Cipher Transposisi Menggunakan Indeks Pemetaan sebagai Kunci

Bayu Kumoro Yakti[1], Ragiel Hadi Prayitno[2] dan Fauziah[2]

[1]Teknik Elektro, Universitas Gunadarma
[2]Sistem Komputer, Universitas Gunadarma
Jl. Margonda Raya No.100, Depok, Jawa Barat 16424
Email: {bayuyakti, ragielhp*, fauziah87}@staff.gunadarma.ac.id

Abstrak

Keamanan data merupakan salah satu hal yang membutuhkan perhatian khusus agar privasi data tetap terjaga. Salah satu solusi yang dapat menangani keamanan data adalah *cipher* transposisi. Penelitian ini menjelaskan metode *cipher* transposisi yang memiliki algoritma sederhana, dimana setiap masukan dari enkripsi dan dekripsi adalah data dan kunci. Kunci pada metode yang dilakukan adalah indeks pemetaan. Indeks pemetaan sebagai kunci merupakan salah satu metode dalam transposisi *cipher*. Indeks pemetaan menggunakan matriks 8x8 dan ditransposisi dengan pola zig-zag yang menghasilkan 64 kunci dalam sebuah vektor. Panjang data masukan bervariasi dan selalu menggunakan kunci yang sama. Metode dalam penelitian diterapkan pada perangkat lunak MATLAB. Penelitian menggunakan jumlah data masukan masing-masing 8-bit, 64-bit, 129-bit, 351-bit sampai dengan 500 juta-bit. Hasil penelitian menunjukkan bahwa proses tercepat adalah 64-bit yaitu 0,0014 detik untuk enkripsi dan 0,0016 detik untuk dekripsi. Hasil proses terlama adalah 500 juta-bit yaitu 76,5297 detik untuk enkripsi dan 54,8092 detik untuk dekripsi.

Kata kunci: Biner, Indeks Pemetaan, MATLAB, Cipher Transposisi, kecepatan proses

Pendahuluan

Keamanan data adalah salah satu dari banyak masalah penting yang perlu diperhatikan untuk memastikan kerahasiaan informasi. Data menjadi lebih rentan terhadap pembobolan ketika data tersebut beradaptasi ke dalam bentuk digital [1]. Media penyimpanan dan transmisi data digital rentan terhadap peretas. Jika media yang berisi data telah dibobol, maka data tersebut rentan terhadap pencurian, sehingga memungkinkan untuk disalahgunakan. Pembobolan data tersebut berpotensi merugikan pihak-pihak tertentu. Salah satu dari banyak metode yang dapat digunakan untuk mencegah kerugian tersebut adalah metode kriptografi.

Kriptografi adalah sebuah metode keamanan yang sudah ada sejak berabad-abad lalu, terutama digunakan untuk komunikasi, militer, atau komersial. Namun, dengan munculnya internet dan perdagangan elektronik, kriptografi telah memainkan peran penting dalam fungsi ekonomi global dan telah menjadi sesuatu yang digunakan oleh jutaan orang setiap hari. Informasi sensitif seperti uang kertas, laporan kartu kredit, kata sandi, atau komunikasi pribadi, dienkripsi (dan harus) dimodifikasi sehingga hanya dapat diakses oleh orang atau pihak yang berwenang, sehingga tidak dapat diuraikan untuk orang lain [2].

Proses kriptografi adalah algoritma yang dapat dibalik untuk mengubah pesan teks biasa (atau teks yang jelas) menjadi pesan *ciphertext* (atau sandi) berdasarkan algoritma yang diketahui oleh pengirim dan penerima. Pesan dalam bentuk sandi tidak dapat dibaca oleh siapa pun kecuali penerima yang dituju. Pesan dalam bentuk sandi tersebut (*ciphertext*) dapat dikembalikan ke bentuk aslinya, teks biasa.

Penyandian adalah tindakan mengubah pesan teks biasa menjadi bentuk *ciphertext*. Membalikkan tindakan itu (yaitu, bentuk *ciphertext* menjadi pesan teks biasa) disebut penguraian. *Enciphering* dan *Deciphering* lebih sering disebut sebagai enkripsi dan dekripsi [3].

Metode yang paling sering digunakan untuk mengenkripsi data yang diberikan adalah teknik substitusi dan transposisi [4]. Teknik substitusi adalah ketika huruf-huruf dari *plaintext* digantikan dengan huruf lain atau dengan angka atau simbol. Jika *plaintext* dilihat sebagai sebuah urutan bit, maka substitusi menggantikan pola bit *plaintext* dengan pola bit *ciphertext*. Teknik Transpo-

DOI : http://dx.doi.org/10.32409/jikstik.22.2.3364, *) Penulis Korespondensi

sisi: Jenis pemetaan yang sangat berbeda dilakukan dengan melakukan beberapa permutasi pada huruf *plaintext* di mana karakter *plaintext* digeser dalam beberapa pola reguler ke *cipher*. Waktu enkripsi lebih singkat untuk substitusi, tetapi membuka jalan bagi penyusup untuk memecahkan kode dengan mudah.

Kecepatan proses enkripsi dan dekripsi pada transposisi dipengaruhi oleh beberapa faktor berikut:

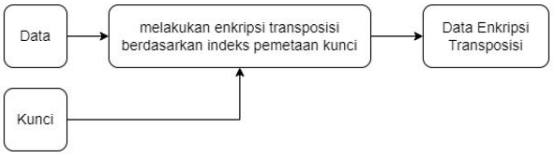
1. Ukuran Pesan: Semakin besar pesan yang akan dienkripsi atau didekripsi, semakin lama waktu yang dibutuhkan untuk melakukan proses transposisi. Proses waktu tersebut berbeda-beda karena semakin banyak karakter yang harus dipindahkan ke posisi yang berbeda dalam pesan.
2. Metode Transposisi: Ada beberapa metode transposisi yang berbeda, seperti kolom atau baris, *rail fence*, dan matriks. Masing-masing metode memiliki kecepatan proses yang berbeda tergantung pada implementasinya. Secara umum, kecepatan proses transposisi relatif lebih cepat daripada teknik kriptografi lainnya seperti substitusi atau enkripsi simetrik. Kecepatan proses pada akhirnya tergantung pada beberapa faktor, dan dapat berbeda-beda tergantung pada kasus yang spesifik [3].

Penelitian ini bertujuan untuk mengembangkan metode transposisi dengan kunci indeks pemetaan. Metode transposisi dengan indeks pemetaan dipilih karena indeks pemetaan merupakan perpindahan data sesuai dengan indeks tersebut. Metode tersebut tidak menggunakan perhitungan atau algoritma yang rumit sehingga proses transposisi enkripsi-dekripsi yang dilakukan juga cepat. Metode indeks pemetaan dalam urutan karakter dalam pesan dapat diubah secara acak sesuai dengan kunci tersebut. Hal ini membuat pola asli dalam teks terenkripsi sulit dikenali dan menganalisis oleh pihak yang tidak berwenang. Keberadaan kunci indeks pemetaan membuat metode transposisi lebih kuat dalam mengacak pesan. Pada penelitian ini, data masukan yang digunakan adalah bit biner sehingga apa pun data nya (teks, audio, gambar) bisa dilakukan pada penelitian ini dikarenakan semua bentuk data tersebut akan diproses dalam bentuk bit biner.

Metode Penelitian

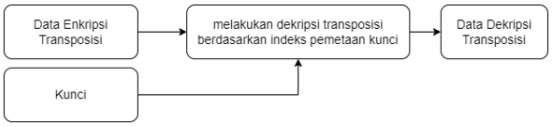
Penelitian ini dibuat dengan menggunakan perangkat lunak MATLAB dan perangkat keras processor intel core i7-1065G7 dengan RAM 16GB untuk menganalisa hasil algoritma Transposisi. Proses transposisi enkripsi membutuhkan dua data masukan yaitu data asli dan kunci dari data pemetaan indeks.

Data asli yang akan dimasukkan sebagai pesan rahasia berupa kode ASCII yang dikonversi ke biner 8-bit. Kunci pemetaan indeks dihasilkan dari pola zig-zag dari matriks 8x8 [10]. Kunci pemetaan data tersebut merupakan kunci indeks standar dalam proses transposisi enkripsi dan transposisi dekripsi. Gambar 1 menjelaskan proses enkripsi transposisi yang digunakan dalam penelitian ini menggunakan MATLAB.



Gambar 1: Alur enkripsi transposisi

Data masukkan pada dekripsi adalah data yang telah dienkripsi dan diterima dari sumbernya. Hasil pada transposisi dekripsi, penerima harus memiliki data sebagai kunci pemetaan indeks yang telah ditentukan. Kunci pemetaan data yang digunakan sama dengan transposisi enkripsi, yaitu pola zig-zag dari matriks 8x8 [10]. Gambar 2 menjelaskan proses dekripsi transposisi yang digunakan dalam penelitian ini menggunakan MATLAB.

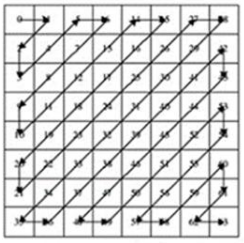


Gambar 2: Alur dekripsi transposisi

Data masukan yang dilakukan pada penelitian ini dapat berupa apa saja (biner atau teks atau kode ASCII) yang telah dikonversi dalam 8-bit dalam format data array vektor. Jumlah data pada penelitian ini dapat melakukan enkripsi dan dekripsi transposisi ke jumlah berapapun hingga satu juta data atau lebih.

Proses enkripsi dan dekripsi membutuhkan sebuah kunci, yang dapat berupa kata atau frasa. Transposisi *cipher* pada penelitian ini menggunakan parameter matriks sebagai kode untuk pemetaan data pada saat proses enkripsi dan dekripsi [10]. Data ditransposisi menggunakan kunci pemetaan indeks untuk enkripsi transposisi. Selanjutnya, data yang telah ditransposisikan diambil pada saat dekripsi menggunakan pemetaan kunci yang tepat. Kunci pemetaan indeks menggunakan matriks 8x8, yang telah ditransposisi dalam pola zig-zag, yang menghasilkan 64 indeks [10] ditunjukkan sebagai berikut Gambar 3.

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



Gambar 3: Ukuran matriks 8x8 dan pola zig-zag

Pola zig-zag dari matriks dapat tersusun menjadi sebuah vektor baris [11,12,13], Oleh karena itu, data vektor sebagai pemetaan indeks memperoleh parameter data input ketika menggunakan proses transposisi enkripsi dan dekripsi. Sebagai contoh, Gambar 4 menunjukkan kunci pemetaan indeks dalam larik vektor.

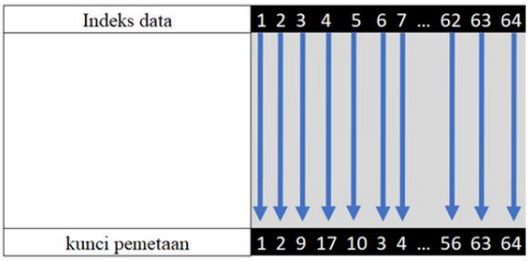
Kolom 1 sampai dengan kolom 15														
1	2	9	17	10	3	4	11	18	25	33	26	19	12	5
Kolom 16 sampai dengan kolom 30														
6	13	20	27	34	41	49	42	35	28	21	14	7	8	15
Kolom 31 sampai dengan kolom 45														
22	29	36	43	50	57	58	51	44	37	30	23	16	24	31
Kolom 46 sampai dengan kolom 60						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

Gambar 4: Pola zig-zag dalam bentuk vektor

Enkripsi Transposisi

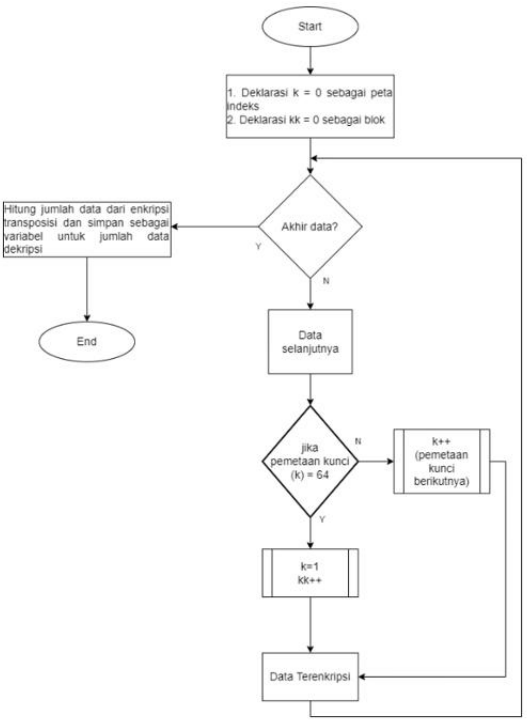
Metode Enkripsi pada metode transposisi adalah memindahkan data dari satu parameter berbasis posisi pada kunci pemetaan indeks. Data hasil transposisi disebut sebagai data enkripsi ketika posisinya sesuai dengan kunci.

Ilustrasi data yang menggunakan indeks pemetaan dalam enkripsi transposisi ada di Gambar 5. Enkripsi data ke-1 ditransposisi ke kunci pemetaan indeks pertama dari data asli pertama juga. Untuk enkripsi data ke-2, ketika indeks pemetaan ke-2 sama dengan 2, maka data asli ke-2 ditransposisikan ke posisi kedua berdasarkan indeks pemetaan. Enkripsi data ke-3 adalah memindahkan data asli dari larik ke-3 ke kunci pemetaan indeks ke-9. Juga, enkripsi data ke-17 adalah mentransposisikan data asli dari larik ke-4 ke posisi pemetaan indeks ke-17. Proses enkripsi transposisi ini dilanjutkan sampai data terakhir.



Gambar 5: Ilustrasi enkripsi indeks pemetaan

Sebagai contoh, jumlah data asli dapat lebih dari jutaan data, sedangkan jumlah kunci pemetaan indeks konstan pada 64 item data dan urutan data tetap dalam pola zig-zag. Secara umum, Gambar 6 menunjukkan alur proses enkripsi.



Gambar 6: Alur proses enkripsi

Transposisi enkripsi untuk mengurangi kompleksitas membutuhkan parameter awal selain data asli dalam format logika bit dan kunci sebagai indeks pemetaan. Parameter-parameter tersebut ditunjukkan pada Gambar 6 yaitu,

1. Deklarasi jumlah data asli yang dimasukkan dalam nilai n dan jumlah data asli yang selalu direkam dalam enkripsi data pertama. Fungsi transposisi enkripsi akan selesai ketika data asli (n = nilai maksimum dari data asli) telah sepenuhnya berubah.

2. k sebagai nilai kunci dari pemetaan indeks. Nilai k=0 adalah awal dari proses, yang menunjukkan indeks pemetaan kunci dari proses transposisi pada 0. Ketika nilai k=64, kunci pemetaan indeks maksimum, in-

deks pemetaan akan kembali ke indeks pertama (k=1), dan nilai kk akan bertambah 1 (kk=kk+1).

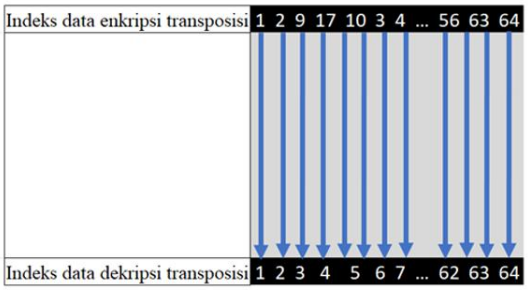
3. Deklarasi parameter terakhir, nilai kk = 0, akan mengeksekusi proses perkalian dengan 64. Kemudian hasilnya akan ditambahkan key(k) sebagai posisi data asli pada data terenkripsi. Enkripsi transposisi didapatkan dari persamaan (1), dimana proses transposisi persamaan ini:

$$\text{DataEnkripsi}[\text{kunci}[k] + (kk * 64) + 1] = \text{DataAsli}[i] \quad (1)$$

Catatan: ditambah 1, karena enkripsi indeks data pertama adalah nomor dari data asli,

Dekripsi Transposisi

Dekripsi pada metode transposisi mengembalikan data hasil enkripsi data ke posisi semula (data sebelum dienkripsi). Dengan demikian, hasil dekripsi transposisi adalah dekripsi data ketika posisinya sesuai dengan kunci. Gambar 7 mengilustrasikan algoritma dekripsi transposisi, yang membutuhkan pemetaan indeks dan enkripsi data.



Gambar 7: Dekripsi ilustrasi indeks pemetaan

Algoritma transposisi dekripsi membutuhkan pemetaan indeks dan enkripsi data. Gambar 8 menunjukkan alur proses dekripsi dengan jumlah kunci pemetaan indeks yang konstan yaitu 64 item data dan urutan data yang tetap dengan pola zig-zag.

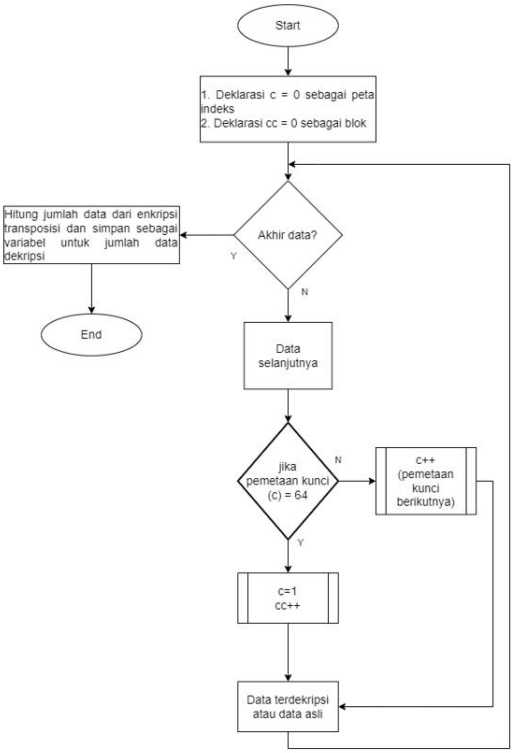
Transposisi dekripsi ditunjukkan pada Gambar 8, yaitu; Deklarasi jumlah data enkripsi yang dimasukkan dalam nilai n sebagai panjang data. Proses transposisi dekripsi akan selesai ketika data enkripsi telah benar-benar berubah, tahapannya:

1. Deklarasi c sebagai nilai kunci dari pemetaan indeks. Nilai c = 0 adalah awal dari proses, yang menandakan indeks pemetaan kunci dari proses transposisi ini adalah 0. Ketika nilai c = 64, kunci pemetaan indeks maksimum, indeks pemetaan akan kembali ke indeks pertama (c = 1), dan nilai cc akan bertambah 1 (cc = cc + 1).

2. Deklarasi terakhir, nilai cc = 0, akan mengeksekusi proses perkalian dengan 64. Kemudian hasilnya akan ditambahkan key(c) sebagai posisi data enkripsi pada data yang didekripsi. Dekripsi transposisi didapatkan dari persamaan (2), dimana proses transposisi persamaan ini.

$$\text{DataDekripsi}[i] = \text{DataEnkripsi}[\text{key}[c] + (cc * 64) + 1] \quad (2)$$

Catatan: ditambah 1, karena enkripsi indeks data pertama adalah nomor dari data asli,diacu.



Gambar 8: Alur proses dekripsi

Hasil dan Pembahasan

Penelitian dilakukan dengan menggunakan perangkat lunak MATLAB. Hasil dari percobaan penelitian kami untuk mengkonduksi lima data yang berbeda berdasarkan jumlah data dan jenis yang digunakan, sebagai berikut:

1. Data input adalah Karakter 'a' yang kemudian dikonversi ke ASCII Biner: {0 1 1 0 0 0 0 0 0 1}.

2. Jumlah data masukan sama dengan angka acak 64 bit.

3. Jumlah data masukan sama dengan 129 bit angka acak.

4. Teks sebagai data masukan dalam jenis data string adalah '*The quick brown fox jumps over the lazy dog*!' Setiap karakter dikonversi ke biner 8-bit. Dengan demikian, jumlah total data masukan adalah 351 bit.

5. Jumlah data masukan sama dengan lima juta bit, seratus juta bit, dan lima ratus juta bit angka acak.

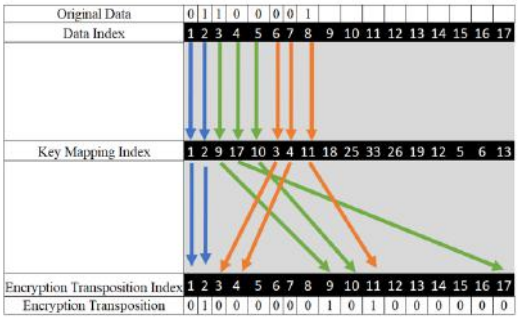
Percobaan untuk Karakter 'a'

Pengujian ini bertujuan untuk menguji data masukan yang jumlahnya di bawah 64 bit, yaitu hanya 8 bit. Percobaan ini menggunakan data input karakter 'a' yang dikonversi ke biner 8 bit pada Gambar 9.

```
DATA =
     0     1     1     0     0     0     0     1
```

Gambar 9: Masukan data karakter 'a' dalam bentuk biner

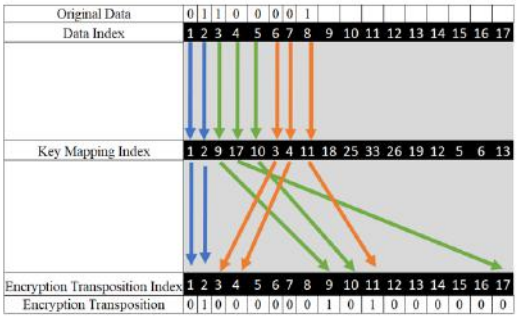
Berdasarkan algoritma enkripsi tersebut, proses transposisi dilakukan sebanyak data asli sebagai data masukan, dimana ilustrasi algoritma enkripsi dan hasil transposisi data enkripsi ditunjukkan pada Gambar 10.



Gambar 10: Ilustrasi proses enkripsi transposisi

Jumlah data yang dienkripsi telah berubah dari 8 bit menjadi 17 bit seperti yang ditunjukkan pada Gambar 10. Data ditransposisi ke jalur pada delapan alokasi pertama pemetaan indeks. Kunci dari pemetaan indeks memiliki 64 bit, namun pada percobaan ini hanya menggunakan 8 bit dari lokasi pertama hingga lokasi kedelapan dari kunci pemetaan indeks, yaitu lokasi yang ditetapkan adalah {1 2 9 17 10 3 4 11}. Beberapa lokasi tidak dapat menempatkan data pada lokasi yang tidak diset, dan lokasi tersebut memiliki data 0. Nilai tertinggi dari himpunan kunci pemetaan indeks adalah 17, sehingga data hasil enkripsi transposisi mengalami penambahan lokasi hingga 17 lokasi pemetaan indeks. Data hasil enkripsi transposisi ditunjukkan pada Gambar 10 (baris terakhir), dan data enkripsi dapat dikirimkan.

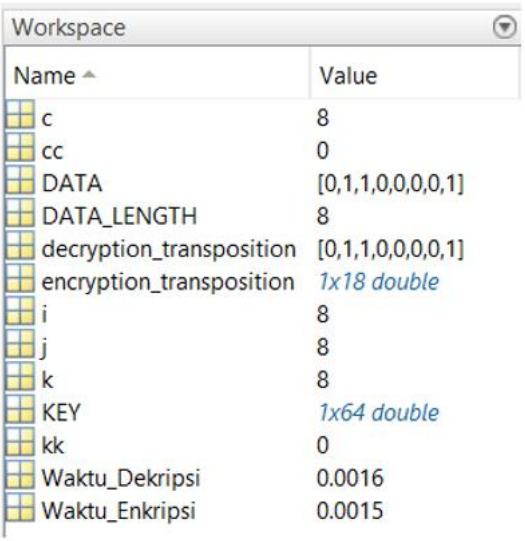
Ketika data enkripsi transposisi telah diterima secara lengkap di level penerima, maka langkah selanjutnya adalah dekripsi transposisi. Pertama, dengan menggunakan set jumlah kunci pemetaan indeks (dengan urutan kunci pemetaan data set yang sama dalam proses enkripsi), dan mendapatkan panjang data yang dienkripsi yang telah ditransmisikan termasuk dalam enkripsi data, proses dekripsi transposisi akan ditransmisikan ke dekripsi data. Gambar 11 merupakan ilustrasi dari proses dekripsi transposisi yang telah dilakukan.



Gambar 11: Ilustrasi proses dekripsi transposisi

Data enkripsi yang berukuran 17-bit akan mengubah data yang dihasilkan sebesar 8 bit pada proses dekripsi. Panjang data dan data yang dihasilkan dari proses dekripsi sama dengan data aslinya. Hal ini berarti metode transposisi telah berhasil untuk dienkripsi dan didekripsi.

Gambar 12 merupakan *workspace* yang dihasilkan dari program enkripsi dan dekripsi transposisi menggunakan MATLAB. Proses kecepatan algoritma tersebut menggunakan fungsi tic toc pada MATLAB. Waktu kinerja pada enkripsi adalah 0,0015 detik dan dekripsi transposisi adalah 0,0016 detik.



Gambar 12: Percobaan Karakter 'a' untuk 8 bit data asli

Percobaan untuk 64 bit data

Pengujian kedua melakukan percobaan enkripsi transposisi dan dekripsi jumlah data masukan sebesar 64 bit. Data masukan berupa kombinasi bit yang dilakukan secara acak pada logika '1' dan '0' (Gambar 13). Kunci data sebagai indeks pemetaan menggunakan pola zig-zag dengan jumlah data 64

bit juga. Percobaan dengan input data 64 bit dan menggunakan kunci dari indeks pemetaan yang telah ditentukan, proses transposisi enkripsi menghasilkan total data enkripsi 64 bit juga (Gambar 14), dan data tersebut telah ditransposisi sesuai dengan pemetaan indeks.

```
DATA =
  Columns 1 through 13
     1     1     1     1     0     0     0     0     1     1     1     1     0
  Columns 14 through 26
     0     0     0     1     1     1     1     0     0     0     0     1     1
  Columns 27 through 39
     1     1     0     0     0     0     1     1     1     1     0     0     0
  Columns 40 through 52
     0     1     1     1     1     0     0     0     0     1     1     1     1
  Columns 53 through 64
     0     0     0     0     1     1     1     1     0     0     0     0
```

Gambar 13: Data masukan 64 bit

Proses transposisi dekripsi, di mana data masukannya adalah data enkripsi yang telah diperoleh sebelumnya sebanyak 64 bit, menggunakan data kunci pemetaan indeks dengan pola zig-zag. Hasil data dekripsi ditunjukkan pada Gambar 15 didapatkan jumlah data dekripsi sebesar 64 bit, yang mentransposisikan data dari pemetaan indeks ke data yang didekripsi. Hasil data dekripsi yang diperoleh sama dengan data masukan seperti data asli.

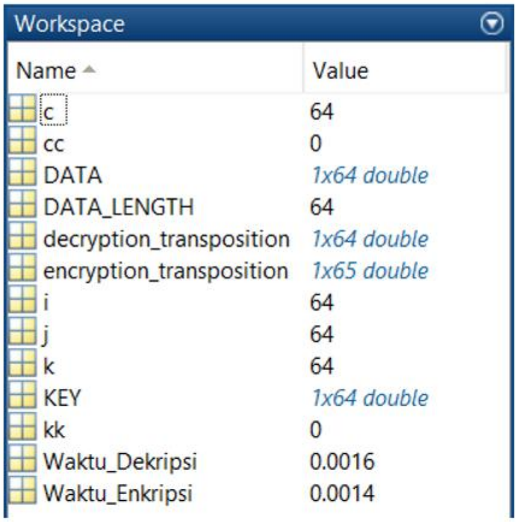
```
encryption_transposition =
  Columns 1 through 13
    64     1     1     0     0     0     0     1     0     1     0     0     0
  Columns 14 through 26
     1     1     0     1     1     1     0     1     1     0     1     1     1
  Columns 27 through 39
     1     1     1     0     1     0     0     1     1     0     1     0     0
  Columns 40 through 52
     0     0     0     0     1     0     0     1     0     0     0     1     0
  Columns 53 through 65
     0     1     1     1     0     1     0     1     1     1     1     0     0
```

Gambar 14: Percobaan Karakter 'a' untuk 8 bit data asli

```
decryption_transposition =
  Columns 1 through 13
     1     1     1     1     0     0     0     0     1     1     1     1     0
  Columns 14 through 26
     0     0     0     1     1     1     1     0     0     0     0     1     1
  Columns 27 through 39
     1     1     0     0     0     0     1     1     1     1     0     0     0
  Columns 40 through 52
     0     1     1     1     1     0     0     0     0     1     1     1     1
  Columns 53 through 64
     0     0     0     0     1     1     1     1     0     0     0     0
```

Gambar 15: Data dekrpsi 64 bit

Hasil enkripsi dan dekripsi transposisi ditunjukkan pada Gambar 16 waktu kinerja pada enkripsi adalah 0,0014 detik dan dekripsi transposisi adalah 0,0016 detik. Jika dibandingkan dengan percobaan sebelumnya yang data masukannya hanya 8 bit, waktu kinerja pada enkripsi adalah 0,0015 detik. Kesimpulannya adalah bahwa pada data masukan 64 bit, algoritma enkripsi transposisi dan dekripsi dapat bekerja lebih baik.

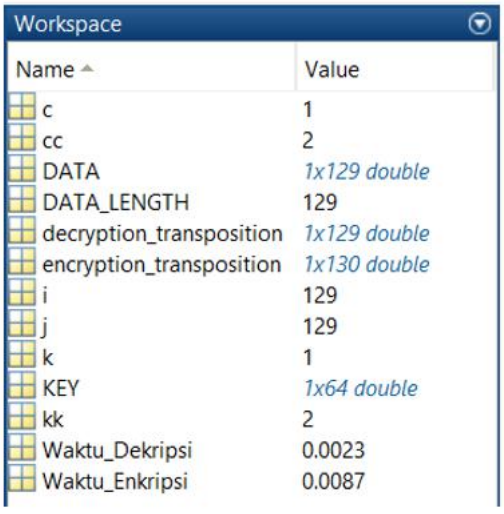


Gambar 16: Percobaan menjalankan dan hasil waktu untuk data asli 64 bit

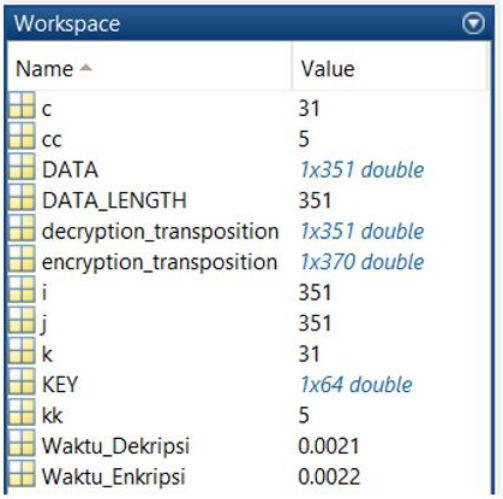
Percobaan untuk 129 bit data biner acak

Data *input* ditransposisi lebih banyak dua kali lipat dari percobaan kedua pada percobaan ini. Proses transposisi membutuhkan tiga kali perulangan pada sebuah kunci pemetaan indeks. Perulangan pertama dilakukan pada pemetaan indeks ke-1 hingga indeks ke-64 pada kunci tersebut. Kemudian, perulangan kedua dilakukan pada pemetaan indeks ke-65 hingga indeks ke-128 pada kunci tersebut. Terakhir, perulangan ketiga dilakukan pada pemetaan indeks ke-129 dengan hanya menggunakan satu kunci pemetaan indeks.

Pada saat yang sama, data *input* sebagai data asli dalam percobaan ini adalah 129 bit. Hasil enkripsi dan dekripsi transposisi ditunjukkan pada Gambar 17 untuk waktu kinerja dalam enkripsi dan dekripsi transposisi adalah 0,002 detik.



Gambar 17: Percobaan menjalankan dan hasil waktu untuk 129 bit data asli.

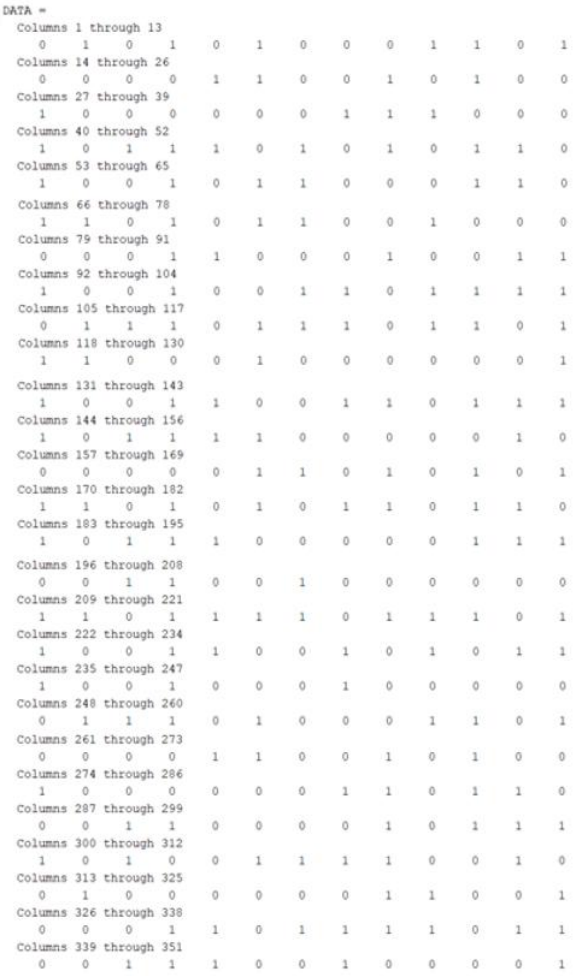


Gambar 18: Percobaan menjalankan dan hasil waktu untuk 351-bit data asli

Percobaan untuk kalimat "*The quick brown fox jumps over the lazy dog*!" dikonversi ke biner 8-bit

Data masukan pada percobaan ini telah dikonversi menjadi 8-bit dalam kode ASCII sehingga total data masukan adalah 351. Proses transposisi memperoleh lima kali loop, yang ditransposisi sesuai dengan kunci baru dan data enkripsi transposisi berubah menjadi 369 bit (lihat Gambar 18; nilainya +1, karena jumlah data yang dienkripsi termasuk data asli) sebagai data terenkripsi. Akhirnya, hasil data yang didekripsi ditransposisikan ke data asli sebesar 351 bit.

Gambar 19 merupakan data masukan kalimat "*The quick brown fox jumps over the lazy dog!*" pada MATLAB. Gambar 20 merupakan hasil enkripsi kalimat tersebut. Gambar 21 merupakan hasil dekripsi kalimat. Hasil data masukan dan hasil dekripsi mempunyai bit biner yang sama. Pada pengujian sebelumnya, data input adalah 129 bit dan hanya mengalami tiga kali loop untuk pemetaan indeks. Selanjutnya, hasil pengujian yang menggunakan data 351 bit dari Gambar 18 pada nilai 'kk' atau nilai 'cc' adalah 5, dan dapat diartikan bahwa hasil iterasi pemetaan indeks sebanyak lima kali pengulangan (pemetaan indeks). Dengan demikian, meskipun perbedaan jumlah data masukan lebih dari dua kali lipat jumlah data masukan dibandingkan dengan pengujian sebelumnya, namun total waktu prosesnya sama dan dapat dilihat pada Gambar 18 untuk waktu kinerja pada kedua enkripsi adalah 0,0022 detik dan dekripsi adalah 0,0022 detik.

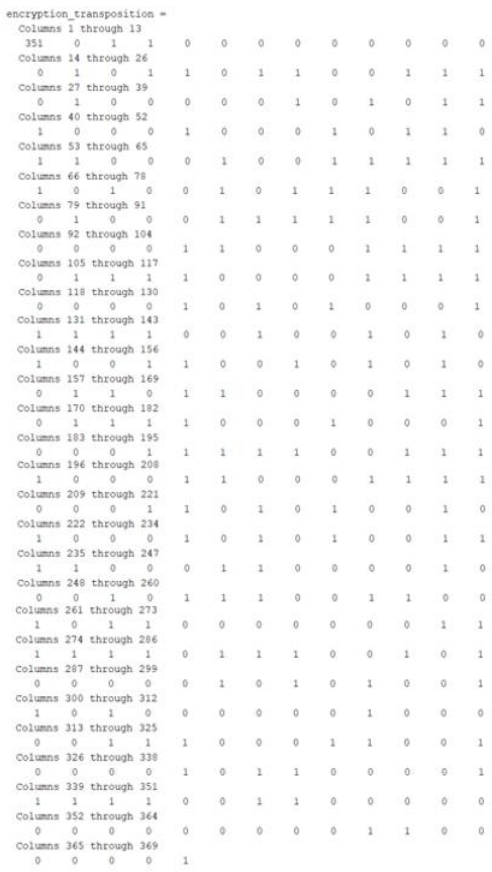


Gambar 19: Data masukan kalimat "The quick brown fox jumps over the lazy dog!"

Percobaan untuk jumlah data masukkan lima juta bit, seratus juta bit, dan lima ratus juta bit bilangan acak

Dalam percobaan ini, data menggunakan fungsi bilangan bulat acak dengan jarak data 0 hingga 1. Dengan demikian, data input adalah lima juta bit (Gambar 22), seratus juta bit (Gambar 23), dan lima ratus juta bit (Gambar 24).

Pada enkripsi dan dekripsi, tiga kali percobaan dengan data masukan yang luas berhasil dilakukan dengan menggunakan algoritma transposisi enkripsi dan dekripsi. Hasil data yang telah dienkripsi ditransposisi ke indeks pemetaan dan ditransposisi lagi menjadi data asli. Kemampuan algoritma untuk memproses pada kecepatan yang sangat tinggi dapat memberikan kepastian dalam mengamankan data. Pengujian berdasarkan jumlah data masukan yang lebih signifikan bertujuan untuk mendapatkan waktu pemrosesan dalam transposisi enkripsi-dekripsi.



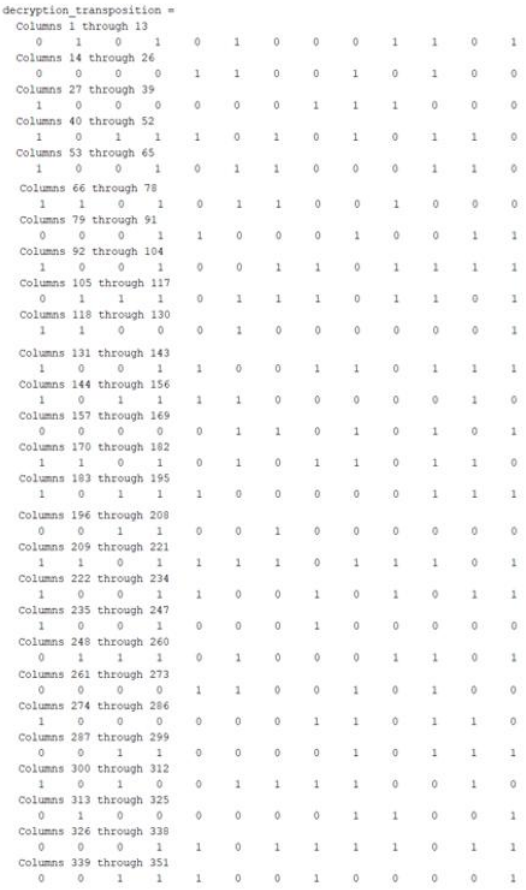
Gambar 20: hasil enkripsi kalimat "The quick brown fox jumps over the lazy dog!"

Hasil pengujian dengan lima juta bit data (Gambar 22), total waktu pemrosesan untuk enkripsi adalah 0,2251 detik, sedangkan untuk proses dekripsi adalah 0,3144 detik. Pengujian dengan seratus juta bit data masukan menghasilkan waktu proses enkripsi 4,3996 detik, namun waktu proses dekripsi 6,3219 detik. Kemudian pada pengujian dengan memberikan data masukan sebesar lima ratus juta bit, durasi proses enkripsi adalah 76,5297 detik, sedangkan proses dekripsi membutuhkan waktu 54,8092 detik. Dengan demikian, selisih waktu untuk mengeksekusi proses transposisi enkripsi selalu lebih kecil dibandingkan dengan proses transposisi dekripsi.

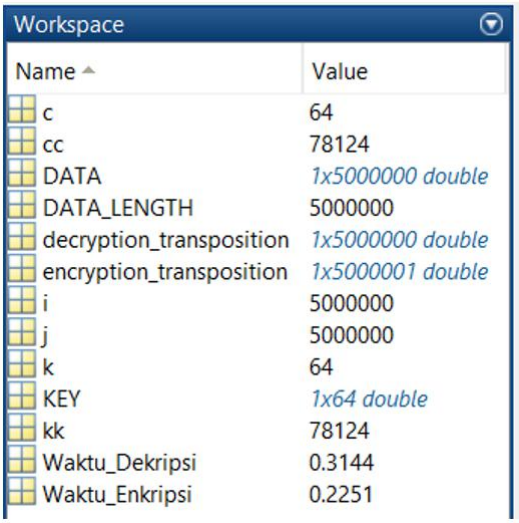
Tabel 1: Hasil kecepatan enkripsi dekripsi kunci indeks pemetaan

Jumlah bit data masukan	Kecepatan enkripsi (detik)	Kecepatan Dekripsi (detik)
8	0,0015	0,0016
64	0,0014	0,0016
129	0,0087	0,0023
351	0,0022	0,0021
5.000.000	0,2251	0,3144
100.000.000	4,3996	6,3219
500.000.000	76,5297	54,8092

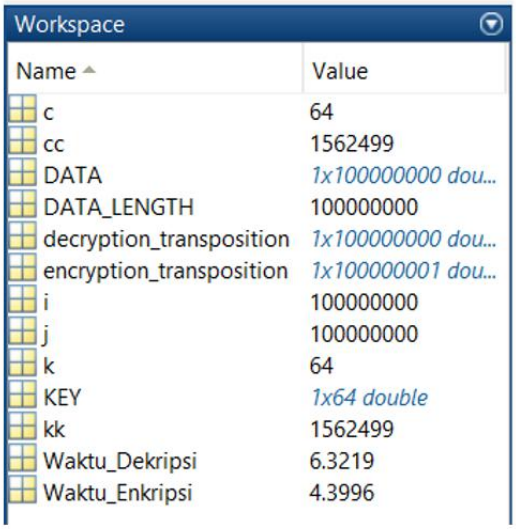
Table 1 menunjukkan hasil kecepatan enkripsi dan dekripsi menggunakan 64 angka kunci indeks pemetaan. Percobaan yang dilakukan kurang dari 1 detik semua menggunakan perangkat MATLAB kecuali jumlah data 100juta dan 500 juta. Jumlah angka 500juta dipakai karena jumlah data tersebut adalah jumlah yang diperbolehkan pada MATLAB.



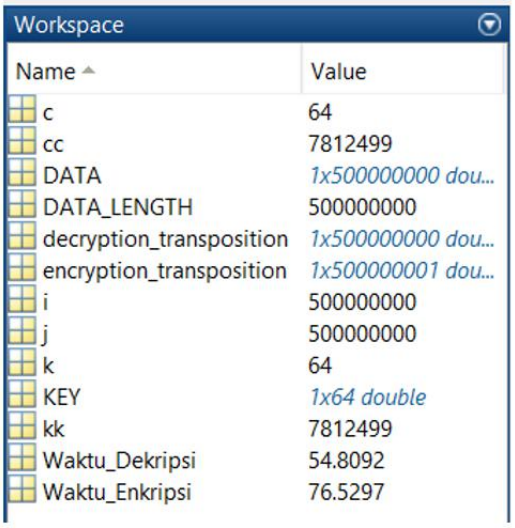
Gambar 21: Hasil dekripsi kalimat "The quick brown fox jumps over the lazy dog!"



Gambar 22: Hasil enkripsi-dekripsi 5 juta bit data pada workspace MATLAB



Gambar 23: Hasil enkripsi-dekripsi 100 juta bit data pada workspace MATLAB



Gambar 24: Hasil enkripsi-dekripsi 500 juta bit data pada workspace MATLAB

Penutup

Pada penelitian ini, transposisi dilakukan dengan menggunakan pola zig-zag sebagai indeks pemetaan. Selain itu, jumlah data yang digunakan juga berbeda, yaitu: 8 bit, 64 bit, 129 bit, 351 bit, dan data masukan lainnya dalam mengkategorikan data yang luas dalam beberapa juta. Berdasarkan pengujian dengan jumlah data pesan dinamis yang tidak terbatas, algoritma transposisi dapat dilakukan pada enkripsi dan dekripsi. Algoritma dan formula yang dihasilkan juga sederhana. Hasil percobaan dengan menggunakan fungsi tic toc pada MATLAB, proses tercepat untuk proses waktu 64-bit adalah 0,0014 detik untuk enkripsi transposisi dan 0,0016 detik untuk dekripsi transposisi. Waktu proses terlama terdapat pada 500 juta bit data dengan durasi proses enkripsi adalah 76,5297 detik, sedangkan proses dekripsi membutuhkan waktu 54,8092 detik.

Berdasarkan hasil percobaan terhadap waktu proses enkripsi atau dekripsi dengan hasil yang cepat, maka pengembangan algoritma transposisi enkripsi dan dekripsi dapat memberikan kontribusi terhadap keamanan data di masa yang akan datang. Di masa depan, metode ini dapat ditambahkan dengan enkripsi multi-level atau diimplementasikan pada chip perangkat keras.

Daftar Pustaka

[1] A. Djamalilleil, M. Muslim, Y. Salim, E. I. Alwi, H. Azis and Herman, " Modified Transposition Cipher Algorithm for Images Encryption", The 2nd East Indonesia Conference on Computer and Informationa Technology (EIConCIT), Makassar, Sulawesi Selatan, 2018.

[2] M. Annalakshmi and P. A. Padmapriya, "Zigzag Ciphers: A Novel Transposition Method", in IJCA Proceedings on International Conference on Computing and Information Technology, (IC2IT-2013), 2013.

[3] P. Poonia and P. Kantha, "Comparative Study of Various Substitution and Transposition Encryption Techniques", International Journal of Computer Applications (0975 - 8887), vol. 145, no. 10, hal. 24-27, Juli 2016.

[4] K. Devi and G.N. Harshini, "Analysis of Substitution and Transposition Cipher", IJRAR-International Journal of Research and Analytical Reviews, vol. 6, no. 2, hal. 549-555, Juni 2019.

[5] J.-S. Kim and H.-O. Lee, "An Algorithm for One-to-One Mapping Matrix-star Graph into Transposition Graph", Journal Korea Institute of Information and Communication Engineering, vol. 18, no. 5, hal. 1110-1115, Mei 2014.

[6] M. A. Budiman, Amalia dan N. I. Chayanie, "Implementasi Algoritma RC4+ dan Algoritma Zig-zag pada Skema Enkripsi Super untuk Keamanan Teks", dalam Konferensi Internasional Komputasi dan Informatika Terapan ke-2, Medan, Indonesia, 2017.

[7] S. Godara, S. Kundu dan R. Kaler, "An Improved Algorithmic Implementation of Rail Fence Cipher",International Journal of Future Generation Communication and Networking, vol. 11, No. 2, hlm. 23-32, 12 Maret 2018.

[8] S. Islam, "Information Encryption by Zigzag Rule with Dynamic Block and Key", Dhaka University of Engineering & Technology, Gazipur, Gazipur, 2011.

[9] A. P. U. Siahaan, "Kriptografi Pagar Rel dalam Mengamankan Informasi", Jurnal Penelitian Ilmiah & Teknik, vol. 7, no. 7, hal. 535-538, Juli 2016.

[10] R. Candra, S. Madenda, S. A. Sudiro and M. Subali, "The implementation of an efficient zigzag scan", Journal of Telecommunication, Electronic and Computer Engineering (JTEC), vol. 9, no. 2, hal. 95-98, April 2017.

[11] M. Y. T. Irsan and S. C. Antoro, "Text Encryption Algorithm based on Chaotic Map", Journal of Physics: Conference Series, Volume 1341 Issue 6 Pages 062023, 2019.

[12] A. Murugan and R. Thilagavathy, "Triple Encryption Scheme with Parallel Zigzag Pattern for Cloud Data Storage Scheme", International Journal of Applied Engineering Research, vol. 13, no. 22, pp. 15766-15772, 2018.

[13] L. Aksoy, P. Flores and J. Monteiro, "A novel method for the approximation of multiplierless constant matrix vector multiplication," EURASIP Journal on Embedded Systems (2016) 2016: 12, vol. 2016, no. 12, hal. 1-11, 20 Mei 2016.