Cipher Blok MalamJumat

Irfan Sofyana Putra[1], **Ahmad Rizal Alifio**[2].

[1,2] Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika (STEI), Institut Teknologi Bandung (ITB), Jalan Ganesha 10, Bandung 40132
E-mail: 13517078@std.stei.itb.ac.id, 13517076@std.stei.itb.ac.id

Abstrak. Makalah ini berisi sebuah proposal cipher blok baru bernama MalamJumat. Algoritma tersebut merupakan hasil modifikasi dari algoritma cipher blok *Data Encryption Standard* (DES). Algoritma ini mengurangi kelemahan yang ada pada DES dengan cara menambahkan kunci baru sebagai *seed* yang digunakan dalam membangkitkan tabel-tabel permutasi serta meningkatkan kompleksitas fungsi internal jaringan feistel pada DES, tanpa meningkatkan kompleksitas perhitungan enkripsi sehingga masih dapat digunakan dengan baik pada perangkat berkemampuan rendah. **Kata Kunci**: cipher blok, DES, Feistel, *seed*, kompleks,enkripsi..

1. Pendahuluan

Dewasa ini, semakin banyak data yang bersifat rahasia harus dikirimkan untuk kebutuhan sehari-hari. Sebagai pengguna internet, tidak mengherankan apabila setiap harinya kita mengirimkan data-data rahasia seperti akun, *password*, kontak, dan sebagainya. Maka dari itu, untuk melindungi data-data rahasia yang ada dibutuhkan pengaplikasian kriptografi. Meskipun banyak algoritma kriptografi yang kuat sudah ada saat ini, algoritma tersebut tidak berarti tak bisa dipecahkan. Sehingga dibutuhkan inovasi baru dalam algoritma kriptografi ataupun modifikasi pada algoritma yang sudah ada. Makalah ini menjelaskan sebuah algoritma cipher blok hasil modifikasi dari *Data Encryption Standard*.

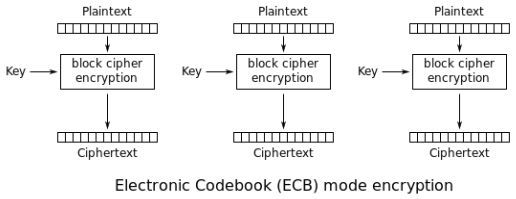
2. Dasar Teori

2.1. Cipher Blok

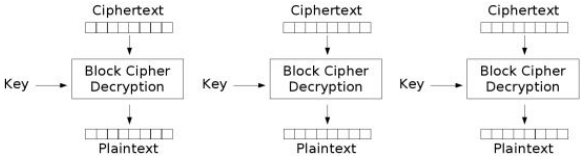
Salah satu cara implementasi kriptografi modern adalah menggunakan cipher blok. Cipher blok bekerja dengan mengelompokkan bit-bit *plainteks* menjadi sebuah ukuran tertentu, misalnya 64 atau 128 bit. Algoritma kriptografi kemudian diaplikasikan terhadap blok-blok bit tersebut menggunakan kunci enkripsi. Panjang kunci yang digunakan akan sama dengan panjang blok, yang kemudian menghasilkan blok-blok *ciphertext* yang dapat disatukan menjadi sebuah *ciphertext* utuh.

2.1.1. *Electronic Code Book*

Electronic Code Book adalah metode paling sederhana dalam cipher blok. Pada metode ini, setiap blok plainteks akan dilakukan enkripsi atau dekripsi dengan kunci yang ada. Kelemahan dari metode ini adalah tidak adanya keterkaitan antar blok, sehingga penyerang cukup memecahkan kunci yang digunakan untuk mengetahui seluruh pesan.



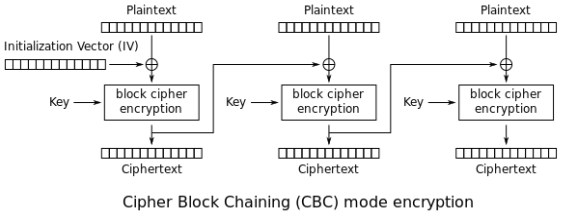
Gambar 1. *Electronic Code Book* mode enkripsi



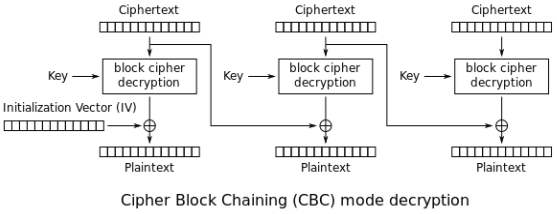
Gambar 2. *Electronic Code Book* mode dekripsi

2.1.2. *Cipher Block Chaining*

Cipher block chaining adalah sebuah metode enkripsi-dekripsi dalam cipher blok dimana hasil ciphertext dari blok sebelumnya digunakan dalam proses enkripsi-dekripsi selanjutnya, sebagai pasangan dalam operasi XOR untuk plainteks yang diterima. Pengecualian dalam proses ini yaitu pada enkripsi-dekripsi blok pertama, dimana plainteks harus menggunakan vektor awal (*initialization vector*) sebagai pengganti input cipherteks dari tahap sebelumnya. Perlu diperhatikan dalam metode ini, error yang terjadi akan terjadi berantai karena ciphertext akan digunakan dalam proses enkripsi-dekripsi selanjutnya.



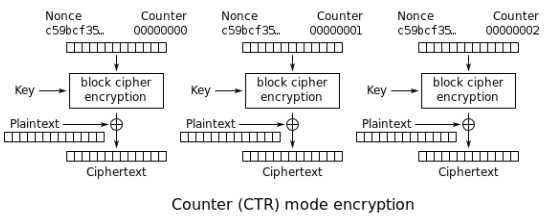
Gambar 3. *Cipher Block Chaining* mode enkripsi



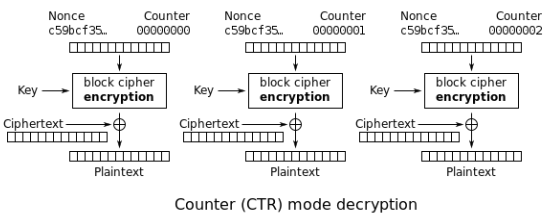
Gambar 4. *Cipher Block Chaining* mode dekripsi

2.1.3. *Counter Mode*

Mode *counter* adalah sebuah metode cipher blok yang tidak membentuk rantai. Serupa seperti *electronic code book*, mode *counter* tidak memiliki keterkaitan antar blok. Perbedaannya terletak pada bagian yang dienkripsi. pada *electronic code book*, plainteks langsung dilakukan enkripsi. Pada *counter mode*, bagian yang dilakukan enkripsi adalah counternya, yang baru kemudian dilakukan XOR terhadap *plainteks* yang dimasukkan.



Gambar 5. *Counter* mode enkripsi



Gambar 6. *Counter* mode dekripsi

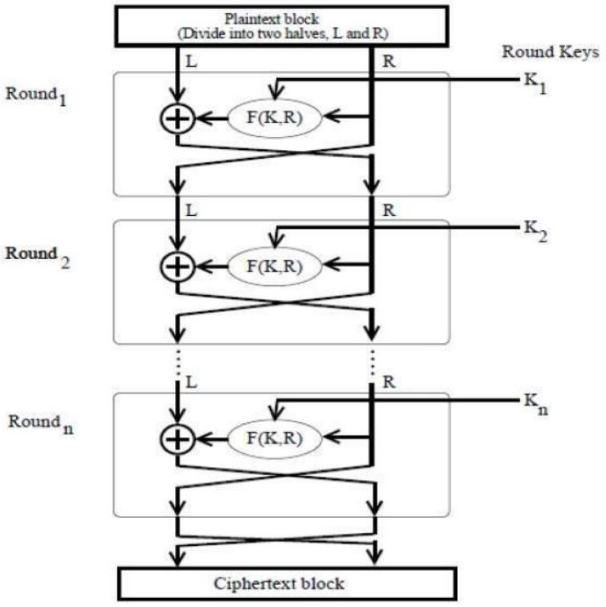
2.2. Prinsip *confusion* dan *diffusion*

Prinsip *confusion* dan *diffusion* pertama kali diperkenalkan oleh Claude Shannon dalam makalah klasiknya yaitu *Communication theory of secrecy systems* pada tahun 1949. Dua prinsip ini kemudian sekarang sering digunakan menjadi panduan dalam merancang algoritma kriptografi. Prinsip *confusion* adalah prinsip yang menyembunyikan hubungan apapun yang ada antara plainteks, cipherteks, dan kunci. Contoh dari algoritma yang menggunakan prinsip *diffusion* ini adalah *one-time pad* (OTP). Prinsip *confusion* dapat direalisasikan dengan menggunakan algoritma substitusi yang kompleks. Sebagai contoh, algoritma DES mengimplementasikan substitusi dengan menggunakan kotak-S.

Prinsip *diffusion* adalah prinsip yang menyebarkan pengaruh satu bit plainteks atau kunci ke sebanyak mungkin cipherteks. Sehingga pengubahan kecil yang terjadi pada plainteks sebanyak satu atau dua bit dapat menghasilkan perubahan pada cipherteks yang tidak dapat diprediksi. Contoh penggunaan prinsip *diffusion* adalah penggunaan operasi permutasi pada algoritma DES.

2.3. Jaringan Feistel

Jaringan Feistel adalah sebuah struktur yang banyak digunakan sebagai prinsip perancangan cipher blok. Kelebihan dari struktur jaringan Feistel adalah operasi enkripsi dan dekripsi yang sangat mirip dan dalam banyak kasus hanya perlu membalikan urutan kunci yang digunakan. Dalam Jaringan Feistel, terdapat fungsi internal yang akan dijalankan setiap ronde pada enkripsi atau dekripsi. Untuk membuat cipher blok yang baik, fungsi inilah yang harus dibuat serumit mungkin. Ilustrasi dari jaringan Feistel dapat dilihat pada gambar berikut.



Gambar 7. Bentuk Struktur Jaringan Feistel (sumber: https://www.tutorialspoint.com/cryptography/feistel_block_cipher.htm)

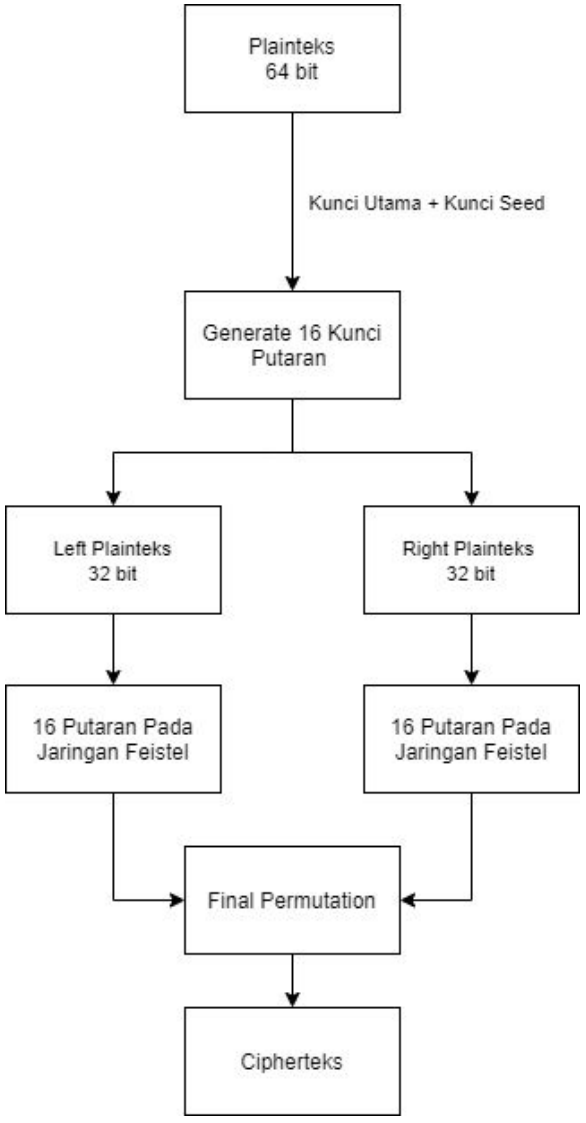
2.4. *Data Encryption Standard* (DES)

Data Encryption Standard merupakan sebuah algoritma kriptografi cipher blok kunci simetri yang populer. Diciptakan pada awal 1970 an dan diadopsi sebagai standar kriptografi *Federal Information Processing Standard* di Amerika Serikat pada 1977, DES dianggap cukup aman hingga pada akhir tahun 1990 an dimana DES berhasil dipecahkan dengan waktu yang relatif singkat. Setelah DES, standar enkripsi yang digunakan adalah *Advanced Encryption Standard* (AES).

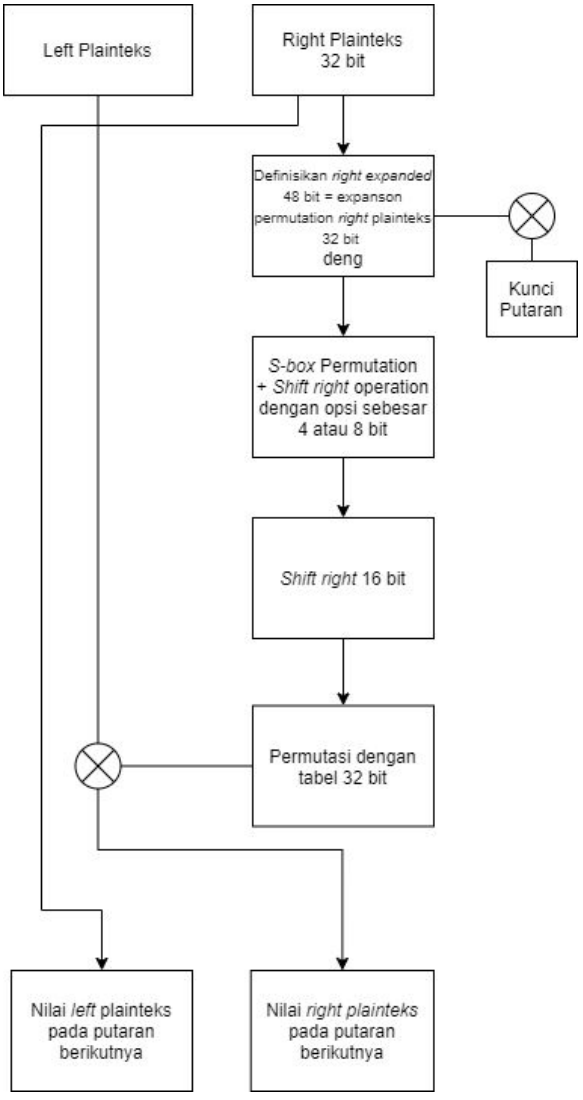
DES bekerja dengan menggunakan permutasi berulang kali, kunci putar, serta jaringan feistel untuk memenuhi prinsip *confusion* dan *diffusion.* panjang kunci DES berukuran 56 bit, namun menggunakan masukan 64 bit dengan 8 bit sebagai *parity bit.*

3. Rancangan Algoritma

Cipher blok yang diusulkan gunakan berukuran 64 bit. Dengan dua buah kunci utama. Kunci pertama berukuran 64 bit yang digunakan sebagai kunci dasar pembangkitan kunci putaran dan kunci kedua berukuran bebas. Kunci kedua digunakan sebagai *seed* untuk melakukan *random* beberapa tabel bantuan yang digunakan dalam algoritma.



Gambar 8. Skema umum algoritma usulan



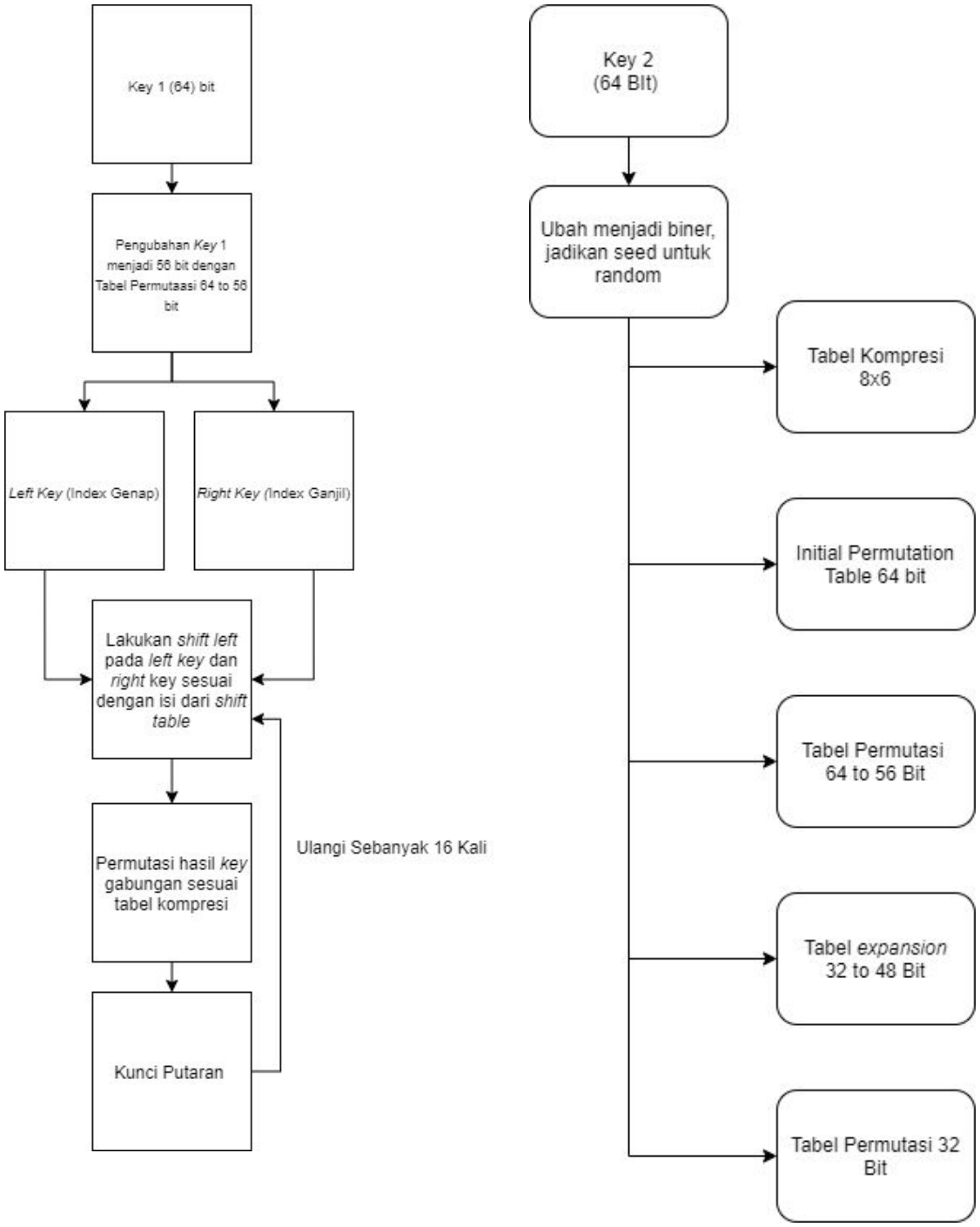
Gambar 9. Skema jaringan feistel usulan

3.1. Pembangkitan Kunci Putaran

Kunci putaran yang digunakan dalam algoritma yang diusulkan tidak lagi dibangkitkan menggunakan tabel yang telah ditentukan sebelumnya (*hardcoded*), melainkan dengan algoritma *pseudo-random* yang menggunakan kunci kedua sebagai *seed* untuk pembangkitan tabel-tabel berikut:

1. Permutasi 64 bilangan yang digunakan sebagai *initial permutation* dari plainteks atau cipherteks.
2. Tabel urutan permutasi kunci 64 bit menjadi kunci 56 bit. Pengubahan 64 bit menjadi 56 bit dilakukan dengan menghilangkan setiap bit pada posisi kelipatan 8.

3. Tabel *expansion* berukuran $6 x 8$ yang digunakan sebagai bantuan untuk mengubah 32 bit menjadi 48 bit.
4. Tabel permutasi berisi 32 bilangan untuk mengacak plainteks atau cipherteks 32 bit pada setiap putaran
5. Tabel *compression* berukuran $8 x 6$ yang digunakan sebagai permutasi untuk mengacak kunci putaran sebelum menjadi kunci putaran final.



Gambar 10. Skema Pembangunan Kunci Putaran dan Tabel pada Algoritma

Dengan menggunakan kunci sebagai *seed* pembangkitan tabel-tersebut, algoritma yang diusulkan berhasil menambahkan 5 buah variabel baru sehingga meningkatkan kompleksitas yang ada pada algoritma secara utuh. Konsekuensinya, diharapkan hasil enkripsi lebih sulit dipecahkan menggunakan serangan analisis plainteks.

3.2. Jaringan Feistel

Jaringan Feistel pada rancangan algoritma menerima input bit sebanyak 64. Kemudian bit tersebut akan dibagi menjadi dua bagian yang sama (*L* dan *R*). Jaringan Feistel yang dirancang terdiri dari 16 buah ronde, dengan setiap ronde akan melakukan hal berikut:

1. Ubah *R* menjadi 48 bit dengan melakukan *expand* dengan bantuan tabel yang sudah didefinisikan pada program. *R* yang telah menjadi 48 bit ini kemudian disebut sebagai $R_{expanded}$.
2. Lakukan XOR variabel $R_{expanded}$ dengan kunci putaran yang telah dihasilkan sebelumnya.
3. Substitusi hasil XOR pada langkah sebelumnya sehingga menjadi 32 bit dengan menggunakan *s-box table*.
    a. Untuk setiap karakter, setelah dilakukan substitusi dengan menggunakan *s-box* akan dilakukan operasi *shift right* dengan ketentuan:
        i. Jika karakter tersebut menempati posisi ganjil, maka dilakukan *shift right* sebanyak 4 bit.
        ii. Jika karakter tersebut menempati posisi genap, maka dilakukan *shift right* sebanyak 8 bit.
    b. Langkah sebelumnya dilakukan agar fungsi jaringan Feistel menjadi semakin kompleks dan tidak hanya mengandalkan substitusi saja.
4. Lakukan operasi *shift right* pada hasil substitusi langkah sebelumnya sebanyak 16 bit. Langkah ini juga dibuat untuk membuat algoritma semakin kompleks dan sulit untuk ditebak.
5. Lakukan prinsip *diffusion* dengan cara mempermutasikan bit yang telah didapatkan sebelumnya dengan bantuan tabel yang telah didefinisikan sebelumnya.
6. Lakukan XOR bit hasil permutasi pada langkah sebelumnya dengan *L*.
7. Nilai *R* untuk ronde berikutnya adalah hasil XOR pada langkah sebelumnya sementara nilai *L* untuk ronde berikutnya adalah nilai *R* pada ronde ini.

4. Percobaan dan Analisis

4.1. Percobaan

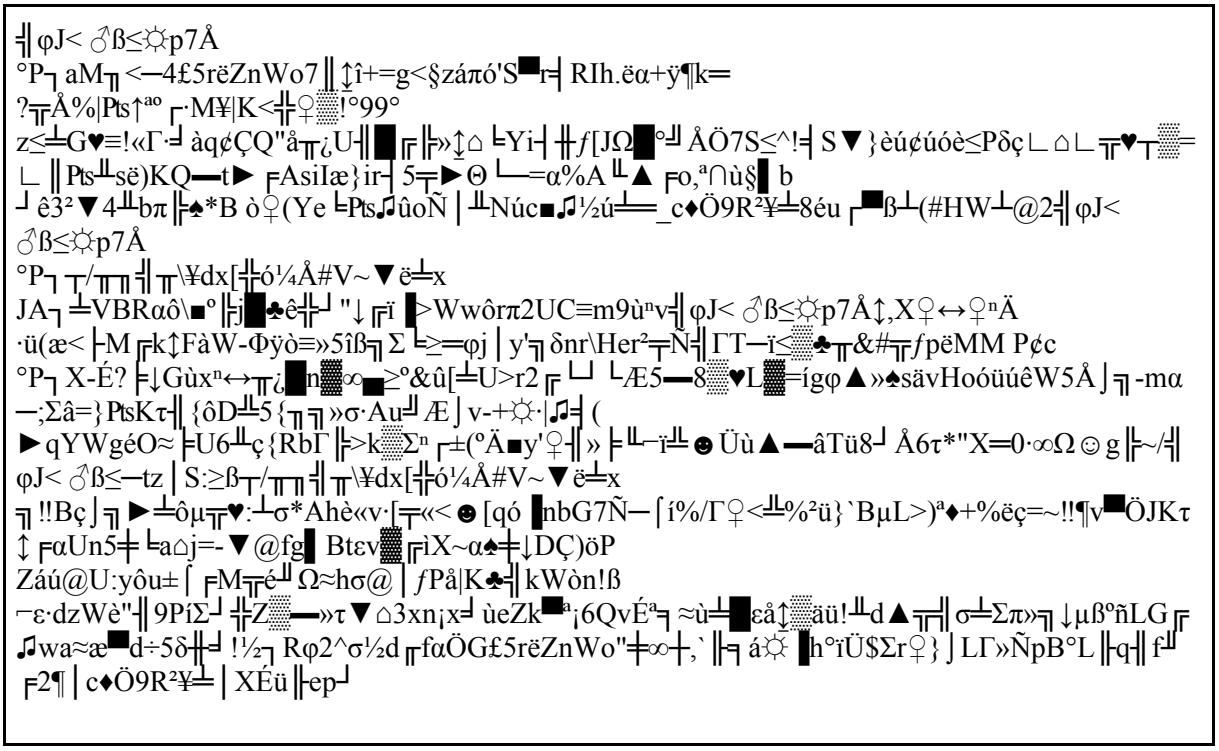
Bahasa yang digunakan dalam implementasi cipher blok adalah bahasa C++. Implementasi dilakukan dalam bit lalu akan diubah menjadi untaian karakter. *Testing* yang digunakan pada percobaan adalah plainteks berikut:

```
Prinsip confusion dan diffusion pertama kali diperkenalkan oleh
Claude Shannon dalam makalah klasiknya yaitu Communication theory
of secrecy systems pada tahun 1949. Dua prinsip ini kemudian
sekarang sering digunakan menjadi panduan dalam merancang
algoritma kriptografi. Prinsip confusion adalah prinsip yang
menyembunyikan hubungan apapun yang ada antara plainteks,
cipherteks, dan kunci.  Contoh dari algoritma yang menggunakan
prinsip diffusion ini adalah one-time pad (OTP). Prinsip
confusion dapat direalisasikan dengan menggunakan algoritma
substitusi yang kompleks. Sebagai contoh, algoritma DES
mengimplementasikan substitusi dengan menggunakan kotak-S.
```

```
Prinsip diffusion adalah prinsip yang menyebarkan pengaruh satu
bit plainteks atau kunci ke sebanyak mungkin cipherteks. Sehingga
pengubahan kecil yang terjadi pada plainteks sebanyak satu atau
dua bit dapat menghasilkan perubahan pada cipherteks yang tidak
dapat diprediksi. Contoh penggunaan prinsip diffusion adalah
penggunaan operasi permutasi pada algoritma DES.
```

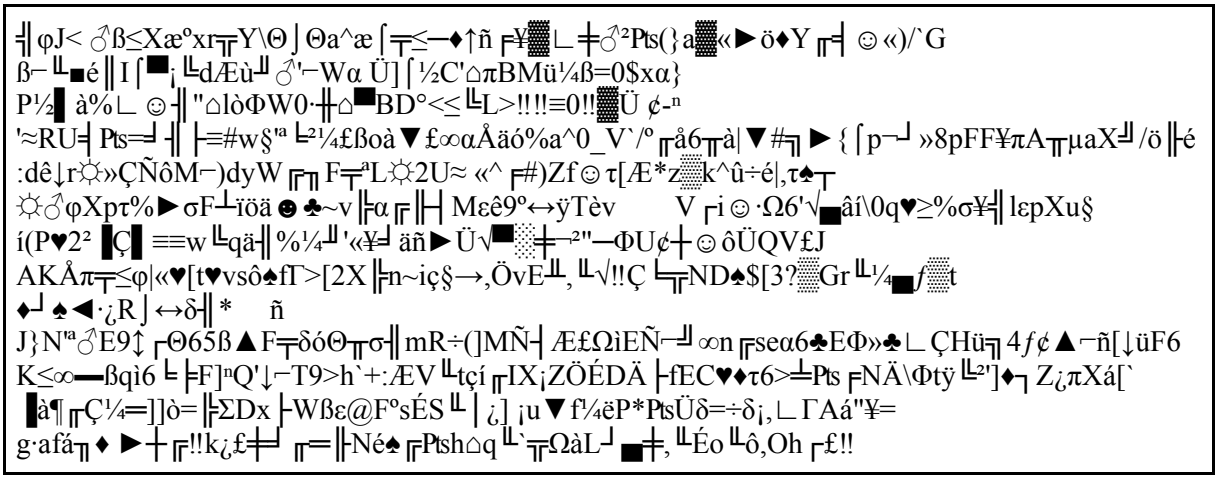
Percobaan dilakukan pada mode operasi ECB, CBC, dan Counter. Berikut adalah hasil enkripsi dari setiap mode tersebut.

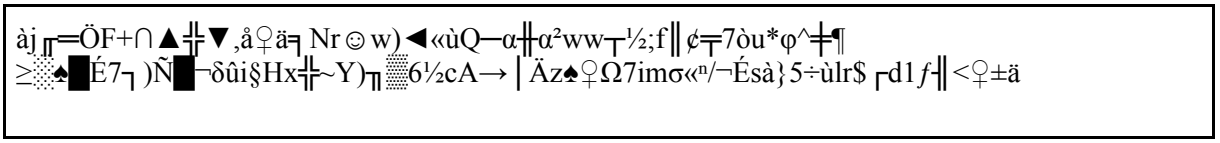
4.1.1. Hasil Enkripsi Mode ECB



Gambar 11. Hasil Enkripsi Mode ECB.

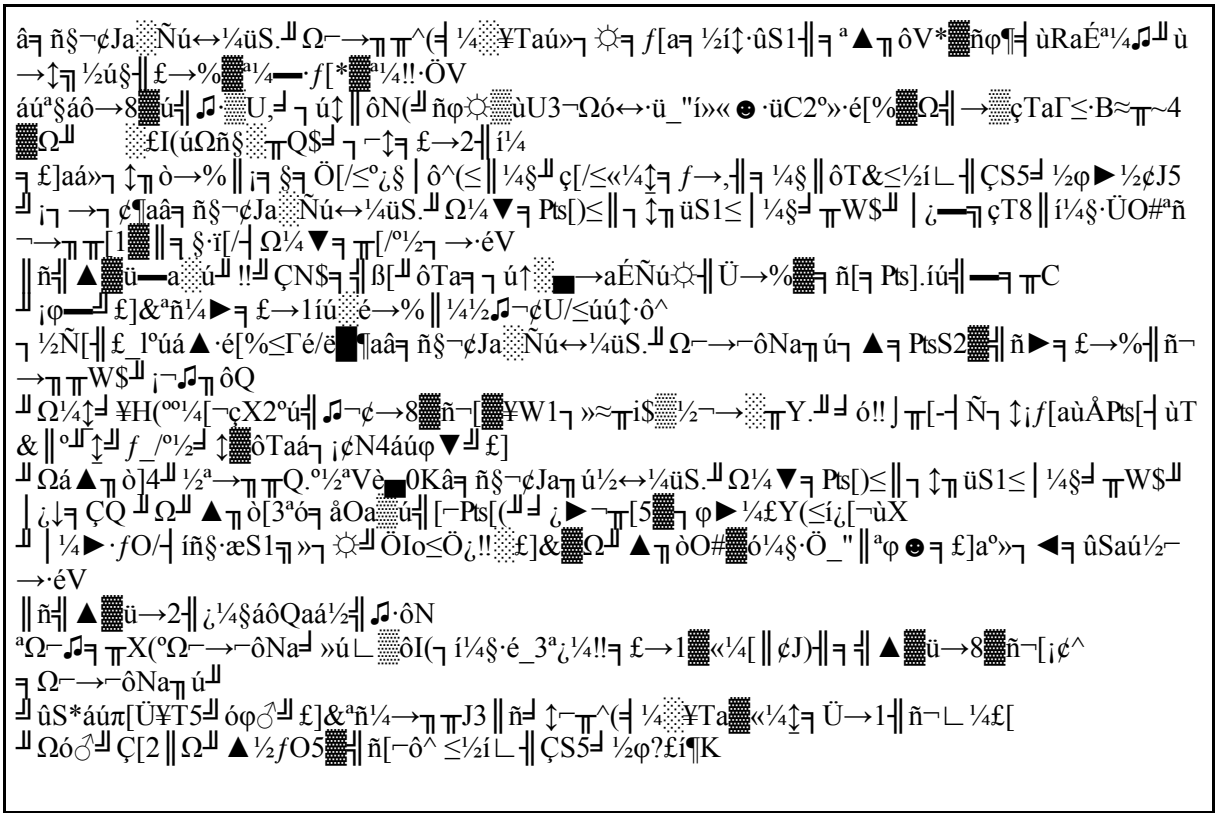
4.1.2. Hasil Enkripsi Mode CBC





Gambar 12. Hasil Enkripsi Mode CBC

4.1.3. Hasil Enkripsi Mode Counter



Gambar 13. Hasil Enkripsi Mode Counter

4.2. Analisis *Brute Force*

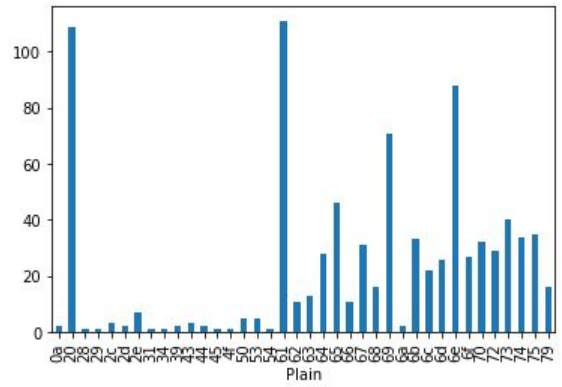
Kelemahan algoritma kriptografi modern ada pada kunci yang digunakan. Semakin panjang kunci maka semakin tinggi juga kemungkinan kunci yang harus dicoba. Dikarenakan tidak adanya perubahan panjang kunci dibandingkan dengan algoritma DES, maka kompleksitas yang dimiliki tetap sepanjang kunci, yaitu 2^{64} bit. Dengan angka tersebut, apabila dapat dilakukan 10^8 kunci per detik maka dibutuhkan sekitar 5845 tahun. Dari jumlah tersebut dapat disimpulkan bahwa algoritma ini masih cukup sulit dipecahkan.

Di sisi lain, algoritma yang kami usulkan juga meningkatkan ketahanan *ciphertext* terhadap serangan plainteks seperti analisis frekuensi, serangan *known-plaintext*, dan serangan *chosen-plaintext.*

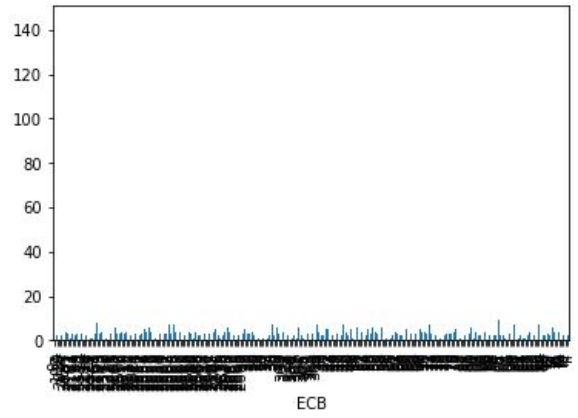
4.3. Analisis *Confusion* dan *Diffusion*

4.3.1. *Confusion*

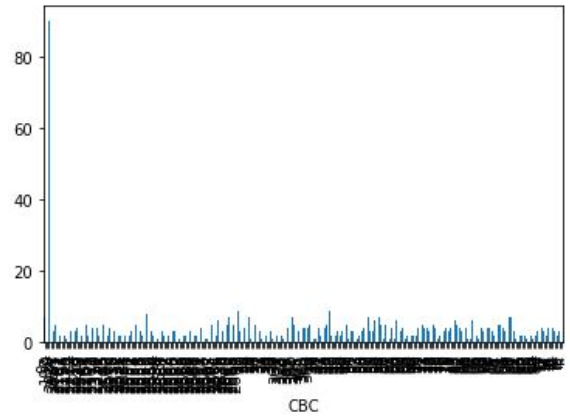
Analisis *confusion* dievaluasi dengan membandingkan frekuensi kemunculan karakter hexadesimal dari plainteks dengan cipherteks.



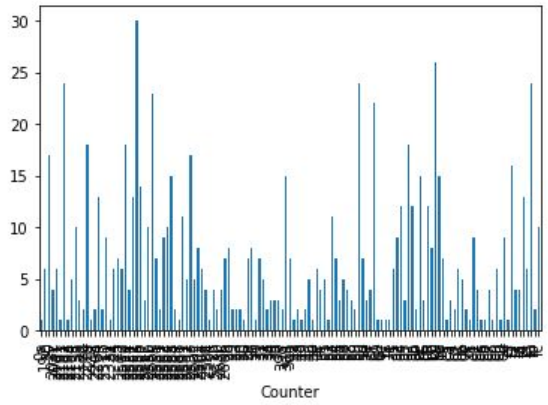
Grafik 1. Frekuensi karakter hexadesimal plainteks



Grafik 2. Frekuensi karakter hexadesimal ECB



Grafik 3. Frekuensi karakter hexadesimal CBC



Grafik 4. Frekuensi karakter hexadesimal *Counter*

Dari keempat grafik tersebut, terlihat bahwa mode ECB dan CBC dapat mengubah frekuensi karakter plainteks menjadi lebih merata sehingga lebih sulit dilakukan analisis plainteks. Di sisi lain, mode *Counter* melakukan enkripsi secara cukup baik karena menambah karakter hexadesimal yang ada namun masih menunjukkan frekuensi tinggi di beberapa karakter. Dengan demikian, dapat diambil kesimpulan bahwa algoritma yang diusulkan sudah memenuhi prinsip *confusion* pada mode ECB dan CBC.

4.3.2. *Diffusion*

Analisis *diffusion* dilakukan akan dilakukan dengan cara mengubah 1 bit pada kunci yang digunakan untuk proses enkripsi dan dekripsi. Kemudian akan dilakukan perbandingan cipherteks yang dihasilkan pada masing-masing proses enkripsi dan dekripsi.

Plainteks yang digunakan pada proses analisis adalah sebagai berikut

```
Ini adalah contoh plainteks yang digunakan sebagai proses
analisis diffusion pada tugas pembuatan cipher blok 'baru'!!!!
```

Dengan menggunakan kunci

```
inikunci
```

Dengan menggunakan mode *counter*, hasil enkripsi dari plainteks di atas dan direpresentasikan dalam hexadesimal adalah

```
2E A2 CE 31 DF E2 D5 7B 73 82 FB 55 F0 B5 E0 EA D5 D6 F0 41 42 65
91 41 FA 2B 92 25 90 2C DE BA ED 8B ED A6 08 2F 49 01 41 67 0D 05
0F BB 3F A5 0A 13 D0 CB 3D E3 99 F7 67 DC DD A9 AA C1 E6 C1 A2 14
B7 0F 2C 08 31 35 C0 29 AD 2C 3E D7 E0 A6 E2 A6 05 EC A3 87 AA 3D
C9 F4 C8 BE F0 32 3B 35 C5 63 99 DD 9E E3 EE 24 40 A0 E9 75 1C 01
9D 85 CD F9 E5 33 97 F7 68 73
```

Misalkan kita mengubah 1 bit pada kunci lama menjadi kunci seperti berikut

```
inikumci
```

Dengan menggunakan mode *counter*, hasil enkripsi dari plainteks di atas berubah menjadi seperti berikut

```
9D E7 00 74 58 11 EA 30 61 CA F0 31 AE 06 43 0D 4B F8 E3 DB 60 8B
9B CC AF 16 7E A2 BB 2F FF 30 0D 53 5D 41 5F 2B 27 06 BF F0 09 84
62 91 D1 12 23 BE F4 EA A6 FE F2 17 22 28 2F 11 55 3B F3 C1 46 4A
A3 4C 0B 18 2F B6 5C B7 B6 EC 15 B3 E1 B8 6A CC FC 01 BB 13 67 EA
40 B5 4F 37 47 4B CC 0C 31 C7 B1 7A CD 8F 31 A8 4A DD 67 A4 3D 02
C0 14 0A 56 69 BC BE 74 9F 0F
```

Dapat dilihat bahwa hasil enkripsi yang dihasilkan sangat berbeda jauh walaupun hanya mengubah 1 karakter kunci. Maka dari itu, algoritma sudah memenuhi prinsip *diffusion.*

5. Kesimpulan dan Saran

Dari evaluasi yang telah dilakukan, algoritma yang diusulkan dirasa sudah cukup baik dalam hasil enkripsi dan dekripsi pesan. Algoritma pun telah memenuhi prinsip *confusion* dan *diffusion*, memiliki panjang kunci yang cukup panjang, serta lebih tahan dari serangan plainteks dibandingkan algoritma DES standar. Evaluasi waktu yang dibutuhkan dalam enkripsi dan dekripsi pun tidak signifikan sehingga dapat digunakan secara umum.

Sebagai saran untuk pengembangan lebih lanjut, sebaiknya dilakukan peningkatan kompleksitas pada key untuk mengurangi *vulnerability* algoritma dari serangan *brute force.* Selain itu, perlu dilakukan analisis lebih mendalam pada kelemahan lain yang mungkin ada di dalam algoritma.

6. Referensi

[1] Grabbe, J. Orlin. 2006. *The DES Algorithm Illustrated*. http://page.math.tu-berlin.de/~kant/teaching/hess/krypto-ws2006/des.htm. (diakses pada tanggal 22 Oktober pukul 15.00 WIB)

[2] Munir, Rinaldi. 2020. Slide Kuliah IF4020: Kriptografi Modern (Bagian 3: Block Cipher)

[3] Munir, Rinaldi. 2020. Slide Kuliah IF4020: Kriptografi Modern (Bagian 4)

[4] Munir, Rinaldi. 2020. Slide Kuliah IF4020 Kriptografi: Serangan terhadap kriptografi