

BAB II TINJAUAN PUSTAKA

1.1 Algoritma

Ditinjau dari asal usul katanya, kata algoritma sendiri mempunyai sejarah yang aneh. Orang hanya menemukan kata *algorism* yang berarti proses menghitung dengan angka arab. Para ahli bahasa berusaha menemukan asal kata ini namun hasilnya kurang memuaskan. Akhirnya para ahli sejarah matematika menemukan asal kata tersebut yang berasal dari penulis buku arab yang terkenal yaitu Abu Ja'far Muhammad Ibnu Musa Al-Khuwarizmi, menulis buku yang berjudul Kitab Aljabar Walmuqabala yang artinya “buku pemugaran dan pengurangan”

Dari judul buku itu juga diperoleh akar kata “Aljabar” (*Algebro*) perubahan dari kata *algorism* menjadi *algorithm* muncul karena kata *algorism* sering dikelirukan dengan *arithmetic*, sehingga akhiran *-sm* berubah menjadi *-thm*. Lambat laun kata *algorithm* berangsur-angsur dipakai sebagai metode perhitungan (komputasi) secara umum, sehingga kehilangan makna kata aslinya. Dalam bahasa Indonesia, kata *algorithm* diserap menjadi algoritma. “Algoritma adalah urutan langkah-langkah logis penyelesaian masalah yang disusun secara sistematis dan logis”. Kata logis merupakan kata kunci dalam algoritma dimana langkah-langkah dalam algoritma harus logis dan harus dapat ditentukan bernilai salah atau benar (Munir, 2006).

1.2 File Teks

File teks yaitu *file* yang di dalamnya berisi informasi-informasi dalam bentuk teks. Masukan dari data teks terdiri dari karakter, angka, huruf dan tanda baca. *Input* dan *output* data teks direpresentasikan sebagai sistem kode atau set karakter yang dikenal oleh sistem komputer (Latifah *et al.*, 2017). Pada sistem komputer terdapat beberapa macam set karakter yang biasa digunakan seperti *ASCII*, *Unicode*, *EBCDIC*. Salah satu standar internasional dalam kode huruf dan simbol yang bersifat *universal* yaitu *ASCII* (*American Code for Information Interchange*).

Berikut ini merupakan beberapa ekstensi dari *file* teks:

1. Teks Biasa (*.txt)

Teks Biasa (*.txt) merupakan jenis berkas yang di dalamnya mengandung *editor* teks yang diformat menggunakan sistem kode pada *ASCII*. Berkas ini hanya terdiri dari angka, karakter, tanda baca, tabulasi, dan pemisah baris untuk sistem *input* dan *output*.

2. File (*.doc)

File (*.doc) pertama kali muncul pada tahun 1980 yang merupakan singkatan dari dokumen. Ukuran file (*.doc) lebih besar dibandingkan file (*.docx). File (*.doc) tidak mudah dikonversi tanpa bantuan *software* atau *external converter*.

3. File (*.docx)

File (*.docx) merupakan file yang dikembangkan setelah versi (*.doc). Ukuran dokumen dari file (*.docx) lebih kecil dibandingkan dengan file (*.doc) sehingga lebih cepat dalam proses pengiriman. File (*.docx) dapat dengan mudah dikonversi ke format doc, html, rtf dan format lainnya.

1.3 Kriptografi

Kriptografi (*cryptography*) berasal dari bahasa Yunani, *cryptos* artinya rahasia, sedangkan *graphein* artinya tulisan. Jadi, kriptografi berarti tulisan rahasia. Kriptografi merupakan bidang ilmu yang mempelajari tentang metode untuk mengirim pesan secara rahasia (yaitu di enkripsi atau disamarkan) sehingga hanya penerima pesan yang dituju yang dapat menghapus penyamaran dan membaca pesan. Dalam kriptografi, proses menyandikan plainteks menjadi cipherteks disebut dengan enkripsi (*encryption*). Sedangkan proses pengembalian cipherteks menjadi plainteks semula dinamakan dekripsi (*decryption*). Parameter yang digunakan untuk transformasi enkripsi dan dekripsi disebut dengan kunci (*key*) (Munir, 2006).

Kriptografi bertujuan untuk memberikan layanan keamanan sebagai berikut:

1. *Confidentiality*. Informasi dirahasiakan dari semua pihak yang tidak berwenang.
2. *Integrity*. Memungkinkan penerima pesan memeriksa bahwa data tersebut tidak dimodifikasi selama pengiriman; penyusup tidak dapat mengganti pesan yang salah dengan yang asli.
3. *Authentication*. Memungkinkan penerima pesan menegaskan keaslian dari data tersebut; penyusup tidak dapat menyamar sebagai orang lain.
4. *Non-repudiation*. Setiap entitas yang berkomunikasi tidak dapat menolak atau menyangkal atas data yang telah dikirim atau diterima.

Kriptografi adalah ilmu dan seni untuk menjaga kerahasiaan pesan dengan cara menyandikannya ke dalam bentuk yang tidak dapat dimengerti lagi maknanya. Dalam ilmu kriptografi, terdapat dua buah proses yaitu melakukan enkripsi dan dekripsi. Pesan yang akan dienkripsi disebut sebagai *plainteks* (teks biasa). Disebut demikian karena informasi ini dengan mudah dapat dibaca dan dipahami oleh siapa saja (Pabokory, 2016). Algoritma yang dipakai untuk mengenkripsi dan mendekripsi sebuah *plainteks* melibatkan penggunaan suatu bentuk kunci. Pesan *plainteks* yang telah dienkripsi (atau dikodekan) dikenal sebagai *ciphertext* (teks sandi).

Di dalam kriptografi kita akan sering menemukan berbagai istilah atau *terminology*. Beberapa istilah yang harus diketahui yaitu :

1. Pesan, Plainteks dan Cipherteks

Pesan (*message*) adalah data atau informasi yang dapat dibaca dan dimengerti maknanya. Nama lain untuk pesan adalah (*plainteks*) atau teks jelas (*cleartext*).

2. Pengirim dan Penerima

Komunikasi data melibatkan pertukaran pesan antara dua entitas. Pengirim (*sender*) adalah entitas yang mengirim pesan kepada entitas lainnya. Penerima (*receiver*) adalah entitas yang menerima pesan.

3. Enkripsi dan dekripsi

Proses menyandikan plainteks menjadi cipherteks disebut enkripsi (*encryption*) atau *enciphering* (standard nama menurut ISO 7498-2). Sedangkan proses mengembalikan cipherteks menjadi plainteks semula disebut dekripsi (*decryption*) atau *deciphering* (standard nama menurut ISO 7498-2).

4. *Cipher* dan kunci

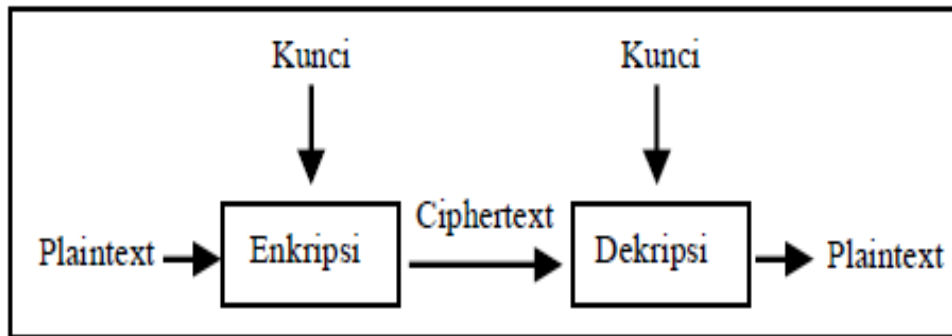
Algoritma kriptografi disebut juga *cipher*, yaitu aturan untuk enkripsi dan dekripsi, atau fungsi matematika yang digunakan untuk enkripsi dan dekripsi. Beberapa *cipher* memerlukan algoritma yang berbeda untuk enkripsi dan dekripsi. Konsep matematis yang mendasari algoritma kriptografi adalah relasi antara dua buah himpunan yang berisi elemen-elemen plainteks dan himpunan yang berisi cipherteks. Enkripsi dan dekripsi merupakan fungsi yang memetakan elemen-elemen antara dua himpunan tersebut. Misalkan P menyatakan plainteks dan C menyatakan cipherteks, maka :

$E(P) = C \rightarrow$ fungsi enkripsi E memetakan P ke C

$D(C) = P \rightarrow$ fungsi dekripsi D memetakan C ke P

Karena proses enkripsi kemudian dekripsi mengembalikan pesan ke pesan asal, maka persamaan $D(E(P)) = P$ harus benar.

Kriptografi mengatasi masalah keamanan data dengan menggunakan kunci, yang dalam hal ini algoritma tidak dirahasiakan lagi, tetapi kunci harus tetap dijaga kerahasiaannya. Kunci (*key*) adalah parameter yang digunakan untuk transformasi enkripsi dan dekripsi. Kunci biasanya berupa *string* atau deretan bilangan. Dengan menggunakan kunci K , maka fungsi enkripsi dan dekripsi dapat ditulis sebagai skema diperlihatkan pada gambar 2.1.



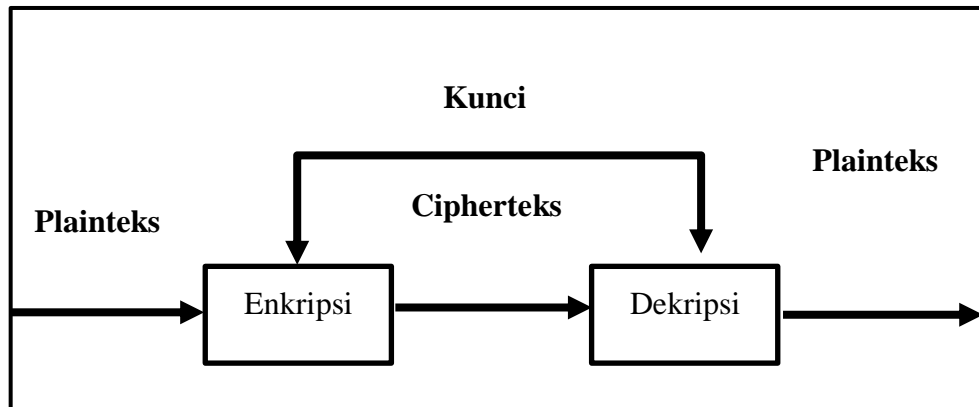
Gambar 2.1 Skema Enkripsi dan Deskripsi dengan Menggunakan Kunci
(Pabokory, 2016)

1.3.1 Jenis-Jenis Algoritma Kriptografi

Ada dua jenis algoritma kriptografi yaitu algoritma simetris dan algoritma asimetris. Pada kriptografi simetris maupun kriptografi asimetris tidak ada keamanan yang bisa terjamin tanpa syarat. Suatu metode enkripsi tidak ada yang praktis. Maka, untuk semua protokol pada kriptografi, keamanan bergantung pada asumsi perhitungan (Setyaningsih, 2015).

1. Algoritma Simetris

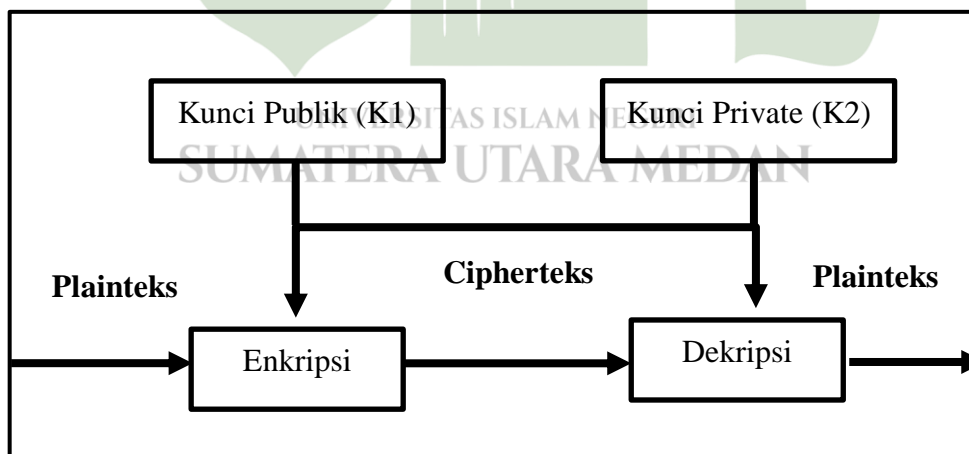
Algoritma simetris juga sering disebut algoritma klasik, dimana untuk melakukan proses enkripsi dan dekripsi dapat menggunakan kunci yang sama. Algoritma ini disebut juga algoritma kunci-privat (*secret-key*), algoritma kunci-tunggal (*single-key*), algoritma satu kunci (*one-key*), dimana pengirim dan penerima sepakat dengan sebuah kunci sebelum berkomunikasi dengan aman. Keamanan algoritma simetris terletak pada kunci. Algoritma simetri dapat dibagi dua, yaitu *stream cipher* dan *block cipher* (Setyaningsih, 2015). Contoh algoritma simetris yaitu *Caesar Cipher*, *Affine Cipher*, *Vigenere Cipher*, dll. Skema proses enkripsi dan dekripsi algoritma simetris dapat dilihat pada gambar 2.2.



Gambar 2.2 Skema Algoritma Simetris (Arifah & Basuki 2017)

2. Algoritma Asimetris

Algoritma asimetris (*public key*) merupakan algoritma kriptografi yang menggunakan sepasang kunci dalam melakukan enkripsi dan dekripsi, yaitu kunci publik (*public key*) dan kunci privat (*private key*). Kunci publik digunakan untuk proses enkripsi, dan kunci privat digunakan untuk proses dekripsi. Dalam algoritma asimetris, kunci publik bisa disebar-luaskan sedangkan kunci privat harus tetap dirahasiakan (Ratna, 2018). Contoh algoritma asimetris yaitu Rivest-Shamir-Adleman (RSA), Diffie-Helman, dan Elgamal. Skema proses enkripsi dan dekripsi algoritma asimetris dapat dilihat pada gambar 2.3.



Gambar 2.3 Skema Algoritma Asimetris (Arifah & Basuki 2017)

1.3.2 Sejarah Kriptografi

Sejarah kriptografi sebagian besar merupakan sejarah kriptografi klasik, yaitu metode enkripsi yang menggunakan kertas dan pensil atau mungkin dengan bantuan alat mekanik sederhana. Secara umum algoritma kriptografi klasik dikelompokkan menjadi dua kategori, yaitu algoritma transposisi (*transposition cipher*) dan algoritma substitusi (*substitution cipher*). *Cipher* transposisi mengubah susunan huruf-huruf di dalam pesan, sedangkan *cipher* substitusi mengganti setiap huruf atau kelompok huruf dengan sebuah huruf atau kelompok huruf lain. Adapun tujuan kriptografi untuk memberi layanan keamanan (Afifah, 2018). Yang dinamakan aspek-aspek keamanan adalah:

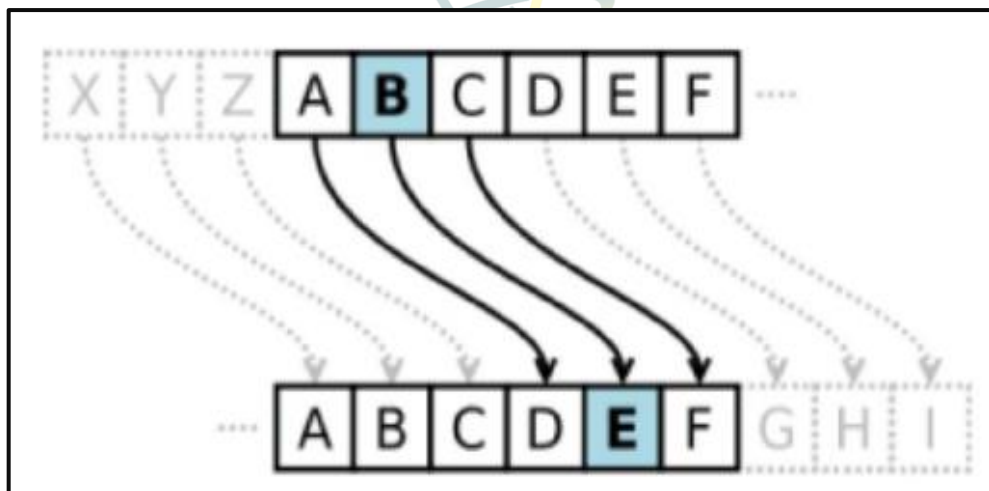
1. Kerahasiaan (confidentiality)
Adalah layanan yang ditujukan untuk menjaga agar pesan tidak dapat dibaca oleh pihak-pihak yang tidak berhak.
2. Integritas data (data integrity)
Adalah layanan yang menjamin bahwa pesan masih asli atau belum pernah dimanipulasi selama pengiriman.
3. Otentikasi (authentication)
Adalah layanan yang berhubungan dengan identifikasi, baik mengidentifikasi kebenaran pihak-pihak yang berkomunikasi (user authentication).
4. Non-repudiation
Adalah layanan untuk menjaga entitas yang berkomunikasi melakukan penyangkalan.

1.3.3 Algoritma Kriptografi *Caesar Cipher*

Substitusi kode yang pertama dalam dunia penyandian dikenal dengan *Caesar cipher*, karena penyandian ini terjadi pada saat pemerintahan Yulius Caesar. Dengan mengganti posisi huruf awal dengan alfabet atau disebut dengan algoritma ROT13 (Ariyus, 2006). *Caesar cipher* merupakan salah satu algoritma *cipher* tertua dan paling diketahui dalam perkembangan ilmu kriptografi. *Caesar cipher* merupakan salah satu jenis *cipher* substitusi yang membentuk *cipher* dengan cara melakukan penukaran karakter pada plaintext menjadi tepat satu

karakter pada cipherteks. Teknik seperti ini disebut juga sebagai *cipher* abjad tunggal. *Caesar cipher* sangat mudah untuk digunakan. Inti dari algoritma kriptografi ini adalah melakukan pergeseran terhadap semua karakter pada plainteks dengan nilai pergeseran yang sama. Adapun langkah-langkah yang dilakukan untuk membentuk cipherteks dengan *Caesar cipher* adalah :

1. Menentukan besarnya pergeseran karakter yang digunakan dalam membentuk cipherteks ke plainteks.
2. Menukarkan karakter pada plainteks menjadi cipherteks dengan berdasarkan pada pergeseran yang telah ditentukan sebelumnya. Misalnya diketahui bahwa pergeseran = 3, maka huruf A akan digantikan oleh huruf D, huruf B menjadi huruf E, dan seterusnya.



Gambar 2.4 Proses Pergantian Huruf Pada Caesar Cipher

Susunan alphabet setelah digeser sejauh 3 huruf membentuk sebuah tabel substitusi sebagai berikut :

Tabel 2.1 Tabel Substitusi ROT13

Indeks	0	1	2	3	4	5	6	7	8	9	10	11	12
P	A	B	C	D	E	F	G	H	I	J	K	L	M
C	D	E	F	G	H	I	J	K	L	M	N	O	P
Indeks	13	14	15	16	17	18	19	20	21	22	23	24	25
P	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
C	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

Untuk menyandikan sebuah pesan, cukup mencari setiap huruf yang hendak disandikan di alfabet biasa, lalu tuliskan huruf yang sesuai pada alfabet sandi. Untuk memecahkan sandi tersebut gunakan cara sebaliknya. Contoh penyandian sebuah pesan adalah sebagai berikut :

Plainteks : SURATDIBERIKANKEPADADILA

Cipherteks : VXUDWGLEHULNDQNHSDGDGLOD

Dengan mengkodekan setiap huruf alfabet dengan integer : 'A'= 0 , 'B'= 1,..., 'Z'= 25, maka secara matematis pergeseran 3 huruf alfabetik ekuivalen dengan melakukan operasi modulo terhadap plainteks P menjadi cipherteks C dengan persamaan :

$$C = E (P) = (P + 3) \bmod 26 \dots\dots\dots (2.1)$$

Karena ada 26 huruf didalam alphabet. Penerima pesan mengembalikan lagi cipherteks dengan operasi kebalikan, secara matematis dapat dinyatakan dengan persamaan :

$$P = D (C) = (C - 3) \bmod 26 \dots\dots\dots (2.2)$$

Dapat diperhatikan bahwa fungsi D adalah balikan (invers) dari fungsi E, yaitu :

$$D (C) = E^{-1} (P) \dots\dots\dots (2.3)$$

Penggunaan dari *Caesar cipher* ini dapat dimodifikasi dengan mengubah jumlah geseran (bukan hanya 3) dan juga arah geseran. Hal ini dilakukan untuk lebih menyulitkan orang yang ingin menyadap pesan karena penyadap harus mencoba semua kombinasi (26 kemungkinan geser).

Salah satu pengembangan dari *Caesar cipher* adalah ROT13. ROT13 (*Rotate* 13) adalah enkripsi *cipher* substitusi yang umum digunakan di sistem operasi UNIX. Pada sistem ini sebuah huruf digantikan dengan huruf yang letaknya 13 posisi darinya. Sebagai contoh, huruf “A” digantikan dengan huruf “N”, huruf “B” digantikan dengan huruf “O”, dan seterusnya. Secara matematis, hal ini dapat dituliskan sebagai :

$$C = \text{ROT13} (P) \dots\dots\dots (2.4)$$

Untuk mengembalikan kembali ke bentuk semulanya (dekripsi) dilakukan proses enkripsi ROT13 dua kali.

$$P = \text{ROT13} (\text{ROT13} (P)) \dots\dots\dots(2.5)$$

Tabel 2.2 Susunan Alfabet ROT13

A	B	C	D	E	F	G	H	I	J	K	L	M
N	O	P	Q	R	S	T	U	V	W	X	Y	Z

Contoh operasi *Caesar cipher* ROT13 :

$$\text{ROT13} (\text{'HELLO'}) = \text{'URYYB'}$$

$$\text{ROT13} (\text{'URYYB'}) = \text{'HELLO'}$$

$$\text{ROT13} (\text{ROT13} (\text{'HELLO'})) = \text{'HELLO'}$$

ROT13 memang tidak didesain untuk keamanan tingkat tinggi. ROT13, misalnya digunakan untuk menyelubungi isi dari artikel (*posting*) di *Usenet news* yang berbau ofensif. Sehingga hanya orang yang betul-betul ingin membaca dapat melihat isinya. Contoh penggunaan lain adalah untuk menutupi jawaban dari sebuah teka teki (*puzzle*).

Dasar keilmuan dari *Caesar cipher* sebagian besar adalah matematika yang antara lain mencakup teori bilangan, aljabar dan fungsi. Subbab matematika tersebut sudah diajarkan sejak pendidikan sekolah bahkan diperluas lagi di perguruan tinggi. Rumus *Caesar cipher* secara umum :

$$C = E (P) = (P + k) \bmod 26 \dots\dots\dots (2.6)$$

Dan fungsi dekripsi adalah :

$$P = D (C) = (C - k) \bmod 26 \dots\dots\dots(2.7)$$

Catatan:

1. Pergeseran 0 sama dengan pergeseran 26 (susunan huruf tidak berubah).
2. Pergeseran lain untuk $k > 25$ dapat juga dilakukan namun hasilnya akan kongruen dengan bilangan bulat dalam modulo 26. Misalnya $k=37$ kongruen dengan 11 dalam modulus 26, atau $37 \equiv 11 \pmod{26}$.

Persamaan di atas menggunakan subbab matematika teori bilangan khususnya dengan modulus. Operasi modulus adalah sebuah operasi yang menghasilkan sisa pembagian dari suatu bilangan terhadap bilangan lainnya.

Contoh modulus :

$$1 = 7 \pmod{2}$$

$$1 = 5 \pmod{3}$$

1.3.4 Algoritma Rail Fence Cipher (RFC)

Algoritma *Rail Fence Cipher* (rel pagar) adalah merupakan salah satu variasi implementasi *cipher* transposisi. *Rail fence cipher* merupakan salah satu teknik kriptografi yang menggunakan pergeseran posisi dengan menggunakan kata kunci sebagai inti dari algoritma untuk enkripsi dan dekripsi teks. Pada algoritma ini, plainteks dituliskan secara vertikal kebawah sepanjang *n-rails* dan menulis lagi ke kolom baru ketika telah mencapai karakter ke-*n*. Cipherteks yang dihasilkan adalah urutan karakter yang dibaca secara horizontal (Girsang *et al.*, 2019).

Algoritma *Rail Fence Cipher* memiliki baris atas dan baris bawah yang terpisah dalam penulisan *plainteks*. Cara penulisan *plainteks* dilakukan dengan mengurutkan karakter pada baris atas yang kemudian diikuti oleh karakter selanjutnya pada baris bawah dan seterusnya sebanyak *plainteksnya*. Pada penulisan baris bawah dilakukan sebanyak jumlah kunci, apabila penulisan ke baris bawah sudah mencapai banyaknya kunci maka dilanjutkan ke baris atas dan seterusnya. Ketika penulisan ke atas juga telah mencapai banyaknya kunci, maka penulisan dilakukan seperti awal (Ratna, 2018).

Rail Fence Cipher (RFC) adalah algoritma klasik model algoritma transposisi, cipher ini hampir mirip sistem kerjanya dengan algoritma *Zig-zag Cipher*. *Rail Fence Cipher* menggunakan teknik perubahan posisi berdasarkan tingkatan, dimana nilai tingkatan disebut kunci enkripsi dan dekripsi dalam algoritma ini. Dengan menggunakan nilai Kunci = 3, maka proses enkripsi Rail Fence Cipher adalah:

Tabel 2.3 Proses Enkripsi Rail Fence Cipher

N	Plainteks= SELAMAT DATANG UINSU MEDAN									
1	S	A	T	T	G	N	M	M	A	
2		E	M	D	A	U	S	U	E	N
3			L	A	A	N	I	U	T	D

Hasil Enkripsi dengan algoritma *Rail Fence Cipher* adalah:

Ciphertext = SATTGNMMA EMDAUSUEN LAANIUTD

Proses dekripsi adalah nilai kunci yang digunakan adalah dengan menghitung jumlah karakter *ciphertext*, selanjutnya bagikan dengan nilai kunci enkripsi, maka hasilnya sebagai kunci dekripsi.

Ciphertext = SATTGNMMA EMDAUSUEN LAANIUTD

Kunci dekripsi = $26 / 3 = 8.66 \sim 9$

Maka susun karakter *ciphertext* dengan jumlah baris 9, seperti pada gambar berikut ini:

Tabel 2.4 Susunan Ciphertext Baris 9

Baris-1	S	E	L
Baris-2	A	M	A
Baris-3	T	D	A
Baris-4	T	A	N
Baris-5	G	U	I
Baris-6	N	S	U
Baris-7	M	U	T
Baris-8	M	E	D
Baris-9	A	N	

Selanjutnya, menentukan nilai plainteks dengan cara membaca karakter secara baris, seperti pada gambar berikut ini:

Hasil Dekripsi Rail Fence Cipher:

Plainteks = SELAMATDATANGUINSUMEDAN

Plainteks = SELAMAT DATANG UINSU MEDAN

Tabel 2.5 Hasil Dekripsi Rail Fence SELAMAT DATANG UINSU MEDAN

Baris-1	S	E	L
Baris-2	A	M	A
Baris-3	T	D	A
Baris-4	T	A	N
Baris-5	G	U	I
Baris-6	N	S	U
Baris-7	M	U	T
Baris-8	M	E	D
Baris-9	A	N	

Berikut contoh dari *Algoritma Rail Fence Cipher* pada sebuah karakter:

1. Sebagai contoh untuk enkripsi, kita mempunyai kunci sebanyak $n=3$ dan sebuah pesan (*plainteks*) yaitu "SITUMORANG". Maka proses enkripsi karakter dapat dilihat seperti pada tabel 2.6 sebagai berikut:

Tabel 2.6 Proses Enkripsi Rail Fence Cipher

N	Plainteks = SITUMORANG				
1	S	U	R	G	
2		I	M	A	
3			T	O	N

Hasil Enkripsi dengan algoritma *Rail Fence Cipher* adalah:

Ciphertext = SURG IMA TON

Proses dekripsi adalah nilai kunci yang digunakan adalah dengan menghitung jumlah karakter *ciphertext*, selanjutnya bagikan dengan nilai kunci enkripsi, maka hasilnya sebagai kunci dekripsi.

Ciphertext = SURG IMA TON

Kunci dekripsi = $10 / 3 = 3.33 \sim 4$

Maka susun karakter *ciphertext* dengan jumlah baris 3, seperti pada gambar berikut ini:

Tabel 2.7 Susunan Ciphertext Baris 4

Baris-1	S	I	T
Baris-2	U	M	O
Baris-3	R	A	N
Baris-4	G		

Selanjutnya, menentukan nilai plainteks dengan cara membaca karakter secara baris, seperti pada gambar berikut ini:

Hasil Dekripsi Rail Fence Cipher:

Plainteks = SITUMORANG

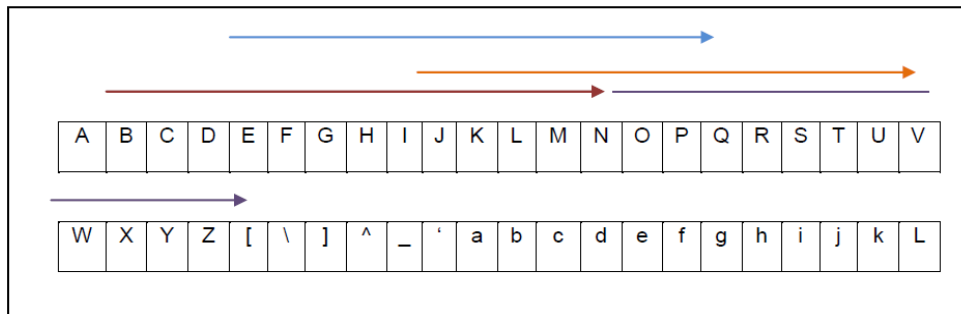
Algoritma *Rail Fence Cipher* mempunyai kelebihan dibandingkan algoritma lainnya dalam proses penulisan plainteks menjadi ciphertexts karena penulisan dapat dilakukan pada baris mana saja. Hal ini akan menambah kerumitan dalam proses enkripsi maupun dekripsi.

1.3.5 Algoritma Rotate 13 (ROT13)

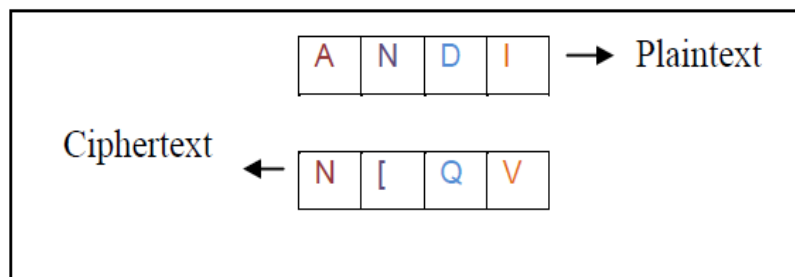
Substitusi kode yang pertama dalam dunia penyandian dikenal dengan *Caesar cipher*, karena penyandian ini terjadi pada saat pemerintahan Yulius Caesar. Dengan mengganti posisi huruf awal dengan alfabet atau disebut dengan algoritma ROT13. *Caesar cipher* merupakan salah satu algoritma *cipher* tertua dan paling diketahui dalam perkembangan ilmu kriptografi. *Caesar cipher* merupakan salah satu jenis *cipher* substitusi yang membentuk *cipher* dengan cara melakukan penukaran karakter pada plainteks menjadi tepat satu karakter pada ciphertexts. Teknik seperti ini disebut juga sebagai *cipher* abjad tunggal. *Caesar cipher* sangat mudah untuk digunakan. Inti dari algoritma kriptografi ini adalah melakukan pergeseran terhadap semua karakter pada plainteks dengan nilai pergeseran yang sama (Aresta *et al.* 2020).

Algoritma ROT13 bekerja dengan menggantikan setiap huruf dengan 13 karakter di depan atau dibelakangnya sesuai dengan alfabet. Pergeseran $k=13$ (huruf A diganti dengan N) agar informasi tidak terbaca dengan sekilas.

Pergeseran karakter pada tabel ASCII dengan menggeser mundur sebanyak 13 karakter.



Gambar 2.5 Alur Pergeseran $k = 13$, Berdasarkan Tabel ASCII



Gambar 2.6 Hasil Pergeseran $k = 13$

Adapun langkah-langkah yang dilakukan untuk membentuk cipherteks dengan *Caesar cipher* adalah :

1. Menentukan besarnya pergeseran karakter yang digunakan dalam membentuk cipherteks ke plainteks.
2. Pergeseran key 13, sehingga abjad "A" menjadi "N" dan sebaliknya abjad "N" menjadi "A". Misalkan plainteks „AKU“. Setelah dilakukannya enkripsi menggunakan ROT13, maka plainteks „AKU“ menjadi „NXH“.

Tabel 2.8 Enkripsi ROT13

PLAINTEXT	A	K	U
CHIPERTEXT	N	X	H

3. Setiap abjad (A-Z) ataupun (a-z) tersebut di dalamnya terdapat kode ASCII. Jadi, setiap plainteks yang menggunakan huruf kapital kita ubah ke huruf kecil dahulu dengan tujuan untuk mempersingkat

representasi nilai dari ASCII tersebut. Namun jika plainteksnya menggunakan huruf kecil maka tidak perlu diubah ke huruf kecil lagi.

4. Kemudian hasil dari pesan yang telah dienkripsi menggunakan ROT 13 kita lakukan kembali substitusi ke dalam bentuk desimal ASCII.

Tabel 2.9 Substitusi Ciphertext ke ASCII

ASCII CHAR	A	K	U
ASCII DECIMAL	78	88	72

5. Hasil dari diatas akan kita lakukan pengenkripsian kembali dengan melakukan pensubtitusian nilai dari ASCII decimal tersebut ke dalam pengindeksan bentuk kedalam abjad.

Tabel 2.10 Substitusi ASCII ke Abjad

Index	Abjad
7	H
8	I
8	I
8	I
7	H
2	C

6. Hasil dari pengenkripsian ke dalam indeks abjad, dilakukan pengenkripsian kembali ke dalam bentuk hexadecimal.

Tabel 2.11 Enkripsi Abjad ke Hexadecimal

ASCII CHAR	H	I	I	I	H	C
ASCII HEX	48	49	49	49	48	3

7. Hasil yang telah disubstitusi menggunakan pensubtitusian ASCII character ke bentuk hexadecimal ASCII, kita lakukan kembali pensubtitusian pengindeksan bentuk kedalam abjad.

Tabel 2.12 Substitusi ASCII Hexa ke Abjad

4	8	4	9	4	9	4	9	4	8	3
E	I	E	J	E	J	E	J	E	I	D

8. Setelah itu dilakukan enkripsi kembali menggunakan ROT 13 dan menghasilkan plainteks yang baru yaitu: EIEJEJEJEIED. Plainteks yang baru inilah yang akan digeser sebanyak 13 kali, maka plainteksnya akan menjadi :

Tabel 2.13 Enkripsi ROT13

E	I	E	J	E	J	E	J	E	I	D
---	---	---	---	---	---	---	---	---	---	---

Hasil dari Enkripsi ROT 13 seperti terlihat pada tabel 2.15

Tabel 2.14 Dekripsi ROT13

R	V	R	W	R	W	R	W	R	V	Q
---	---	---	---	---	---	---	---	---	---	---

1.4 Visual Basic NET 2010

Visual Basic 2010 adalah bahasa pemrograman terbaru yang memudahkan programmer VB/VB .NET beralih dari VB 2008 (Budiharto & Lisangan. 2012). Visual Basic 2010 merupakan salah satu aplikasi pemrograman visual yang dibuat oleh Microsoft. Visual Basic 2010 merupakan bagian dari sebuah suite aplikasi pemrograman bernama Visual Studio 2010. Suite aplikasi ini adalah *suite* aplikasi paling mutakhir yang dibuat oleh Microsoft. Visual Basic 2010 sudah mendukung konsep pemrograman berorientasi objek (*Object Oriented Programming*). Dalam Visual Basic 2008 akan dikenal konsep objek, kelas (*class*), pewarisan (*inheritance*), *name space* dan lain-lain.

Program Visual Basic adalah bahasa pemrograman yang paling mudah dikuasai oleh para pemula. Dalam program Visual Basic 2010 (disingkat VB 2010), menawarkan banyak kemudahan lagi dibandingkan versi-versi sebelumnya, antara lain teknik pemrograman dapat dibuat lebih terstruktur dan lebih banyak bantuan dalam pemrograman. Jauh lebih mudah untuk menguasainya dibandingkan dengan versinya yang terdahulu, yaitu Visual Basic 6 (disingkat VB6).

Ada banyak perubahan dalam VB 2010 ini dibandingkan VB6, antara lain:

1. Bahasa pemrograman sekarang benar-benar bahasa berbasis objek (*Object Oriented Programming*), sedangkan VB6 bukan bahasa berbasis objek.
2. Aplikasi dan komponen yang ditulis di VB 2010 mempunyai akses penuh ke *Net Framework*. Sedangkan di VB6 tidak dikenal atau tidak menggunakan *Net Framework*.
3. Semua Semua aplikasi yang dibuat beroperasi dalam manajemen *Common Language Runtime (CLR)*.

Net Framework sendiri, yang sekarang sudah versi 4.0 adalah suatu himpunan file-file pustaka yang telah terorganisir dan berguna sebagai fasilitas untuk sistem dan aplikasi. Sehingga seorang *programmer* tidak perlu lagi menghafal fungsi-fungsi Windows API untuk akses sistem, seperti di dalam bahasa VB6 karena sudah diorganisir oleh *Net Framework*. Hampir semua fungsi Windows API tersebut telah dijadikan *object-object* yang dapat dengan mudah digunakan dan ditemukan oleh *programmer* VB 2010. Pemrograman berbasis objek (OOP) sendiri adalah suatu pendekatan ke arah struktur pengembangan aplikasi berdasarkan objek. Objek tersebut dapat berupa prosedur, *event*, ataupun *variable*. Objek satu dapat menjadi bawahan *object* lainnya berdasarkan susunan fungsinya. Artinya suatu objek terdepan terdiri atas beberapa Objek yang memiliki tugas lebih sempit, dan antar objek dapat saling berinteraksi dalam melaksanakan tugas tertentu.

Contoh kode Visual Basic yang OOP adalah:

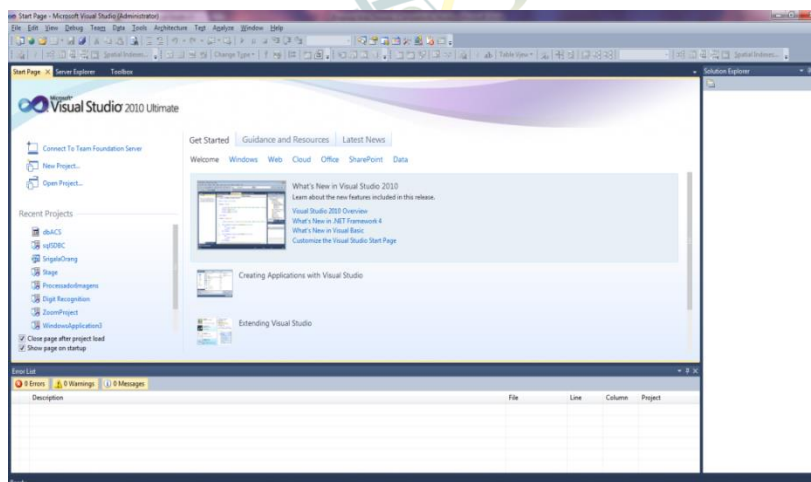
```
Dim Masukan as String= "Selamat Membaca"
```

```
Dim nilai as String = Strings.Left(Masukan, 3).
```

Objek masukan bertipe *string*, yang isi *text*-nya adalah "Selamat Membaca". Kemudian pada baris berikutnya digunakan *object left* untuk memprosesnya. *Object left* sendiri dapat diakses melalui *object Strings*. Hasil proses *object left* terhadap *object* masukan, yaitu mengambil 3 karakter *string* kirinya untuk kemudian hasil proses tersebut dimasukkan dalam *object* nilai yang bertipe *string* pula. *Common Language Runtime (CLR)* adalah suatu *runtime*

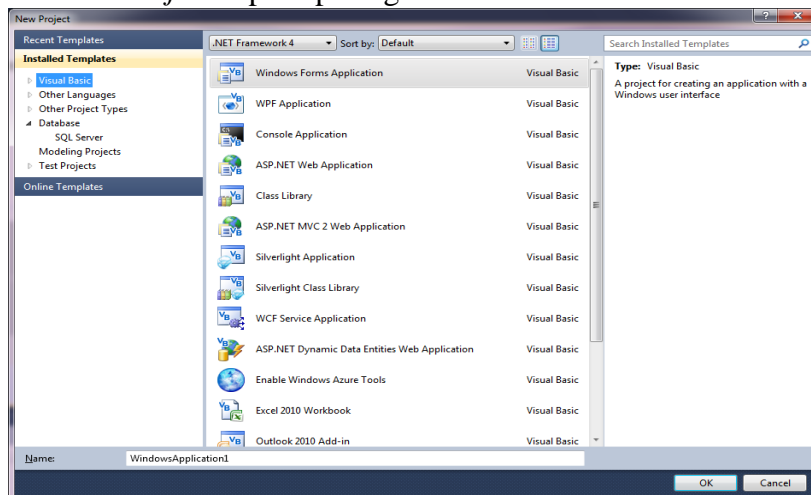
lingkungan yang memproses, melaksanakan, dan mengatur kode dasar Visual Basic. Mirip dengan *runtime* Visual Basic tradisional, yaitu VBRUN300.dll atau MSVBVM60.dll, tetapi kemampuannya saja lebih ditingkatkan sehingga jalannya program yang dibuat lebih stabil dan penanganan kesalahan lebih baik dengan tujuan supaya program dapat berjalan secara optimum. Untuk membuat suatu *project* pada Microsoft Visual Basic 2010 dapat dilakukan dengan langkah sebagai berikut:

1. Klik tombol *Start* pada *Windows taskbar*
2. Pilih menu program Microsoft Visual Studio 2010 > Microsoft Visual Studio 2010.
3. Setelah itu akan muncul halaman *Start Page* seperti pada gambar 2.



Gambar 2.7 Halaman Start Page Visual Studio 2010 Ultimate
(Budiharto & Lisangan, 2012)

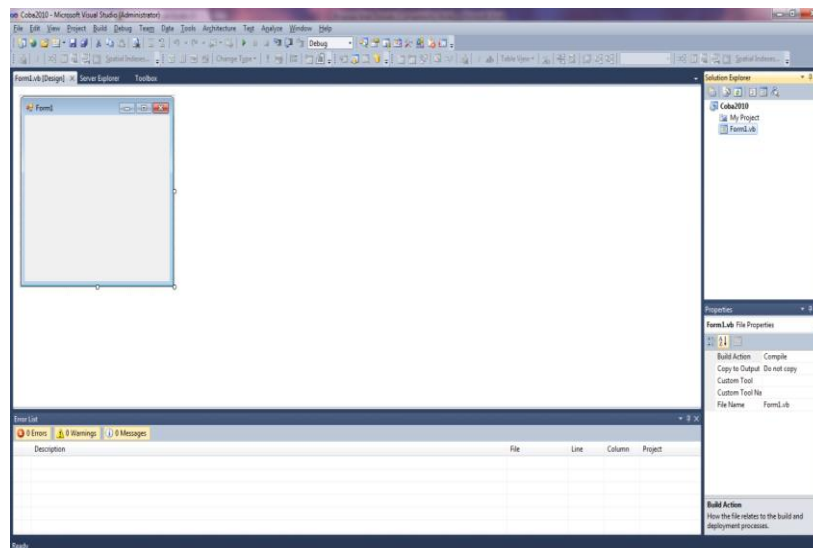
4. Pada halaman *Start Page* pilih *New Project* maka akan muncul jendela *New Project* seperti pada gambar 2.8.



Gambar 2.8 Jendela *New Project* (Budiharto & Lisangan, 2012)

5. Ketika muncul jendela *New Project*, pilih *Visual Basic* dan *Windows Form Application*, ketikkan nama *project* pada kotak isian *Name* dan pilih tombol *OK*, maka muncul *IDE (Integrated Development Environment)* berupa *form* desain untuk memulai membangun aplikasi baru.

Adapun antarmuka yang dimiliki oleh VB2010 adalah antarmuka yang berupa ruang kerja yang terpadu dan disebut *IDE (Integrated Development Environment)*. Antarmuka VB2010 dapat diatur sesuai selera dan kebutuhan pengguna (Budiharto & Lisangan, 2012). Namun, biasanya VB2010 memiliki tampilan antarmuka seperti gambar 2.9.


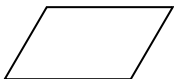
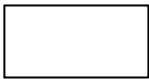
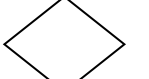


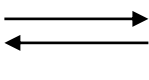
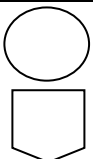


Gambar 2.9 IDE Microsoft Visual Basic NET 2010
(Budiharto & Lisangan 2012)

2.5 *Flowchart*

Flowchart adalah bagan alir yang menggambarkan arus data dari program. Fungsi dari bagan alir ini adalah untuk memudahkan programmer di dalam perancangan program aplikasi (Jogiyanto, 2005). Simbol-simbol yang digunakan pada bagan *flowchart* ini antara lain seperti pada tabel 2.15.

Tabel 2.15 Simbol-Simbol Flowchart Program (Jogiyanto, 2005)

Simbol	Fungsi
	<i>Terminator</i> Menunjukkan awal dan akhir suatu proses.
	<i>Data</i> Digunakan untuk mewakili data <i>input/output</i> .
	<i>Process</i> Digunakan untuk mewakili proses.
	<i>Decision</i> Digunakan untuk suatu seleksi kondisi didalam program.
	<i>Predefined Process</i> Menunjukkan suatu operasi yang rinciannya ditunjukkan di tempat lain.
	<i>Preparation</i> Digunakan untuk memberi nilai awal variabel.
	<i>Flow Lines Symbol</i> Menunjukkan arah dari proses.
	<i>Connector</i> Menunjukkan penghubung ke halaman yang sama. Menunjukkan penghubung ke halaman yang baru.

1.5 Penelitian Terdahulu

Penelitian terdahulu diambil dari beberapa jurnal yang berkaitan dengan penelitian penulis dan dapat dilihat pada tabel 2.16

Tabel 2.16 Penelitian Terdahulu

No	Judul Penelitian	Penerbit	Isi Penelitian
1	Implementasi algoritma kriptografi <i>Rail Fence Cipher</i> dan algoritma <i>myszkowski transposition</i> dan Algoritma kompresi <i>fibonacci code</i> . Andriyani, S. Y. (2019)	Skripsi Program Studi S1 Ilmu Komputer, Fakultas Ilmu Komputer dan Teknologi Informasi, Universitas Sumatera Utara	Berhasil dalam pengamanan <i>file</i> teks, sehingga <i>ciphertext</i> yang dihasilkan menjadi lebih sulit untuk dipecahkan. Dari hasil yang diperoleh disimpulkan bahwa melakukan kompresi menggunakan algoritma <i>Fibonacci Code</i> dapat berjalan dengan baik karena berhasil mengembalikan keutuhan pesan seperti semula serta hasil dari kompresi dapat memperkecil ukuran <i>file</i> .
2	Kombinasi Algoritma Kriptografi Transposisi <i>Rail Fence Cipher</i> dan <i>Route Cipher</i> , Girsang <i>et al.</i> , (2019)	Prosiding Seminar Nasional Teknologi Informatika Volume 2 Nomor 1 November 2019.	Dilakukan kombinasi algoritma <i>Rail Fence Cipher</i> dengan <i>Route Cipher</i> untuk mengetahui seberapa persen tingkat keamanan pesan yang dihasilkan. Kombinasi <i>Rail Fence Cipher</i> dan <i>Route Cipher</i> bekerja dengan cara mengenskripsikan pesan terlebih dahulu dengan <i>Rail</i>


No	Judul Penelitian	Penerbit	Isi Penelitian
			<i>Fence Cipher</i> , selanjutnya cipherteks di enkripsi kembali menggunakan <i>Route Cipher</i> .
3	Kombinasi Algoritma Huffman dan Algoritma ROT13 dalam Pengamanan File Docx, Hendrik (2020)	Journal of Information Sistem Research (JOSH) Volume 2, No. 1, October 2020.	File rtf akan diamankan menggunakan 2 algoritma, yaitu algoritma huffman dan algoritma ROT13. adapun cara pengamanan file rtf yaitu dengan cara mengkompresi file rtf menggunakan algoritma huffman, setelah file rtf dikompresi dan mendapatkan sebuah karakter tertentu, maka dilanjutkan dengan pengamanan menggunakan algoritma ROT 13 sehingga file rtf dapat terjaga kerahasiaannya.
4	Implementasi Algoritma Rail Fence Cipher Dalam Keamanan Data Gambar 2 Dimensi, Ratna (2018)	Jurnal Pelita Informatika, Volume 7, Nomor 1, Juli 2018 ISSN 2301-9425 (Media Cetak) Hal: 38-42.	Dilakukan proses pengamanan citra digital dengan mengubah citra asli menjadi citra yang keabuan menggunakan algoritma <i>Rail Fence Cipher</i> dapat digunakan mengamankan citra digital dengan cara mentransposisi piksel pada

No	Judul Penelitian	Penerbit	Isi Penelitian
			citra digital. Aplikasi ini dirancang dengan menggunakan <i>Microsoft Visual Basic .net</i> 2008 ini dapat digunakan untuk proses pengamanan citra dengan Algoritma <i>Rail Fence Cipher</i>
5	Kombinasi Algoritma ROT13 dan Vigenere Cipher Pada Alamat Directory File Untuk Keamanan Dokumen, Pramudya <i>et al.</i> (2020).	Seminar Nasional Hasil Penelitian dan Pengabdian pada Masyarakat V Tahun 2020 Pengembangan Sumber Daya Menuju Masyarakat Madani Berkearifan Lokal LPPM Universitas Muhammadiyah Purwokerto. ISBN: 978-602-6697-660.	Hasil pengujian sistem menunjukkan bahwa kombinasi algoritma ROT 13 dan Vigenere Cipher mampu mengenkripsi jenis laporan dan nama file serta hasil dari enkripsi dapat dijadikan format file directori dengan ditambahkan index dokumen pada akhir nama file. Implementasi sistem dengan keamanan dokumen menggunakan algoritma <i>ROT13</i> dan <i>Vinegere Cipher</i> berhasil diterapkan sehingga nama file dan lokasi dokumen terenskripsi sehingga sulit untuk dilakukan pengambilan dokumen oleh pengguna yang tidak sah.

No	Judul Penelitian	Penerbit	Isi Penelitian
6	Perancangan Aplikasi Penyandian Teks Dengan Algoritma ROT13 Dan Triangle Chain Cipher (TCC), Hondro & Fau (2018)	Jurnal Mahajana Informasi, Vol.3 No. 2, 2018 e-ISSN: 2527-8290.	Mengubah huruf ke huruf berbaring posisi 13 dari surat asli. Algoritma kriptografi segitiga jaringan Cipher adalah algoritma yang dibuat untuk memperbaiki algoritma kriptografi klasik terutama algoritma substitusi tunggal alfabet sangat rentan dengan teknik analisis frekuensi. Kekuatan cipher terletak di kunci yaitu nilai integer yang menunjukkan pergeseran karakter-karakter sesuai dengan operasi pada caesar-cipher.
7	Analysis Stego-Image Extraction Using ROT13 and <i>Least Significant Bit</i> (LSB), Akhyar, H., <i>et al.</i> (2017).	Jurnal Ilmiah FIFO P-ISSN 2085-4315 / E-ISSN 2502-8332.	Algorithm Method on Text Security dilakukan teknik pengamanan Steganografi pesan rahasia dengan keamanan berlapis, dengan menambahkan Kriptografi ke pesan rahasia yang akan disisipkan ke dalam citra digital dan kemudian pesan disisipkan ke dalam citra digital melalui Steganografi menggunakan metode LSB. Pada penelitian ini

No	Judul Penelitian	Penerbit	Isi Penelitian
			diterapkan pengamanan pada steganografi dengan menambahkan kriptografi ROT13 yang membuat pesan rahasia kemudian digeser 13 karakter.
8	Implementasi Multi Enkripsi ROT13 Pada Symbol Whatsapp, Aresta, R. M., <i>et al.</i> (2020)	Jurnal Of Information System Management e-ISSN: 2715-3088 Vol 2., No. 1. (2020).	Dilakukan enkripsi teks pesan dengan algoritma ROT13 Perubahan plainteks yang sebelumnya telah di enkripsi dengan metode ROT13 kemudian disempurnakan dengan pengenkripsian dalam bentuk emoticon. Emoticon tersebut akan mewakili dari setiap abjad dari „N-A dan n-a“.
9	Implementasi Metode Rail Fence Chiper Dan Row Transposition Chiper, Rusmala & Prasti, D. (2019)	Pada Mata Kuliah Kriptografi Jurnal Ilmiah d'Computare Volume 9 Edisi Januari 2019.	Dilakukan implementasi metode Rail Fence Chiper dan Row Transposition Chiper untuk enkripsi file teks dengan hasil bahwa kedua algoritma diatas dapat melakukan enkripsi dan dekripsi file teks dengan baik.

No	Judul Penelitian	Penerbit	Isi Penelitian
10	Implementasi Algoritma One Time Menggunakan Algoritma Chipper Transposition, Huda et al. (2022)	Jurnal J-COM (Jurnal Informatika dan Teknologi Komputer) Vol. 03 No. 01 (2022) 40 – 48.	Dalam pengamanan data yang penting atau rahasia dan penggunaan kunci adalah hal yang paling dibutuhkan untuk menjaga kerahasiaan dalam pemakaiannya sehingga sangat penting dalam enkripsi dan dekripsi dan penggunaan pemisah kata (spasi) pada proses enkripsi sangat berpengaruh terhadap pembentukan karakter matriks sehingga menghasilkan plainteks yang sesuai dengan pesan aslinya.
11	Combination of Caesar Cipher Modification with Transposition Cipher, Lubis et al. (2017)	Advances in Science, Technology and Engineering Systems Journal Vol. 2, No. 5, 22-25 (2017).	Dilakukan enkripsi file teks dengan mengkombinasikan algoritma Caesar Cipher dengan Transposisi Cipher dan dapat dilakukan dekripsi kembali.

No	Judul Penelitian	Penerbit	Isi Penelitian
12	Rail Fence Cryptography in Securing Information., Siahaan, A.P.U. 2016.	<i>International Journal of Science & Engineering Research (IJSER)</i> . Vol 7, Issue 7. 	Hasil ciphertext diturunkan dari huruf-huruf yang menggeser setiap karakter pada plaintext. Sedangkan pada Rail Fence diambil dari matriks pembentukan blok ciphertext secara diagonal. Tingkat keamanan pada metode ini memiliki kelebihan dibandingkan dengan metode sebelumnya.



UNIVERSITAS ISLAM NEGERI
 SUMATERA UTARA MEDAN