

Block Cipher Traveler

Ahmad Rizqee Nurhani¹, Ariel Ansa Razumardi².

^{1,2} Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika (STEI), Institut Teknologi Bandung (ITB), Jalan Ganesha 10, Bandung 40132
NIM : 13517058,13517040
E-mail: 13517058@std.stei.itb.ac.id, 13517040@std.stei.itb.ac.id

Abstrak. Makalah ini berisi sebuah usulan pada perancangan block cipher, Algoritma *block cipher* adalah sebuah algoritma kriptografi modern yang sangat populer. Menggunakan beberapa prinsip-prinsip dalam perancangannya seperti prinsip *confusion* dan *diffusion* Shanon, jaringan *feistel*, dan *cipher* berulang. Karena *block cipher* mengimplementasi prinsip-prinsip tersebut, *block cipher* dapat dikatakan memiliki keamanan yang cukup tinggi. Selain itu karena *block cipher* menggunakan jaringan *feistel*, kompleksitas pada enkripsi dapat dibuat sekompleks mungkin tanpa harus membuat algoritma lain untuk melakukan proses dekripsi. Dalam perancangan block cipher Traveler, penulis menarik inspirasi pada block cipher terkenal yang bernama DES. Secara umum, algoritma Traveler menaruh fokusnya pada pembangkitan bilangan acak dan proses transposisi untuk membantu proses enkripsi. Berdasarkan analisis yang telah penulis lakukan algoritma Traveler aman untuk digunakan. Hal ini dikarenakan algoritma tersebut tahan terhadap *brute force*, frekuensi analisis, dan teknik-teknik kriptanalisis lain, **Keywords:** *block cipher*, Traveler, DES, transposisi, bilangan acak.

1. Pendahuluan

Dengan berkembangnya teknologi, teknik-teknik kriptografi tentu juga berkembang pesat. Kriptografi adalah sebuah teknik yang digunakan untuk menyembunyikan sebuah naskah, tulisan, ataupun surat, yang biasa disebut sebagai “plainteks”, menjadi sebuah naskah atau tulisan yang tidak dapat dibaca, naskah yang berisi tulisan yang tidak dapat dibaca biasa disebut dengan nama “cipherteks”. Kata kriptografi berasal dari bahasa Yunani “cryptós”, yang artinya rahasia, dan “gráphein”, yang artinya tulisan, sehingga kata kriptografi dapat diterjemahkan menjadi “tulisan rahasia”. Menurut Menez (1996), kriptografi adalah ilmu yang mempelajari teknik-teknik matematika yang berhubungan dengan aspek keamanan informasi seperti kerahasiaan, integritas data, serta otentikasi. Kriptografi digunakan untuk menjaga keamanan dan kerahasiaan dari sebuah teks rahasia. Kriptografi memiliki aspek “aman”, yang berarti terjaga kerahasiaan, terjaga keasliannya, yakin pengirim pesan bukan pihak ketiga, dan pesan tidak dapat disangkal oleh penerima.

Dalam rekor historis, kriptografi untuk menjaga kerahasiaan sebuah pesan penting telah digunakan oleh kerajaan Romawi kuno. Kriptografi yang digunakan dikenal dengan nama “Caesar Cipher”. Cipher ini digunakan oleh Julius Caesar untuk menjaga kerahasiaan dari pesan-pesan pribadi yang dia ingin kirimkan. Caesar cipher merupakan contoh dari enkripsi substitusi, dimana huruf-huruf alfabet pada sebuah plainteks disubstitusi dengan huruf-huruf alfabet lain menghasilkan sebuah cipherteks.

Pada zaman sekarang, enkripsi-enkripsi klasik seperti enkripsi substitusi sudah tidak digunakan dalam pengenkripsian sebuah pesan rahasia. Hal ini dikarenakan enkripsi-enkripsi klasik dapat dengan mudah untuk di decipher oleh pihak ketiga menggunakan teknik frekuensi analisis. Oleh sebab itu kriptografi harus berkembang untuk dapat mengikuti zaman modern. Kriptografi yang berkembang ini melakukan operasi pada level byte atau bit. Prinsip kriptografi modern sama dengan kriptografi klasik

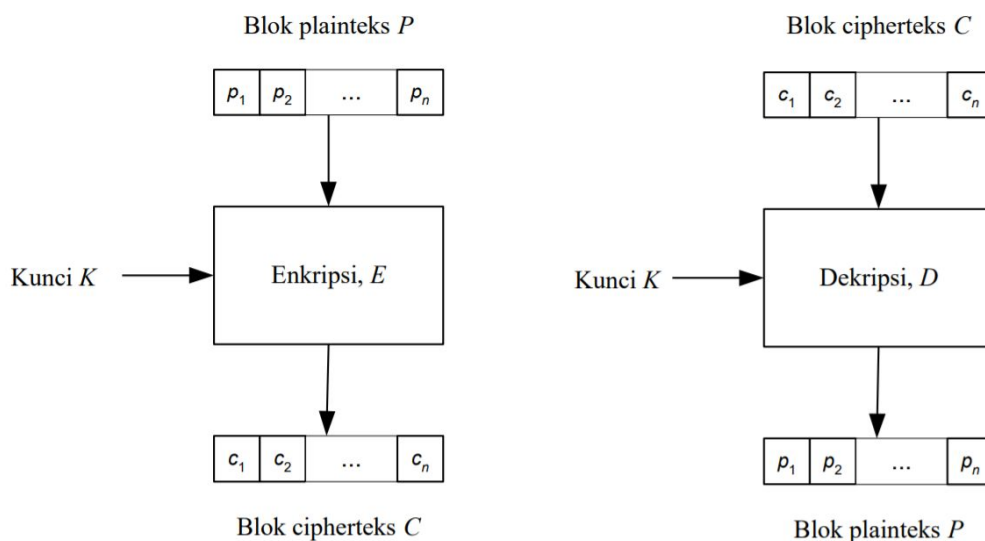
dimana kriptografi tersebut menggunakan teknik-teknik seperti substitusi dan transposisi, tetapi kompleksitas dari teknik-teknik tersebut lebih kompleks dibandingkan kriptografi klasik. Salah satu teknik kriptografi yang digunakan pada zaman modern adalah block cipher.

Pada makalah ini penulis mengembangkan sebuah block cipher baru bernama Block Cipher Traveler. Block cipher yang dikembangkan penulis dibuat berdasarkan algoritma pada *Data Encryption Standard* (DES) atau yang disebut dengan nama *Data Encryption Algorithm* (DEA). Penulis memilih DEA sebagai dasar dikarenakan DEA telah memenuhi seluruh prinsip perancangan block cipher. Block cipher Traveler melakukan proses enkripsi dengan fokus kepada pembangkitan bilangan acak dan proses transposisi.

2. Studi Pustaka

2.1. Block Cipher

Block cipher adalah sebuah teknik enkripsi dimana bit-bit pada sebuah plaintext dibagi menjadi beberapa block-bit dengan ukuran sama, biasanya 64 atau lebih bit, lalu block-bit tersebut dienkripsikan menggunakan sebuah bit kunci.

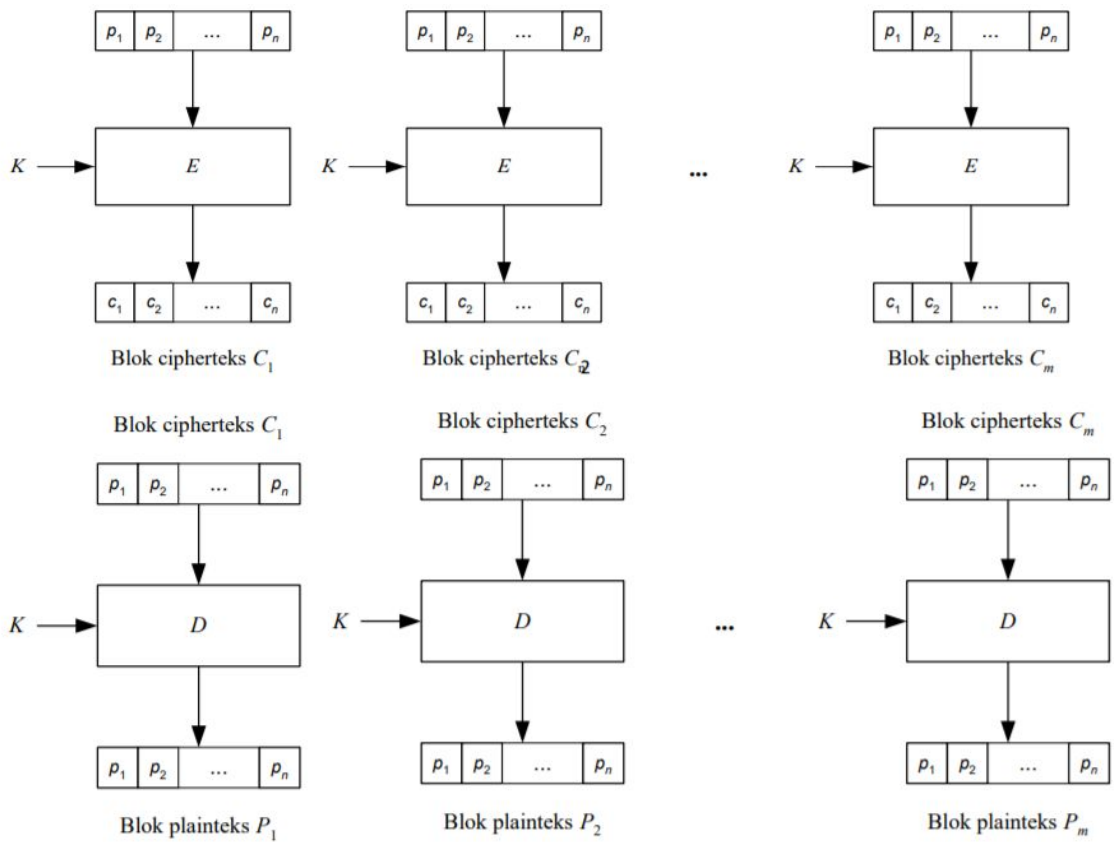


Gambar 1. Gambaran Enkripsi Block Cipher

Sebelum sebuah blok dienkripsikan, terdapat 5 mode pengoperasian blok tersebut. Setiap mode memiliki kelebihan dan kekurangan masing-masing. Dalam perancangan block cipher, penulis menggunakan 3 dari 5 mode, mode yang digunakan adalah:

1. Electronic Code Book

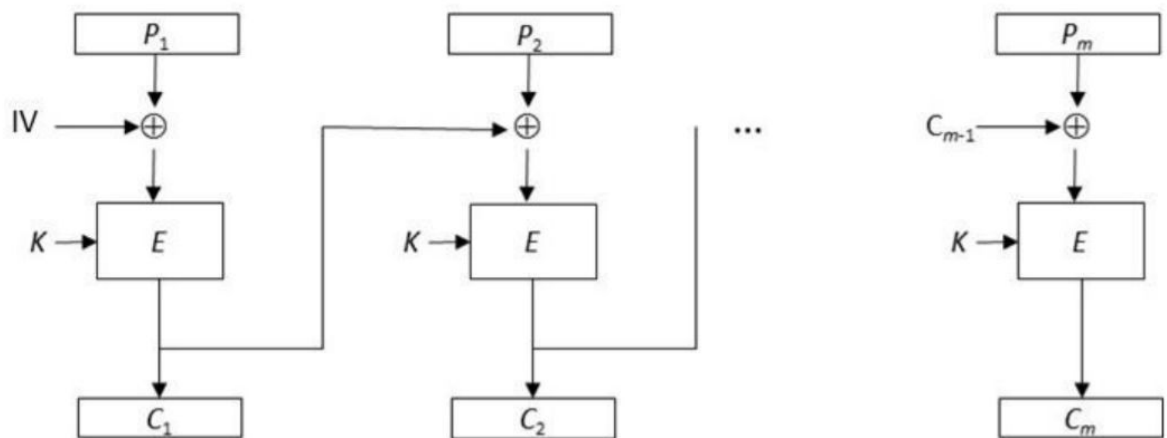
Mode ini adalah mode paling simpel dari pengoperasian blok. Pada mode ini setiap blok plaintext di enkripsi independen dari blok lainnya. Setelah blok-blok dienkripsi, blok-blok digabungkan menjadi satu cipherteks. Hal yang sama dilakukan pada proses dekripsi.



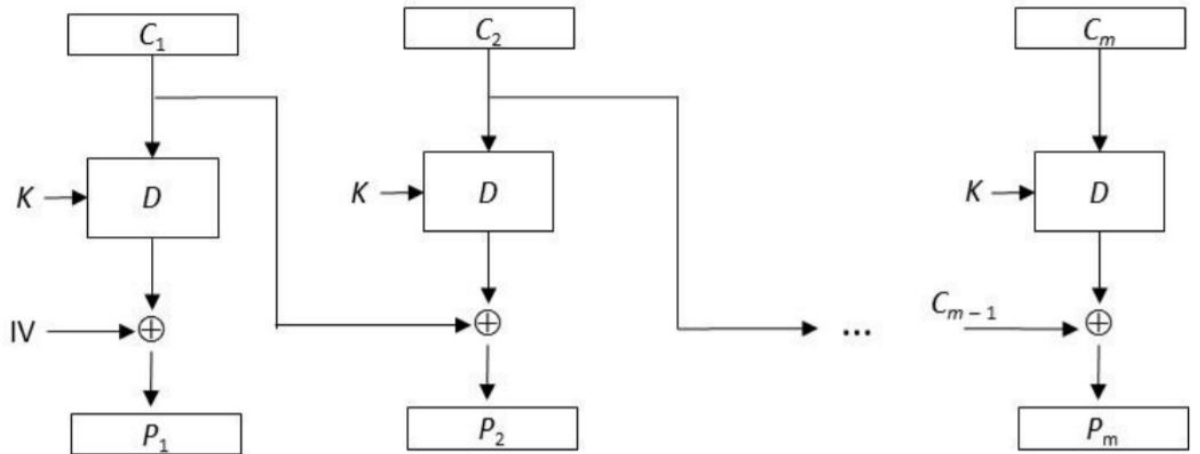
Gambar 2. Gambaran enkripsi dan dekripsi ECB

2. Cipher Block Chaining

Mode Cipher Block Chaining adalah mode enkripsi dimana blok yang telah dienkripsi bergantung terhadap blok lain. Mode ini memiliki tujuan untuk membuat ketergantungan antara blok-blok agar isi dari sebuah plaintext tidak dapat diserang oleh pihak ketiga.



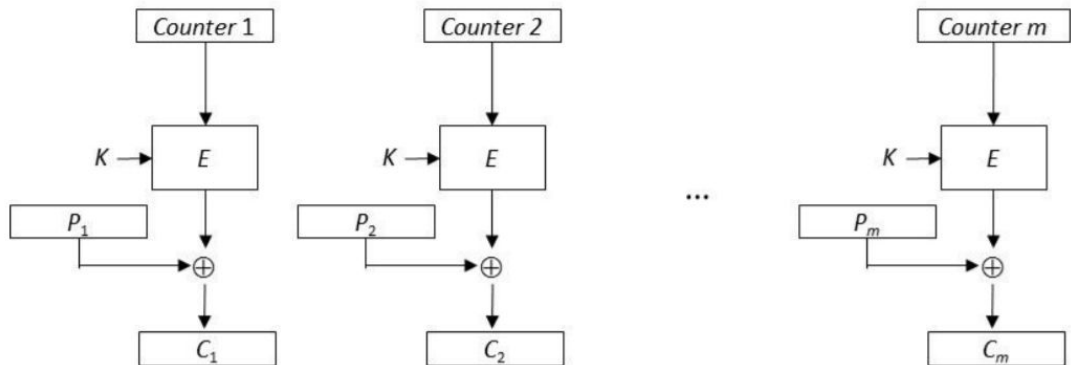
Gambar 3. Gambaran enkripsi CBC



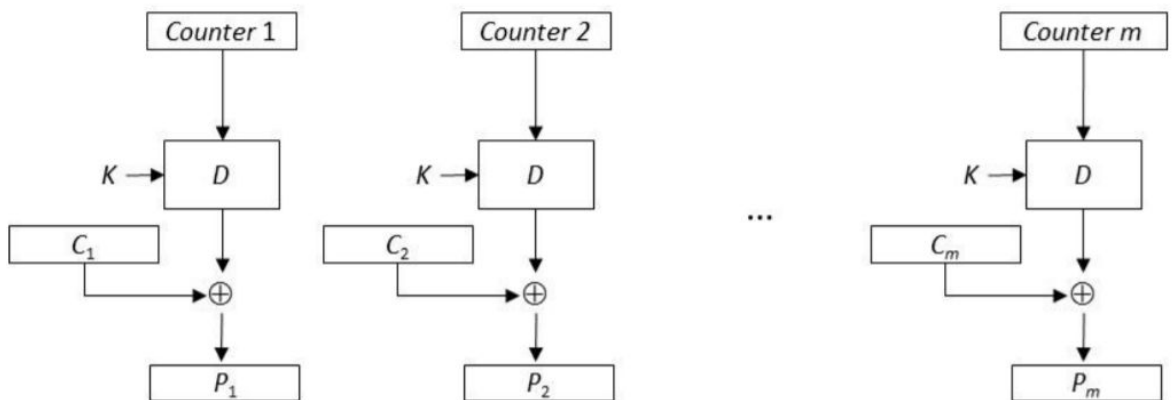
Gambar 4. Gambaran dekripsi CBC

3. Counter Mode

Pada mode ini enkripsi dilakukan pada nilai counter bukan pada blok-blok. Nilai counter adalah sebuah blok bit yang berukuran sama dengan blok plainteks yang berjumlah sama dengan jumlah blok bit plainteks, nilai counter berbeda untuk tiap blok plainteks. Untuk menghasilkan cipherteks, dilakukan operasi XOR pada blok plainteks dengan hasil nilai counter yang telah dienkripsi.



Gambar 5. Gambaran enkripsi Counter

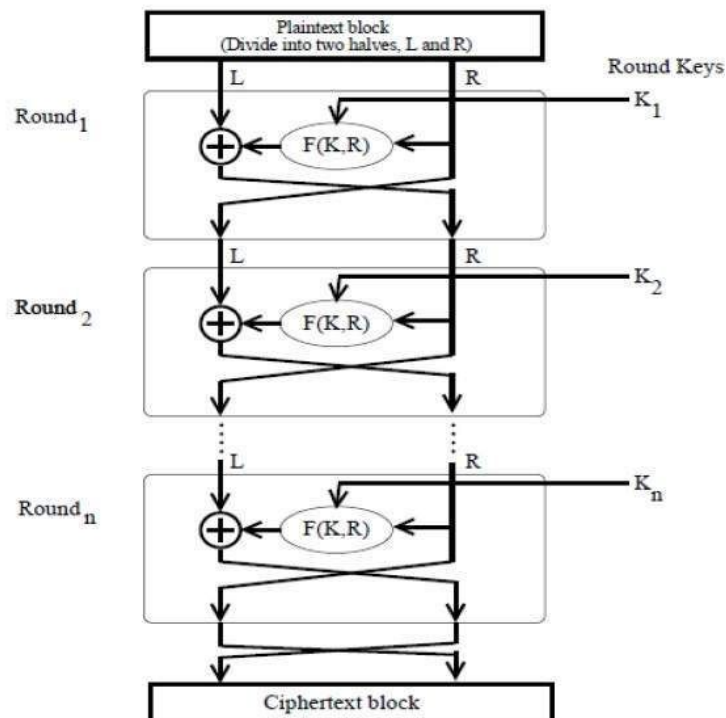


Gambar 6. Gambaran dekripsi Counter

Dalam pembangunannya algoritma block cipher menggunakan prinsip-prinsip untuk membuat enkripsinya lebih sulit untuk ditebak. Prinsip-prinsip tersebut bernama Confusion dan Diffusion dari Shannon, Cipher berulang, Jaringan Feistel, dan Kotak-S. Prinsip-prinsip tersebut membuat sebuah block cipher menjadi lebih aman.

2.2. Jaringan Feistel

Jaringan feistel adalah sebuah desain pada pembuatan block cipher yang memungkinkan enkripsi dan dekripsi dapat dilakukan menggunakan algoritma yang sama. Cara kerja jaringan feistel adalah pada putaran pertama input blok plainteks akan dibagi menjadi dua. Bagian kanan akan diubah menggunakan sebuah fungsi F dan sebuah kunci K_1 lalu dilakukan operasi XOR dengan bagian kiri. Hasil XOR akan menjadi bagian kanan untuk round selanjutnya dan bagian kanan lama yang belum diubah akan menjadi bagian kiri untuk round selanjutnya. Round akan diulang sebanyak n kali. Karena operasi XOR tersebut jaringan feistel bersifat reversible, sehingga fungsi F dapat dibuat sekompleks mungkin.



Gambar 7. Jaringan feistel

2.3. Shanon's Principle of Confusion and Diffusion

Shanon's Principle of Confusion and Diffusion adalah sebuah prinsip yang dikemukakan oleh Claude Shannon pada tahun 1949 dalam makalah berjudul "Communication theory of secrecy systems" yang berisi sebuah usulan untuk melawan serangan statistik pada enkripsi. Prinsip confusion dan diffusion menjadi panduan dalam perancangan algoritma enkripsi. Berikut adalah penjelasan mengenai prinsip confusion dan diffusion :

1. Confusion

Prinsip confusion digunakan untuk menyembunyikan hubungan apapun yang terdapat pada plainteks. Prinsip ini akan mempersulit kriptanalis dalam mencari pola pada cipherteks. prinsip ini dapat direalisasikan dengan menggunakan algoritma substitusi yang kompleks.

2. Diffusion

Prinsip diffusion digunakan untuk menyebarkan pengaruh dari sebuah bit pada plainteks atau kunci ke sebanyak mungkin cipherteks. Prinsip ini bertujuan agar perubahan sedikitpun pada

plaintexts akan menyebabkan perubahan yang tidak dapat terprediksi pada ciphertexts. Prinsip ini dapat direalisasikan dengan menggunakan permutasi.

2.4. Kotak-S

Kotak-S adalah sebuah matriks substitusi sederhana yang berisi pemetaan satu atau lebih bit dengan satu atau lebih bit lain. Kotak-S memetakan m bit masukan menjadi n bit keluaran. Kotak-S merupakan satu-satunya langkah nirlanjar dalam perancangan sebuah algoritma block cipher. Hal ini dikarenakan kotak-S merupakan sebuah operasi *look-up table*. Masukan dari operasi *look-up table* dijadikan sebagai indeks kotak-S, dan ukurannya adalah entry dalam kotak-S.

2.5. Cipher Berulang

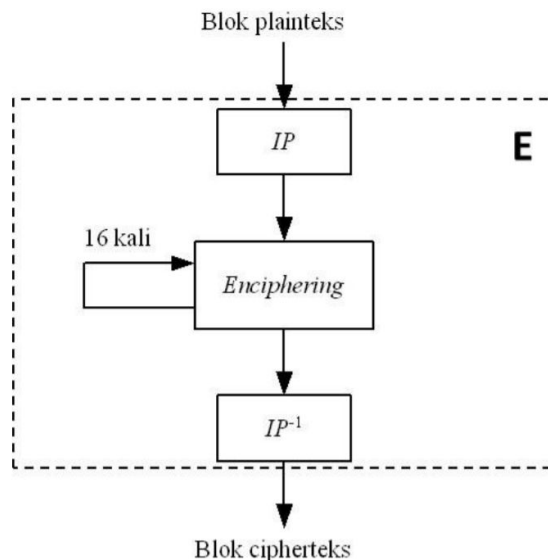
Cipher berulang adalah sebuah fungsi transformasi sederhana yang melakukan proses enkripsi pada sebuah plaintext berkali-kali sebanyak n kali. Pada setiap putaran, digunakan upa-kunci yang dikombinasikan dengan plaintexts. Cipher berulang dinyatakan sebagai

$$C_i = f(C_{i-1}, K_i)$$

dimana i adalah jumlah putaran, K_i adalah upa-kunci pada putaran ke i , dan f adalah fungsi transformasi.

2.6. DES

Salah satu Block cipher yang terkenal adalah block cipher dengan nama Data Encryption Standard (DES). Block cipher tersebut dikembangkan oleh IBM pada tahun 1972 dan menjadi standar pada algoritma enkripsi data. DES dikategorikan sebagai algoritma kriptografi kunci-simetri. DES menggunakan kunci eksternal dengan jumlah 64 bit. Cara enkripsi DES pada dasarnya adalah



Gambar 8. Gambaran algoritma DES

DES memenuhi seluruh prinsip-prinsip perancangan block cipher. Prinsip Cipher berulang dipenuhi pada 16 kali *Enciphering*, prinsip jaringan feistel diimplementasikan dalam proses *Enciphering*, prinsip confusion dan diffusion shanon diimplementasikan pada permutasi awal (IP), dan kotak-S telah diimplementasi dalam proses confusion permutasi awal.

3. Proposed Block Cipher

Algoritma Traveler ini adalah algoritma *block cipher* yang menggunakan jaringan *feistel* agar fungsi yang dipakai saat enkripsi dan dekripsi dapat sama. Jumlah putaran dengan jaringan *feistel* pada algoritma Traveler ini adalah 8 sampai 12 kali dengan ukuran masing-masing *block* 64 bit. Algoritma

ini didasari oleh DES (Data Encryption Standard) dan beroperasi dalam 3 mode, yaitu ECB, CBC, dan *Counter*. Salah satu ciri khas dari algoritma ini adalah proses *cipher* transposisi dan pembangkitan bilangan acak yang dilakukan pada berbagai bagian.

A. Penentuan *Seed* untuk Pembangkitan Bilangan Acak

Pada beberapa bagian dari algoritma Traveler ini, diperlukan sebuah *seed* untuk membangkitkan sebuah bilangan acak yang digunakan untuk berbagai proses. *Seed* ini ditentukan dengan menggunakan kunci yang dimasukkan oleh pengguna algoritma. Cara penentuan *seed* berdasarkan kunci adalah dengan menjumlahkan nilai ASCII dari masing-masing karakter pada kunci. Berikut adalah contoh penentuan nilai *seed* dari sebuah kunci:

Tabel 1. Proses penentuan *seed*

Kunci	IniKunci
<i>Seed</i>	$I(73) + n(110) + i(105) + K(75) + u(117) + n(110) + c(99) + i(105) = 794$

B. Penentuan Jumlah Putaran

Seperti yang sudah dijelaskan sebelumnya, jumlah putaran jaringan *feistel* pada algoritma Traveler adalah 8 sampai 12 kali. Jumlah putaran ini ditentukan dengan menggunakan bilangan acak di antara 8 sampai dengan 12 yang dibangkitkan menggunakan *seed* yang sudah ditentukan menggunakan kunci. Berikut adalah contoh jumlah putaran yang dibangkitkan dengan pembangkit bilangan acak yang terdapat pada bahasa *python*:

Tabel 2. Proses penentuan jumlah putaran

Kunci	IniKunci
<i>Seed</i>	794
Jumlah Putaran	10

C. Penentuan Matriks Permutasi Awal dan Akhir

Algoritma ini menggunakan proses permutasi pada tahap paling awal dan tahap paling akhir dari proses enkripsi dan dekripsi sebagai realisasi dari prinsip *diffusion* dari Claude Shannon. Matriks permutasi ini bukanlah sebuah matriks yang konstan seperti pada algoritma DES, melainkan dapat berbeda berdasarkan *seed*. Matriks permutasi awal akan dibangkitkan dengan cara membangkitkan bilangan acak di antara 1 sampai dengan 64 sebanyak 64 kali sampai matriks sudah berisi bilangan 1 sampai dengan 64 dengan urutan yang teracak. Matriks permutasi akhir akan dibangkitkan sebagai kebalikan dari matriks permutasi awal, seperti jika indeks pertama dari matriks permutasi awal bernilai 9, maka indeks ke-9 dari matriks permutasi akhir akan bernilai 1. Berikut adalah contoh matriks permutasi awal dan matriks permutasi akhir yang dibangkitkan oleh algoritma dengan Kunci “Ini Kunci”:

Tabel 3. Matriks permutasi awal

40	21	12	41	54	53	9	6	1	32	45	3	13	26	10	36
48	23	5	42	46	49	19	14	39	61	37	60	2	62	63	22

43	56	4	15	8	27	55	16	64	28	31	17	7	11	18	20
58	24	57	29	44	59	25	35	30	33	34	38	47	50	51	52

Tabel 4. Matriks permutasi akhir

9	29	12	35	19	8	45	37	7	15	46	3	13	24	36	40
44	47	23	48	2	32	18	50	55	14	38	42	52	57	43	10
58	59	56	16	27	60	25	1	4	20	33	53	11	21	61	17
22	62	63	64	6	5	39	34	51	49	54	28	26	30	31	41

D. Proses Transposisi

Salah satu keunikan dari algoritma Traveler adalah proses transposisi. Proses transposisi ini dilakukan menggunakan *cipher* transposisi dengan detail sebagai berikut:

Tabel 5. Proses transposisi

Teks Awal	Proses Transposisi	Teks Akhir
Transposisi	Tasoiirnpss	Tasoiirnpss

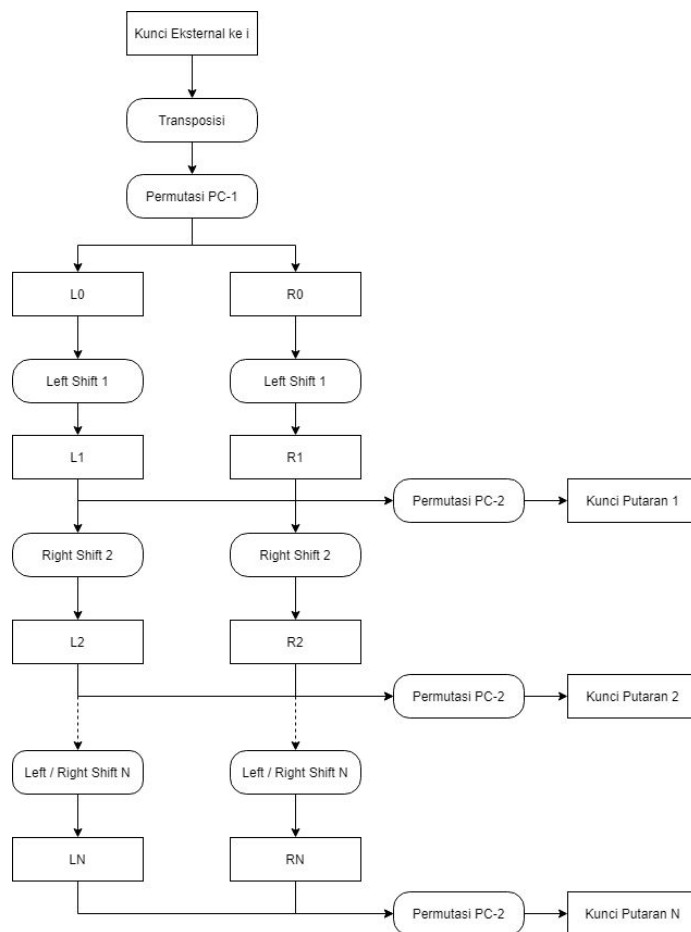
Selain seperti yang dapat dilihat pada tabel di atas, proses transposisi ini dapat dibayangkan memisahkan sebuah teks menjadi anggotanya yang berada di indeks genap dan anggotanya yang berada di indeks ganjil. Selain proses transposisi, pada algoritma Traveler ini juga terdapat proses detransposisi yang merupakan kebalikan dari proses transposisi. Kalau mengacu pada tabel di atas, proses detransposisi ini akan mengubah Teks Akhir kembali menjadi Teks Awal.

E. Proses Substitusi

Pada algoritma Traveler, mekanisme substitusi dibuat sama seperti yang dilakukan pada algoritma DES. Pertama sebuah bagian dari block yang berukuran 32-bit akan diekspansi menggunakan matriks ekspansi menjadi 48-bit. Lalu, saat ingin dikembalikan menjadi 32 bit, setiap 6 bit dari 48 bit akan disubstitusi dengan menggunakan kotak-s yang berbeda-beda. Matriks ekspansi dan 8 kotak-s yang digunakan juga sama seperti yang ada pada DES.

F. Pembangkitan Kunci Putaran

Pada algoritma Traveler, proses pembangkitan kunci putaran adalah sebagai berikut:



Gambar 9. Pembangkitan kunci putaran algoritma Traveler

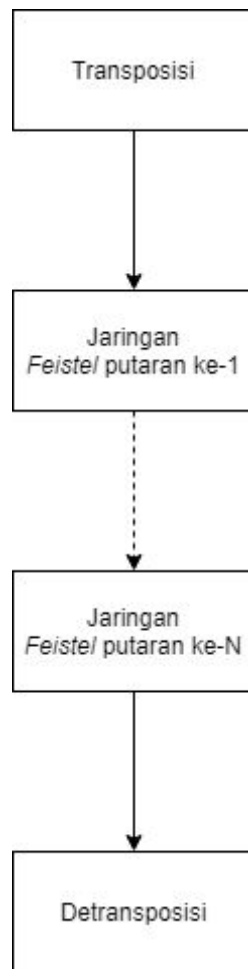
Pada diagram di atas, dapat dilihat bahwa terdapat matriks PC-1 dan PC-2. Kedua matriks tersebut merupakan matriks PC-1 dan PC-2 yang juga digunakan pada algoritma DES. Pembangkitan Kunci Putaran ini didasarkan oleh DES.

Hal yang membedakan antara pembangkitan di algoritma Traveler dan di DES adalah adanya proses Transposisi sebelum Permutasi dengan PC-1. Selain itu, perbedaan lainnya adalah dalam proses perubahan L_i ke L_{i+1} . Pada DES, proses ini dilakukan dengan Left Shift saja, sedangkan pada algoritma Traveler, proses yang dilakukan bergantian antara Left Shift dan Right Shift dengan jumlah pergeseran bit yang meningkat setiap kalinya. Perbedaan terakhir adalah Kunci Eksternal yang dimiliki oleh algoritma Traveler tidak hanya 1 tetapi bergantung kepada sepanjang apa kunci yang dimasukkan oleh pengguna. Setiap 8 byte pada kunci akan menjadi 1 buah kunci eksternal. Kunci-kunci eksternal ini akan dipakaikan

Pada diagram di atas juga dapat dilihat kalau pembangkitan kunci putaran dilakukan sampai mendapatkan N kunci. Hal tersebut disebabkan oleh jumlah putaran di algoritma Traveler dapat bervariasi dari 8 sampai 12.

G. Algoritma Enkripsi dan Dekripsi

Berikut adalah garis besar algoritma enkripsi/dekripsi yang digunakan pada algoritma Traveler:

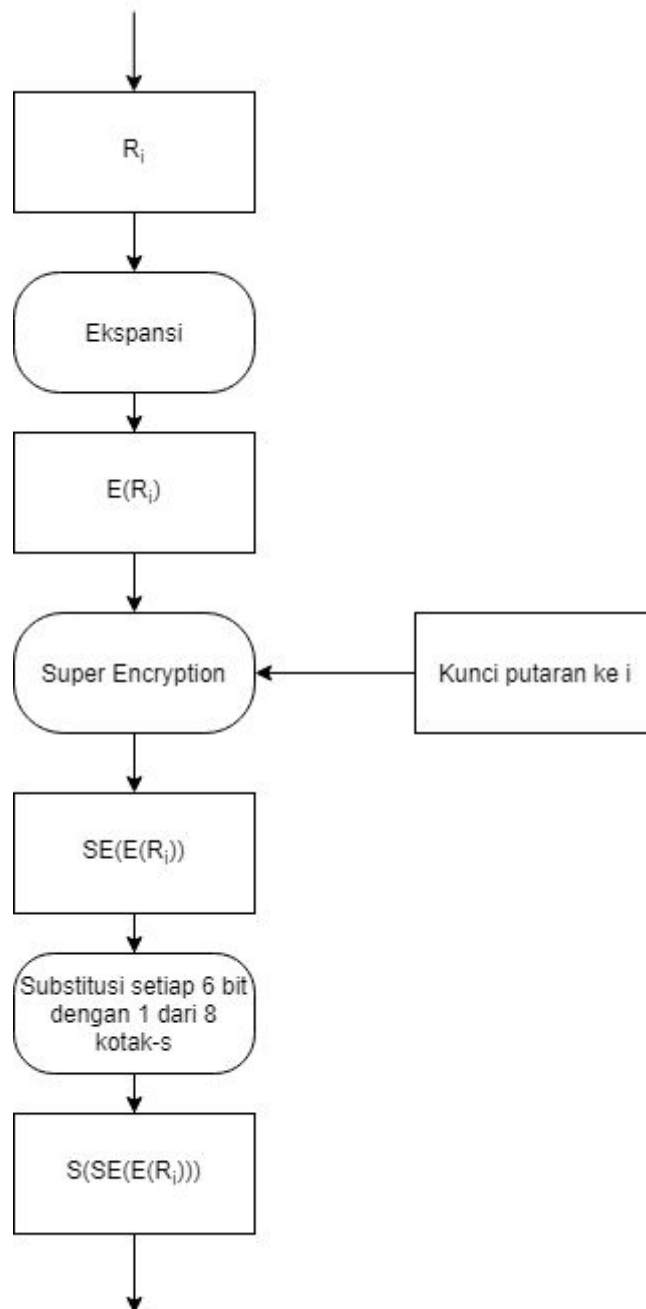


Gambar 10. Gambaran enkripsi algoritma Traveler

Algoritma enkripsi/dekripsi yang digunakan oleh Traveler secara garis besar adalah jaringan *feistel* sebanyak N kali putaran. Namun, terdapat sedikit perbedaan, yaitu pada proses transposisi dan detransposisi yang ditambahkan di awal dan akhir algoritma enkripsi/dekripsi. Dengan cara ini bagian kiri dan kanan teks pada jaringan *feistel* akan berubah.

H. Fungsi F yang Digunakan

Berikut adalah Fungsi F yang digunakan penulis pada algoritma Traveler:



Gambar 11. Gambaran fungsi f dari jaringan feistel

Pada diagram di atas terlihat ada proses ekspansi dan substitusi yang telah dijelaskan sebelumnya. Selain itu, saat bagian blok berukuran 48-bit ada proses enkripsi blok 48-bit tersebut dengan menggunakan *super encryption* yang menggunakan kunci putaran tersebut. *Super encryption* adalah algoritma enkripsi yang menggabungkan *cipher* transposisi dan *vigenere cipher*. *Cipher* transposisi yang digunakan pada Traveler sama seperti proses transposisi yang lain pada algoritma ini. *Vigenere cipher* yang digunakan adalah *full vigenere cipher*.

I. Mode CBC dan Counter

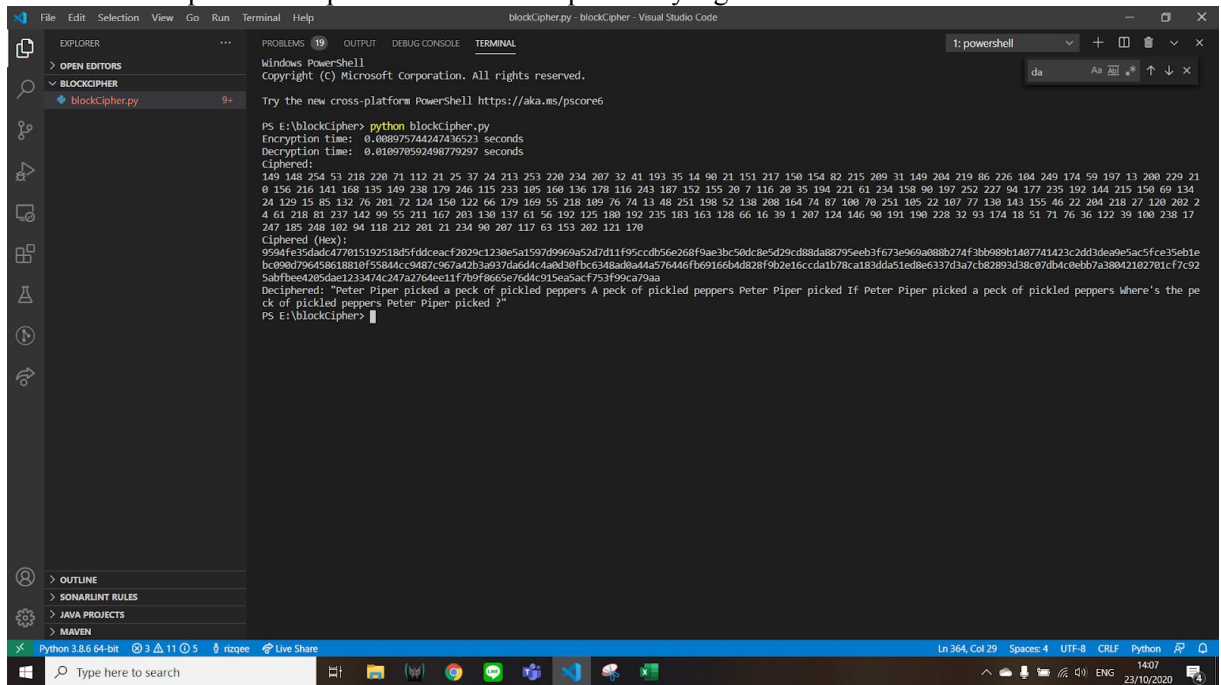
Jika menggunakan mode CBC atau *Counter*, maka terdapat nilai-nilai tambahan yang perlu diinisialisasi, yaitu IV pada mode CBC dan nilai awal *counter* pada mode *Counter*. Kedua hal ini didapatkan pada algoritma Traveler dengan menggunakan pembangkit bilangan acak

4. Eksperimen dan Analisis Hasil

A. Eksperimen

Dalam pengetesan kode algoritma Traveler, penulis menggunakan kunci “KRIPTOGRAFI_RAHASIA” dan sebuah *tongue twister* yang berisi “Peter Piper picked a peck of pickled peppers A peck of pickled peppers Peter Piper picked If Peter Piper picked a peck of pickled peppers Where's the peck of pickled peppers Peter Piper picked?”. Penulis memilih sebuah *tongue twister* sebagai kata plainteks dikarenakan terdapat banyak karakter berulang pada *tongue twister*. Sehingga dapat melakukan pengetesan pada prinsip-prinsip perancangan block cipher.

Gambar 12 sampai 14 memperlihatkan hasil eksperimen yang dilakukan.



```
File Edit Selection View Go Run Terminal Help blockCipher - blockCipher - Visual Studio Code
EXPLORER PROBLEMS 19 OUTPUT DEBUG CONSOLE TERMINAL
OPEN EDITORS
BLOCKCIPHER
blockCipher.py 9+
Try the new cross-platform PowerShell https://aka.ms/powershell
PS E:\blockCipher> python blockCipher.py
Encryption time: 0.088975744247436523 seconds
Decryption time: 0.010970592498779297 seconds
Ciphertext:
149 148 254 53 218 220 71 112 21 25 37 24 213 253 220 234 207 32 41 193 35 14 90 21 151 217 150 154 82 215 209 31 149 204 219 86 226 104 249 174 59 197 13 200 229 21
0 156 216 141 168 135 149 238 179 246 115 233 105 160 136 178 116 243 187 152 155 20 7 116 20 35 194 221 61 234 158 90 197 252 227 94 177 235 192 144 215 150 69 134
24 129 15 85 132 76 201 72 124 150 122 66 179 169 95 218 109 76 74 13 48 251 198 52 138 208 164 74 87 100 70 251 105 22 107 77 130 143 155 46 22 204 218 27 120 202 2
4 61 218 81 237 142 99 55 211 167 203 130 137 61 56 192 125 180 192 235 163 163 128 66 16 39 1 207 124 146 90 191 190 228 32 93 174 18 51 71 76 36 122 39 100 238 17
247 185 248 102 94 118 212 201 21 234 90 207 117 63 153 202 121 170
Ciphertext (Hex):
9594fe35dad477815192518d5fddceacf2029c1230e5a1597d9969a52d7d11f95cddb5e268f9ae3bc50dc8e5d29cd88da88795eeb3f673e969a088b274f3bb989b1407741423c2dd3dea9e5ac5fce35eb1e
bc90e0796458018810f55844c3087c967a72b3a937a6dcfca8030fbc6308a0a4a576446fb69166b4d828f9b2e1ccda1b78ca183dda51ed8e6337d3a7cb2893d38c07db4c0ebb7a38042102781cf7c92
5abf8ea205dae1233474c247a2764ee11f709f8669e7d4c915eaaacf753f99ca79aa
Deciphered: "Peter Piper picked a peck of pickled peppers A peck of pickled peppers Peter Piper picked If Peter Piper picked a peck of pickled peppers Where's the pe
ck of pickled peppers Peter Piper picked ?"
PS E:\blockCipher>
```

Gambar 12. Hasil tangkap layar Algoritma Traveler dengan metode CBC

```

PS E:\blockCipher> python blockCipher.py
Copyright (C) Microsoft Corporation. All rights reserved.
Try the new cross-platform PowerShell https://aka.ms/powershell

PS E:\blockCipher> python blockCipher.py
Encryption time: 0.00897574424736523 seconds
Decryption time: 0.018971784591674895 seconds
Ciphertext:
148 215 91 239 27 132 164 188 212 43 32 214 46 123 238 78 237 146 190 217 98 238 219 122 49 181 238 89 17 36 44 136 20 147 23 115 192 88 180 64 212 46 141 251 77 135
213 210 230 218 140 38 139 211 286 170 223 148 161 248 1 61 159 140 233 143 151 151 98 104 60 86 249 95 39 92 58 221 195 193 125 14 210 228 82 24 232 185 233 50 157
89 194 114 38 93 91 137 76 157 185 221 160 219 231 17 8 259 146 200 117 129 81 27 228 251 195 68 75 133 34 228 62 213 220 186 116 17 170 162 216 47 101 126 287 9 16
8 117 204 119 80 44 25 55 249 192 101 113 181 105 205 70 22 247 95 102 48 149 227 60 35 229 32 24 57 224 254 281 154 71 82 237 17 246 151 54 233 92 5 254 102 249 218
230 18 232 64 158 189 105 29 139 8 178 97 161 243 43 21 124
Ciphertext (Hex):
94d73ef1d8a4b4d42b78d52e7bee4eed92bde962eedb7a31b5ee5911242c8814931773c058b440d42e8dfb4d87d5d2fadac268bd3ceb34f94a1f8013dc78ce98f979762683c56f95f275c3addc3c170ed
245218e80e9329d59c272265d33894cc5b9dda0db67118fa92c875b3511be4fbc3444b8522d3ced5dcb4711aaa2d82f657ecf09a875cc77592c1937f9c06571b569c4616f75f663095e33c23e5201839
e0fec99a4752ed11f69736e95c05fe66f9dae12e8409ebd691d8b88261a1f32b157c
Deciphered: "Peter Piper picked a peck of pickled peppers A peck of pickled peppers Peter Piper picked If Peter Piper picked a peck of pickled peppers Where's the peck of pickled peppers Peter Piper picked ?"
PS E:\blockCipher>

```

Gambar 13. Hasil tangkapan layar Algoritma Traveler dengan metode ECB

```

PS E:\blockCipher> python blockCipher.py
Copyright (C) Microsoft Corporation. All rights reserved.
Try the new cross-platform PowerShell https://aka.ms/powershell

PS E:\blockCipher> python blockCipher.py
Encryption time: 0.008978605270385742 seconds
Decryption time: 0.010456323623657227 seconds
Ciphertext:
136 234 145 216 148 39 194 201 177 147 246 175 238 147 0 158 25 55 226 200 161 53 253 24 170 75 180 175 10 28 92 248 225 174 240 62 196 145 61 22 86 137 92 162 220 2
36 194 135 184 195 9 214 197 4 188 74 229 60 179 231 64 171 93 82 55 44 129 176 254 39 234 179 59 253 215 248 77 196 222 196 106 253 242 196 103 128 115 202 217 8 22
8 169 127 96 92 76 72 68 3 21 112 155 150 182 82 44 204 141 77 174 216 39 222 6 251 197 170 246 161 204 234 13 37 132 88 46 141 209 243 27 242 220 230 88 44 189 176
93 149 28 34 218 84 33 218 162 115 44 148 152 232 131 146 188 172 221 72 49 105 13 235 21 189 94 233 99 154 195 152 54 178 36 145 29 216 107 164 121 76 25 227 225 12
206 95 185 153 235 38 153 170 118 30 210 181 160 225 149 31 138
Ciphertext (Hex):
88e991d894277cc9b193f6afee93009e1937e2c8a135fd18aa4b4d4f0a1c5cf8e1aef03ec4913d1656895ca2dccc28768c309d6c5046cdae53cb3e740bab5d52372c81bbfe27eab33bfd7f84dc46a6dfdf
2c4678073ca908e4a97f605c4c4844031570b996b6522cc8d4daed827de06fbc5aaf6a1ccea8d258450e2e8dd1f31bf2c0e582c0db05d951c22da5421daa2732c9498e88392bcacdd4831690deb15bd5ee9
639ac39836b224911dd8eb4794c19e3e10cc5fb999eb2699aa761ed2b5a0e1951f8a
Deciphered: "Peter Piper picked a peck of pickled peppers A peck of pickled peppers Peter Piper picked If Peter Piper picked a peck of pickled peppers Where's the peck of pickled peppers Peter Piper picked ?"
PS E:\blockCipher>

```

Gambar 14. Hasil tangkapan layar Algoritma Traveler dengan metode Counter

B. Analisis Serangan Brute Force

Serangan brute force adalah salah satu cara untuk memecahkan algoritma block cipher Traveler. Serangan ini bertujuan untuk menebak kunci yang digunakan dalam enkripsi Traveler. Serangan ini mungkin merupakan satu-satunya cara untuk melewati enkripsi yang telah dilakukan oleh algoritma block cipher Traveler. Tetapi serangan brute force memiliki kelemahan yaitu waktu yang akan digunakan untuk menebak kunci. Proses tebak kunci untuk algoritma block cipher Traveler memiliki waktu kurang lebih sama dengan waktu yang dibutuhkan untuk menebak password. Dalam pembuatan password, untuk sebuah password

yang memiliki panjang 8 karakter terdapat 128^8 kemungkinan kombinasi karakter yang dapat dipakai. Untuk mengestimasi waktu, sebuah komputer modern dengan 8 core dan 2.8 GHz dapat membuat 1 kemungkinan kunci dalam 0.0017 milisecond [3]. Untuk melakukan tebakan pada kunci dibutuhkan waktu kurang lebih 14177.9988269 hari atau sekitar 38 tahun. Waktu ini tentu akan lebih lama kala

Oleh sebab itu dapat disimpulkan bahwa serangan brute force tidak mungkin dapat dilakukan untuk menemukan kunci yang digunakan untuk melakukan enkripsi block cipher Traveler. Kunci algoritma Traveler dapat memiliki panjang yang melebihi 8 karakter, oleh karena itu algoritma tersebut akan susah untuk di-brute force.

C. Analisis Frekuensi Kemunculan Karakter

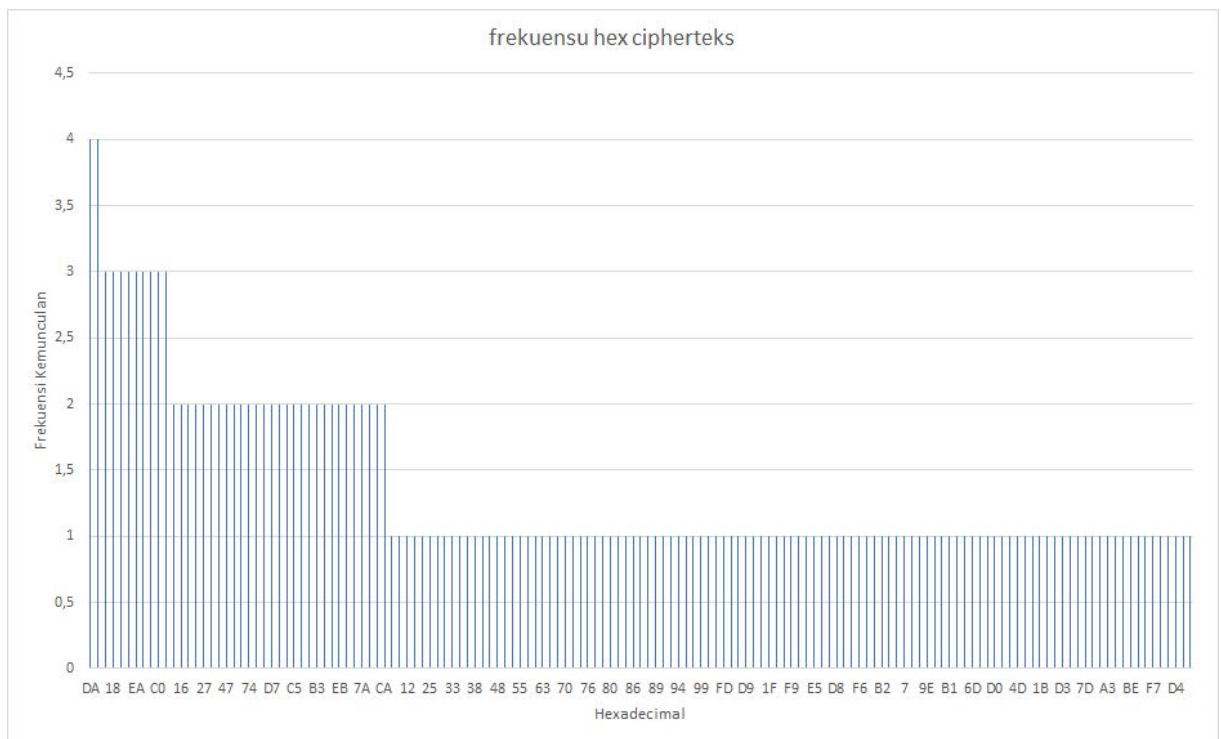
Analisis frekuensi kemunculan karakter adalah sebuah cara penyerangan yang dilakukan oleh kriptanalisis untuk menemukan pola pada sebuah cipherteks. Serangan ini merupakan serangan yang efektif terhadap kriptografi klasik seperti caesar cipher, vigenere cipher, atau cipher sejenis yang menggunakan substitusi. Frekuensi analisis menghitung kemunculan karakter pada sebuah cipherteks sehingga tanpa perlu mengetahui kunci dari enkripsi, isi plainteks dapat ditebak. Sebagai contoh jika cipher memetakan “E” ke huruf “J” maka seluruh huruf yang berdekatan dengan J pada cipherteks dapat ditebak dan dibuat tabel substitusi untuk memecahkan cipherteks.

Algoritma Traveler yang telah penulis buat merupakan algoritma yang memiliki prinsip confusion dan diffusion Shanon. Prinsip ini akan mengacaukan teknik frekuensi analisis sehingga cipherteks tidak dapat diketahui tabel substitusinya. Analisis frekuensi pada algoritma Traveler dilakukan menggunakan sebuah kata *tongue twister* yang memiliki banyak kemunculan karakter yang sama.

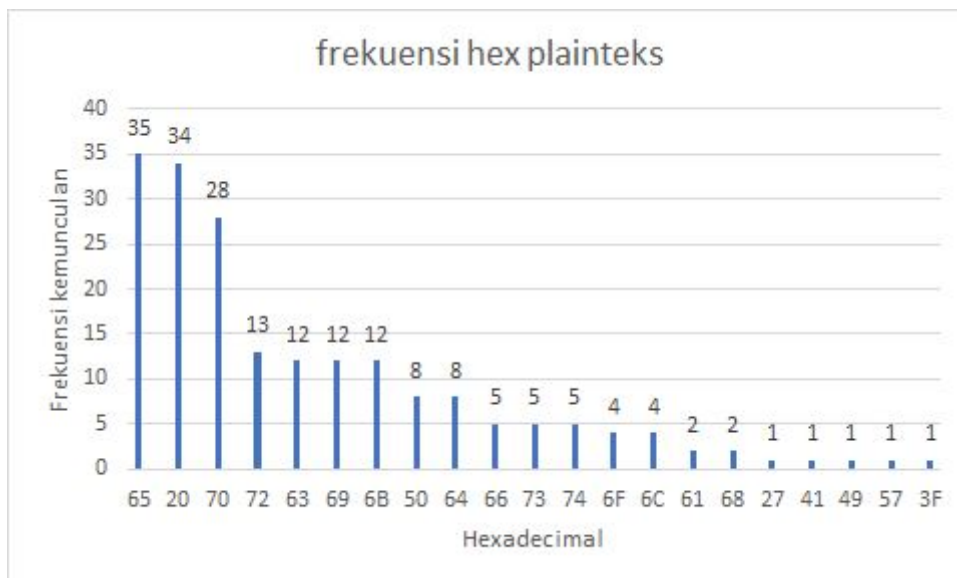
Uji analisis dilakukan menggunakan kunci dan plainteks yang terdapat dibawah. Mode enkripsi yang digunakan adalah CBC

Tabel 6. Tabel enkripsi CBC

kunci	plainteks	cipherteks
KRIPTOGRAFI_RAHASIA	Peter Piper picked a peck of pickled peppers A peck of pickled peppers Peter Piper picked If Peter Piper picked a peck of pickled peppers Where's the peck of pickled peppers Peter Piper picked ?	9594fe35dadcd477015192518d5fddceacf2029c1230e5a1597d9969a52d7d11f95ccdb56e268f9ae3bc50dc8e5d29cd88da88795eeb3f673e969a088b274f3bb989b1407741423c2dd3dea9e5ac5fce35eb1ebc090d796458618810f55844cc9487c967a42b3a937da6d4c4a0d30fbc6348ad0a44a576446fb69166b4d828f9b2e16ccda1b78ca183dda51ed8e6337d3a7cb82893d38c07db4c0ebb7a38042102701cf7c925abfbee4205dae1233474c247a2764ee11f7b9f8665e76d4c915ea5acf753f99ca79aa



Graf 1. Graf frekuensi hex cipherteks



Graf 2. Graf analisis frekuensi plainteks

Dilihat dari graf diatas bahwa jumlah kemunculan karakter pada plainteks dan cipherteks tidak sama, selain itu penyebaran hexadesimal sesuai dengan prinsip konfusi. Sehingga jika dilakukan analisis frekuensi pada cipherteks, plainteks tidak dapat diketahui.

D. Analisis Perubahan Sedikit Karakter pada Kunci

Dengan melakukan perubahan sedikit saja pada kunci, cipherteks yang dihasilkan akan memiliki perbedaan yang signifikan. Hal ini membuat kunci yang dipakai sulit untuk ditebak. Karena perbedaan kecil sekalipun seperti huruf kapital atau angka dapat menyebabkan cipher yang berbeda.

tabel 7 sampai 9 menggambarkan enkripsi dan dekripsi menggunakan mode ECB,CBC, dan Counter jika kunci berbeda hanya pada 1 karakter.

Tabel 7. Tabel enkripsi dan dekripsi CBC

Kunci	plainteks	cipherteks
KRIPTOGRAFI_RAHASIA	Peter Piper picked a peck of pickled peppers A peck of pickled peppers Peter Piper picked If Peter Piper picked a peck of pickled peppers Where's the peck of pickled peppers Peter Piper picked ?	9594fe35dad477015192518 d5fddceacf2029c1230e5a15 97d9969a52d7d11f95ccdb56 e268f9ae3bc50dc8e5d29cd8 8da88795eeb3f673e969a088 b274f3bb989b1407741423c 2dd3dea9e5ac5fce35eb1ebc 090d796458618810f55844c c9487c967a42b3a937da6d4c 4a0d30fbc6348ad0a44a5764 46fb69166b4d828f9b2e16cc da1b78ca183dda51ed8e6337 d3a7cb82893d38c07db4c0e bb7a38042102701cf7c925ab fbee4205dae1233474c247a2 764ee11f7b9f8665e76d4c91 5ea5acf753f99ca79aa
KrIPTOGRAFI_RAHASIA	Peter Piper picked a peck of pickled peppers A peck of pickled peppers Peter Piper picked If Peter Piper picked a peck of pickled peppers Where's the peck of pickled peppers Peter Piper picked ?	aaa88f92ab56a5962c9a45f6 7e8108576edd6bc4eda66d0e d3f80b82c982ef66958ff793 bc1bdc8e447b7d3815cbb30 ccd688beae236126e598896e a3c563b827995366e63c536 8ab9de33477d8c8acc1a9130 24f568f6930bbba8d5e679ba fea636329dec251a9c631743 a565c17692ac151777876e6 5b6072b9122a152cd460d53 f6fad395f899971bff7e9c8f3 df08b8d5d309a7dc0f917e5a 9e9e892b7d4d186f9b44a333 b625cf415eb3c04dab533020 5200540b10ea10fc5469862a add583461dba1bb7c2c

Tabel 8. Tabel enkripsi dan dekripsi ECB

Kunci	plainteks	cipherteks
KRIPTOGRAFI_RAHASIA	Peter Piper picked a peck of pickled peppers A peck of pickled peppers Peter Piper picked If Peter Piper picked a peck of pickled peppers	94d75bef1b84a4bcd42b20d6 2e7bee4eed92bed962eedb7a 31b5ee5911242c881493177 3c058b440d42e8dfb4d87d5 d2fada8c268bd3ceb3df94a1f

	Where's the peck of pickled peppers Peter Piper picked ?	8013dc78ce98f979762683c56f95f275c3addc3c17d0ed2e45218e8b9e9329d59c272265d33894cc5b9dda0dbe71108fa92c875b3511be4fbc3444b8522dc3ed5dcba7411aaa2d82f657ecf09a875cc77502c1937f9c06571b569cd4616f75f663095e33c23e5201839e0fec99a4752ed11f69736e95c05fe66f9dae612e8409ebd691d8b08b261a1f32b157c
KrIPTOGRAFI_RAHASIA	Peter Piper picked a peck of pickled peppers A peck of pickled peppers Peter Piper picked If Peter Piper picked a peck of pickled peppers Where's the peck of pickled peppers Peter Piper picked ?	b25fe8e456935e710b81b99738ed6fde40fd0c573546ce99f6c71229e7dc65bd5bc459a7b709024c9fe8a323b0bdcc0eeb58b3fe7b7ff01372cc0e0dfbfcfb958cadb0d400d8be552468f78e8bb68921829e6e0b3003bac533bb2a6349b83f0c3219e0675495a7e452c8806aec47281e315695d9086c3b8324b33802415b64b05be49972825e0005a91c33f787af6985d435a93e424fd0478172cf1c937920b075aa82188579e1f9ef5748ba3ad55f5f871ef7302662f2247837ccfadd7e9de54dd40ae95e1161b

Tabel 9. Tabel enkripsi dan dekripsi Counter

Kunci	plainteks	cipherteks
KRIPTOGRAFI_RAHASIA	Peter Piper picked a peck of pickled peppers A peck of pickled peppers Peter Piper picked If Peter Piper picked a peck of pickled peppers Where's the peck of pickled peppers Peter Piper picked ?	88ea91d894277cc9b193f6af ee93009e1937e2c8a135fd18aa4bb4af0a1c5cf8e1aef03ec4913d1656895ca2dcecc28768c309d6c5046c4ae53cb3e740ab5d52372c81b0fe27eab33bfdd7f84dc4dec46afdf2c4678073cad908e4a97f605c4c48440315709b96b6522ccc8d4daed827de06fbc5aaf6a1ccea0d2584502e8dd1f31bf2dce6582cbdb05d951c22da5421daa2732c9498e88392bcacdd4831690deb15bd5ee9639ac39836b224911dd86ba4794c19e3e10cce5fb999eb2699aa7

		61ed2b5a0e1951f8a
KrIPTOGRAFI_RAHASIA	Peter Piper picked a peck of pickled peppers A peck of pickled peppers Peter Piper picked If Peter Piper picked a peck of pickled peppers Where's the peck of pickled peppers Peter Piper picked ?	9487ae09c3a536c65018405c bffcfa4ce52642e059aa653b 7df4087940be0be9e661fd90 438f75c31d769538195d8f50 ba13954cfe630951669ac20a 5f7692f70a618eb0cac17ed9 17085229f9db0daa0c6bcaffe a064aa1b5067b0fd48eb85f0 ced2af60057e0ec6109ee3bc bdf9398a12449179faf3445a 9bdc334f26c994497c17f832 e9caae9d210dbac5d3cd8310 047b5be7f42c9476ef8e224f b9922653094d10b3b2d5c81 ba0ace02b7faf974db916b56 bb090a4f8de550a489a8748f 2c1148317b75ece

E. Analisis Perubahan Sedikit Karakter pada Plainteks

Pada algoritma Traveler perubahan sedikit karakter pada plainteks dapat menghasilkan perubahan yang signifikan pada cipherteks. Hal ini menghasilkan isi plainteks yang sulit ditebak.

tabel 10 sampai 12 menggambarkan enkripsi dan dekripsi menggunakan mode ECB,CBC, dan Counter jika 1 karakter pada plainteks diubah.

Tabel 10. Tabel enkripsi dan dekripsi CBC

Kunci	plainteks	cipherteks
KRIPTOGRAFI_RAHASIA	Peter Piper picked a peck of pickled peppers A peck of pickled peppers Peter Piper picked If Peter Piper picked a peck of pickled peppers Where's the peck of pickled peppers Peter Piper picked ?	9594fe35dadca77015192518 d5fddceacf2029c1230e5a15 97d9969a52d7d11f95ccdb56 e268f9ae3bc50dc8e5d29cd8 8da88795eeb3f673e969a088 b274f3bb989b1407741423c 2dd3dea9e5ac5fce35eb1ebc 090d796458618810f55844c c9487c967a42b3a937da6d4c 4a0d30fbc6348ad0a44a5764 46fb69166b4d828f9b2e16cc da1b78ca183dda51ed8e6337 d3a7cb82893d38c07db4c0e bb7a38042102701cf7c925ab fbee4205dae1233474c247a2 764ee11f7b9f8665e76d4c91 5ea5acf753f99ca79aa
KRIPTOGRAFI_RAHASIA	Peter Paper picked a peck of	1e462e0f83bcae6ad42b20d6

	pickled peppers A peck of pickled peppers Peter Piper picked If Peter Piper picked a peck of pickled peppers Where's the peck of pickled peppers Peter Piper picked ?	2e7bee4eed92bed962eedb7a 31b5ee5911242c881493177 3c058b440d42e8dfb4d87d5 d2fada8c268bd3ceb3df94a1f 8013dc78ce98f979762683c5 6f95f275c3addc3c17d0ed2e 45218e8b9e9329d59c27226 5d33894cc5b9dda0dbe7110 8fa92c875b3511be4fbc3444 b8522dc3ed5dcb7411aaa2d 82f657ecf09a875cc77502c1 937f9c06571b569cd4616f75 f663095e33c23e5201839e0f ec99a4752ed11f69736e95c0 5fe66f9dae612e8409ebd691 d8b08b261a1f32b157c
--	---	---

Tabel 11. Tabel enkripsi dan dekripsi ECB

Kunci	plaintexts	cipherteks
KRIPTOGRAFI_RAHASIA	Peter Piper picked a peck of pickled peppers A peck of pickled peppers Peter Piper picked If Peter Piper picked a peck of pickled peppers Where's the peck of pickled peppers Peter Piper picked ?	9594fe35dadca77015192518 d5fddeacaf2029c1230e5a15 97d9969a52d7d11f95ccdb56 e268f9ae3bc50dc8e5d29cd8 8da88795eeb3f673e969a088 b274f3bb989b1407741423c 2dd3dea9e5ac5fce35eb1ebc 090d796458618810f55844c c9487c967a42b3a937da6d4c 4a0d30fbc6348ad0a44a5764 46fb69166b4d828f9b2e16cc da1b78ca183dda51ed8e6337 d3a7cb82893d38c07db4c0e bb7a38042102701cf7c925ab fbee4205dae1233474c247a2 764ee11f7b9f8665e76d4c91 5ea5acf753f99ca79aa
KRIPTOGRAFI_RAHASIA	Peter Paper picked a peck of pickled peppers A peck of pickled peppers Peter Piper picked If Peter Piper picked a peck of pickled peppers Where's the peck of pickled peppers Peter Piper picked ?	2038337cb7381cbfa9c469be 8058c4113a59fb1408c62e5b 8d5990081f2cc92a69ec3dd9 3474b6b705d78cf17665e86a 6ee4a3d0e1b59e5563eb9bb3 e25833987292f0ba334ae91f d5e58a4852606f29baa33401 1108fabb16b210169619a5ea 009a6fe854a402cacbad8e75 fa17c8f8ae432aaa7e2f8bede b5cadf689364c9fab43ee0d6 e8e6924e54aa8fd72fd7e758 7bdcd581d1bd62f72e9e9ba7

		53df538febcb62ffa5f5c91b fce3122cde200370dfeefb3b8 2f4c17a5c165b0106d92a25e b401b7447bbb9
--	--	--

Tabel 12. Tabel enkripsi dan dekripsi Counter

Kunci	plainteks	cipherteks
KRIPTOGRAFI_RAHASIA	Peter Piper picked a peck of pickled peppers A peck of pickled peppers Peter Piper picked If Peter Piper picked a peck of pickled peppers Where's the peck of pickled peppers Peter Piper picked ?	9594fe35dadcd477015192518 d5fddceacf2029c1230e5a15 97d9969a52d7d11f95ccdb56 e268f9ae3bc50dc8e5d29cd8 8da88795eeb3f673e969a088 b274f3bb989b1407741423c 2dd3dea9e5ac5fce35eb1ebc 090d796458618810f55844c c9487c967a42b3a937da6d4c 4a0d30fbc6348ad0a44a5764 46fb69166b4d828f9b2e16cc da1b78ca183dda51ed8e6337 d3a7cb82893d38c07db4c0e bb7a38042102701cf7c925ab fbee4205dae1233474c247a2 764ee11f7b9f8665e76d4c91 5ea5acf753f99ca79aa
KRIPTOGRAFI_RAHASIA	Peter Paper picked a peck of pickled peppers A peck of pickled peppers Peter Piper picked If Peter Piper picked a peck of pickled peppers Where's the peck of pickled peppers Peter Piper picked ?	88ea91d894277cc1b193f6af ee93009e1937e2c8a135fd18 aa4bb4af0a1c5cf8e1aef03ec 4913d1656895ca2dcecc2876 8c309d6c5046c4ae53cb3e74 0ab5d52372c81b0fe27eab33 bfdd7f84dc4dec46afdf2c467 8073cad908e4a97f605c4c48 440315709b96b6522ccc8d4 daed827de06fbc5aaf6a1ccea 0d2584502e8dd1f31bf2dce6 582cbdb05d951c22da5421d aa2732c9498e88392bcacdd4 831690deb15bd5ee9639ac3 9836b224911dd86ba4794c1 9e3e10cce5fb999eb2699aa7 61ed2b5a0e1951f8a

5. Kesimpulan dan Saran

Dapat disimpulkan dari percobaan-percobaan dan hasil analisis di atas, bahwa algoritma Traveler yang dibuat oleh penulis kuat dalam melawan teknik-teknik kriptanalisis. Hal ini dikarenakan block cipher yang telah dibuat oleh penulis dapat menggunakan kunci dengan panjang apapun, memenuhi semua prinsip perancangan block cipher, dan memiliki fungsi f pada jaringan feistel yang cukup kompleks.

6. Referensi

- [1] <http://informatika.stei.itb.ac.id/~rinaldi.munir/> diakses pada 22 Oktober 2020, 20.00
- [2] https://www.tutorialspoint.com/cryptography/feistel_block_cipher.htm diakses pada 22 Oktober 2020, 20.53
- [3] <https://thycotic.force.com/support/s/article/Calculating-Password-Complexity#:~:text=Now%20lets%20assume%20you%20use,%2F%202%2C%20or%201.44%20years.> diakses pada 22 Oktober 2020, 22.34

7. Acknowledgement

Penulis berterima kasih kepada Tuhan Yang Maha Esa dengan berkat dan rahmatnya penulis dapat menyelesaikan makalah ini dalam tenggat waktu yang sesuai. Penulis tidak lupa juga menyampaikan terima kasih yang sebesar-besarnya kepada dosen mata kuliah IF4020, Bapak DR. Ir. Rinaldi Munir, tanpa bimbingannya dalam mengajarkan mata kuliah tersebut, penulis tidak mungkin dapat mengerjakan makalah ini dengan baik. Selain itu penulis juga ingin menyampaikan terima kasih sebesar besarnya kepada kedua orang tua penulis, yang selalu mendukung penulis menempuh masa-masa sulit kuliah. Penulis juga turut berterima kasih kepada teman-teman penulis yang telah memberikan penulis inspirasi dalam mengerjakan algoritma ini.