# Basit: Algoritma Cipher Blok dengan Menggunakan Fungsi Hash Quark

Jauhar Arifin-13515049[1] Fadhil Imam Kurnia-13515146[2]
*Program Studi Teknik Informatika*
*Sekolah Teknik Elektro dan Informatika*
*Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia*
[1]*jauhararifin10@gmail.com* [2]*fadhilimamk@gmail.com*

**Abstrak—Dalam makalah ini diajukan sebuah algoritma cipher blok (*block cipher*) baru yang menggunakan fungsi hash Quark yang ringan. Algoritma ini dinamakan Basit, yang dapat mengenkripsi data dengan ukuran blok sebesar 64-bit. Keunikan algoritma Basit adalah masukan kunci yang ukurannya tidak ditentukan, pengguna dapat memasukan kunci yang mudah diingatnya. Algoritma Basit juga mudah diimplementasikan karena hanya menggunakan operasi-operasi sederhana, terdapat penggunaan fungsi hash Quark yang ringan dalam algoritma ini. Algoritma Basit juga dapat menghasilkan cipherteks yang sulit untuk dipecahkan dengan menggunakan analisis frekuensi, karena cipherteks yang dihasilkan memiliki kemunculan frekuensi yang relatif sama untuk setiap karakter. Oleh karena itu, Basit cocok digunakan untuk enkripsi data yang memerlukan keamanan yang tinggi.**

**Kata Kunci—Cipher Blok, Hash, Jaringan Fiestel, Quark**

## I. PENDAHULUAN

Salah satu aspek penting dalam komunikasi di era digital adalah masalah keamanan, terutama jika pesan yang dikomunikasikan merupakan pesan rahasia yang tidak boleh diketahui oleh pihak lain. Keamanan menjadi sangat penting karena pada umumnya komunikasi dilakukan melalui saluran publik yang digunakan oleh banyak orang. Bentuk serangan yang terjadi pada proses komunikasi dapat berupa serangan pasif atau serangan aktif. Pada serangan pasif, pihak ketiga berusaha mendapatkan sebanyak banyaknya informasi dengan melakukan penyadapan. Sedangkan pada serangan aktif, pihak penyerang berusaha mengintervensi komunikasi dan ikut mempengaruhi untuk kepentingan penyerang tersebut. Penyerang dapat mengintervensi dengan menghapus atau mengubah pesan, menyisipkan tambahan pesan, atau mengirim ulang pesan yang lama.

Teknik yang dapat digunakan untuk mengamankan pesan adalah dengan menggunakan kriptografi. Pesan yang dikirimkan dapat diubah terlebih dahulu menggunakan algoritma kriptografi tertentu sehingga hanya pengirim dan penerima saja yang dapat membaca pesan tersebut. Seiring dengan berkembangnya teknologi komputer digital, teknik kriptografi juga sudah dikembangkan untuk komunikasi dengan komputer. Kriptografi yang memanfaatkan komputer biasa disebut dengan kriptografi modern. Kriptografi modern memanfaatkan teori matematis dan aplikasi komputer, dengan pengoperasian dalam mode bit atau biner. Penggunaan kriptografi modern sudah sangat luas, teknik tersebut dapat ditemukan dalam sistem perbankan, koneksi internet aman (HTTPS), dan lain sebagianya.

Salah satu metode pada enkripsi digital adalah *block cipher*. *Block cipher* membagi bit-bit plainteks menjadi blok-blok bit dengan panjang yang sama, misalkan satu blok terdiri dari 64 bit. Proses enkripsi yang dilakukan menggunakan kombinasi operasi bit sederhana seperti permutasi dan substitusi, operasi tersebut juga dilakukan berulang-ulang dalam beberapa putaran. Beberapa contoh block cipher yang terkenal diantaranya AES, DES, Blowfish, dan lain sebagainya.

Pembuatan algoritma *block cipher* mengharuskan penentuan ukuran kunci yang akan digunakan untuk proses enkripsi dan dekripsi. Namun hal tersebut dapat membuat pengguna merasa kesulitan untuk menentukan kunci yang mudah diingat dan aman untuk digunakan. Oleh karena itu kami berinisiatif untuk menggunakan fungsi hash pada pemrosesan kunci yang diberikan oleh pengguna. Penggunaan fungsi hash tersebut memungkinkan pengguna untuk menggunakan kunci yang mudah diingat tanpa mengurangi aspek keamanan. Agar proses enkripsi yang dilakukan tetap ringan, kami menggunakan fungsi hash Quark yang dipublikasikan oleh Jean-Philippe Aumasson, Luca Henzen, Willi Meier, dan Mar´ıa Naya-Plasencia pada tahun 2012[1].

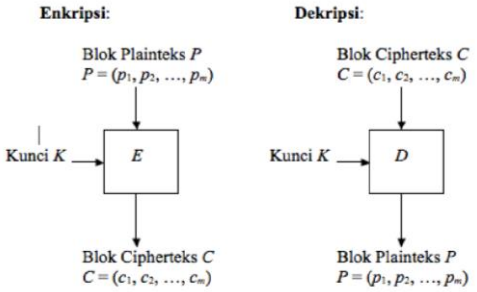
Pada makalah ini pembahasan dasar teori akan dijelaskan pada bagian II, kemudian akan dibahas mengenai rancangan Basit pada bagian III. Pada bagian IV akan dijelaskan hasil percobaan dan analisis dari algoritma *block cipher* Basit yang sudah diimplementasikan. Terakhir pada bagian V akan disimpulkan hasil penelitian yang kami lakukan.

## II. DASAR TEORI

### A. Cipher Blok

Cipher blok merupakan algoritma kriptografi yang beroperasi dengan memroses data dalam satuan blok yang sudah ditentukan. Setiap blok yang diproses dapat berisi sejumlah bit pesan atau byte pesan yang panjangnya sudah ditentukan oleh pembuat cipher blok. Hasil pemrosesan cipher blok adalah cipherteks dengan ukuran blok yang sama seperti panjang blok masukan yang diberikan, atau

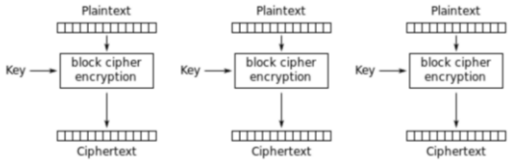
dapat dikatakan bahwa panjang blok cipherteks sama dengan panjang blok plainteks. Skema enkripsi dan dekripsi pada *blok cipher* dapat dilihat pada Gambar 1.




**Gambar 1.** Skema enkripsi dan dekripsi pada *block cipher*

Pada cipher blok dikenal lima mode operasi, yaitu Electronic Code Book (ECB), Cipher Block Chaining (CBC), Cipher Feedback (CFB), Output Feedback (OFB), dan mode counter.

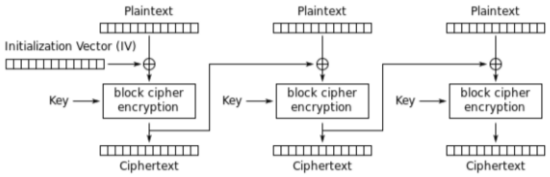
1. Electronic Code Book (ECB)
   Dengan mode ECB, setiap blok dienkripsi dan didekripsi secara independen satu persatu. Setiap blok menjadi input dari suatu fungsi yang akan menghasilkan cipher. Hasil enkripsi untuk suatu blok tidak akan mempengaruhi proses enkripsi blok yang lain.




**Gambar 2.** Operasi *block cipher* dengan mode ECB

   Karena sifat pemrosesannya yang independen, mode ini akan menghasilkan cipher yang sama untuk masukan blok yang sama. Hal tersebut dapat mempermudah pemecahan pesaan menggunakan metode kriptanalis, terutama dengen mendeteksi kemunculan cipher yang sama.

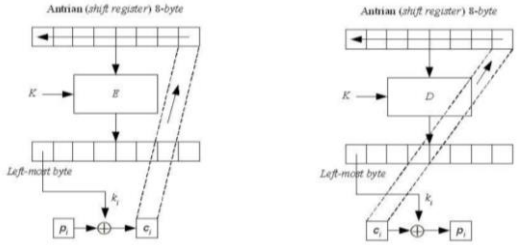
2. Cipher Block Chaining (CBC)




**Gambar 3.** Operasi *block cipher* dengan mode CBC

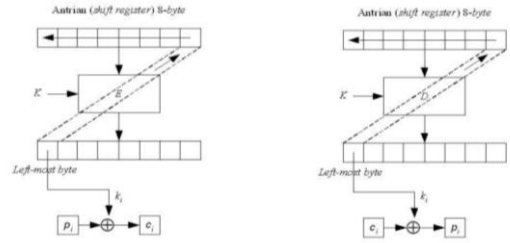
   Menggunakan mode CBC, setiap blok akan dienkripsi dengan memanfaatkan hasil enkripsi (cipher) blok sebelumnya. Cipher yang dihasilkan pada proses enkripsi akan di-XORkan dengan blok selanjutnya, kemudian hasil operasi tersebut akan dienkripsi. Metode ini sering digunakan karena blok yang sama tidak akan dienkripsi menjadi cipher yang sama, sehingga proses kriptanalisis dapat menjadi lebih sulit. Namun jika terdapat kesalahan satu bit saja pada suatu proses enkripsi, maka kesalahan tersebut akan merambat ke pemrosesan blok-blok selanjutnya.

3. Cipher Feedback (CFB)
   Mode ECB dan CBC harus dilakukan jika jumlah bit dalam satu blok sudah lengkap, sehingga proses akan menunggu terlebih dahulu hingga jumlah bit lengkap. Mode CFB berusaha mengatasi kelemahan tersebut dengan melakukan proses dalam unit yang lebih kecil daripada ukuran blok. Ukuran data yang diproses dapat berupa bit per bit, 2 bit, 3 bit, dan sebagainya. Penggunaan mode CFB ini memerlukan struktur antrian (*queue*) yang berukuran sama dengan ukuran blok masukan. Contoh penggunaan mode CFB 8-bit pada blok berukuran 64-bit dapat dilihat pada Gambar 4.




**Gambar 4.** Operasi *block cipher* dengan mode CFB

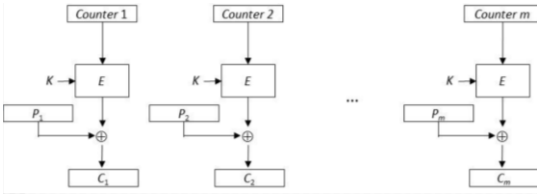
4. Output Feedback (OFB)
   Pada mode OFB, operasi yang digunakan mirip dengan operasi pada mode CFB, namun n-bit dari hasil enkripsi terhadap antrian disalin menjadi elemen posisi paling kanan di antrian. Contoh penggunaan mode OFB 8-bit pada blok berukuran 64-bit dapat dilihat pada Gambar 5.




**Gambar 5.** Operasi *block cipher* dengan mode OFB

5. Mode *counter*
   Mode *counter* diusulkan oleh Diffie dan Hellman pada tahun 1979. Pada mode ini tidak dilakukan proses perantaian (*chaining*) seperti pada mode CBC. Untuk melakukan proses enkripsi, digunakan *counter* berupa blok bit yang ukurannya sama dengan ukuran blok plainteks. Nilai awal *counter* tersebut harus berbeda dari pada setiap blok yang dienkripsi, kemudian nilai *counter* tersebut dinaikan nilainya satu persatu pada tiap proses

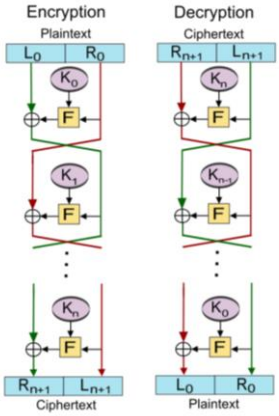
enkripsi. Skema mode *counter* ini dapat dilihat pada Gambar 6.




**Gambar 6.** Operasi *block cipher* dengan mode *counter*

## B. Jaringan Fiestel (*Fiestel Network*)

Jaringan Fiestel merupakan struktur simetris yang digunakan dalam konstruksi Block Cipher. Jaringan ini ditemukan oleh kriptografer Horst Feistel selama pelaksanaan penelitiannya di IBM. Jaringan Fiestel bersifat *reversible*, karena operasi untuk melakukan proses enkripsi dan dekripsi sama, sehingga tidak perlu membuat algoritma baru untuk mendekripsi cipherteks menjadi plainteks. Ilustrasi operasi yang dilakukan pada diagram Fiestel dapat dilihat pada Gambar 7.




**Gambar 7.** Ilustrasi operasi pada jaringan Fiestel

Jaringan Feistel terbentuk dari sejumlah putaran yang terdiri dari opweasi-operasi berulang, seperti permutasi, substitusi, dan operasi aljabar menggunakan XOR. Fungsi yang digunakan pada setiap putaran ini disebut *round function*.

## C. Properti Konfusi dan Difusi

Konfusi (*confusion*) dan difusi (*diffusion*) merupakan dua properti dalam kriptografi yang menjamin keamanan cipherteks dengan menyulitkan proses analisis statistik. Kedua prinsip ini diperkenalkan oleh Claude Shannon pada tahun 1949 dalam makalah yang dipublikasikannya.

Prinsip konfusi bekerja dengan menyembunyikan hubungan apapun yang ada antara plainteks, cipherteks, dan kunci. Hal tersebut dapat direalisasikan dengan menggunakan algoritma substitusi yang kompleks.

Sedangkan prinsip difusi menyebarkan pengaruh satu bit plainteks atau kuncu ke sebanyak mungkin cipherteks. Misalnya jika perubahan dilakukan pada satu bit plainteks maka akan ada banyak bit pada cipherteks yang juga berubah, begitu pula sebaliknya. Kedua prinsip ini merupakan panduan dalam merancang berbagai algoritma kriptografi, dan juga menjadi konsep yang penting dalam merancang fungsi *hash* dan *pseudorandom generator*.

## D. Fungsi Hash Quark

Fungsi hash adalah fungsi yang menghasilkan data keluaran dengan panjang yang selalu sama dari data yang panjangnya sembarang. Hasil keluaran fungsi hash disebut *message digest*. Fungsi hash bersifat satu arah karena kita tidak bisa mengembalikan *message digest* ke data atau string awal. Perubahan sedikit saja dalam data dapat mengakibatkan nilai hash yang berubah drastis. Dalam kriptografi, dikenal beberapa fungsi hash antara lain MD5, SHA-1, dan lain sebagainya.

Salah satu fungsi hash yang ringan adalah Quark. Fungsi hash tersebut membutuhkan kemampuan komputasi yang ringan[1]. Penggunaan fungsi hash Quark dapat diterapkan pada penggunaan teknologi RFID, NFC, dan *smartcard* yang membutuhkan algoritma hash yang cepat dan ringan.

# III. RANCANGAN ALGORITMA

Algoritma *block cipher* Basit merupakan modifikasi algoritma *block cipher* yang memanfaatkan jaringan Fiestel. Besar blok pesan yang diterima adalah 64-bit atau 8 karakter, dan ukuran kuncinya dibebaskan. Panjang kunci dibebaskan karena ada tahap pemrosesan yang menggunakan fungsi hash, sehingga kunci yang digunakan akan selalui menghasilkan kunci baru yang panjangnya sama. Penggunaan fungsi hash juga membuat kunci mudah diingat tanpa mengurangi kekuatan algoritma *block cipher* Basit. Terdapat 14 (empat belas) putaran yang dilakukan pada *block cipher* ini. Secara garis besar, algoritma ini terdiri dari dua bagian utama yaitu tahap pembangkitan kunci internal dan tahap pemrosesan *round function* pada jaringan Fiestel. Proses dalam round function juga menggunakan S-Box serta P-Box yang sudah didefinisikan. Algoritma ini dapat berjalan relatif cepat karena menggunakan operasi bitwise sederhana seperti shift right, operasi XOR, substitusi, dan permutasi.

## A. Pembangkitan Kunci Internal (*subkey*)

Setiap iterasi pada Basit memerlukan kunci yang berbeda-beda, maka dari kunci yang diberikan oleh pengguna perlu dihasilkan 14 kunci internal (*subkey*) yang akan digunakan pada masing-masing putaran. Langkah pembangkitan *subkey* adalah sebagai berikut:

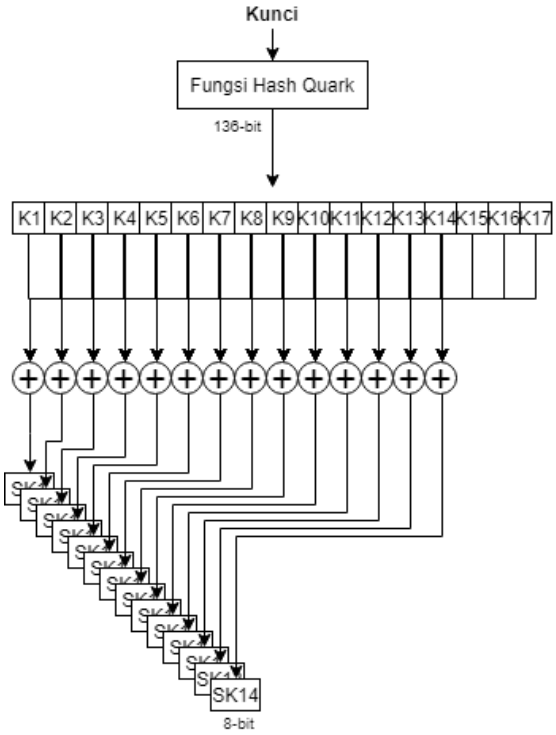
1. Kunci yang diberikan oleh pengguna dimasukan dalam fungsi hash quark yang relatif ringan. Fungsi hash tersebut akan menghasilkan 136-bit kunci baru.
2. Kemudian dari 136-bit kunci baru tersebut akan dibagi-bagi menjadi 17 bagian yang masing-masing sebesar 8-bit.

Misalkan bagian kunci baru ke-n kita sebut dengan notasi Kn, dan subkey ke-n yang dihasilkan pada

pembangkitan kunci kita sebut dengan notasi SKn.

3. Dari 17 bagian yang ada kemudian akan dibentuk 14 subkey dengan menggunakan operasi XOR pada 3 bagian yang bersebelahan. Hasil XOR K1, K2, dan K3 akan menjadi SK1, lalu hasil XOR K2, K3, dan K4, akan menjadi SK2, begitu seterusnya hingga dihasilkan SK1 hingga SK14 yang akan digunakan pada fungsi putaran.

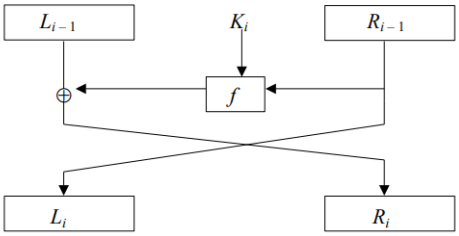
Proses pembangkitan kunci internal ini dapat dilihat pada Gambar 8.




**Gambar 8.** Proses pembangkitan kunci pada Basit

## B. Fungsi Putaran (*round function*)

Fungsi putaran *f* merupakan bagian dalam jaringan Fiestel yang digunakan pada Basit. Jaringan Fiestel yang digunakan pada algoritma ini membagi blok plainteks menjadi dua bagian sama besar yaitu 64-bit pada bagian kanan, dan 64-bit pada bagian kiri. Lalu 64-bit plainteks bagian kanan akan digunakan sebagai input dari fungsi putaran beserta *subkey* yang dihasilkan pada tahap pembangkitan kunci internal. Kemudian hasil fungsi putaran tersebut di-XOR-kan dengan 64-bit plainteks bagian kiri untuk kemudian menjadi plainteks bagian kanan pada putaran selanjutnya. Plainteks bagian kiri pada putaran selanjutnya diambil dari plainteks bagian kanan putaran sebelumnya. Proses tersebut dapat dilihat pada Gambar 9. Terdapat 14 putaran yang akan dilakukan hingga proses enkripsi sebuah blok selesai dilakukan.




**Gambar 9.** Penggunaan jaringan Fiestel pada Basit

Fungsi putaran (*round function*) *f* yang digunakan pada Basit memanfaatkan beberapa operasi bitwise ringan seperti *shift right*, operasi XOR, substitusi, dan permutasi. Operasi substitusi dan permutasi yang dilakukan memanfaatkan S-Box dan P-Box yang sudah didefinisikan sedemikian rupa sehingga properti difusi dapat diperoleh. Urutan proses pada fungsi putaran dalam Basit adalah sebagai berikut:

1. Lakukan operasi XOR terhadap *subkey* dengan 32-bit plainteks bagian kanan yang dimasukan ke dalam fungsi putaran.
2. Lakukan proses substitusi pada hasil yang didapatkan dari operasi XOR sebelumnya. S-Box yang digunakan adalah sebagai berikut:

| |
|---|
| d9 9f c4 6d a8 d2 67 66 51 1e ba d5 b2 a1 95 10 |
| f4 b0 4a 25 a9 ca 4 15 c0 9a 70 2a 49 1d 92 b6 |
| 21 87 f6 c6 b9 ae 83 5f 1a 1f 41 a6 27 5d 6a 3b |
| 1b 5c b8 eb b1 b4 4e 60 2 55 a0 62 76 d8 d1 94 |
| b3 ad f5 77 79 ee 9e 4d 7c f9 90 34 8a 7a 8e 63 |
| c9 89 99 b 1c 73 8d f1 16 6e d7 43 72 17 bb 9b |
| 4c f8 39 2f 20 22 80 ce 18 45 3f 3e 81 e6 e5 56 |
| 82 dc bc d4 7e cd 65 38 5 ab da 6 f2 30 1 36 df |
| 54 bf 78 ef b5 e2 c1 0 86 98 24 96 be fb e0 ff 7 |
| 12 3c 50 61 85 6f 3d a3 cb a cf ea db 5b e9 e7 |
| f7 fe c8 c7 40 84 6b d0 97 3 42 2e 31 2b e8 29 |
| 2d ac ec 13 74 19 8f 8b a2 37 7f f 58 9c a5 cc |
| 75 88 4b e b7 46 f0 bd c 7d 93 8c e1 23 5e 44 |
| d3 4f a4 47 a7 11 e3 6c 26 aa af 9 52 c3 68 ed fc |
| c2 c5 14 de 59 fa d 71 35 9d 2c 57 8 dd 69 5a f3 |
| e4 fd d6 48 64 53 7b 33 28 3a |

3. Lakukan proses cyclic shift right pada 32-bit yang diperoleh dari hasil substitusi dengan aturan sebagi berikut.
    a. Bagi 32-bit hasil proses sebelumnya menjadi 4 bagian, masing-masing bagian memiliki panjang 8-bit.

        Misalkan bagian ke-n disebut dengan menggunakan notasi Pn.

b. Untuk setiap Pn, lakukan *shift right* n (>>n), sehingga

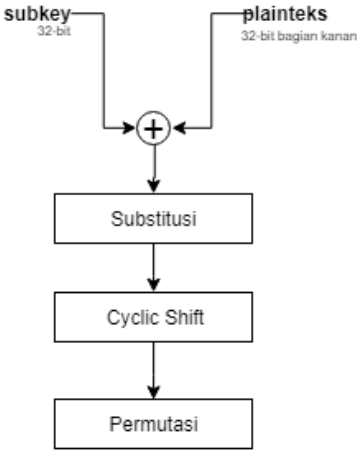
P1 ← P1 >> 1
P2 ← P2 >> 2
P3 ← P3 >> 3
P4 ← P4 >> 4

c. Gabungkan 4 bagian 8-bit tersebut menjadi 32-bit kembali (P1|P2|P3|P4).

4. Lakukan proses permutasi menggunakan P-Box yang sudah didefinisikan sebagai berikut:

```
{39 2 3f 20 12 0 33 15 f b 32 3b 3c 27 1e 2a 2e 38 7 2c 1b 36 1a 1d e 10 3d 1 1c 25 14 26 13 21
3e 2d 2f 30 d 1f 17 35 23 4 19 34 3a a 9 16 6 29 31 8 24 3 c 2b 22 11 28 37 5 18}
{13 2b 24 37 29 10 28 33 d 3c 17 1a 14 15 1f 6 b 30 20 1b 36 12 9 1d 31 21 3e 1c 3f 2a 7 1e 3b
32 c 0 38 4 16 26 8 25 3 e f 35 2c 27 18 23 11 2e 2d 1 22 39 a 5 3d 19 3a 2 2f 34}
{31 29 23 1 18 c 2 3e 2a 3c 36 33 0 22 30 21 3d 2d 2b 3b 32 1b 1c 24 39 2e 10 17 1d 38 2f 11 1f
19 13 8 37 3a 14 25 3f 16 34 d 27 12 b 9 f 2c 7 6 a 3 1e 5 15 28 35 26 20 1a 4 e}
{31 9 14 1e 34 39 2 2e 2c e 10 1f 18 11 3 1a 37 21 20 1d 3d c 4 3e 30 d 3a 22 6 2d 5 1b 2f 27 24
12 f a 2b 3c 17 3b 3f 2a 23 19 33 35 38 32 b 16 29 13 1c 15 7 8 28 26 1 36 25 0}
{36 d 10 1e 16 1d 35 14 29 1f e 9 7 2d 2c 5 38 6 27 15 c 1c 17 21 32 2a 3a 3 26 3b 1b 8 39 30
25 28 20 2e 3c 24 31 13 23 12 b 34 2f 37 2b 4 11 22 1 3e 18 f 1a 0 3f 19 3d a 33 2}
{15 29 1a 22 37 11 26 d 17 27 7 21 16 30 33 12 20 4 3a 2 f 39 31 e 2c b 14 9 0 6 28 3b 2f 2a 24
a 3e 1 1f 3f 25 1e 1b 5 2e 3 10 3d 38 1d 36 8 2b 23 32 13 2d 35 34 18 19 3c 1c c}
{33 17 2d 13 14 11 28 3f 8 24 31 34 2b 1 1d 1e 2 30 1f 2c 22 7 c e b 1a 26 3c 29 2e 23 5 36 3e
10 2a 3b f 27 12 20 38 37 3 6 35 3d 0 15 3a 39 2f 4 19 16 9 1c d 32 21 18 25 a 1b}
{2e 2f 33 20 1a 10 3b 35 2b 39 1f 30 28 1b 29 3d e 1 13 34 18 3c 38 37 1d 23 d 32 9 31 12 b 2c
2 2d 14 6 4 8 3f 21 1c 1e 24 19 26 5 0 3e 27 15 16 7 36 11 f 3a a 25 22 17 3 c 2a}
{36 13 34 32 3e 27 20 3b 19 28 c 2 33 22 2c 2a 3f 1 9 d 2f 7 8 1a 38 18 12 31 2b 14 37 1f 3 1e
3c 16 23 3a b 5 11 0 26 a 39 2d 2e 25 1c 30 21 6 35 1d f 4 3d e 29 1b 10 17 24 15}
{1f 3f 35 1d 34 37 20 3 2d 24 28 30 3d 2b 39 c 13 32 23 1c 3b e 17 36 4 5 18 19 3e 2e 16 21 29
2 0 3a 15 1b 1e 22 d 3c 11 38 6 25 2f 7 10 a 27 9 1 2c 1a 14 31 33 8 12 2a b f 26}
{14 16 15 2a 4 b 36 2e 13 2c 17 37 27 19 2f 22 c 1f 24 23 7 2d 31 3c 25 38 20 29 1e 2 35 3e 32
3f 2b 18 5 3 3d 1a f 9 21 1c 3a 1b 10 6 34 e 26 8 1 28 d 39 3b 12 11 33 30 a 1d 0}
{d 20 39 2d a 30 3b 2b 3f 6 22 2c 8 2e 37 4 36 18 13 7 21 1a 3 3d 2a 28 3a c 14 10 12 19 1e 32
11 2f 24 35 1 2 1c e 1b 27 26 38 0 f 29 31 16 17 9 1f 3c 3e b 5 25 23 33 34 1d 15}
{e 22 4 13 1 2a f 2f 3a 31 7 39 36 2 23 1c 1f 15 19 26 1e 18 21 34 b a 37 17 12 3b 8 5 14 6 0 2c
10 16 25 3d 1a 33 2e 2b 24 28 20 3 38 30 11 32 1d 29 1b 3e 35 3c c 27 2d 3f 9 d}
{0 e 25 38 3f 33 3d 1f 13 28 2e 31 22 21 36 17 1e 18 c 1 26 2f 1b a 34 f 3e 1c 12 19 3b 23 37 39
1d 2 2a 35 29 3 2c 10 b 9 27 d 32 7 30 3a 15 8 3c 2d 16 1a 20 2b 11 14 6 4 5 24}
```

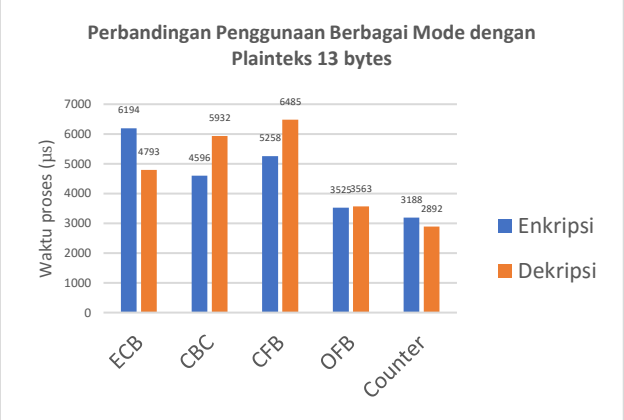
Proses fungsi putaran tersebut dapat dilihat pada Gambar 10.



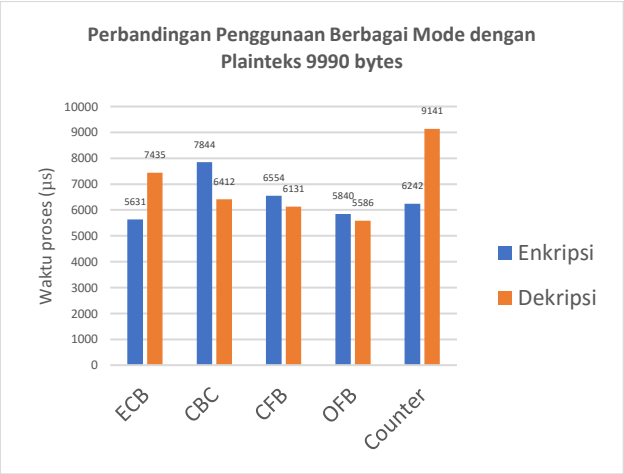

**Gambar 10.** Proses fungsi putaran pada Basit

## IV. Percobaan dan Analisis

Untuk melakukan percobaan terhadap algoritma Basit yang sudah dirancang, kami mengimplemantasikannya menggunakan bahasa c dengan tambahan program quark hash yang telah dibuat sebelumnya[3]. Proses pengujian dilakukan dengan mode ECB, CBC, CFB, OFB, dan counter. Kami melakukan 2 kali percobaan, yaitu dengan data kecil (13 bytes) dan dengan data yang lebih besar (9990 bytes). Proses enkripsi pada data kecil menghasilkan waktu rata-rata untuk enkripsi selama 4552,2 µs, sedangkan untuk dekripsi selama 4733 µs. Percobaan pada proses enkripsi data besar membutuhkan waktu 6422 µs, sedangkan untuk dekripsi membutuhkan waktu 6941,2 µs.




**Gambar 10.** Hasil percobaan penggunaan Basit pada data yang ukurannya kecil




**Gambar 11.** Hasil percobaan penggunaan Basit pada data yang ukurannya besar

Dari percobaan tersebut, kenaikan jumlah data lebih dari 100 kali lipat tidak menaikan waktu eksekusi terlalu signifikan. Pemrosesan data kecil hanya memerlukan waktu 4.6 ms, sedangkan pada data besar memerlukan waktu 6.7 ms.

Selain menjalankan Basit dengan banyak mode, kami juga mencoba membandingkan properti konfusi dan difusi dari *block cipher* Basit. Kami melakukan analisis terhadap frekuensi kemunculan pada plainteks dan cipherteks. Hasil analisis yang kami lakukan sangat memuaskan. Dari teks yang berisi cerita berbahasa inggris, terdapat beberapa karakter yang cukup sering muncul seperti huruf A, E, dan lain sebagainya. Menggunakan Basit, kami berhasil mendapatkan cipherteks yang relatif sama, untuk kemunculan setiap karakter ASCII. Hasil analisis ini dapat dilihat pada Gambar 12.

As a child, I loved sitting on my grandfather's lap while he read me stories. I remember most of them even though I am now a grandparent, too! As a child, I was blissfully unaware that, as I listened to the stories, I was also learning new words and ways in which those new words combined to communicate ideas and life lessons.

A good story encourages us to turn the next page and read more. We want to find out what happens next and what the main characters do and what they say to each other. We may feel excited, sad, afraid, angry or really happy. This is because the experience of reading or listening to a story is much more likely to make us 'feel' that we are part of the story, too. Just like in our 'real' lives, we might love or hate different characters in the story. Perhaps we recognise ourselves or others in some of them. Perhaps we have similar problems.

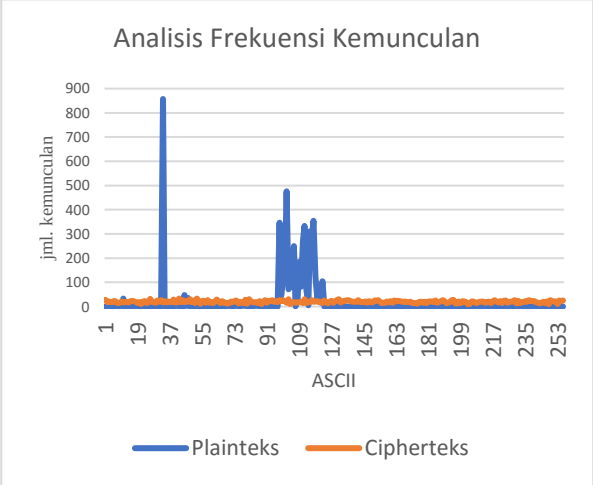
Because of this natural empathy with the characters, our brains process the reading of stories differently from the way we read factual information. Our brains don't always recognise the difference between an imagined situation and a real one so the characters become 'alive' to us. What they say and do is therefore more meaningful. This is why the words and structures that relate a story's events, descriptions and conversations are processed in this deeper way.

.....

But, of course, stories don't only offer the young reader a chance to read. The experience also creates an opportunity to talk about the story. As a parent, you can encourage your child to describe their favourite person, part of the story or picture. Their creativity might be developed by drawing new story pictures or even by writing their own short stories as a result.

If your child is reluctant to read or has little confidence in their ability to read in another language, you might help them by reading the story to them, stopping where necessary to interact and ask questions like 'What do you think will happen next?' If you read to your children in a relaxed and fun way, they will subconsciously relate to the reading and language learning process more confidently and positively. Of course, being read to by a parent, for whatever reason, is also simply a lovely way to share quiet and close time.

The experience of reading or listening to a story allows us to escape our own lives for a moment and live in another one in a fun and safe way. In the same magical experience, a goldmine of language may be learned, so do encourage your child to read stories in their second language as well as their first!




**Gambar 12.** Hasil analisis frekuensi menggunakan algoritma *cipher block* Basit

Kami juga mencoba melakukan perubahan 1 bit pada plainteks dan didapatkan rata-rata ada 18 bit perubahan pada cipherteks. Sedangkan 1 bit perubahan pada kunci akan mengakibatkan 15 bit perubahan pada cipherteks. Dengan cipherteks yang kemunculan setiap karakter ASCII-nya relatif sama, maka serangan yang menggunakan analisis frekuensi akan semakin sulit dilakukan.

## V. KESIMPULAN

Algoritma *block cipher* Basit merupakan alternatif algoritma yang mudah untuk diimplementasi karena menggunakan operasi-operasi yang sederhana. Walaupun demikian, algoritma Basit dapat menghasilkan cipherteks yang sulit untuk dipecahkan menggunakan analisis frekuensi. Waktu yang digunakan untuk pemrosesan data yang kecil dan besar menggunakan algoritma Basit juga relatif sama. Penggunaan fungsi hash Quark membuat algoritma ini dapat menerima berbagai kunci yang mudah diingat, tanpa ada ketentuan ukuran bit kunci.

## REFERENCES


[1] Jean-Philippe, Aumasson, Luca Henzen, Willi Meier, dan Mar´ıa Naya-Plasencia. 2012. *Quark: a lightweight hash*.

[2] Ariyus, Dony. 2008. *Pengantar Ilmu Kriptografi: Teori Analisis & Implementasi*. Yogyakarta: Penerbit Andi

[3] Munir, Rinaldi. 2015. Slide Kuliah IF4020 Kriptografi: Algoritma Kriptografi Modern

[4] Munir, Rinaldi. 2015. Slide Kuliah IF4020 Kriptografi: Serangan terhadap Kriptografi.


## 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.

Bandung, 15 Maret 2018

Jauhar Arifin (13515049)

Fadhil Imam Kurnia (13515146)