# ConvBC : Convolutional Block Cipher

Daniel Pintara
*Teknik Informatika / Sekolah Teknik Elektro dan Informatika*
*Institut Teknologi Bandung*
*Bandung, Indonesia*
13515071@std.stei.itb.ac.id

Dery Rahman Ahaddienata
*Teknik Informatika / Sekolah Teknik Elektro dan Informatika*
*Institut Teknologi Bandung*
*Bandung, Indonesia*
13515097@std.stei.itb.ac.id

**Abstak – Algoritma *block cipher* merupakan algoritma penyandian yang beroperasi pada mode bit secara blok per blok. ConvBC merupakan algoritma *block cipher* simetri yang beroperasi pada ukuran blok 128 bit dengan menggunakan teknik *confusion* dan *diffusion*. ConvBC juga menggunakan jaringan feistel, dengan jumlah putaran sebanyak 24 kali. Teknik pembangkitan *internal key* pada ConvBC menggunakan operasi konvolusi sederhana yang terinspirasi dari *Convolutional Neural Network* dimana sebagian *key* digunakan sebagai kernel yang akan mengkonvolusi sebagian *key* yang lain. ConvBC dapat dioperasikan pada mode ECB, CBC, CFB, OFB, dan mode *CRT*.**

***Kata kunci – block, cipher, convolution, feistel, ECB, CRT***

## I. Pendahuluan

Sebelum kriptografi modern seperti *cipher* alir dan *cipher* blok ditemukan, kriptografi pada awalnya hanya menggunakan operasi yang sangat sederhana. Teknik yang sering digunakan pada kriptografi klasik menggunakan teknik substitusi dan transposisi [1]. Contohnya seperti Caesar *cipher* yang menggunakan teknik substitusi. Namun, dalam perkembangannya, kriptografi klasik sangat mudah dipecahkan dengan teknik-teknik kriptanalisis seperti analisis frekuensi, metode *kasiski*, *known-plaintext attack*, dan lain sebagainya. Sehingga kriptografi klasik tidak dapat digunakan untuk penyandian pada masa kini.

Dalam perkembangannya kebutuhan kriptografi untuk keamanan pesan pada komputer digital mendorong lahirnya kriptografi modern [2]. Pada dasarnya kriptografi modern menggunakan teknik substitusi dan transposisi, namun sangat kompleks sehingga sukar dipecahkan. Selain itu kriptografi modern beroperasi pada mode bit, dibanding dengan kriptografi klasik yang hanya beroperasi pada mode karakter saja. Sehingga kombinasi yang dihasilkan dapat lebih banyak. Pada kriptografi modern, data direpresentasikan dalam biner, sehingga dapat dilakukan operasi xor pada tiap bitnya.

Terdapat 2 kategori algoritma cipher berbasis bit, yaitu *cipher* alir dan *cipher* blok [2]. Pada *cipher* alir enkripsi maupun dekripsi dilakukan pada bit tunggal, sedangkan *cipher* blok dilakukan pada blok bit. Berbagai jenis algoritma ditemukan dari kombinasi berbagai teknik dan metode yang ada, seperti DES, TripleDES, AES, RC4, A5, dan sebagainya. Tingkat keamanannya dan penggunaannya juga berbeda-beda. Dalam *cipher* blok, untuk menghasilkan tingkat keamanan yang cukup tinggi dapat menggunakan prinsip *confussion* dan *diffusion* seperti yang dikemukakan oleh Claude Shannon [2].

Teknik perancangan *cipher* blok dapat diimplementasikan dengan berbagai macam cara. Contohnya pada algoritma DES. DES menggunakan prinsip *confusion* dengan menggunakan matriks substitusi bernama *S-box* dimana input 6-bit disubstitusi dengan 4-bit. Sedangkan untuk prinsip *diffusion*nya menggunakan *P-box* atau *permutation box*. DES dioperasikan dengan jaringan *feistel*.

ConvBC menggunakan prinsip *confussion* dan *diffusion* didalam fungsi jaringan feistelnya, sehingga dapat menambah kekuatan keamanannya. Putaran yang digunakan pada ConvBC sejumlah 24 kali. ConvBC menggunakan operasi konvolusi sederhana ketika melakukan pembangkitan *internal key* yang terinspirasi dari *Convolutional Neural Network*.

## II. Dasar Teori

Bab ini akan membahas mengenai dasar teori kriptografi yang menjadi bagian dalam perancangan algoritma ConvBC, antara lain *cipher* blok, jaringan feistel, prinsip *confussion* dan *diffusion* Shannon, dan *convolution kernel*.

### *A. Cipher* Blok

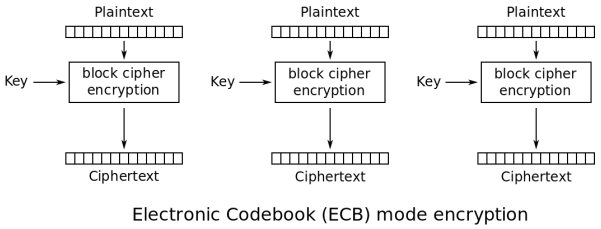
Dalam algoritma *cipher* blok bit-bit *plain text* dibagi kedalam blok-blok bit dengan panjang yang sama. Jika blok terakhir memiliki panjang yang tidak sama, maka dapat ditambahkan *padding bits* yang bernilai 0. Contohnya untuk blok berukuran 8 bit maka *plain text* dengan panjang 12 bit

akan mempunyai 2 blok dimana blok terakhir menggunakan *padding bits*.

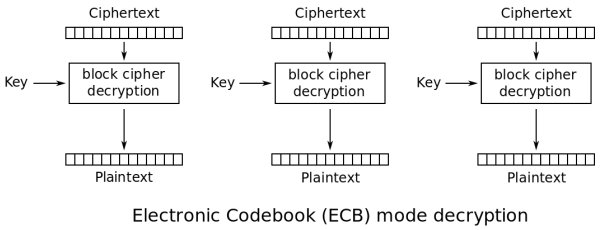
Mode operasi yang digunakan pada *cipher* blok berkaitan dengan bagaimana blok dioperasikan didalam fungsi enkripsi (E) maupun dekripsi (D). Terdapat 5 buah mode operasi pada *cipher* blok, mode operasi tersebut antara lain adalah ECB, CBC, CFB, OFB, dan CTR.

a) *Electronic Code Book* (ECB)

ECB merupakan mode yang paling sederhana diantara mode lainnya. ECB dioperasikan secara independen pada setiap bloknya. Sehingga ECB dapat diparalelkan karena antara 1 blok dengan blok lain tidak ada keterhubungan. Perubahan bit pada enkripsi maupun dekripsi hanya akan mempengaruhi pada blok yang berkaitan saja.




**Gambar 1**. Enkripsi pada ECB



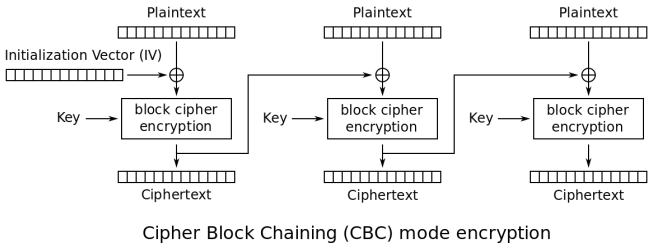

**Gambar 2.** Dekripsi pada ECB

Oleh karena blok-blok pada mode ECB tidak berkaitan antara satu dengan yang lainnya, maka penyerang dapat mengubah 1 atau beberapa blok pada *cipher text* yang dianggap krusial. Ketika dilakukan dekripsi bagian blok lain akan tetap dapat terdekripsi dengan baik, sehingga penerima tidak sadar bahwa *cipher text* tersebut telah diubah.

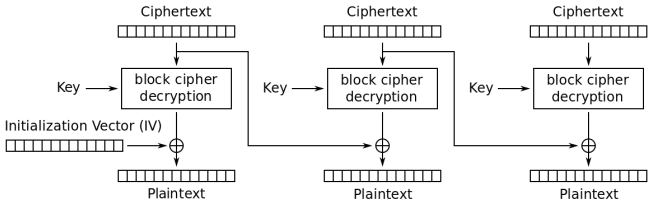
a) *Cipher Block Chaining* (CBC)

Pada mode CBC, blok-blok memiliki ketergantungan antara satu dengan yang lainnya. Berbeda dengan ECB, pada CBC perubahan 1 bit saja pada *plain text* dapat menghasilkan *cipher text* yang jauh berbeda.

CBC membuat ketergantungan antara 1 blok dengan blok selanjutnya. Hasil enkripsi pada blok sebelumnya akan dilakukan operasi xor dengan *plain text*nya. Kemudian kasil operasi xor tersebut menjadi masukan untuk fungsi enkripsi yang akan menghasilkan *cipher text*. Untuk enkripsi blok *plain text* pertama (P1) diperlukan *cipher* blok semu (C0) disebut sebagai *initialization vector* (IV).




**Gambar 3.** Enkripsi pada CBC

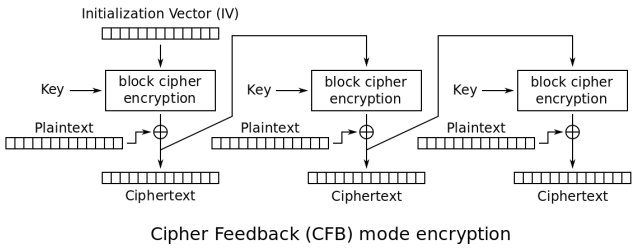



**Gambar 4.** Dekripsi pada CBC

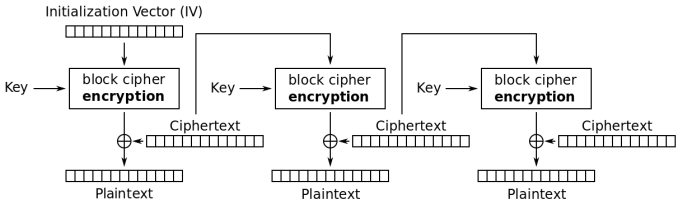
Pada mode CBC, kesalahan dekripsi pada 1 blok hanya akan mempengaruhi blok selanjutnya. Berbeda dengan enkripsi, pada enkripsi, kesalahan 1 blok akan berpengaruh pada blok-blok selanjutnya hingga akhir.

b) *Cipher Feedback* (CFB)

Berbeda dengan mode sebelumnya, CFB menggunakan *cipher text* sebelumnya sebagai masukan untuk fungsi enkripsi. Kemudian hasil dari enkripsi ini akan dilakukan operasi xor dengan *plain text*nya menjadi *cipher text*. Seperti pada CFB, untuk melakukan enkripsi blok *plain text* pertama (P1) diperlukan *cipher* blok semu (C0) disebut sebagai *initialization vector* (IV).




**Gambar 5.** Enkripsi pada CFB



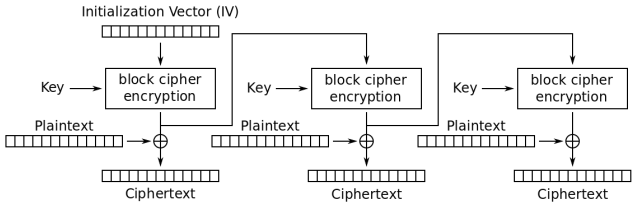

Cipher Feedback (CFB) mode decryption

**Gambar 6.** Dekripsi pada CFB

Kesalahan dekripsi pada 1 blok hanya akan mempengaruhi blok selanjutnya, namun pada enkripsi, kesalahan 1 blok akan berpengaruh pada blok-blok *cipher text* yang berkoresponsen dan blok-blok *cipher text* selanjutnya.

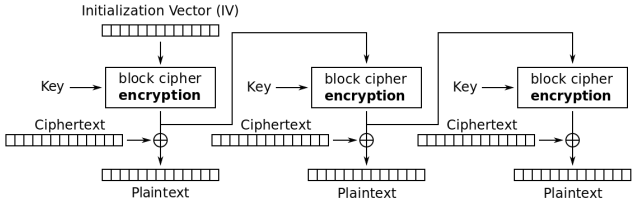
c) *Output Feedback* (OFB)

Mode OFB pada dasarnya sama seperti CFB. Hanya saja *cipher text* sebelumnya yang menjadi masukan pada fungsi enkripsi merupakan cipher text yang belum dilakukan operasi xor dengan *plain text*nya.




Output Feedback (OFB) mode encryption

**Gambar 7.** Enkripsi pada OFB

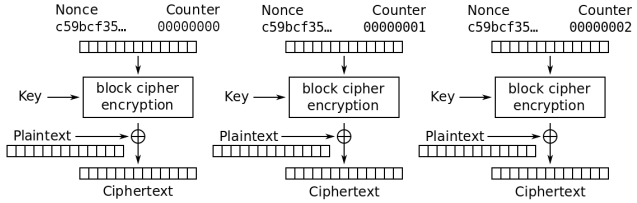



Output Feedback (OFB) mode decryption

**Gambar 8.** Dekripsi pada OFB

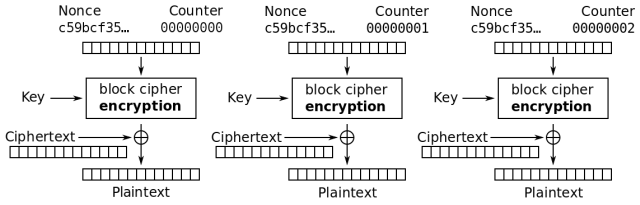
d) *Counter* (CTR)

Pada mode CTR, masukan dari fungsi enkripsi merupakan counter. Kemudian hasilnya akan dilakukan operasi xor dengan *plain text*nya.




Counter (CTR) mode encryption

**Gambar 9.** Enkripsi pada CTR

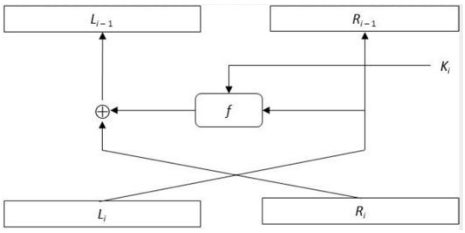



Counter (CTR) mode decryption

**Gambar 10.** Dekripsi pada CTR

*B.* Jaringan *Feistel*

Jaringan *feistel* pertama kali diperkenalkan oleh Horst Feistel. Dalam proses enkripsi maupun dekripsi jaringan Feistel memiliki keunggulan, dimana model ini bersifar reversible untuk proses enkripsi maupun dekripsi. Sehingga tidak perlu membuat model yang berbeda antara enkripsi maupun dekripsi. Contoh algoritma yang menggunakan jaringan *feistel* adalah DES.




**Gambar 11.** Jaringan *feistel*

Jaringan *feistel* akan dioperasikan berulang-ulang sebanyak putaran yang telah didefinisikan. Sebelum *plain text* masuk ke dalam jaringan *feistel, plain text* dibagi menjadi 2 bagian dengan panjang yang sama.

1) Pada putaran ke-i, jaringan *feistel* akan menerima bagian $L_{i-1}$ dan $R_{i-1}$.
2) Selanjutnya, $R_{i-1}$ dan key $K_i$ akan dimasukan kedalam fungsi f.
3) Hasilnya dilakukan operasi xor dengan $L_{i-1}$. Hasil ini akan menjadi $R_i$ sebagai masukan dari putaran selanjutnya

4) $R_{i-1}$ akan menjadi $L_i$ sebagai masukan dari putaran selanjutnya

*B.* Prinsip *confusion* dan *diffusion* Shannon

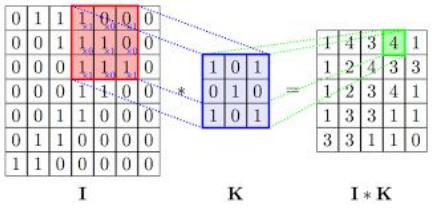
Untuk menghasilkan keamanan *cipher* blok yang kuat diperlukan prinsip *confusion* dan *diffusion*. Pinsip ini dikemukakan oleh Claude Shannon dalam publikasinya yang berjudul "*Communication Theory of Secrecy Systems*" pada tahun 1949. Dengan menggunakan kedua prinsip ini proses analisis statistik menjadi lebih susah.

*Confusion* merupakan prinsip dimana menyembunyikan hubungan antara *plain text, cipher text*, dan kuncinya. Caranya adalah dengan membuat keterhubungan antara kunci dan *cipher text* serumit mungkin. Salah satu algoritma yang dapat mendukung prinsip ini adalah substitusi. Contohnya adalah S-box pada algoritma DES.

*Diffusion* merupakan prinsip dimana perubahan pada 1 bit *plain text* dapat memengaruhi sebanyak mungkin *cipher text*. Mode CBC dan CFB menggunakan prinsip ini. Sedangkan pada algoritma DES, prinsip *diffusion* ini direalisasikan dengan menggunakan operasi permutasi.

*C. Convolution Kernel*

Secara sederhana sebuah matriks dapat dilakukan operasi konvolusi dengan menggunakan *convolution kernel*. Misalnya, untuk matriks A berdimensi 7x7 dan *convolution kernel* berdimensi 3x3, operasi yang dilakukan adalah dengan menggeser *convolution kernel* pada matriks A. Setiap pergeseran dilakukan operasi perkalian kemudian dijumlahkan menghasilkan suatu nilai yang dijadikan output. Ilustrasi dari operasi ini dapat dilihat pada gambar 12.




**Gambar 12.** Operasi *convolution matrix* [3]

Operasi konvolusi inilah yang nantinya akan digunakan untuk membangkitkan *internal key* pada setiap putarannya.

## III. Rancangan Algoritma

Algoritma ConvBC menggunakan jaringan *feistel* dengan jumlah putaran sebanyak 24 kali. Ukuran blok yang digunakan sepanjang 128 bit.

*A.* Pembangkitan Kunci Putaran

Dalam pembangkitan kunci internal untuk setiap putaran, ConvBC pertama-tama kunci masukan dari pengguna akan dibagi menjadi 2 bagian sama panjang. Jika panjang kunci masukan ganjil maka akan di*padding* dengan menambahkan byte 0 pada bagian belakang kunci. Pengguna dapat memasukan kunci dengan panjang yang berbeda-beda.

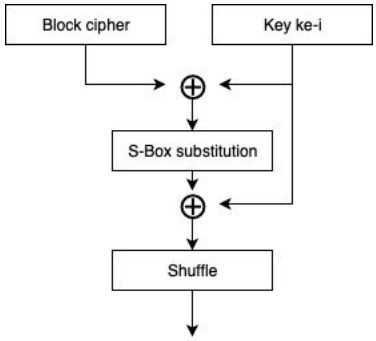
Selanjutnya, ConvBC akan melakukan hash pada 2 bagian tersebut dengan menggunakan MD4. MD4 dipilih karena lebih sederhana dan lebih cepat daripada MD5. Bagian yang dihash ini akan menghasilkan hash dengan panjang 16 bytes (128 bit) masing-masing.

| **Ekspansi kunci** | |
|---|---|
| Kunci : "makanayamgoreng" | |
| Kunci+padding : "makanayamgoreng\x00" | |
| Kunci kiri: "makanaya"<br>Kunci kanan: "mgoreng\x00" | |
| Hash kunci kiri:<br>65fb6a0a0f6d6d5382f79364a3bc7a35<br>Hash kunci kanan:<br>**f242cb543ca49b114b183ed5c83c3816** | |
| Hash kunci kiri 16 byte dijadikan 25 byte, dengan cara mengambil 9 bytes awal pada hash kunci kanan, kemudian dibuat matrix 5x5. Matrix ini dinamakan matrix data. | 65 fb 6a 0a 0f<br>6d 6d 53 82 f7<br>93 64 a3 bc 7a<br>35 **f2 42 cb 54**<br>**3c a4 9b 11 4b** |
| Hash kunci kanan 16 byte dijadikan 4 byte, dengan cara mengambil 4 byte paling kanan, kemudian dibuat matrix 2x2 byte. Matrix ini dinamakan matrix kernel. | **c8 3c**<br>**38 16** |
| Hasil konvolusi antara matrix data dengan matrix kernel akan menghasilkan matrix berukuran 4x4. Matrix ini akan diflatten menjadi array yang akan dijadikan kunci internal pada step ke i | f2 ea 7c fe<br>74 7e 20 10<br>ac f0 4a 18<br>58 ba 82 72 |
| Kemudian hasil kunci internal pada step ke i ini akan ditambahkan padding sehingga ukurannya kembali menjadi 5x5. | f2 ea 7c fe 74<br>7e 20 10 ac f0<br>4a 18 58 ba 82<br>72 00 00 00 00<br>00 00 00 00 00 |
| Kemudian dilakukan operasi xor dengan matrix data 5x5 sebelumnya. Hasil ini dijadikan matrix data untuk proses konvolusi untuk kunci internal selanjutnya. | 65 fb fa 0a 0f<br>6d 9f b9 fe 09<br>e7 1a 83 ac d6<br>c5 b8 5a 93 ee<br>be d6 9b 11 4b |

Konvolusi dilakukan hingga 24 kali, sehingga menghasilkan 24 kunci internal yang nantinya akan digunakan pada jaringan feistel.

*B.* Fungsi pada jaringan *feistel*.

Operasi-operasi yang dibuat untuk fungsi pada jaringan *feistel* antara lain menggunakan S-box dan *shuffle* karakter. S-box yang digunakan merupakan Rijndael S-box yang sama digunakan pada algoritma AES. Sedangkan *shuffle* menggunakan *pseudo random* dengan *seed* yang diperoleh dari *key*.




**Gambar 13.** Fungsi *internal* jaringan *feistel*

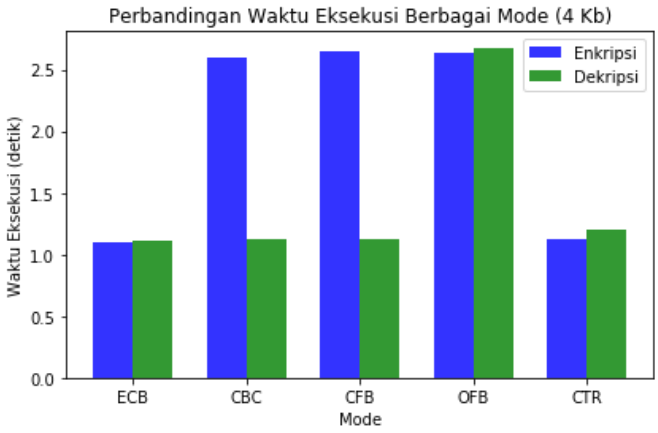
Operasi s-box dapat diimplementasikan dengan cara menggunakan tabel substitusi oleh Rijndael. Tabel substitusi Rijndael itu sendiri digunakan oleh karena tabel substitusi Rijndael dirancang sedemikian rupa sehingga cipherteks yang dihasilkan menjadi lebih sulit untuk dilakukan *cryptanalysis*.

Operasi *shuffle* merupakan operasi mengacak masukan berdasarkan dengan *key* yang dimasukkan. Teknik *shuffle* dilakukan dengan memanfaatkan algoritma Mersenne Twister. Pemilihan algoritma ini didasari karena algoritma ini cukup cepat dan menghasilkan distribusi nilai *random* yang baik.

## IV. Pengujian dan Analisis

*A.* Analisis kecepatan operasi pada *block cipher* pada mode EBC, CBC, CFB, OFB, dan CTR

Pengujian kecepatan eksekusi *block cipher* pada berbagai macam mode dilakukan dengan cara melakukan proses enkripsi dan dekripsi pada data sampel berupa yang memiliki ukuran sebesar 4 kb. Enkripsi dilakukan pada prosesor Intel® Processor 5Y10 CPU 4 x 0.80 GHz dengan kemampuan akselerasi sampai dengan 2.0 GHz.

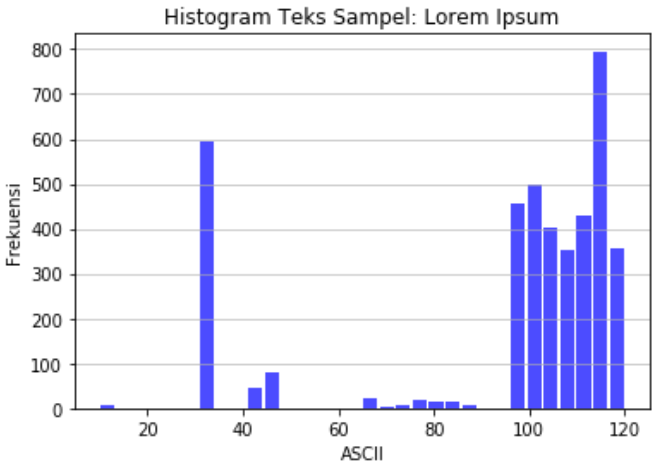



**Gambar 14.** Bagan perbandingan waktu eksekusi

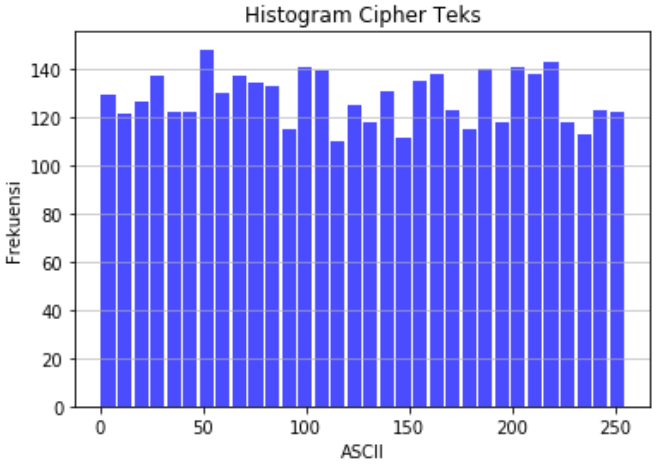
Pada gambar 14, terdapat bahwa mode ECB dan CTR menghasilkan performa yang lebih cepat, baik dalam operasi enkripsi maupun dalam operasi dekripsi. Hal ini terjadi karena proses enkripsi atau dekripsi satu blok terhadap blok yang lain tidak saling bergantung, sehingga paralelisasi dapat dilakukan. Sedangkan pada mode CBC dan CFB, operasi enkripsi lebih lambat dibandingkan dengan operasi dekripsi. Hal ini disebabkan oleh karena tahapan enkripsi pada blok selanjutnya bergantung pada tahapan enkripsi pada blok sebelumnya, sehingga tahapan enkripsi tidak dapat diparalelisasikan. Namun, pada tahapan dekripsi itu sendiri tidak saling bergantung tiap blok. Operasi OFB merupakan mode yang lebih lambat, baik tahapan enkripsi maupun tahapan dekripsi. Hal ini disebabkan karena adanya ketergantungan pada blok sebelumnya saat melakukan operasi pada blok selanjutnya, baik operasi enkripsi maupun operasi dekripsi.

*B.* Analisis distribusi karakter pada cipher teks

Pengujian distribusi karakter pada cipherteks dilakukan dengan cara menggunakan teks Lorem Ipsum yang dibuat dengan menggunakan Lorem Ipsum generator sebanyak 4 kb. Teks tersebut dianalisis distribusi frekuensi karakternya dan ditampilkan dengan menggunakan diagram histogram. Selanjutnya, teks tersebut dienkripsi dan dianalisis kembali distribusi frekuensi karakternya dan dibandingkan dengan distribusi frekuensi dari plainteks.




**Gambar 15.** Histogram analisis frekuensi plainteks




**Gambar 14.** Histogram analisis frekuensi cipherteks

Melalui analisis frekuensi yang dilakukan, terlihat bahwa cipherteks yang dihasilkan merupakan cipherteks dengan distribusi yang cukup rata. Hal ini dapat teknik cryptanalisis berdasarkan analisis frekuensi menjadi sulit untuk dilakukan untuk cipherteks yang dienkripsikan menggunakan ConvBC.

*B.* Analisis *avalanche effect* dari perbedaan plain teks

Pengujian terhadap *avalance effect* yang dihasilkan terhadap perbedaan plain teks dilakukan dengan cara melakukan enkripsi teks sebesar satu blok, lalu mengubah satu huruf pada blok itu dan dienkripsi kemudian, lalu dibandingkan hasilnya dengan cipherteks sebelumnya.

| Plain teks | Cipher teks |
|---|---|
| the brown fox jumps over the | c8 3d b0 19 31 c6 89 86 71 80 51 aa c5 c9 6d a2 |
| dog | 1b 5b 28 2d 93 02 63 3b 82 8c 05 51 27 7a 21 bd |
| the brown **c**ox jumps over the dog | **ca 9f b5 df 1c 6a 6a f6 f0 c8 38 92 76 fc 1f 63 50 d0 0d** 2d **e3 49 1b 2d f4 50 bf 29 20 03 20 91** |

Melalui hasil percobaan tersebut, diketahui bahwa kedua teks tersebut memiliki perbedaan sebanyak 96.88 %. Hal ini dapat menyebabkan korelasi terhadap cipher teks dan plain teks menjadi lebih sulit untuk ditemukan sehingga *block cipher* menjadi lebih aman.

Perlu diketahui bahwa apakah *avalanche effect* terjadi hanya pada satu blok saja atau pada banyak blok bergantung pada mode *block cipher* yang digunakan. Jika enkripsi dilakukan dengan menggunakan mode yang mengaitkan satu blok dengan blok lainnya, maka avalanche effect yang terjadi dapat merambat pada blok-blok lainnya.

Berikut ini adalah contoh enkripsi dengan mode EBC:

| Plain teks (*no space*) | Cipher teks |
|---|---|
| 1234 5678 1234 5678 1234 5678 1234 5678 1234 5678 | 52 a8 74 77 63 3b 18 66 21 c6 3b 1c 41 8f 9f 3b 4d 0f ac 6f 90 0c c6 5e f6 08 e1 30 c5 9e 47 dc 52 a8 74 77 63 3b 18 66 21 c6 3b 1c 41 8f 9f 3b 4d 0f ac 6f 90 0c c6 5e f6 08 e1 30 c5 9e 47 dc |
| **!**234 5678 1234 5678 1234 5678 1234 5678 1234 5678 | **b8 80 b5 a2 f5 27 01 ad 1a 50 c8 6a 50 a3 da 2e a2 63 c6 50 09 b0 09 2d 7e 53 ff 37 4b cc d9 4c** 52 a8 74 77 63 3b 18 66 21 c6 3b 1c 41 8f 9f 3b 4d 0f ac 6f 90 0c c6 5e f6 08 e1 30 c5 9e 47 dc |

Pada hasil di atas, terlihat bahwa kedua cipher teks tersebut memiliki berbedaan sejumlah 50%. Mode EBC tidak melanjutkan *Avalanche effect* pada blok-blok berikutnya.

Berikut ini adalah contoh enkripsi dengan mode CBC:

| Plain teks | Cipher teks |
|---|---|
| 1234 5678 1234 | 32 3e ff 98 3c c6 8f e2 |

| | |
|---|---|
| 5678 1234 5678<br>1234 5678 1234<br>5678 1234 5678<br>1234 5678 1234<br>5678 | b9 d2 d7 b1 aa 62 59 5f<br>96 f7 5f 2d 46 cd bf 7c<br>09 57 5e a7 c5 ca cd fd<br>9d 2a 60 dc 43 ed 53 a6<br>ea 87 69 1d a6 0b 3c b5<br>d2 54 ca 9f 4b 1b 1a f3<br>bf ee 82 da 37 04 37 ae |
| **!**234 5678 1234<br>5678 1234 5678<br>1234 5678 1234<br>5678 1234 5678<br>1234 5678 1234<br>5678 | **e0 60** ff **48 da 91 57 c4**<br>**b2 62 fc 0a 5e f7 55 76**<br>**53 34 3e 4b 04 21 d1 57**<br>**1a 76 a7 4d d2 60 02 e7**<br>**c8 0a db 1d e2 c4 59 16**<br>**c8 1f 0b a6 02 43 cb 1b**<br>**be c2 6b dc 99 24 ce 85**<br>**02 71 2f 4f 64 ee e3 45** |

Pada hasil di atas, terlihat bahwa perbedaan kedua cipher teks tersebut adalah 98.43%. *Avalanche effect* yang terjadi pada blok sebelumnya ikut terjadi pada blok-blok setelahnya.

*C.* Analisis *avalanche effect* dari perbedaan cipher teks

Pengujian terhadap *avalance effect* yang dihasilkan dari perbedaan cipher teks dilakukan dengan cara melakukan enkripsi teks sebesar satu blok, dan melakukan dekripsi. Selanjutnya satu huruf pada cipher teks dari blok itu diubah dan didekripsi. Perbandingan dilakukan terhadap kedua plain teks yang dihasilkan.

| **Cipher teks** | **Plain teks** |
|---|---|
| 32 3e ff 98 3c c6 8f e2<br>b9 d2 d7 b1 aa 62 59 5f<br>96 f7 5f 2d 46 cd bf 7c<br>09 57 5e a7 c5 ca cd fd<br>9d 2a 60 dc 43 ed 53 a6<br>ea 87 69 1d a6 0b 3c b5<br>d2 54 ca 9f 4b 1b 1a f3<br>bf ee 82 da 37 04 37 ae | 74 68 65 20 62<br>72 6f 77 6e 20<br>66 6f 78 20 6a<br>75 6d 70 73 20<br>6f 76 65 72 20<br>74 68 65 20 64<br>6f 67<br>*(the brown fox jumps over the dog)* |
| 32 3e ff **00** 3c c6 8f e2<br>b9 d2 d7 b1 aa 62 59 5f<br>96 f7 5f 2d 46 cd bf 7c<br>09 57 5e a7 c5 ca cd fd<br>9d 2a 60 dc 43 ed 53 a6<br>ea 87 69 1d a6 0b 3c b5<br>d2 54 ca 9f 4b 1b 1a f3<br>bf ee 82 da 37 04 37 ae | **72 c6 46 5c 84**<br>**0d 67 93 cd c4**<br>**b4 d7 fb ef b0**<br>**43 b5 73 1d dc**<br>**62 11 1f 1c 7e**<br>**8c d0 ca b6 b4**<br>**45 29** |

Pada hasil di atas, terlihat bahwa perbedaan kedua plain teks tersebut adalah 100%.

## VI. Kesimpulan

Algoritma ConvBC menerapkan operasi konvolusi pada pembangkitan kunci internalnya. ConvBC menerapkan prinsip *confusion* dan *diffusion* dari Shannon, sehingga hasil persebaran statitik dari ConvBC uniform secara menyeluruh. Selain itu ConvBC juga menggunakan jaringan *feistel*, sehingga operasi enkirpsi maupun dekripsi tidak banyak perbedaannya. Fungsi yang terdapat pada jaringan *feistel* menggunakan operasi s-box substitusi dan *shuffle* saja. Hasil pengujian dan analisis untuk algoritma ConvBC mempunyai tingkat keamanan yang cukup tinggi. Secara garis besar, ConvBC mempunyai konfigurasi sebagai berikut:

- Blok yang digunakan sepanjang 128 bit
- Menggunakan jaringan *feistel* dengan 24 kali putaran
- Kunci internal dibangkitkan dengan hash MD4 dan konvolusi sebanyak 24 kali
- Fungsi pada jaringan *feistel* menggunakan operasi substitusi s-box dan *shuffle*


## Ucapan Terimakasih

Penulis mengucapkan terimakasih kepada Tuhan Y.M.E, karena atas berkat karuniaNya kami penulis dapat menyelesaikan makalah ini. Tidak lupa penulis mengucapkan terima kasih kepada Bapak Dr. Ir. Rinaldi Munir selaku dosen mata kuliah kriptografi. Atas berkat ilmu beliau, penulis dapat memahami konsep dari kriptografi dalam pembuatan makalah ini.


## References


[1] Munir, Rinaldi. 2019. Slide Kuliah IF4020 Kriptografi: Pengantar Kriptografi

[2] Munir, Rinaldi. 2019. Slide Kuliah IF4020 Kriptografi: Algoritma Kriptografi Modern

[3] Karpathy, Andrej. (n.d). Convolutional Neural Networks (CNNs / ConvNets). Diakses tanggal 12 Maret 2019. http://cs231n.github.io/convolutional-networks/