# Hanzo Cipher

M Galih R R[1], M Iqbal Sigid[2]
[1,2] Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): 13519017@std.stei.itb.ac.id[1], 13519152@std.stei.itb.ac.id[2]

***Abstract*—Hanzo cipher adalah sebuah blok cipher yang memanfaatkan *feistel network* untuk melakukan proses *ciphering*-nya. Seperti kebanyakan blok cipher lainnya, Hanzo cipher menggunakan prinsip *confusion, diffusion,* dan *key expansion.***

***Keywords—block cipher, S-box, P-box, key, feistel network***

## I. Pendahuluan

Kriptografi berguna untuk menjaga suatu informasi agar informasi tersebut hanya dapat dibaca, diubah, dan dimanfaatkan oleh suatu pihak yang berwenang untuk melakukannya. Saat ini informasi sangat penting, seperti banyaknya layanan yang sangat bergantung dengan autentifikasi pengguna atau setiap orang memiliki informasi rahasia yang harus dilindungi. Untuk menjaga informasi ini dapat digunakan Kriptografi.

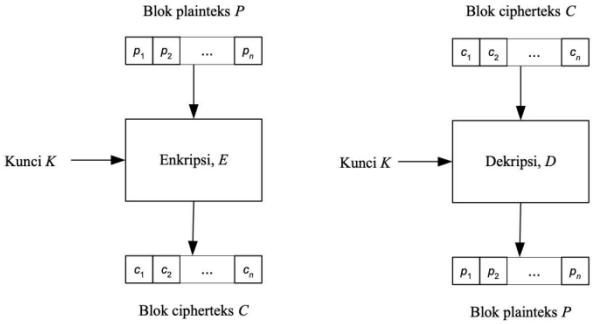
Kriptografi menjaga kerahasiaan data dengan mensubstitusi data *plaintext* menjadi *ciphertext* yang tidak bermakna apa-apa. Untuk memanfaatkan informasi *ciphertext* ini, pengguna memerlukan kunci untuk mendekripsi *ciphertext* menjadi *plaintext*. Awal perkembangan kriptografi yang dinamakan kriptografi klasik, digunakan teknik substitusi dan pergeseran untuk menciptakan *ciphertext*. Namun, kriptografi klasik ini sudah dinyatakan tidak aman karena *ciphertext* dapat dipecahkan tanpa kunci dengan berbagai metode sehingga digantikan oleh kriptografi modern.

Kriptografi modern memproses data secara bit atau byte, berbeda dengan kriptografi klasik yang memproses karakter dan kriptografi modern juga banyak memanfaatkan operasi xor. Kriptografi modern banyak digunakan untuk keamanan komunikasi seperti pengiriman pesan dan sebagainya. Beberapa contoh algoritma kriptografi modern seperti A5, RC5, dan AES. Selama adanya kriptografi, maka ada pula pihak yang berusaha untuk memecahkan *ciphertext* yang dihasilkan oleh algoritma kriptografi. Hal ini menjadi motivasi untuk membuat algoritma kriptografi yang lebih aman dari yang telah ada sebelumnya.

## II. Dasar Teori

### A. Block Cipher

*Block cipher* merupakan salah satu algoritma kriptografi modern kunci simetris yang memproses *plaintext* dengan membaginya menjadi blok-blok dengan panjang yang sama. Ukuran blok yang umum contohnya 64 bit, 128 bit, 256 bit. Cara kerja *block cipher* pada umumnya akan mengenkripsi blok *plaintext* dengan berbagai operasi dengan menggunakan kunci enkripsi. Panjang blok dan kunci yang digunakan akan berbeda sesuai dengan algoritma kriptografi yang digunakan. Gambar 2.1. menunjukkan skema umum enkripsi dan dekripsi *block cipher*.




Gambar 2.1. Skema enkripsi dan dekripsi *block cipher*

(Sumber: Rinaldi Munir)

*Block cipher* sendiri memiliki berbagai operasi diantaranya:

1. *Electronic Code Book (ECB)*
2. *Cipher Block Chaining (CBC)*
3. *Cipher Feedback Block (CFB)*
4. *Output Feedback Block (OFB)*
5. *Counter*

### B. Prinsip Confusion dan Diffusion Shannon

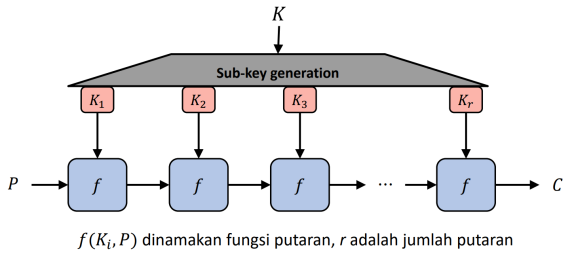
Prinsip *confusion* dan *diffusion* berguna untuk menjadi panduan dalam merancang algoritma kriptografi. Kriptografi klasik dapat dipecahkan karena adanya hubungan statistik antara *ciphertext* dan *plaintext*. Hubungan statistik, seperti jumlah kemunculan karakter, dapat digunakan untuk menebak *plaintext* pada metode analisis frekuensi. Untuk menyembunyikan hubungan statistik ini digunakan kedua prinsip tersebut.

Prinsip *confusion* menyembunyikan hubungan statistik dengan substitusi non-linear, seperti pada algoritma *one time pad* dimana setiap karakter memiliki kunci yang tidak berhubungan satu sama lain. Prinsip *diffusion* menyebarkan

pengaruh satu bit *plaintext* atau kunci ke banyak bit pada *ciphertext*. Ini berarti penggantian satu bit pada *plaintext* tidak hanya akan mempengaruhi satu bit pada *ciphertext*, tetapi banyak bit. *Diffusion* dapat dilakukan dengan permutasi dan transposisi berulang-ulang.

### C. Cipher Berulang

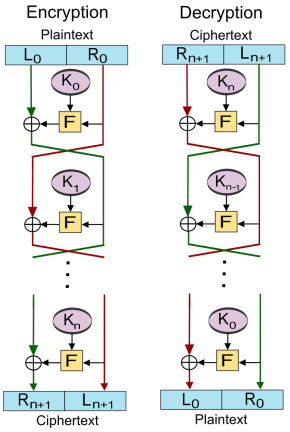
Cipher berulang bertujuan untuk memperkuat algoritma kriptografi dengan melakukan pengkodean secara berulang. Pada setiap putaran, digunakan *subkey* yang berbeda yang sebelumnya dibangkitkan dari kunci eksternal. Jumlah putaran yang digunakan harus mengkonsiderasikan keamanan dan efisiensi. Gambar 2.2. menunjukkan skema cipher berulang.




Gambar 2.2. Ilustrasi cipher berulang
(Sumber: Rinaldi Munir)

### D. Jaringan Feistel

Jaringan Feistel merupakan sebuah struktur *enciphering* pada *block cipher*. Struktur ini dapat merepresentasikan proses enkripsi maupun dekripsi karena struktur ini bersifat *reversible*. Salah satu algoritma kriptografi yang menggunakan jaringan feistel adalah DES yang dapat dilihat pada Gambar 2.3.




Gambar 2.3. Jaringan Feistel pada algoritma DES
(Sumber: Wikipedia)

### E. Substitution Box

Substitution box adalah sebuah *lookup table* yang digunakan dalam tahap substitusi pada algoritma ini. Substitution yang baik seharusnya bersifat non-linear dengan memiliki distribusi derivatif berarah yang seimbang [1]. Perancangan substitution box juga mengikuti cara perancangan substitution box pada AES yang akan dijelaskan pada bagian selanjutnya.

## III. Rancangan Block Cipher

[Past] Cipher bekerja dengan ukuran blok 128-bit dan kunci 128-bit. Cipher ini memiliki Jaringan Feistel dengan fungsi putaran yang melakukan operasi XOR dengan kunci putaran serta substitusi dan difusi. Jumlah putaran yang dilakukan sebanyak 16 kali. Cipher ini juga memanfaatkan S-box dan P-box untuk melakukan substitusi dan permutasi bit terhadap *plaintext*.

### A. Fungsi Putaran

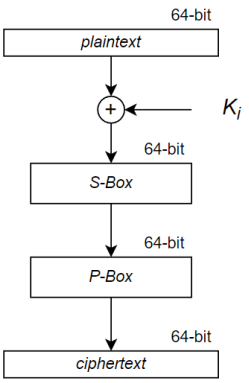
Fungsi putaran cipher ini terdiri dari tiga operasi, yaitu operasi XOR, substitusi S-box, dan permutasi P-box. Fungsi putaran menerima masukan 64-bit. Blok ini akan dilakukan operasi XOR dengan kunci putaran. Kunci putaran yang digunakan akan dijelaskan pada bagian III.B. Blok hasil operasi XOR kemudian disubstitusi menggunakan S-box. Berikut adalah S-box yang digunakan.

Tabel 3.1. S-Box

| | 00 | 01 | 10 | 11 |
|---|---|---|---|---|
| 00 | 0011 | 1100 | 0100 | 1001 |
| 01 | 1000 | 1010 | 1110 | 0001 |
| 10 | 0110 | 1101 | 0111 | 0000 |
| 11 | 0101 | 1111 | 0010 | 1011 |

Dengan sumbu y adalah 2 bit MSB dan sumbu x adalah 2 bit LSB. Jadi sebagai contoh, 0xF (1111) akan diubah menjadi 0xB (1011). Setelah substitusi, dilakukan operasi permutasi menggunakan P-box. P-box akan menukar posisi bit dari blok dengan tabel yang berisi 64 angka yang menunjukkan posisi bit *i* dan posisi dimana bit tersebut akan dipindahkan *P(i)*. Isi dari P-box akan dijelaskan pada bagian III.D

Blok 64-bit hasil permutasi tersebut kemudian dilanjutkan ke jaringan feistel. Gambar 3.1. menunjukkan skema fungsi putaran.



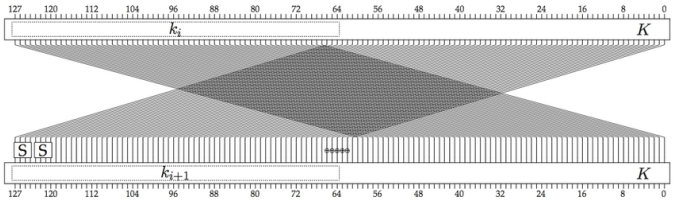

Gambar 3.1. Skema fungsi putaran

### B. Key Expansion

Algoritma *key expansion* yang digunakan adalah algoritma *key expansion* dari PRESENT cipher [4] dengan kunci 128-bit. Kunci 128-bit direpresentasikan sebagai $k_{127}k_{126}\ldots k_0$. Kunci putaran yang digunakan hanyalah 64-bit MSB $K_i = k_{127}k_{126}\ldots k_{64}$. Pada setiap putaran, kunci external akan dilakukan operasi sebagai berikut

1. $[k_{127}k_{126}\ldots k_0] = [k_{66}k_{65}\ldots k_{67}]$
2. $[k_{127}k_{126}k_{125}k_{124}] = S[k_{127}k_{126}k_{125}k_{124}]$
   $[k_{123}k_{122}k_{121}k_{120}] = S[k_{123}k_{122}k_{121}k_{120}]$
3. $[k_{66}k_{65}k_{64}k_{63}k_{62}] = [k_{66}k_{65}k_{64}k_{63}k_{62}] \oplus rc$

Pertama, kunci external akan diputar 61-bit ke kiri. Kemudian 8-bit MSB pada kunci external akan disubstitusi dengan dua S-box masing-masing berukuran 4-bit. Terakhir bit $k_{66}k_{65}k_{64}k_{63}k_{62}$ akan dilakukan operasi XOR dengan *round counter*. Skema *key expansion* dapat dilihat pada Gambar 3.2.




Gambar 3.2. Skema *key expansion*

### C. Jaringan Feistel

Desain jaringan feistel yang digunakan adalah jaringan feistel yang digunakan oleh DES yang digambarkan oleh Gambar 2.3. Masukan *plaintext* 128-bit dipecah menjadi dua blok 64-bit yang akan disebut L untuk 64-bit MSB dan R untuk 64-bit LSB. R akan diproses oleh fungsi putaran dengan kunci putaran, kemudian hasil dari fungsi putaran ini akan dilakukan operasi XOR dengan L. Hasil operasi XOR dari L kemudian digabungkan dengan blok R dengan posisi yang ditukar. Proses ini akan diulang sebanyak jumlah putaran, yaitu 16 kali. Hasil putaran terakhir ini juga akan kemudian dioperasikan secara XOR dengan hasil *key expansion* terakhir. Hal ini dilakukan agar proses substitusi terakhir tidak bersifat trivial untuk proses cryptanalysis yang mengikuti proses enkripsi dari akhir ke bagian awal [3].

### D. Desain S-box dan P-box

S-box didesain dengan cara yang mirip dengan S-box pada AES. Perbedaan terbesar dengan AES terletak pada jumlah bit yang digunakan. Pada AES 'S-box'-nya menggunakan 8 bit sementara S-box pada algoritma ini menggunakan 4 buah bit. Pada AES semua anggotanya dari 0x0…0xFF diubah menjadi *multiplicative inverse*-nya pada GF(256) dengan *irreducible polynomial* $x^8 + x^4 + x^3 + x + 1$ dan kemudian diubah lagi dengan transformasi affine tertentu [2]. Pada S-box yang kami gunakan perubahan anggota dari 0x0..0xF pada GF(16) dilakukan dengan polinomial $x^4 + x + 1$ dan kemudian ditransformasikan dengan transformasi affine yang mirip dengan yang dilakukan pada AES. P-box tidak didesain dengan cara tertentu, melainkan hanya dengan melakukan *mapping* dari semua 64 bit secara acak. Proses ini dilakukan dengan metode *random* dari *python* menggunakan *seed* yang sudah ditetapkan (1). P-box dapat dilihat pada Gambar 3.3.

| $i$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $P(i)$ | 0 | 16 | 32 | 48 | 1 | 17 | 33 | 49 | 2 | 18 | 34 | 50 | 3 | 19 | 35 | 51 |
| $i$ | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| $P(i)$ | 4 | 20 | 36 | 52 | 5 | 21 | 37 | 53 | 6 | 22 | 38 | 54 | 7 | 23 | 39 | 55 |
| $i$ | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 |
| $P(i)$ | 8 | 24 | 40 | 56 | 9 | 25 | 41 | 57 | 10 | 26 | 42 | 58 | 11 | 27 | 43 | 59 |
| $i$ | 48 | 49 | 50 | 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 | 60 | 61 | 62 | 63 |
| $P(i)$ | 12 | 28 | 44 | 60 | 13 | 29 | 45 | 61 | 14 | 30 | 46 | 62 | 15 | 31 | 47 | 63 |

Gambar 3.3. P-Box

## IV. Eksperimen dan Analisis

Implementasi [Past] Cipher dilakukan menggunakan bahasa Python.

### A. Confusion Shannon

Nilai kualitas *confusion* dari S-box akan dilakukan dengan menerka *linear approximation function* dari S-box-nya. *Linear approximation function* atau LAT digunakan dalam *linear cryptanalysis* memanfaatkan kelinearan dari S-box yang digunakan [3].

Tabel 4.1. *Linear approximation table*

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | a | b | c | d | e | f |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | +8 | 0 | 0 | 0 | +1 | +1 | -1 | -1 | +1 | +1 | -1 | -1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | -3 | +1 | -1 | +3 | +3 | -1 | -3 | +1 | -2 | -2 | -2 | -2 |
| 2 | 0 | +2 | -2 | 0 | -1 | +1 | -1 | +1 | -1 | -3 | -1 | -3 | 0 | -2 | -2 | +4 |
| 3 | 0 | -2 | +2 | 0 | +3 | -3 | +3 | +5 | +1 | -2 | +1 | -1 | -2 | 0 | 0 | +2 |
| 4 | 0 | +2 | 0 | +2 | -1 | -3 | +1 | -1 | +3 | +1 | -3 | +3 | 0 | +2 | 0 | +2 |
| 5 | 0 | -2 | -4 | +2 | -1 | -3 | +1 | -1 | +1 | -1 | +3 | +1 | -2 | -4 | +2 | 0 |
| 6 | 0 | 0 | -2 | -2 | +1 | -3 | -3 | +1 | +1 | +1 | -3 | -3 | +4 | 0 | +2 | -2 |
| 7 | 0 | 0 | -2 | -2 | +1 | +1 | +1 | +1 | -1 | +3 | -1 | +3 | +2 | -2 | 0 | +4 |
| 8 | 0 | 0 | +2 | -2 | +3 | -1 | -1 | -1 | -1 | -1 | -1 | +3 | 0 | -4 | -2 | -2 |
| 9 | 0 | 0 | +2 | +6 | -1 | -1 | -1 | +3 | -3 | +1 | +1 | +1 | +2 | -2 | 0 | 0 |
| a | 0 | +2 | -4 | +2 | +5 | +3 | -1 | +1 | +1 | -1 | -1 | +1 | 0 | +2 | 0 | -2 |
| b | 0 | -2 | 0 | +2 | +1 | -1 | +3 | -3 | -1 | -3 | -3 | -1 | +2 | 0 | -2 | 0 |
| c | 0 | -2 | -2 | 0 | +1 | -1 | -3 | -1 | +1 | +3 | +1 | -1 | 0 | +2 | -6 | 0 |
| d | 0 | +2 | +2 | 0 | +1 | -1 | -3 | -1 | +3 | -3 | +3 | +1 | +2 | 0 | 0 | +2 |
| e | 0 | -4 | 0 | 0 | -1 | +3 | +1 | +1 | +3 | -1 | +1 | +1 | +4 | 0 | 0 | 0 |
| f | 0 | -4 | 0 | 0 | -1 | -1 | -3 | +1 | -3 | -3 | -1 | +3 | -2 | +2 | +2 | +2 |

S-box yang linear akan menghasilkan angka-angka yang besar, baik positif maupun negatif, pada tabel LAT dan dalam frekuensi yang besar juga. Pada tabel di atas bisa dilihat bahwa ada 1 buah angka 8 pada tabel dengan koordinat [0,0]. Pada LAT sel ini akan selalu berisi nilai $2^{n-1}$ dengan n sebagai jumlah bit yang digunakan S-box, jadi kita bisa abaikan nilai ini. Nilai terbesar kedua dan ketiga pada LAT adalah 6 dan 5, yang termasuk besar akan tetapi keduanya hanya muncul 2 kali, jadi bisa diambil kesimpulan bahwa S-box yang digunakan memiliki kenonlinearan yang cukup.

*B. Avalanche Effect dan Diffusion Shannon*

Eksperimen untuk *avalanche effect* dilakukan dengan melakukan dua buah enkripsi barisan bit yang mirip, perbedaannya hanya ada pada LSB mereka. Enkripsi dilakukan menggunakan kunci yang sama dan seluruh prosesnya dituliskan dalam bentuk heksadesimal.

Tabel 4.2. *Avalanche effect* dari enkripsi pada setiap putaran

| | | |
|---|---|---|
| Input | F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0 | F0F0F0F0F0F0F0F0F0F0F0F0F0F0F1 |
| Round 0 | f0f0f0f0f0f0f0f0f0f0f0f0f0f0f0f | f0f0f0f0f0f0f0f1f0f1f0f10f0e0f0e |
| Round 1 | f0f0f0f0f00f0f0f0f700fb00fcff070f0 | f0f1f0f10f0e0f0e611eb01ecfe161e0 |
| Round 2 | 700fb00fcff070f0f4c743b04f8f70f8 | 611eb01ecfe161e007f0838f3c86b0c7 |
| Putaran berikutnya tidak akan dilabeli karena kedua proses sudah memiliki perbedaan yang signifikan. | | |
| Round 3 | f4c743b04f8f70f8a8af1309dedf4a09 | 7f0838f3c86b0c74bffa01275d40ac9 |
| Round 4 | a8af1309dedf4a097d2b9e50ec39664b | 4bffa01275d40ac940d458edb1eaad2c |
| Round 5 | 7d2b9e50ec39664bedcfb273fa256eef | 40d458edb1eaad2cbf776515573445ae |
| Round 6 | edcfb273fa256eef6e514ea7d97327d9 | bf776515573445ae6d53577bc95a2803 |
| Round 7 | 6e514ea7d97327d99057c307b7fca5fc2 | 6d53577bc95a280305c2e42fcd4b2292 |
| Round 8 | 57c307b7fca5fc2a037563f09ae0fa6 | 5c2e42fcd4b2292b69965351c110dca |
| Round 9 | a037563f09ae0fa666f019f6aadbd493 | b69965351c110dcacbdd6064d1f87d24 |
| Round 10 | 66f019f6aadbd493a3dd55c6923c97e2 | cbdd6064d1f87d24c53891f5ce16b7ea |
| Round 11 | a3dd55c6923c97e21b460a352196ca55 | c53891f5ce16b7eac0bb928e708d43ab |
| Round 12 | 1b460a352196ca55d26a113fbfbaa8db | c0bb928e708d43ab4cf05fc33e34dd2d |
| Round 13 | d26a113fbfbaa8db06a9091062d6ecb2 | 4cf05fc33e34dd2dc68217a80fdc0ea4 |
| Round 14 | 6a9091062d6ecb2cdc3de04538514e6 | c68217a80fdc0ea4c99aa425ec44d217 |
| Round 15 | cdc3de04538514e61477013b6ba4103e | c99aa425ec44d21789bd296a1010d3e7 |
| Hasil akhir akan dilabeli untuk menunjukkan perbedaan antar ciphertext. | | |
| Ciphertext | cdc3de04538514e673cdaa6b6f0020ca | c99aa425ec44d217ee07823a14b4e313 |

Hasil dari *ciphertext* menghasilkan bit yang berbeda jauh, walaupun pada masukan kedua *plaintext* hanya memiliki perbedaan 1-bit. Hasil ini menunjukkan prinsip difusi Shannon sudah diterapkan dengan baik pada cipher ini.

*C. Kinerja*

Percobaan dilakukan dengan tiga buah file teks dengan ukuran berbeda untuk melakukan enkripsi kemudian dekripsi. Hasil percobaan ditunjukkan pada Tabel 4.3.

Tabel 4.3. Hasil percobaan kinerja enkripsi dan dekripsi pada file teks

| Ukuran File | Waktu eksekusi |
|---|---|
| 100 KB | 26.5 detik |
| 50 KB | 12.7 detik |

| | |
|---|---|
| 23 KB | 5.5 detik |
| 12 KB | 2.7 detik |

Dari hasil percobaan di atas, waktu eksekusi semakin meningkat secara linear dengan pertambahan ukuran file. Kinerja ini dapat dikatakan cukup baik.

## V. Kesimpulan dan Saran

Algoritma yang dibuat bisa dikatakan telah memiliki karakteristik *confusion* dan *diffusion* yang baik. Hal ini bisa terjadi karena banyak bagian dari algoritma ini yang merupakan modifikasi dari algoritma lainnya yang telah terbukti memiliki evaluasi yang baik (AES dan PRESENT), akan tetapi untuk kriteria *avalanche*, terutama *strict avalanche criterion,* algoritma ini tidak memiliki kinerja yang kurang ideal. Eksperimen dan evaluasi lanjut diperlukan untuk membuktikan karakteristik yang benar-benar konkret untuk algoritma ini.

## Referensi


[1] Nyberg K. Perfect nonlinear S-boxes. InAdvances in Cryptology—EUROCRYPT'91: Workshop on the Theory and Application of Cryptographic Techniques Brighton, UK, April 8–11, 1991 Proceedings 10 1991 (pp. 378-386). Springer Berlin Heidelberg.

[2] https://csrc.nist.gov/csrc/media/projects/cryptographic-standards-and-guidelines/documents/aes-development/rijndael-ammended.pdf. Diperiksa pada 5 Maret 2023.

[3] Heys HM. A tutorial on linear and differential cryptanalysis. Cryptologia. 2002 Jul 1;26(3):189-221.

[4] Bogdanov A, Knudsen LR, Leander G, Paar C, Poschmann A, Robshaw MJ, Seurin Y, Vikkelsoe C. PRESENT: An ultra-lightweight block cipher. InCryptographic Hardware and Embedded Systems-CHES 2007: 9th International Workshop, Vienna, Austria, September 10-13, 2007. Proceedings 9 2007 (pp. 450-466). Springer Berlin Heidelberg.