# Algoritma Pattimura (v0.1)

**Pendahuluan**
Algoritma ini tidak menggunakan sub kunci dalam proses enkripsinya, melainkan menggunakan tabel yang dibangkitkan dari kunci. Dasar ide dalam pengembangan algoritma enkripsi ini adalah dari algoritma Blowfish, IDEA dan RC4.

**Pembangkitan Tabel**
Panjang kunci yang digunakan pada algoritma Pattimura adalah 128 bit, 192 bit dan 256 bit.
Komponen dalam pembangkitan tabel adalah sebagai berikut:
Kunci (K): kunci yang digunakan pada algoritma Pattimura (128 bit, 192 bit, 256 bit)
Tabel Pengguna (TP): merupakan tabel yang ditentukan oleh pengguna. Tujuannya adalah agar setiap pengguna dapat menentukan tabel mereka sendiri secara rahasia. Dengan demikian, dengan algoritma yang sama, belum tentu dapat melakukan skripsi - dekripsi dengan sembarang pengguna.
Aturan TP adalah sebagai berikut:

- besarnya 256x8 bit;
- harus berisi angka 0 – 255, tidak ada angka yang muncul lebih dari 1 kali; dan
- posisi angka dapat pada tabel tersebut diacak sesuai dengan keinginan pengguna.

Tabel Pemutih (TW): merupakan tabel untuk pemutihan sebelum proses enkripsi dan sesudah proses enkripsi. Besar tabel 256x8 bit, sama dengan TP yaitu berisi angka 0 – 255 dan tidak ada angka yang muncul lebih dari 1 kali.
Tabel Kiri (TX): tabel yang digunakan untuk subtitusi bagian kiri (lihat proses enkripsi). Besar tabel 256x8 bit, berisi angka 0 – 255 dan tidak ada angka yang muncul lebih dari 1 kali.
Tabel Kanan (TY): tabel yang digunakan untuk subtitusi bagian kanan (lihat proses enkripsi). Besar tabel 256x8 bit, berisi angka 0 – 255 dan tidak ada angka yang muncul lebih dari 1 kali.

Proses pengisian tabel TW, TX, TY adalah sebagai berikut (untuk kunci 128 bit):

1. isi masing-masing tabel TW, TX, dan TY dengan angka 0 – 255 secara berurutan (0, 1, 2, 3, 4, ..., 255);
2. buat variabel Z dengan ukuran 256x8 bit;
3. salin 16x8 bit K ke 16x8 bit pertama variabel Z (Z[0]=K[0], Z[1]=K[1],Z[2]=K[2], ..., Z[15]=K[15]);
4. isi Z[16] hingga Z[255] dengan cara sebagai berikut:
   for i = 16 to 255 do:
   Z[i] = (((Z[i-4] ^ Z[i-3]) + Z[i-2]) ^ Z[i-1]) + 0xAB;
5. perbaharui Z dengan cara sebagai berikut:
   for i = 255 to 0 do:
   Z[i] = (((TP[Z[i]] ^ TP[Z[(i+1)mod256]]) + TP[Z[(i+2)mod256]]) ^
   TP[Z[(i+3)mod256]]) + 0xAB;
6. perbaharui Z (langkah 5)
7. permutasi tabel TW dengan cara sebagai berikut:
   for i = 0 to 255 do:
   tmp = TW[i];
   TW[i] = TW[Z[i]];
   TW[Z[i]] = tmp;
8. lakukan langkah 6 dan 7 untuk tabel TX dan TY;

*) semua array dimulai dari 0 untuk memudahkan modulus);
**) semua operasi penjumlahan dalam algoritma ini menggunakan operasi penjumlahan nodulus $2^8$.

Untuk K 192 bit dan 256 bit, perbedaan hanya pada pada penyalinan Z (langkah 3), yaitu langsung 24x8 bit dan 32x8 bit, sehingga pada langkah 4 dimulai dari i=24 (untuk K 192 bit) dan i=32 (untuk K 256 bit).

**Proses Enkripsi dan Dekripsi**

Semua proses, yaitu proses enkripsi dan dekripsi, dilakukan dalam ukuran 8 bit (byte). Panjang blok masukan (IN) dan keluaran (OUT) adalah 128 bit (16 byte). Jumlah round adalah 8. Berikut adalah pseudocode untuk **proses enkripsi**:

1. Substitusi dengan tabel TW;

```
OUT[0] = TW[IN[0]];
OUT[1] = TW[IN[1]];
OUT[2] = TW[IN[2]];
OUT[3] = TW[IN[3]];
OUT[4] = TW[IN[4]];
OUT[5] = TW[IN[5]];
OUT[6] = TW[IN[6]];
OUT[7] = TW[IN[7]];
OUT[8] = TW[IN[8]];
OUT[9] = TW[IN[9]];
OUT[10] = TW[IN[10]];
OUT[11] = TW[IN[11]];
OUT[12] = TW[IN[12]];
OUT[13] = TW[IN[13]];
OUT[14] = TW[IN[14]];
OUT[15] = TW[IN[15]];
```

2. Siapkan counter -> CNT = 0;
3. Lakukan proses enkripsi;

```
for i = 0 to 8 do
        OUT[0]  ^= TY[(TX[OUT[4]]  + TX[OUT[11]] + TY[CNT++])];
        OUT[1]  ^= TY[(TX[OUT[5]]  + TX[OUT[8]]  + TY[CNT++])];
        OUT[2]  ^= TY[(TX[OUT[6]]  + TX[OUT[9]]  + TY[CNT++])];
        OUT[3]  ^= TY[(TX[OUT[7]]  + TX[OUT[10]] + TY[CNT++])];
        OUT[12] ^= TY[(TX[OUT[8]]  + TX[OUT[7]]  + TY[CNT++])];
        OUT[13] ^= TY[(TX[OUT[9]]  + TX[OUT[4]]  + TY[CNT++])];
        OUT[14] ^= TY[(TX[OUT[10]] + TX[OUT[5]]  + TY[CNT++])];
        OUT[15] ^= TY[(TX[OUT[11]] + TX[OUT[6]]  + TY[CNT++])];

        OUT[4]  ^= TY[(TX[OUT[0]]  + TX[OUT[15]] + TY[CNT++])];
        OUT[5]  ^= TY[(TX[OUT[1]]  + TX[OUT[12]] + TY[CNT++])];
        OUT[6]  ^= TY[(TX[OUT[2]]  + TX[OUT[13]] + TY[CNT++])];
        OUT[7]  ^= TY[(TX[OUT[3]]  + TX[OUT[14]] + TY[CNT++])];
        OUT[8]  ^= TY[(TX[OUT[12]] + TX[OUT[3]]  + TY[CNT++])];
        OUT[9]  ^= TY[(TX[OUT[13]] + TX[OUT[0]]  + TY[CNT++])];
        OUT[10] ^= TY[(TX[OUT[14]] + TX[OUT[1]]  + TY[CNT++])];
        OUT[11] ^= TY[(TX[OUT[15]] + TX[OUT[2]]  + TY[CNT++])];
```

4. substitusi dengan tabel TW;

```
OUT[0] = TW[OUT [0]];
OUT[1] = TW[OUT [1]];
OUT[2] = TW[OUT [2]];
OUT[3] = TW[OUT [3]];
OUT[4] = TW[OUT [4]];
OUT[5] = TW[OUT [5]];
OUT[6] = TW[OUT [6]];
OUT[7] = TW[OUT [7]];
OUT[8] = TW[OUT [8]];
OUT[9] = TW[OUT [9]];
OUT[10] = TW[OUT [10]];
OUT[11] = TW[OUT [11]];
OUT[12] = TW[OUT [12]];
OUT[13] = TW[OUT [13]];
OUT[14] = TW[OUT [14]];
OUT[15] = TW[OUT [15]];
```

Untuk proses dekripsi tidak berbeda jauh dengan enkripsi, secara struktur mirip hanya saja ada perbedaan untuk TW (inversi TW, dijelaskan setelah bagian ini) dan urutan counter. Berikut **proses dekripsi**:

1. Substitusi dengan tabel TW;

```
OUT[0] = TW[IN[0]];
OUT[1] = TW[IN[1]];
OUT[2] = TW[IN[2]];
OUT[3] = TW[IN[3]];
OUT[4] = TW[IN[4]];
OUT[5] = TW[IN[5]];
OUT[6] = TW[IN[6]];
OUT[7] = TW[IN[7]];
OUT[8] = TW[IN[8]];
OUT[9] = TW[IN[9]];
OUT[10] = TW[IN[10]];
OUT[11] = TW[IN[11]];
OUT[12] = TW[IN[12]];
OUT[13] = TW[IN[13]];
OUT[14] = TW[IN[14]];
OUT[15] = TW[IN[15]];
```

2. Siapkan counter -> CNT = 128;
3. Lakukan proses enkripsi;

```
for i = 0 to 8 do
        OUT[11] ^= TY[(TX[OUT[15]] + TX[OUT[2]]  + TY[--CNT])];
        OUT[10] ^= TY[(TX[OUT[14]] + TX[OUT[1]]  + TY[--CNT])];
        OUT[9]  ^= TY[(TX[OUT[13]] + TX[OUT[0]]  + TY[--CNT])];
        OUT[8]  ^= TY[(TX[OUT[12]] + TX[OUT[3]]  + TY[--CNT])];
        OUT[7]  ^= TY[(TX[OUT[3]]  + TX[OUT[14]] + TY[--CNT])];
        OUT[6]  ^= TY[(TX[OUT[2]]  + TX[OUT[13]] + TY[--CNT])];
```

```
OUT[5]  ^= TY[(TX[OUT[1]]  + TX[OUT[12]] + TY[--CNT])];
OUT[4]  ^= TY[(TX[OUT[0]]  + TX[OUT[15]] + TY[--CNT])];

OUT[15] ^= TY[(TX[OUT[11]] + TX[OUT[6]]  + TY[--CNT])];
OUT[14] ^= TY[(TX[OUT[10]] + TX[OUT[5]]  + TY[--CNT])];
OUT[13] ^= TY[(TX[OUT[9]]  + TX[OUT[4]]  + TY[--CNT])];
OUT[12] ^= TY[(TX[OUT[8]]  + TX[OUT[7]]  + TY[--CNT])];
OUT[3]  ^= TY[(TX[OUT[7]]  + TX[OUT[10]] + TY[--CNT])];
OUT[2]  ^= TY[(TX[OUT[6]]  + TX[OUT[9]]  + TY[--CNT])];
OUT[1]  ^= TY[(TX[OUT[5]]  + TX[OUT[8]]  + TY[--CNT])];
OUT[0]  ^= TY[(TX[OUT[4]]  + TX[OUT[11]] + TY[--CNT])];
```

4. substitusi dengan tabel TW;

```
OUT[0] = TW[OUT [0]];
OUT[1] = TW[OUT [1]];
OUT[2] = TW[OUT [2]];
OUT[3] = TW[OUT [3]];
OUT[4] = TW[OUT [4]];
OUT[5] = TW[OUT [5]];
OUT[6] = TW[OUT [6]];
OUT[7] = TW[OUT [7]];
OUT[8] = TW[OUT [8]];
OUT[9] = TW[OUT [9]];
OUT[10] = TW[OUT [10]];
OUT[11] = TW[OUT [11]];
OUT[12] = TW[OUT [12]];
OUT[13] = TW[OUT [13]];
OUT[14] = TW[OUT [14]];
OUT[15] = TW[OUT [15]];
```

Inversi TW yang digunakan pada proses dekripsi dilakukan dengan cara sebagai berikut:

1. Salin TW ke TMP
2. Lakukan proses berikut:

```
for i=0 to 255 do
    TW[TMP[i]] =  i;
```