Modifikasi Algoritma Caesar Cipher pada Kode ASCII dalam Meningkatkan Keamanan Pesan Teks

Mira[1], Hindriyanto Dwi Purnomo[2] , Irwan Sembiring[2]
[1,2,3] Universitas Kristen Satya Wacana
Jl. Diponegoro No.52-60, Salatiga, Kec. Sidorejo, Kota Salatiga, Jawa Tengah 50711
[1]miracle.yaa@gmail.com, [2]hindriyanto.purnomo@uksw.edu, [3]irwan@uksw.edu

***Abstract*— Theft of personal data is a criminal act and violates the law in the field of information technology. Therefore, the security of personal data must have a higher level of security. Personal data security can be done by one of them providing a password to the data file. To avoid password burglary, you can encrypt the password. This study uses the Caesar Cipher algorithm in the encryption process. However, some classical cryptographic algorithms that have been widely known have weaknesses that can be solved by cryptanalysis. Therefore, to avoid this, before encrypting, the caesar cipher algorithm is modified first and then converted into ASCII code. The results of the modified algorithm provide a level of message security complexity, when compared to the caesar algorithm without modification. This is shown from the results of the ciphertext that produces ciphertext in the form of combined characters, making the ciphertext more difficult to read and understand by humans, so as to increase message security and can minimize the occurrence of hacking by irresponsible parties.**

***Intisari*— Pencurian data pribadi merupakan tindakan kriminal dan melanggar hukum pada bidang teknologi informasi. Oleh karena itu, pengamanan data pribadi harus memiliki tingkat pengamanan yang lebih. Pengaman data pribadi dapat dilakukan dengan salah satunya memberikan password terhadap file data. Untuk menghindari pembobolan password dapat melakukan enkripsi terhadap password. Penelitian ini menggunakan algoritma Caesar Chiper dalam melakukan proses enkripsi. Namun beberapa algoritma kriptografi klasik yang telah banyak diketahui secara luas memiliki kelemahan yang dapat dipecahkan oleh criptanalysis. Oleh karena itu, untuk menghindari hal tersebut, sebelum melakukan enkripsi, algoritma caesar cipher dimodifikasikan terlebih dahulu kemudian dikonversikan kedalam code ASCII. Hasil modifikasi algoritma memberikan tingkat kompleksitas keamanan pesan, bila dibandingkan dengan caesar algoritma tanpa modifikasi. Hal ini ditunjukan dari hasil ciphertext yang menghasil ciphertext yang berupa karakter gabungan, membuat ciphertext semakin sulit untuk dibaca dan dipahami oleh manusia, sehingga dapat meningkatkan keamanan pesan dan dapat meminimalisir terjadinya peretasan oleh pihak yang tidak bertanggungjawab.**

Kata Kunci— Kriptografi, ASCII, Modifikasi, Caesar Cipher.

I. PENDAHULUAN

Pencurian data pribadi merupakan tindakan kriminal dan melanggar hukum pada bidang teknologi informasi. Oleh karena itu, pengamanan data informasi yang berharga yang bersifat pribadi harus memiliki tingkat pengamanan yang lebih. Tingkat pengamanan data sebagai upaya dalam melindungi data atau informasi yang bersifat rahasia agar tidak disalahgunakan. Pengaman data atau informasi pribadi dapat dilakukan dengan salah satunya memberikan password terhadap file data yang bersifat rahasia. Untuk menghindari password yang dapat diketahui oleh pihak yang tidak bertanggungjawab dapat melakukan enkripsi terhadap password file data sebelum diberikan kepada pihak yang dianggap layak mengetahui password dari file data tersebut. Upaya meningkatkan keamanan data dapat menggunakan kriptografi, agar pihak lain tidak dapat memahami maksud pesan yang telah dienkripsikan terlebih dahulu untuk menjaga kerahasiaan pesan.

Kriptografi merupakan teknik untuk menjaga keamanan rahasia data atau informasi dengan mengubah pesan asli (dapat dibaca) menjadi pesan yang teracak (sulit dibaca) [1]. Kriptografi merupakan pihak ketiga dari proses enkripsi dan deskripsi [2]. Enkripsi merubah pesan asli menjadi pesan yang diacak dengan algoritma khusus untuk menjaga kerahasiaan pesan [3]. Setelah pesan dienkripsi, untuk dapat dipahami kembali oleh penerima pesan maka perlu dilakukan deskripsi terhadap pesan yang telah dienksripsi. Oleh karena itu, untuk meningkatkan kerahasian pesan teks digunakan beberapa algoritma. Algoritma pengaman data informasi terdiri dari banyak kategori, namun dalam penelitian ini algoritma yang akan digunakan yaitu Caesar Chiper, Caesar Cipher merupakan salah satu algoritma tertua, dan merupakan salah satu jenis cipher substitusi yang menyusun huruf dalam *plaintext* yang digeser dan diganti dengan huruf beberapa posisi tetap di bawah alfabet [4][5]. Namun beberapa algoritma kriptografi klasik yang telah banyak diketahui secara luas memiliki kelemahan yang dapat diketahui dan dipecahkan oleh *criptanalysis*. Kriptanalisis melakukan pengecekan serangan teks biasa dengan mempelajari pengaruh hasil putaran pasangan teks cipher iteratif [6].

Pada penelitian terdahulu sebelumnya oleh Lubis dkk, yang melakukan modifikasi algoritma caesar cipher dan mengkombinasikan dengan transposition cipher sehingga menghasilkan proses enkripsi sebanyak tiga kali, proses modifikasi pada caesar cipher dengan pergeseran sesuai abjad pada *ASCII* sehingga menghasilkan *plaintext* akhir yang komplek sehingga untuk mengetahui pesan aslinya *cryptanalysis* harus melakukan banyak percobaan [7].

Penelitian terdahulu kedua oleh Rachmawati dkk yang melakukan kombinasi algoritma *columnar transposition* cipher caesar cipher dan *lempel ziv welch* dalam keamanan dan kompresi gambar. Dengan proses gambar akan dienkripsi menggunakan algoritma *columnar transposition cipher*, kemudian file yang telah dienkripsi dengan algoritma caesar cipher menggunakan kombinasi linear *congruential generators* dan dimodifikasi. kemudian file file gambar yang

telah dienkripsi akan dikompres menggunakan algoritma kedua *lempel ziv welch*. Dengan hasil penelitian dapat menjaga kerahasiaan, keutuhan, dan keamanan citra. [5].

Sedangkan dalam penelitian ini, algoritma caesar cipher dimodifikasi dengan menambah perkalian terhadap proses enkripsi yang kemudian hasil enkrispi dikonversikan kembali kedalam *code ASCII*. Dikarenakan bahwa algoritma kriptografi klasik memiliki tingkat pengamanan yang rentan yang dapat dipecahkan oleh *cryptanalysis.* Oleh karena itu, untuk menghindari hal tersebut, sebelum melakukan enkripsi, algoritma caesar cipher dimodifikasikan terlebih dahulu. Selain itu, peneliti melakukan perbandingan kinerja terhadap algoritma Caesar Cipher klasik dengan algoritma Caesar Cipher yang di modifikasi denga melakukan modifikasi penambahan perkalian 4 dengan kunci yang diinputkan kemudian dikonversi ke dalam kriptografi era modern yaitu *code ASCII* dengan data *decimal*, biner, hexa, *decimal* dan karakter cipher.

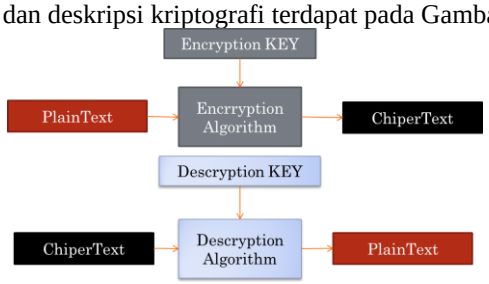
II. TINJAUAN PUSTAKA

1. Pesan Teks

   Pesan merupakan materi yang memiliki maksdud dan tujuan antara pengirim dan penerima. Pesan teks terdiri dari susunan huruf atau abjad alfabet menjadi sebuah kalimat yang dapat dideskripsikan, dibaca dan memiliki nilai. Penyampaian pesan terdiri dari beberapa kategori, seperti teks, gambar, audio dan video.

2. Kriptografi

   Kriptografi merupakan seni dan teknik dalam menjaga kerahasiaan pesan dengan metode enkripsi dan deskripsi. Proses enkripsi merupakan mengubah pesan biasa (teks biasa) menjadi pesan yang tida dikenali yang disebut dengan *ciphertext.* Sedangkan deskripsi merupakan proses mengembalikan *ciphertext* menjadi pesan awal atau pesan asli yang disebut dengan *plaintext* [8]. adapun proses enkripsi dan deskripsi kriptografi terdapat pada Gambar 1.



Gambar 1. Proses Enkripsi Deskripsi Kriptografi

3. Caesar cipher

   Caesar cipher merupakan algoritma kriptografi simetris sebelum asimetris ditemukan. Caesar cipher menggunakan proses dengan teknik substitusi, substitusi digunakan oleh kaisar Romawi pertama yaitu Julius Caesar untuk melakukan enkripsi pesan yang akan dikirim kepada Gubernur. Proses enkripsi setiap huruf abjad dengan bilangan bulat : A=0, B=1, C=2,...Z=25 [9]. Sehingga proses enkripsi menggunakan persamaan 1.

$$C=E(K,P)=(P+K)\text{mod}26 \quad (1)$$

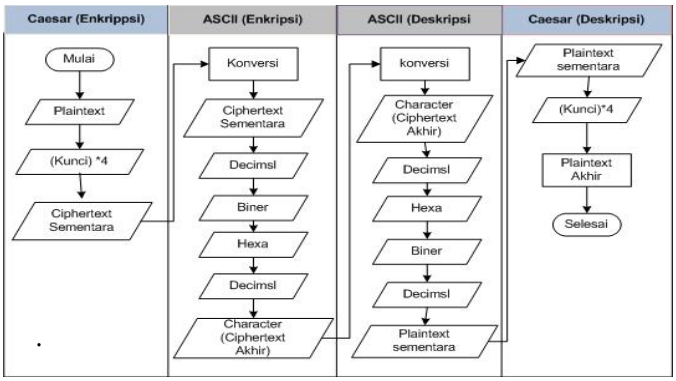
Sedangkan proses deskripsi menggunakan persamaan 2.

$$P=D(K,C)=(C-K) \text{ mod } 26 \quad (2)$$

III. METODOLOGI PENELITIAN

Terminologi dasar yang digunakan dalam kriptografi adalah sebagai berikut: *Plaintext* secara sinonim adalah pesan asli yang ingin dikirimkan pengirim kepada penerima. Enkripsi adalah teknik yang digunakan untuk mengubah teks biasa menjadi teks tersandi (cipher). Cipher Text adalah teks yang diperoleh setelah menerapkan enkripsi pada teks biasa. Dekripsi adalah teknik untuk mengubah teks sandi menjadi teks biasa [10].

Metode enkripsi dalam Caesar Cipher ini merupakan enkripsi berjenis subtitusi, dimana setiap huruf pada *plaintext* diganti dengan huruf lain. Dalam penelitian ini, melakukan modifikasi terhadap algoritma dalam melakukan enkripsi pesan yang kemudian akan di konversi ke *code ASCII*. *ASCII* adalah singkatan dari *American Standard Code for Information Interchange*, dengan kode karakter 7bit di mana setiap bit mewakili karakter unik dan 8 bit yang terdiri dari 256 karakter [11]. Hasil enkripsi pertama akan dikonversi ke *decimal*, biner, hexa dan karakter. Sedangkan untuk deskripsi merupakan kebalikan dari proses enkripsi. Memodifikasi algoritma Caesar Cipher sebagai upaya dalam meningkatkan kompleksitas keamanan pesan teks, untuk menghindari peretasan oleh pihak yang tidak bertanggungjawab. Adapun langkah-langkah atau skema dalam penelitian ini dapat dilihat pada Gambar 2.



Gambar 2. Alur Penyandian dan Konversi

Berdasarkan gambar 1. Dapat diuraikan sebagai berikut :

1. Menginput pesan atau karakter *plaintext*.
2. Melakukan modifikasi algoritma caesar cipher dengan mengalikan kunci yang diinputkan untuk mendapatkan *ciphertext* sementara, dalam penelitian ini, perkalian yang digunakan 4.
3. *Ciphertext* sementara yang dihasilkan dari enkripsi pertama akan di enkrispi kembali dengan jumlah karakter kunci sama dengan jumlah karakter pesan yang akan dikonversi ke data *decimal*, biner, hexa dan karakter *cipher* sebagai *ciphertext* terakhir.
4. Kemudian *ciphertext* tersebut dideskripsikan kembali dengan kunci yang sama saat melakukan enkripsi kedua untuk dikonversikan kembali ke dalam bentuk *decimal*, *hexa*, biner dan *decimal* agar menghasilkan *plaintext* sementara.

5. *Plaintext* sementara hasil deskripsi pertama, akan di deskripsikan kembali dengan kunci yang sama saat proses enkripsi sehingga didapatkan kembali *plaintext* pertama yang merupakan pesan asli diawal.

IV. HASIL DAN PEMBAHASAN

Proses enkripsi dan deskripsi menggunakan algorima Caesar Cipher klasik menggunakan *plaintext* "MODIFIKASI" dengan kunci "2" dengan menggunakan persamaan 1.

1 Enkripsi
Diketahui :
Plaintex : MODIFIKASI
Kunci : 2
Proses :
M = (12+2) mod 26 = (14) mod 26 = 14 mod 26 = O
O = (14+2) mod 26 = (16) mod 26 = 16 mod 26 = Q
Dan seterusnya...
Sehingga *ciphertext* dari *plaintext* MODIFIKASI dengan kunci 2 menggunakan algoritma caesar cipher meghasilkan *ciphertext* OQFKHKMCUK. Selanjutnya melakukan deskripsi untuk mengembalikan pesan yang diacak (penyandian) ke pesan asli dengan menggunakan persamaan 2.

2 Deskripsi
Diketahui :
Ciphertext : OQFKHKMCUK
Kunci : 2
Proses :
O = (14-2) mod 26 = (12) mod 26 = 12 mod 26 = M
Q = (16-2) mod 26 = (14) mod 26 = 14 mod 26 = O
Dan seterusnya...
Sehingga *plaintext* dari *ciphertext* OQFKHKMCUK dengan kunci 2 menggunakan algoritma caesar cipher menghasilkan *plaintext* MODIFIKASI.

Proses modifikasi enkripsi dalam penelitian ini, melakukan perubahan posisi huruf dengan melakukan perkalian dari setiap kunci yang di input dengan 4. Adapun persamaan enkripsi yang dimodifikasi terdapat pada persamaan 3.

$$C=E(K,P)=(P+K*4)\bmod 26 \quad (3)$$

Sedangkan proses deskripsi yang dimodifikasi terdapat pada persamaan 4.

$$P=D(K,C)=(C-K*4)\bmod 26 \quad (4)$$

Proses enkripsi algoritma Caesar Cipher yang dimodifikasi :
Plaintext : MODIFIKASI
Perkalian : 4
Kunci : 2
Proses enkripsi untuk menghasilkan *ciphertext* sementara :
M = (12+2*4) mod 26 = (12+8) mod 26 = 20 mod 26 = U
O = (14+2*4) mod 26 = (14+8) mod 26 = 22 mod 26 = W
D = (3+2*4) mod 26 = (3+8) mod 26 = 11 mod 26 = L
I = (8+2*4) mod 26 = (8+8) mod 26 = 16 mod 26 = Q
F = (5+2*4) mod 26 = (5+8) mod 26 = 13 mod 26 = N
I = (8+2*4) mod 26 = (8+8) mod 26 = 16 mod 26 = Q
K = (10+2*4) mod 26 = (10+8) mod 26 = 18 mod 26 = S
A = (0+2*4) mod 26 = (0+8) mod 26 = 8 mod 26 = I
S = (18+2*4) mod 26 = (18+8) mod 26 = 26 mod 26 = A
I = (8+2*4) mod 26 = (8+8) mod 26 = 16 mod 26 = Q

Sehingga *ciphertext* dari *plaintext* MODIFIKASI dengan kunci 2 yang dikalikan 4 dengan algoritma Caesar Cipher menghasilkan *ciphertext* sementara yaitu UWLQNQSIAQ. Selanjutnya *ciphertext* yang telah dihasilkan menjadi *ciphertext* sementara yang akan dikonversikan kembali ke dalam tabel *ASCII* menjadi data *decimal*, biner, *hexa* dan karakter dengan jumlah karakter kunci yang sama dengan jumlah karakter *plaintext*, dalam penelitian ini menggunakan kunci "ALGORITMAA" sebagaimana terdapat pada tabel 1.

Tabel 1. Hasil Konversi

Plain text	***Decimal***	**Biner**	***Key***	***Decimal***	**Biner**
U	85	01010101	A	65	0b1000001
W	87	01010111	L	76	0b1001100
L	76	01001100	G	71	0b1000111
Q	81	01010001	O	79	0b1001111
N	78	01001110	R	82	0b1010010
Q	81	01010001	I	73	0b1001001
S	83	01010011	T	84	0b1010100
I	73	01001001	M	77	0b1001101
A	65	01000001	A	65	0b1000001
Q	81	01010001	A	65	0b1000001

Kemudian pesan biner dan kunci biner di XOR kan sehingga menghasilkan XOR yang terdapat pada tabel 2.

Tabel 2. Hasil Konversi

Plaintext	**Kunci**	**XOR**
01010101	01010101	00010100
01010111	01010111	00011011
01001100	01001100	00001011
01010001	01010001	00011110
01001110	01001110	00011100
01010001	01010001	00011000
01010011	01010011	00000111
01001001	01001001	00000100
01000001	01000001	00000000
01010001	01010001	00010000

Selanjutnya hasil XOR dikonversikan kembali ke hexa dan *decimal* untuk menghasilkan karakter *ciphertext* akhir yang terdapat pada tabel 3.

Tabel 3. Hasil Konversi

XOR	***Hexa***	***Decimal***
00010100	0x14	20
00011011	0x1b	27
00001011	0xb	11
00011110	0x1e	30
00011100	0x1c	28
00011000	0x18	24
00000111	0x7	7

00000100	0x4	4
00000000	0x0	0
00010000	0x10	16

Setelah di XOR dikonversi ke *hexa* dan *decimal*, selanjutnya akan dikonversi ke karakter cipher sehingga menghasilkan *ciphertext* terakhir dengan karakter cipher.

Sedangkan proses deskripsi merupakan kebalikan dari proses enkripsi, proses deskripsi dimulai dengan mengkonversi karakter cipher yang dijadikan *ciphertext* terakhir dari enkripsi kedua ke data *decimal*, kemudian hexa, biner dan *decimal*, dari data *decimal* akan dikonversi ke karakter huruf, karakter huruf akan menjadi *plaintext* sementara yang akan dideskripsikan kembali dengan algoritma Caesar Cipher modifikasi, proses deskripsi dengan algoritam Caesar Cipher modifikasi merupakan proses kebalikan dari enkripsi, yaitu dengan mengurangi nilai urutan abjad dengan hasil perkalian kunci dengan angka 4, maka akan menghasilkan *plaintext* awal atau pesan semula yaitu MODIFIKASI.

Selanjutnya mengimplementasikan proses enkripsi deskripsi menggunakan bahasa python dengan bantuan *google colab*. Python adalah bahasa pemrograman tingkat tinggi modern yang berfokus pada keterbacaan kode[12], berorientasi objek, yang mencakup beberapa karakteristik umum seperti bahasa yang bersih dan sederhana, bahasa ekspresif, diketik secara dinamis, manajemen memori otomatis, dan ditafsirka [13]. *Google Colaboratory (a.k.a Colab)* adalah proyek yang bertujuan untuk menyebarluaskan pendidikan dan penelitian *machine learning* [14]. Adapun logika pemrosesan enkripsi dan deskripsi dengan Caesar Cipher yang dimodifikasi dan dikonversi ke *code ASCII* menggunakan python dan bantuan *google colab* terdapat pada kode program 1, 2, 3 dan 4.

Kode Program 1. Proses Enkrispi Menggunakan Caesar Cipher Modifikasi

```
from string import ascii_lowercase,ascii_uppercas
e,digits
def caesar_en(text, key):
    result = []
    for c in text:
        if c in ascii_lowercase:
            enkripsi = ascii_lowercase.index(c)
            enkripsi = (enkripsi + key) % 26
            result.append(ascii_lowercase[enkrips
i])
        elif c in ascii_uppercase
            enkripsi = ascii_uppercase.index(c)
            enkripsi = (enkripsi + key) % 26
            result.append(ascii_uppercase[enkrips
i])
        elif c in digits
            enkripsi = digits.index(c)
            enkripsi = (enkripsi + key) % 10
            result.append(digits[enkripsi])
        else:
            result.append(c)
    return "".join(result)
def main():
  text = input("Masukkan Plaintext :")
  kunci = input("Masukkan Kunci    : ")
  key = int(kunci) * 4
  print ("\nProgram Modifikasi Algoritma Caesar C
ipher Dengan Mengalikan kata kunci yang diinputka
n dengan 4")
  print ("-------------------------------------")
  print ("Kasus Caesar Cipher Modifikasi")
  print ("\nPesan : "+text)
  print ("Kunci : "+str(kunci))
  print ('\n Enkripsi \"' + text + '\" dengan Kun
ci  ' + str(kunci) + "  ")
  print ("-------------------------------------")
  print ("\t Chiperteks Sementara : "+caesar_en(t
ext, key))
  print ("-------------------------------------")
if   name   == '  main  ':
  main()
```

Kode Program 2. Proses Konversi *Ciphertext* Sementara Ke Code Ascii

```
import random
import string
from operator import xor
def konvers_enc(pesan):
    key1 = input("Masukkan Kunci Konversi Cipher

    print ("---------------------------------\n")
    convdecimal = []
    for i in range(len(pesan)):
        convdecimal.append(str(ord(pesan[i])))
    decimal = ' '.join(convdecimal)
    print ("Hasil Konversi ke Desimal Karakter Pe
san : ")
    decimal = ' '.join(convdecimal)
    print (' '.join(convdecimal))
    convbiner = []
    for i in range(len(convdecimal)):
        convbiner.append(str(bin(int(convdecimal[
i]))[2:].zfill(8)))
    biner = ''.join(convbiner)
    print ("hasil konversi ke biner karakter pesa
n : ")
    print (' '.join(convbiner))
    convkeydecimal = []
    for i in range(len(key1)):
        convkeydecimal.append(str(ord(key1[i])))
    keydecimal = ' '.join(convkeydecimal)
    print ("\nHasil Konversi ke Desimal Karakter
Kunci : ")
    print (' '.join(convkeydecimal))
    convkeybiner = []
    for i in range(len(convkeydecimal)):
        convkeybiner.append(str(bin(int(convkeyde
cimal[i]))))
    binerkey = ''.join(convkeybiner)
    print ("Hasil Konversi ke biner karakter kunc
i :")
    print (' '.join(convkeybiner))
    binerpesan =biner.replace("b","")
    print ("\n Sehingga ")
    print ("Biner pesan : \n"+binerpesan)
    binerkey =binerkey.replace("b","")
    print ("Biner Key : \n"+binerkey)
    ciphertext = []
    for i in range(len(binerpesan)):
        ciphertext.append(xor(int(binerpesan[i]),
int(binerkey[i])))
    cipher =''.join(map(str,ciphertext))
    print ("Hasil XOR Biner Pesan dan Biner Key :
 ")
    print (cipher)
    splits=[cipher[x:x+8] for x in range(0,len(ci
phertext),8)]
    print ("\nHasil XOR Akhir : \n"+str(splits))
    hexa = []
```

```
    for i in range(len(splits)):
        hexa.append(hex(int(splits[i],2)))
    print ("\n==>> Konversi Hasil dari XOR ke Hex
a : \n"+str(hexa))
    dec = []
    for i in range(len(hexa)):
        dec.append(int(hexa[i], 16))
    print ("\n==>> Konversi Hasil dari Hexa ke De
simal : \n"+str(dec))
    teks = []
    for i in range(len(dec)):
        teks.append(chr(dec[i]))
    save2file = ''.join(teks)
    print ("\n---------------------------------")
    print ("\t Cipherteks Pesan : "+''.join(teks)
)
    print ("---------------------------------")
      outfile = open('ciphertext.txt','w')
    print ("\n\nBukti ::")
    outfile.write(save2file)
    return save2file
def main():
    print ("Program Konversi Hasil Enkripsi Algor
itma Caesar Cipher Yang Telah di Modifikasi Denga
n Menambah Nominal Perkalian")
    print ("---------------------------------")
    print ("konversi Klasik Ke ASCII ")
    pesan = input("Masukkan Cipherteks sementara
:")
    print (konvers_enc(pesan))
if __name__ == '__main__':
     main()
```

Kode Program 3. Proses Deskripsi Konversi Ciphertex Ke *Plaintext* Sementara

```
import random
import string
from operator import xor
def konvers_dec(key):
    textcipher = open('ciphertext.txt','r').read(
)
    print ("\nCipheteks yang digunakan adalah : "
+textcipher)
    key = input("Kunci yang digunakan adalah
: ")
    print ("---------------------------------")
    convdecimal = []
    for i in range(len(textcipher)):
        convdecimal.append(str(ord(textcipher[i])
))
    decimal = ' '.join(convdecimal)
    print ("Hasil konversi ke decimal karakter Ci
phertext : ")
    print (' '.join(convdecimal))
    convbiner = []
    for i in range(len(convdecimal)):
        convbiner.append(str(bin(int(convdecimal[
i]))[2:].zfill(8)))
    biner = ''.join(convbiner)
    print ("hasil konversi ke biner karakter Ciph
ertext : ")
    print (' '.join(convbiner))
    convkeydecimal = []
    for i in range(len(key)):
        convkeydecimal.append(str(ord(key[i])))
    print ("\nHasil konversi ke desimal karakter
kunci : ")
    keydecimal = ' '.join(convkeydecimal)
    print (" ".join(convkeydecimal))
    convkeybiner = []
    for i in range(len(convkeydecimal)):
        convkeybiner.append(str(bin(int(convkeyde
cimal[i]))))
    binerkey = ''.join(convkeybiner)
    print ("Hasil Konversi ke biner karakter kunc
i :")
    print (' '.join(convkeybiner))
    print ("==> Didapatkan :")
    binercipher =biner.replace("b","")
    print ("Biner Ciphertext : \n"+binercipher)
    binerkey =binerkey.replace("b","")
    print ("Biner kunci : \n"+binerkey)
    plaintext = []
    for i in range(len(binercipher)):
        plaintext.append(xor(int(binercipher[i]),
int(binerkey[i])))
    plain =''.join(map(str,plaintext))
    print ("Hasil XOR : ")
    print (plain)
    splits=[plain[x:x+8] for x in range(0,len(pla
intext),8)]
    print ("\nHasil XOR variabel ciphertext dan k
unci : \n"+str(splits))
    hexa = []
    for i in range(len(splits)):
        hexa.append(hex(int(splits[i],2)))
    print ("\nKonversi hasil dari XOR ke Hexa : \
n"+str(hexa))
    dec = []
    for i in range(len(hexa)):
        dec.append(int(hexa[i], 16))
    print ("\nKonversi Hasil dari Hexa ke Desimal
 : \n"+str(dec))
    teks = []
    for i in range(len(dec)):
        teks.append(chr(dec[i]))
    print ("\n---------------------------------")
    print ("Didapatkan Hasil Plaintext Pesan Seme
ntara : ")
    return ''.join(teks)
def main():
    print ("Program Konversi HasiL Enkripsi ASCII
 Ke Algoritma Caesar Cipher")
    print ("---------------------------------")
    print ("konversi ASCII Ke Klasik")
    key=("")
    print (konvers_dec(key))
if __name__ == '__main__':
     main()
```

Kode Program 4. Proses Deskripsi *Plaintext* Sementara Ke Pesan Awal

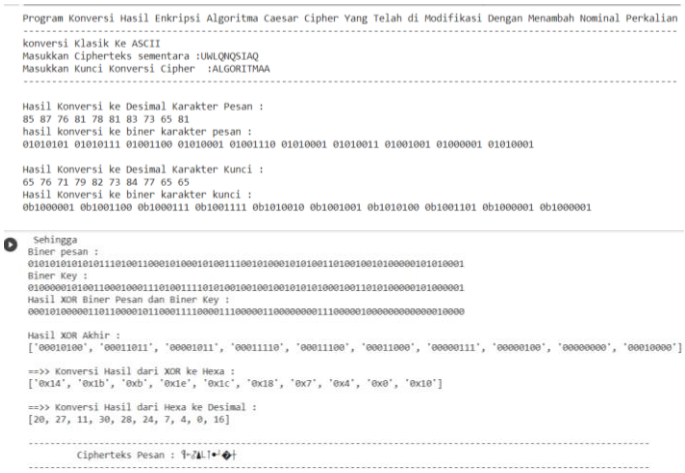
```
from string import ascii_lowercase,ascii_uppercase
,digits
def caesar_dec(ciphertext, key_dec):
  hasil = []
  for d in ciphertext:
    if d in ascii_lowercase:
      dekripsi = ascii_lowercase.index(d)
      dekripsi = (dekripsi - key_dec) % 26
      hasil.append(ascii_lowercase[dekripsi])
    elif d in ascii_uppercase:
      dekripsi = ascii_uppercase.index(d)
      dekripsi = (dekripsi - key_dec) % 26
      hasil.append(ascii_uppercase[dekripsi])
    elif d in digits:
      dekripsi = digits.index(d)
      dekripsi = (dekripsi - key_dec) % 10
      hasil.append(digits[dekripsi])
    else:
      hasil.append(d)
```

```
  return "".join(hasil)
def main():
  print ("Program Modifikasi Algoritma Caesar Ciph
er Dengan Mengalikan kata kunci yang diinputkan de
ngan 4")
  print ("----------------------------------------
")
  print ("Program Konversi HasiL Enkripsi ASCII Ke
 Algoritma Caesar Cipher")
  textcipher = open('ciphertext.txt','r').read()
  ciphertext = input("Masukkan Plainteks Sementara
 : ")
  kunci awal = input("Masukan Kunci
 : ")
  key_dec = int(kunci_awal) *4
  print ('\nDeskripsi Cipherteks :  \"' + textciph
er + '" dengan Kunci ' + str(kunci_awal) + "    ")
  print ("Didapatkan pesan      :  ")
  print (caesar_dec(ciphertext, key_dec))
  print ("----------------------------------------
")
if __name__ == '__main__':
  main()
```

Adapun hasil eksekusi program terdapat pada Gambar 3, 4, 5 dan 6.

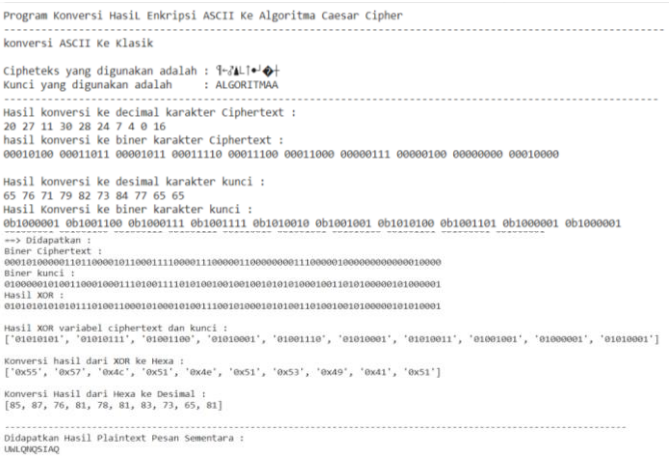
Gambar 3. *Ciphertext* Sementara

Enkripsi kedua dengan mengkonversi hasil dari modifikasi algoritma caesar cipher ke *decimal*, biner hexa dan *decimal* untuk menghasilkan karakter *ciphertext* akhir.



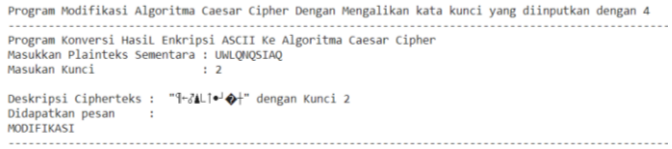
Gambar 4. *Ciphertext* Akhir

Deskripsi pertama dengan mengkonversi kembali karakter *ciphertext* akhir ke decimal, hexa, biner dan hexa untuk mendapatkan *plaintext* sementara.



Gambar 5. *Plaintext* Sementara

Deskripsi kedua dari hasil deskripsi pertama dengan kunci dan perkalian yang sama untuk mendapatkan plainteks akhir atau pesan asli.



Gambar 6. *Plaintext* Akhir

V. KESIMPULAN DAN SARAN

Berdasarkan hasil enkripsi dan deskripsi Caesar Cipher standar atau tanpa modifikasi yang bergantung pada panjang karakter *plaintext*, *ciphertext* yang dihasilkan hanya berupa huruf abjad yang diacak, sehingga masih ada kemungkinan pihak yang tidak bertanggungjawab untuk melakukan beberapa percobaan untuk dapat mengetahui isi pesan tersebut. Berbeda dengan proses enkripsi algoritma Caesar Cipher yang telah dimodifikasi dan dikonversi ke *ASCII* membuat *ciphertext* semakin sulit untuk dibaca dan dipahami. Hal ini ditunjukan dari *ciphertext* yang dihasil, yaitu berupa karakter gabungan, membuat *ciphertext* semakin sulit untuk dibaca dan dipahami oleh manusia, sehingga dapat meningkatkan keamanan pesan dan dapat meminimalisir terjadinya peretasan oleh pihak yang tidak bertanggungjawab. Oleh karena itu, penyandian pesan dengan menggunakan algoritma Caesar Cipher modifikasi pada *code ASCII* layak untuk diterapkan sebagai upaya dalam meningkatkan kompleksitas pengamanan data atau informasi. Namun dengan modifikasi algoritma Caesar Cipher proses enkripsi dan deskripsi pesan menjadi cukup panjang dan apabila proses dilakukan secara manual akan memakan cukup banyak waktu untuk mengembalikan *ciphertext* ke *plaintext*. Untuk penelitian selanjutnya diharapkan dapat mengembangkan hasil penelitian ini dengan menambahkan algoritma atau mengkobinasikan dengan algoritma lainnya sehingga menghasilkan proses enkripsi dan deskripsi yang semakin komplek dan menghasilkan *ciphertext* yang lebih sulit untuk dideskripsikan.

UCAPAN TERIMA KASIH

Terima kasih penulis sampaikan kepada Tim Jurnal TI ISB yang telah meluangkan waktu untuk melakukan review dan menerbitkan jurnal ini dan juga kepada seluruh pihak yang turut andil memberikan dukungan dan masukan yang positif.

REFERENSI

[1] O. Dakhi, M. Masril, R. Novalinda, and J. Ambiyar, "Analisis Sistem Kriptografi dalam Mengamankan Data Pesan Dengan Metode One Time Pad Chiper," vol. 20, no. 1, pp. 27–36, 2020, doi: 10.24036/invotek.v20i1.647.

[2] R. Politeknik Ganesha Medan, "Implementation of Combination Vigenere Cipher and RSA in Hybrid Cryptosystem for Text Security," *Int. J. Inf. Syst. Technol. Akreditasi*, vol. 4, no. 1, pp. 471–481, 2020.

[3] E. Setyawati, C. E. Widjayanti, R. Ramadhan Siraiz, and H. Wijoyo, "Pengujian keamanan komputer kriptografi pada surat elektronik berbasis website dengan enkripsi metode MD5," *J. Manaj. Inform. Jayakarta*, vol. 1, no. 1, pp. 56–67, 2021, [Online]. Available: http://journal.stmikjayakarta.ac.id/index.php/JMIJayakarta.

[4] A. Saraswat, C. Khatri, P. Thakral, and P. Biswas, "An Extended Hybridization of Vigenere and Caesar Cipher Techniques for Secure Communication," *2nd Int. Conf. Intell. Comput. Commun. Converg.*, vol. 92, pp. 355–360, 2016, doi: 10.1016/j.procs.2016.07.390.

[5] D. Rachmawati, S. Melvani Hardi, and R. Partogi Pasaribu, "Combination of columnar transposition cipher caesar cipher and lempel ziv welch algorithm in image security and compression," *J. Phys. Conf. Ser.*, vol. 1339, no. 1, 2019, doi: 10.1088/1742-6596/1339/1/012007.

[6] M. Cao and W. Zhang, "Related-Key Differential Cryptanalysis of the Reduced-Round Block Cipher GIFT," *IEEE Access*, vol. 7, pp. 175769–175778, 2019, doi: 10.1109/ACCESS.2019.2957581.

[7] F. I. Lubis, H. F. S. Simbolon, T. P. Batubara, and R. W. Sembiring, "Combination of caesar cipher modification with transposition cipher," *Adv. Sci. Technol. Eng. Syst.*, vol. 2, no. 5, pp. 22–25, 2017, doi: 10.25046/aj020504.

[8] M. M. Ali, "Cryptography : A Comparative Analysis for Modern Techniques," vol. 8, no. 6, pp. 442–448, 2017.

[9] T. Limbong, "Testing the Classic Caesar Cipher Cryptography using of Matlab," no. February, 2017, doi: 10.17605/OSF.IO/PEMA5.

[10] S. Godara, S. Kundu, and R. Kaler, "An Improved Algorithmic Implementation of Rail Fence Cipher," *Int. J. Futur. Gener. Commun. Netw.*, vol. 11, no. 2, pp. 23–32, 2018, doi: 10.14257/ijfgcn.2018.11.2.03.

[11] A. Code and A. Code, "ASCII Code - The extended ASCII table," pp. 1–7, 2013, [Online]. Available: http://www.ascii-code.com/.

[12] A. N. Syahrudin and T. Kurniawan, "Input Dan Output Pada Bahasa," *J. Dasar Pemrograman Python STMIK*, no. January, pp. 1–7, 2018.

[13] K. R. Srinath, "Python – The Fastest Growing Programming Language," pp. 354–357, 2017.

[14] Google, "Colaboratory: frequently asked questions ," 2018, [Access: 6-21-2018]. [Online]. Available:https://research.google.com/colaboratory/faq.html