0 menjadi angka 1. Hasil yang diperoleh menunjukkan perubahan chiperteks yang cukup banyak dan tersebar seperti pada gambar.



H. *Dampak Perubahan Bit Cipherteks*

Chipertext yang digunakan pada pengujian ini adalah plainteks yang digunakan di pengujian sebelumnya. Kali ini, operasi yang dilakukan terhadap rangkaian byte input adalah operasi dekripsi. Hasil pengujian menunjukkan perubahan bit hasil dekripsi yang disebabkan perubahan chiperteks cukup banyak dan polanya cukup acak seperti pada gambar berikut.



V. Analisis Keamanan

Berdasarkan percobaan di bagian sebelumnya, lima mode eksekusi Block Chiper menghasilkan distribusi byte yang frekuesi tiap byte-nya menurut penilaian kami cukup merata. Kami menilai demikian karena meskipun frekuensi kemunculan byte pada plainteks bisa lebih dari 20, frekuensi dari tiap byte pada chiperteks tidak lebih dari 4 kemunculan. Distribusi yang cukup merata ini menyebabkan serangan berbasis analisis statistik sulit dilakukan pada chiperteks.

Berdasarkan tiga percobaan berikutnya, tampak bahwa perubahan satu bit plainteks menghasilkan perubahan pada chiperteks yang cukup banyak dan acak. Begitu juga sebaliknya, jika chiperteks diubah satu bit, plainteks mengalami perubahan yang tampak bersifat acak. Perubahan satu bit pada key juga menyebabkan chiperteks berubah cukup signifikan dan tidak berpola. Hal ini menunjukkan bahwa prinsip difusi dan konfusi Shannon telah terimplementasi pada algoritma. Dengan terimplementasinya prinsip ini, serangan berbasis analisis pola akan sulit dilakukan pada algoritma.

Serangan yang paling umum dilakukan pada algoritma kriptografi adalah *brute force attack*. Pada algoritma kami, serangan jenis ini mungkin dilakukan pada dua titik, yaitu key utama dan *round key*. Jika serangan dilakukan pada key utama, jika key yang digunakan sepanjang 16 byte, maka banyak kemungkinan key yang ada adalah sebanyak 2^{128}. Dengan kecepatan komputasi serangan sebesar 10^6 key per detik, waktu yang dibutuhkan untuk mencoba semua key sekitar 10^{25} tahun. Perhitungan tersebut didasarkan pada asumsi panjang key 16 byte. Karena panjang key pada algoritma ini bisa lebih panjang lagi, sebenarnya *domain* dari key memiliki tak hingga kemungkinan. Dari hal tersebut, disimpulkan key utama cukup aman dari serangan *brute force*.

Selain pada key utama, serangan *brute force* juga dapat dilakukan pada *round key*. Domain dari *round key*terbatas karena jumlah round dibatasi antara 8 hingga 15. Ukuran *round key* pada tiap round sebesar 128 bit. Itu berarti ada 2^{128} kemungkinan *round key* per round. Karena jumlah round minimal 8 dan maksimal 15, maka banyak kemungkinan pasangan *round key* antara 2^{1024} hingga 2^{1920}. Jumlah tersebut jauh lebih banyak daripada jumlah kemungkinan key utama 16-bit, sehingga kami menilai melakukan *brute force* pada semua kemungkinan *round key* lebih infeasible meskipun domainnya lebih terbatas.

VI. Kesimpulan dan saran

Algoritma yang kami rancang masih memiliki banyak kekurangan terutama pada proses memunculkan *round* kunci. Kemudian untuk menyembunyikan pesan algoritma ini cukup baik dibuktikan dengan grafik kemunculan karakter yang didapat setelah proses enkripsi didapat grafik yang acak.

Algoritma ini juga menerapkan prinsip difusi dan konfusi sehingga hubungan antar kunci dan teks terjaga membuat semakit kuat.

Saran untuk penggunaan algoritma ini agar optimal adalah panjang kunci minimal 16 byte dan meminimalisir perulangan karakter. Kemudian algoritma ini dapat diperbaiki dalam proses pembangunan *round key* dengan mendesign pembangunan yang lebih kompleks. Agar dapat dipastikan algoritma ini dapat digunakan secara luas maka diperlukan analsis keamanan yang lebih lanjut lagi.

VII. References

[1] http://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2017-2018/Algoritma%20Kriptografi%20Klasik_bag2%20(2018).pdf

[2] *Menezes, Alfred J.; Oorschot, Paul C. van; Vanstone, Scott A. (2001). Handbook of Applied Cryptography (Fifth ed.)*

[3] *Katz, Jonathan; Lindell, Yehuda (2008). Introduction to modern cryptography. CRC Press. ISBN 9781584885511*

VIII. PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.