Analisa Performa Enkripsi-Dekripsi Vigenere Cipher pada Media Teks

Natanael James Santoso[1], Christy Atika Sari[2]
[1,2]Program Teknik Informatika, Fakultas Ilmu Komputer, Universitas Dian Nuswantoro, Semarang, Indonesia

Artikel Info

Kata kunci:

Vigenere Cipher
Bit Error Rate
Character Error Rate
Entropy
Avalanche Effect

ABSTRAK

Vigenere Cipher menjadi salah satu algoritma dalam kriptografi yang telah lama digunakan sampai sekarang. Algoritma ini mudah untuk digunakan dan mudah untuk di implementasikan pada jaman dahulu, ini menjadi alasan yang kuat mengapa vigenere cipher menjadi algoritma yang cukup popular terutama untuk masa kini. Algoritma ini termasu ke dalam salah satu algoritma yang aman, namun setelah di temukan kelemahannya oleh Friedrich Kasiski algoritma ini sekarang sudah mulai jarang untuk digunakan. Kelemahan dari Vigenere Cipher dapat di lihat di hasil tes yang akan di lakukan. Dalam kasus ini, saya ingin menguji kemampuan atau performa dari vigenere cipher. Percobaan akan di lakukan dengan proses enkripsi dan dekripsi, bersamaan dengan pengujian yang akan penulis lakukan menggunakan BER (Bit Error Rate), CER (Cipher Error Rate), AE (Avalanche Effect), dan Entropy. Dari hasil pengujian dapat di lihat bahwa algoritma vigenere cipher ini cukup aman untuk digunakan sebagai enkripsi data, namun masih ada banyak algoritma lain yang lebih aman daripada vigenere, hal ini di karenakan hasil entropy yang tidak begitu tinggi dibandingkan algoritma lain yang lebih modern.

Penulis Korespondensi :

Christy Atika Sari
Program Studi Teknik Informatika
Fakultas Ilmu Komputer
Universitas Dian Nuswantoro, Semarang 50131
Email: christy.atika.sari@dsn.dinus.ac.id

1. PENDAHULUAN (10 PT)

Dengan adanya penggunaan tekhnologi yang berkembang secara pesat dari hari ke hari, maka akan semakin banyak data dan informasi yang dapat kita terima. Bahkan banyak dari kita yang memberikan informasi pribadi ke dalam dunia maya, baik itu nama, Alamat, ataupun dapat dengan nomor KTP, dll. Ketika kita melakukan pertukaran data di dunia maya, data tersebut dapat dikatakan tidak aman dan dapat terbuka untuk para *hacker* atau *cybercriminal* mencuri data tersebut. Di zaman sekarang ini pengamanan data menjadi hal yang penting untuk melindungi data pribadi dari *cybercriminal* atau tindakan illegal lainnya. Kriptografi memainkan peran penting dalam keamanan data. Kriptografi yaitu ilmu yang digunakan untuk mengamankan data dengan mengubah bentuk asli menjadi bentuk yang tidak dapat diuraikan sehingga hanya pihak yang di ijinkan yang dapat mengerti atau membaca data tersebut [1].

Berikut istilah terkait kriptografi di bawah ini :

- *Plaintext*
  Di dalam kriptografi, plaintext ialah teks awal yang belum di ubah menjadi cipher text. Data yang dapat di baca dan di mengerti tanpa adanya perubahan di sebut sebagai plaintext [10].
- *Key / kunci*
  Dapat berupa bilangan atau angka atau keduanya yang akan digunakan bersama dengan plaintext dan cipher text untuk enkripsi dan dekripsi [11].

- *Cipher text*
  Cipher text sebuah teks yang telah berubah dari plaintext menjadi teks yang tidak dapat di baca [13]. Cipher text di dapatkan dari hasil enkripsi plaintext dan kunci atau key.
- Enkripsi
  Enkripsi sebuah proses mengubah plaintext menjadi cipher text. Enkripsi membutuhkan algoritma dan kunci untuk mengubah plaintext menjadi cipher text. Hasil dari enkripsi berbeda-beda tergantung dengan algoritma yang digunakan [12].
- Dekripsi
  Dekripsi sebuah proses membalikan hasil enkripsi atau cipher text menjadi bentuk awal yaitu plaintext [12].

Tujuan dari kriptografi yaitu [5,15] :

- Autentikasi :
  Autentikasi sebuah proses untuk memverifikasi identitas pengirim.
- Privasi atau kerahasiaan
  Memastikan bahwa hanya penerima yang di ijinkan yang dapat membaca pesan.
- Integritas
  Menyakinkan penerima bahwa pesan yang diterima tidak di ubah dari pesan asli nya.
- Tidak dapat di bantah
  Sebuah metode yang memastikan bahwa pengirim benar-benar mengirim pesan.

Kriptografi memerlukan sebuah kunci dan juga algoritma yang akan di gunakan untuk mengubah data tersebut menjadi data yang abstrak. Algoritma yang akan saya gunakan yaitu adalah algoritma *Vigenere Cipher*.

2. METODE

2.1. Vigenere Cipher

Vigenere cipher adalah sebuah metode algoritma untuk meng enkripsi teks yang berupa huruf dan angka dengan menggunakan cara *Caesar cipher* yang berbeda-beda berdasarkan dengan keyword nya [6]. Vigenere cipher merupakan contoh algoritma *polyalphabetic cipher* [3]. *Vigenere ciphe*r termasuk ke dalam algoritma yang aman selama beberapa dekade, namun algoritma tersebut telah di ketahui kelemahannya. *Friedrich Kasiski* menemukan kelemahan *vigenere cipher* melalui sebuah pola [9]. Kelemahan yang kritis dari *vigenere cipher* terletak pada kuncinya/key yang dimana key nya selalu berulang menyesuaikan dengan panjang plaintextnya. Jika kita menemukan seberapa panjang kuncinya maka hasil dari cipher text dapat dengan mudah di ketahui [7]. Metode seperti *Kasiski* dapat membantu untuk menemukan panjang kunci. Algoritma ini sekarang menjadi lebih mudah untuk di tembus. Namun Algoritma ini masih banyak digunakan dalam berbagai algoritma cipher [16] seperti *AES (Advance Encryption Standard) [4]*. Algoritma cipher menggunakan table Bernama tabula recta dimana table tersebut berukuran 26 x 26 yang berisikan huruf dari A sampai Z. Algoritma *Vigenere cipher* juga dapat dihitung jika huruf A sampai Z diubah menjadi 0 sampai 25.

Rumus *Vigenere cipher* enkripsi [8] :

$$Ci = (Pi + Ki) \bmod 26 \quad (1)$$

Rumus *Vigenere Cipher* dekripsi [8] :

$$Pi = (Ci - Ki) \bmod 26 \quad (2)$$

Dimana,
C = *Cipher text*
P = *Plaintext*
K = Key

Proses akan dibagi menjadi dua yaitu berupa proses enkripsi dan dekripsi serta pengujian akan di lakukan menggunakan 4 pengujian yaitu Avalanche Effects, BER (Bit Error Rate), CER (Character Error Rate), dan Entropy. Teks *plaintext* dan *keyword* dapat menggunakan teks *ASCII* dari 32 sampai dengan 128. Proses Enkripsi dilakukan dengan mengambil input dari plaintext dan key yang nantinya dari inputan tersebut plaintext dan key akan mendapatkan hasil enkripsi, sedangkan untuk proses dekripsi, akan di lalui dengan mengambil input dari cipher text atau hasil enkripsi dan key, dari input tersebut akan dihasilkan dekripsi yang seharusnya berupa plaintext atau pesan original. Enkripsi dan dekripsi akan dilakukan dengan Bahasa pemrograman kotlin menggunakan android. Nanti dari hasil yang telah melalui proses enkripsi dan juga dekripsi akan digunakan untuk menilai pengujian yang akan di lakukan. Pengujian juga akan menggunakan plaintext dan juga key sesuai dengan pengujian yang dilakukan.

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

Gambar 1. Tabel *Tabula Recta* [19]

Berikut Pseudo Code dari enkripsi dan dekripsi program :

```
function encryption(plaintext, keyword):
    encryptedText = ""
    keyLength = length(keyword)

    for i = 0 to length(plaintext) - 1:
        plain = plaintext[i]
        key = keyword[i % keyLength]
        encryption = (plain + key - 32) % 95 + 32
        encryptedText = encryptedText + CHAR(encryption)

    return encryptedText

function decryption(encryption, keyword):
    decryptedText = ""
    keyLength = length(keyword)

    for i = 0 to length(encryption) - 1:
        encrypt = encryption[i]
        key = keyword[i % keyLength]
        decryption = (encrypt - key - 32) % 95
        if decryption < 0:
            decryption = decryption + 95
        decryption = decryption + 32
        decryptedText = decryptedText + CHAR(decryption)

    return decryptedText
```

Gambar 2. *Pseudo Code* Enkripsi dan Dekripsi

2.2. ASCII

American Standard Code for Information Interchange (ASCII) adalah acuan atau mewakili dalam melakukan pengkodean karakter untuk karakter dalam komunikasi yang mewakili huruf sampai simbol - sim bol pada perangkat telekomunikasi dan perangkat lain – lainnya [20]. Kode *ASCII* sendiri memilki sebanyak 256 kode, dimana 0 sampai 127 merupakan merupakan kode karakter dan 128 sampai 255 merupakan kode untuk symbol. Di dalam studi ini saya akan menggunakan kode karakter saja yaitu 0 sampai 127. Saya akan menggunakan *ASCII* yang berupa karakter agar hasil dari enkripsi dapat di baca oleh aplikasi, karena beberapa simbol tidak dapat di baca dengan baik oleh aplikasi. Berikut gambar tabel ASCII yang akan digunakan.

2.3. Avalanche Effect

Avalanche Effect menjadi salah satu bagian penting dalam testing atau pengujian enkripsi dalam algoritma kriptografi [18]. Penggunaan *Avalanche Effect* dengan cara mengganti salah satu input baik itu *plaintext* maupun *key,* sehingga *output cipher text* yang dihasilkan dapat berbeda [14]. *Avalanche Effect* sendiri digunakan untuk mengetahui berapa persen perubahan pesan dari pesan awal, dengan cara melihat jumlah bit dari *cipher text* yang telah berubah dan *cipher text* yang belum berubah saat proses enkripsi [20]. Semakin besar persen yang dihasilkan maka akan semakin bagus juga proses enkripsi yang di lalui, proses di anggap baik apabila nilai menunjukan 45 sampai 50 %. Avalanche Effect yang akan saya implementasikan yaitu dengan mengganti key nya, dengan karakter lain sehingga nantinya hasil dari enkripsi yang telah di ganti key nya akan berbeda dengan hasil enkripsi awal. Sebagai contoh jika terdapat *key* "Hello" maka dengan menggunakan avalanche effect, kata tersebut akan sedikit di ubah, misalkan saya ubah menjadi "Kello", dll.

$$Avalanche\ Effect = \frac{No.of\ flipped\ bits\ in\ the\ ciphered\ text}{No.of\ bits\ in\ the\ ciphered\ text} \times 100\ \% \quad (3)$$

No. of flipped bits in the ciphered text dapat dikatakan sebagai jumlah bits yang secara sengaja di buat salah atau error dengan tujuan untuk mengetahui berapa jumlah bit dari cipher text yang telah berubah dari cipher text yang telah berubah. *No. of bits in the ciphered text* dapat dikatakan sebagai berapa jumlah bits pada cipher text yang asli, yang nantinya jumlah bits tersebut akan di bagikan dengan jumlah bits yang *error* lalu di kalikan 100 agar hasil dari *avalanche effect* ini menjadi persen bukan desimal. Berikut *Pseudo Code* dari *Avalanche Effect* :

```
function simulateErrors(message, errorRate) {
    let messageChars = message;
    let random = new Random();
    for (let i = 0; i < messageChars.length; i++) {
        if (random.nextInt(100) < errorRate) {
            // Flip the bit (change character)
            messageChars[i] =
String.fromCharCode(messageChars[i].charCodeAt(0) ^ 1);
        }
    }
    return messageChars.join('');
}

function calculateAE(originalCipher, modifiedCipher) {
    let totalBitChanges = 0;

    for (let i = 0; i < originalCipher.length; i++) {
        if (originalCipher[i] != modifiedCipher[i]) {
            totalBitChanges++;
        }
    }
    return totalBitChanges / originalCipher.length;
}
```

Gambar 5. *Pesudo Code AE*.

2.3. BER (Bit Error Rate)

BER atau *Bit Error Rate* di definisikan sebagai berapa kali *error* atau kesalahan terjadi [21]. *BER* sendiri merupakan total dari jumlah bit yang *error* di bagi dengan jumlah bit yang telah terkirim atau ter proses [23]. Nilai *BER* yang baik yaitu mendekati nilai 0, dengan demikian tidak dapat perbedaan antara nilai hasil dekripsi dan data asli [22]. Pengujian dari BER, akan saya lakukan dengan mensimulasikan *error*, sehingga nantinya *BER* akan meghitung dengan rumus (4). Simulasi *error* bertujuan untuk menguji algoritma dengan cara mensimulasikan *error* pada ciphertext yang nantinya akan di hitung dengan jumlah bits awal, nantinya hasil dari perhitungan menjadi hasil pengujian untuk BER.

$$BER = \frac{Number\ of\ bits\ with\ error}{Total\ number\ of\ bits\ sent} \quad (4)$$

Number of bits with error berupa jumlah *error* pada *bits cipher text* atau hasil enkripsi yang digunakan untuk menguji hasil dari dekripsi yang dilakukan tanpa simulasi *error* dan juga tanpa adanya simulasi *error*. *Total number of bits sent* yaitu berupa jumlah dari bits hasil dekripsi yang aslinya. Hasil dari jumlah *bits* yang *error* dibagi dengan jumlah *bits plaintext* asli akan menjadi hasil pengujian *BER*. Berikut *Pseudo Code* dari BER :

```
function simulateErrors(message, errorRate) {
    let messageChars = message;
    let random = new Random();
    for (let i = 0; i < messageChars.length; i++) {
        if (random.nextInt(100) < errorRate) {
            // Flip the bit (change character)
            messageChars[i] =
String.fromCharCode(messageChars[i].charCodeAt(0) ^ 1);
        }
    }
    return messageChars.join('');
}

function calculateBER(originalMessage, receivedMessage) {
    let errorCount = 0;
    for (let i = 0; i < originalMessage.length; i++) {
        if (originalMessage[i] != receivedMessage[i]) {
            errorCount++;
        }
    }
    return errorCount / originalMessage.length;
}
```

Gambar 7. *Pseudo Code BER*

2.4 CER (Character Error Rate)

Character Error Rate (CER) adalah salah satu metode pengujian yang digunakan untuk mengukur presentase tingkat akurasi sebuah hasil enkripsi dengan cara mencocokan dan membandingkan character (huruf, angka, simbol) yang dimiliki plainteks dibandingkan dengan plainteks yang ditambah atau diubah semakin rendah hasilnya semakin bagus juga hasil enkripsinya dan sebaliknya [20]. Pengujian *CER* akan di jalankan dengan menggunakan hasil dekripsi, yang di mana nantinya hasil dekripsi akan di hitung apakah memiliki *error* atau tidak jika di bandingkan dengan plaintext atau teks awal.

$$Character\ Error\ Rate = \frac{Jumlah\ karakter\ berbeda}{Jumlah\ karakter\ yang\ dikirim} \quad (5)$$

Gambar 8. Rumus *CER.*

Jumlah karakter berbeda sama dengan pengujian yang dilakukan dengan menggunakan BER, yakni dengan menghitung berapa jumlah error pada hasil enkripsi, namun untuk pengujian ini saya tidak menggunakan simulasi error lagi, sehingga seharusnya hasil CER yang benar yaitu 0. Jumlah Karakter yang dikirim seperti pada pengujian BER, yaitu berupa jumlah dari bits hasil dekripsi yang aslinya. Hasil dari

pengujian ini seharusnya memberikan hasil 0, karena tidak ada simulasi error dan juga nilai 0 membuktikan bahwa hasil dekripsi bekerja dengan baik.

```
function calculateCER(originalText, decryptedText) {
    var errorCount = 0;
    for (var i = 0; i < originalText.length; i++) {
        if (originalText[i] != decryptedText[i]) {
            errorCount++;
        }
    }

    return (errorCount / originalText.length) * 100;
}
```

Gambar 9. *Pseudo Code CER.*

2.5. Entropy

Entropy merupakan sebuah metode menghitung berapa jumlah variable yang acak [24]. *Entropy* digunakan untuk menguji tingkat ke acakan suatu karakter [25]. Dalam kriptografi pengujian *entropy* di lakukan dengan mengukur seberapa besar nilai ke acakan hasil *enkripsi* atau *cipher text*. Hasil dari *entropy* dikatakan baik jika hasil atau nilai pengukuran tinggi. Semakin tinggi maka hasil *enkripsi* semakin baik, dikarenakan nilai ke acakan nya yang tinggi maka *cipher text* tersebut akan semakin sulit untuk di baca. Proses dari *entropy* yang akan saya uji, akan mengambil cipher text lalu akan saya hitung seberapa tinggi nilai *entropy* nya. Entropi akan digunakan sebagai salah satu pengujian yang akan di lakukan pada aplikasi yang telah saya buat, pengujian ini menjadi faktor penting untuk mengukur seberapa aman algoritma Vigenere Cipher ini. Dari entropi juga, kita dapat untuk membandingkan algoritma Vigenere cipher dengan algoritma lainnya melalui hasil dari entropi ini.

$$\mathrm{E(x)} = -\sum_{i=l}^{n} P(x_i).\log_2 P(x_i)$$

Dimana :
E(X) = Entropi dari variable x
n = jumlah kemungkinan nilai yang dapat di ambil oleh x
P(xi) = probabilitas munculnya xi pada x

```
function calculateEntropy(text) {
    charFrequency = mutableMapOf<Char, Int>()
    totalChars = 0

    for each character c in text {
        if c is a letter {
            totalChars++
            if c is already in charFrequency {
                charFrequency[c]++
            } else {
                charFrequency[c] = 1
            }
        }
    }

    entropy = 0.0
    for each frequency in charFrequency {
        probability = frequency / totalChars
        entropy -= probability * log2(probability)
    }

    return entropy
}
```

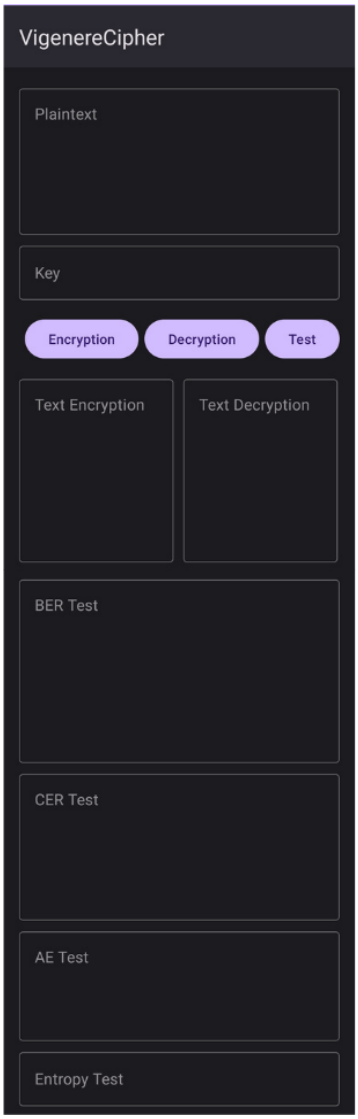
Gambar 11. *Pseudo Code Entropy.*

3. PEMBAHASAN HASIL

Dalam bab ini, saya akan menampilkan hasil dari penelitian yang telah saya buat. Hasil yang akan saya berikan akan di bagi menjadi beberapa sub bab, yaitu proses enkripsi menggunakan algoritma vigenere cipher. Hasil dari proses enkripsi dan dekripsi tersebut akan melalui proses pengujian yang dibagi menjadi 4 pengujian yaitu BER, CER, AE, dan Entropy.

3.1. Proses Enkripsi dan Dekripsi

Enkripsi dan dekripsi akan di lakukan dengan aplikasi *android*, di mana nantinya ketika aplikasi di buka akan di minta untuk mengisi *plaintext* dan juga *key* untuk melalui proses enkripsi dan dekripsi. Setalah di isi tombol *Encryption* dan *Decryption* di tekan sehingga *Text Encryption* dan *Text Decryption* akan terisi hasil proses enkripsi dan dekripsi. Saya akan melakukan 3 skenario proses, yang nanti hasil dari enkripsi dan dekripsi tersebut akan saya gunakan untuk menentukan hasil dari proses pengujian.



Gambar 12. Gambar Tampilan Aplikasi

Berikut langkah kerja dari aplikasi pada Gambar 12 :

- Saat masuk ke dalam aplikasi akan di sambut dengan beberapa hal yaitu *plainteks* dan *key* yang harus di isi terlebih dahulu. Lalu ada 3 tombol yang dapat di tekan yaitu tombol *encryption, decryption, dan test*, masing-masing dari tombol tersebut memiliki fungsi sendiri - sendiri.
- Saat sudah mengisi *plaintext* dan *key*, jika menekan tombol *encryption*, maka *text encryption* akan terisi dengan hasil yang telah melalui proses enkripsi.

- Jika menekan tombol *decryption* maka *text decryption* akan terisi dengan hasil dekripsi dari *cipher text dan key*.
- Jika menekan tombol *test* maka akan melalui semua proses pengujian *BER, CER, AE,* dan *Entropy*. Dan masing - masing text area yaitu *BER test, CER test, AE test, dan Entropy test* akan terisi dengan hasil pengujian nya.

Setelah melalui proses enkripsi dan juga dekripsi dengan cara yang sudah di berikan tadi, kita akan memulai proses semua pengujian dari *BER* sampai kepada *entropy* dengan cara mengisi *plaintext* dan *key*. Pengujan dilakukan dengan *plaintext* dan *key* yang sudah di tentukan, *plaintext* akan di isi secara random berjumlah 3, dan *plaintext* akan di isi dengan *&16a*. Panjang dari *plaintext* akan berbeda-beda namun masih menggunakan *key* yang sama, nantinya hasil akan di lihat bagimana performa dari algoritma *Vigenere Cipher*. Hasil dari pengujian dari ketiga *plaintext* akan kita bandingkan dengan menggunakan sebuah tabel sebagai perhitungan untuk rata – rata dari perhitungan hasil pengujian. Dari rata – rata akan diketahui bagaimana proses dan hasil dari performa dan keamanan *Vigenere Cipher*.

	Skenario		
Keterangan	1	2	3
Plaintext	3CJ`>:6{"x:hs~bcRc X0eFc']vAMR'xT<q" "(gT7VpBOl>V8>H	HCWW<rASG'l(QlmaeI9x zVn4vNKLSaDoauR/W~T P<C)h(3\oI<)Q]E\DimTu(Z94(	HjI4)I[>31njrQbE0Df;NzUVZILD d4y,C7p5Lm9zW0F61BU'2(W9en Z)b4oJCXqU8,P`Z)Vh{U")cv/Dvc `68
Key	&16a	Js1#4	Ki19*71
Hasil Enkripsi	Yt!bdkl}HJpj:P9ex5/ 2,w:)$HwOxXOVbC X$N9+9|BxQ3o-:dy	3W)zp]U%j[W<#0BLyz\ Mej@WK9_}v6/$39'ykPw %'WZ,\}pAlpse/h1/}?wJrn jW\	4tzmS!-*=bH5J#NOa}1r f_(4s$uP>KemnB!V?rE/a2@b{ ^csaj?92ZN>A$m0CAB]*+2ZBr M/L`5b9uP.8g$
Hasil Dekripsi	3CJ`>:6{"x:hs~bcRc X0eFc']vAMR'xT<q" "(gT7VpBOl>V8>H	HCWW<rASG'l(QlmaeI9x zVn4vNKLSaDoauR/W~T P<C)h(3\oI<)Q]E\DimTu(Z94(	HjI4)I[>31njrQbE0Df;NzUVZILD d4y,C7p5Lm9zW0F61BU'2(W9en Z)b4oJCXqU8,P`Z)Vh{U")cv/Dvc `68

Tabel 1.Hasil Enkripsi dan Dekripsi.

3.2. Pengujian BER

Seperti yang sudah di jelaskan, *BER* menghitung jumlah *error bits* dan total *bits*. Di karenakan di sini saya menggunakan *cipher text* untuk menghitung *BER*, dengan cara mengganti beberapa karakter enkripsi sehingga hasil dekripsi akan berbeda. Nilai *BER* dikatakan baik apabila nilai mendekati angka 0 atau nilai 0, jika mendapatkan nilai 0 maka hasil dekripsi sama dengan teks awal. Hasil total dari 3 skenario mencapai 0,43, ini merupakan hasil yang baik karena nilai tersebut mendekati nilai 0.

	Skenario		
Keterangan	1	2	3
Plaintext	3CJ`>:6{"x:hs~bcRc X0eFc']vAMR'xT<q" "(gT7VpBOl>V8>H	HCWW<rASG'l(QlmaeI9x zVn4vNKLSaDoauR/W~T P<C)h(3\oI<)Q]E\DimTu(Z94(	HjI4)I[>31njrQbE0Df;NzUVZI LDd4y,C7p5Lm9zW0F61BU'2(W9enZ)b4oJCXqU8,P`Z)Vh{U ")cv/Dvc`68
Key	&16a	&16a	&16a
Hasil Enkripsi	Yt!bdkl}HJpj:P9ex5/ 2,w:)$HwOxXOVbC X$N9+9|BxQ3o-:dy	nt.YbDwUmXC*w>Dc,zo zA(E6= "Ny3zq(G)1}P+Rbt_jNd3 qom_S$v3F0?+wN,o6N	n< 6Oz2@YbEl9#9GVu==tL,X!z# F+eP.ihG7r?o|}a|8Ws,)XY.:,@1 +)eFLi*HW^]'b!Z- jB'X+*HeF=578^
Hasil Dekripsi	3CJ`>:6{"x:hs~bcRc X0eFc']vAMR'xT<q" "(gT7VpBOl>V8>H	HCWW<rASG'l(QlmaeI9x zVn4vNKLSaDoauR/W~T P<C)h(3\oI<)Q]E\DimTu(Z94(	HjI4)I[>31njrQbE0Df;NzUVZI LDd4y,C7p5Lm9zW0F61BU'2(W9enZ)b4oJCXqU8,P`Z)Vh{U ")cv/Dvc`68
jumlah bit error	3	3	3
Bit Error Rate	0.06	0.04	0.03
Rata-Rata BER	0.043333333		

Tabel 2. Hasil Pengujian BER.

3.3. Pengujian CER

Pengujian CER saya lakukan tanpa mengganti karakter plaintext, sehingga hasil yang saya dapatkan seharusnya bernilai 0 karena hasil dari enkripsi yang nantinya akan di dekripsikan lagi pasti akan Kembali ke plaintext awal. Hasil CER dikatakan baik apabila hasil mendekati nilai 0 ataupun bernilai 0 yang artinya tidak ada kesalahan. Hasil dari scenario yang mencapai nilai 0, maka dapat dikatakan hasil tersebut adalah hasil yang baik.

	Skenario		
Keterangan	1	2	3
Plaintext	3CJ`>:6{"x:hs~bcRc X0eFc']vAMR'xT<q" "(gT7VpBOl>V8>H	HCWW<rASG'l(QlmaeI9x zVn4vNKLSaDoauR/W~T P<C)h(3\oI<)Q]E\DimTu(Z94(	HjI4)I[>31njrQbE0Df;NzUVZIL Dd4y,C7p5Lm9zW0F61BU'2(W 9enZ)b4oJCXqU8,P`Z)Vh{U")cv /Dvc`68
Key	&16a	&16a	&16a
Hasil Enkripsi	Yt!bdkl}HJpj:P9ex5/ 2,w:)$HwOxXOVbC X$N9+9|BxQ3o-:dy	nt.YbDwUmXC*w>Dc,zo zA(E6= "Ny3zq(G)1}P+Rbt_jNd3 qom_S$v3F0?+wN,o6N	n< 6Oz2@YbEl9#9GVu==tL,X!z#F +eP.ihG7r?o|}a|8Ws,)XY.;,@1+) eFLi*HW^]'b!Z- jB'X+*HeF=578^
Hasil Dekripsi	3CJ`>:6{"x:hs~bcRc X0eFc']vAMR'xT<q" "(gT7VpBOl>V8>H	HCWW<rASG'l(QlmaeI9x zVn4vNKLSaDoauR/W~T P<C)h(3\oI<)Q]E\DimTu(Z94(	HjI4)I[>31njrQbE0Df;NzUVZIL Dd4y,C7p5Lm9zW0F61BU'2(W 9enZ)b4oJCXqU8,P`Z)Vh{U")cv /Dvc`68
Karakter Error	0	0	0
CER	0	0	0
Rata-Rata CER	0		

Tabel 3. Hasil Pengujian *CER*

3.4. Pengujian AE

	Skenario		
Keterangan	1	2	3
Plaintext	3CJ`>:6{"x:hs~bcRc X0eFc']vAMR'xT<q" "(gT7VpBOl>V8>H	HCWW<rASG'l(QlmaeI9xzV n4vNKLSaDoauR/W~TP<C) h(3\oI<)Q]E\DimTu(Z94(	HjI4)I[>31njrQbE0Df;NzUVZI LDd4y,C7p5Lm9zW0F61BU'2(W9enZ)b4oJCXqU8,P`Z)Vh{U ")cv/Dvc`68
Key	&16a	&16a	&16a
Hasil Enkripsi	Yt!bdkl}HJpj:P9ex5/ 2,w:)$HwOxXOVbC X$N9+9|BxQ3o-:dy	nt.YbDwUmXC*w>Dc,zoz A(E6="Ny3zq(G)1}P+Rbt_j Nd3qom_S$v3F0?+wN,o6N	n<6Oz2@YbEl9#9GVu==tL,X! z#F+eP.ihG7r?o|}a|8Ws,)XY.;, @1+)eFLi*HW^]'b!Z- jB'X+*HeF=578^
Key yang berubah	&17a	&17a	&17a
Hasil Enkripsi	Yt"bdkm}HJqj:P:ex5 02,w;)$HxOxXPVbC Y$N9,9|ByQ3o.:dy	nt/YbDxUmXD*w>Ec,zpzA (F6=#Ny3{q(G*1}P,Rbt`jN d4qom`S$v4F0?,wN,p6N	n<!6Oz3@YbFl9#:GVu>=tL- X!z$F+eQ.ihH7r?p|}a}8Ws-)XY/;,@2+)eGLi*IW^](b!Z.jB' Y+*HfF=588^
AE	76%	75%	75%
Rata-Rata AE	75%		

Tabel 4. Hasil Pengujian AE

Pengujian AE dilakukan dengan memodifikasi atau mengganti key, dari modifikasi tersebut AE akan dihitung dengan membandingkan hasil enkripsi lama dengan hasil enkripsi baru yang menggunakan key yang telah di ubah. Hasil dari AE menunjukkan 75%, yang dimana hasil ini termasuk ke dalam nilai yang baik, karena kategori nilai yang baik dalam pengujian AE adalah 50% ke atas.

3.5. Pengujian Entropy

Pengujian *Entropy* dilakukan dengan cara menghitung nilai ke acakan hasil enkripsi atau *cipher text*. Hasil dari *entropy* di tabel bawah menunjukkan 4,51. Nilai *entropy* yang baik seharusnya lebih dari hasil ini. Bahkan algoritma yang lebih *modern* dapat menunjukkan nilai *entropy* yang jauh lebih besar, ingat bahwa nilai entropy akan semakin baik apabila nilai nya semakin tinggi. Algoritma vigenere cipher menunjukkan nilai entropy yang rendah di karenakan algoritma ini merupakan algoritma yang cukup lama, dan akan kalah jika di bandingkan dengan algoritma lain yang lebih baru. Sebenarnya untuk nilai *Entropy* sendiri dapat lebih meningkat dari nilai yang sekarang, dapat di tingkatkan lagi karena *plaintext* yang digunakan dengan *key* dapat di ganti menjadi lebih rumit untuk meningkatkan nilai *entropy*, dan *factor* lain seperti Panjang dari *plaintext* dan *key* juga dapat mempengaruhi nilai *entropy*. Inilah mengapa nilai *entropy* yang di hasilkan tidak begitu besar.

	Skenario		
Keterangan	1	2	3
Plaintext	3CJ`>:6{"x:hs~bcRc X0eFc']vAMR'xT<q" "(gT7VpBOl>V8>H	HCWW<rASG'l(QlmaeI9xz Vn4vNKLSaDoauR/W~TP< C)h(3\oI<)Q]E\DimTu(Z94(	HjI4)I[>31njrQbE0Df;NzU VZILDd4y,C7p5Lm9zW0F 61BU'2(W9enZ)b4oJCXqU 8,P`Z)Vh{U")cv/Dvc`68
Key	&16a	&16a	&16a
Hasil Enkripsi	Yt!bdkl}HJpj:P9ex5/ 2,w:)$HwOxXOVbC X$N9+9|BxQ3o-:dy	3W)zp]U%j[W<#0BLyz\Me j@WK9_}v6/$39'ykPw%'W Z,\}pAlpse/h1/}?wJrnjW\	n< 6Oz2@YbEl9#9GVu==tL,X !z#F+eP.ihG7r?o|}a|8Ws,)X Y.;,@1+)eFLi*HW^]'b!Z- jB'X+*HeF=578^
Entropy	4.41	4.51	4.61
Rata-rata Entropy	4.51		

Tabel 5. Hasil Pengujian Entropy

4. KESIMPULAN

Berdasarkan penelitian yang telah dijalankan, dapat diambil kesimpulan bahwa vigenere cipher merupakan algoritma yang cukup baik dalam menyembunyikan atau mengamankan data, walaupun termasuk ke dalam kategori algoritma yang sudah cukup lama. Hasil pengujian menunjukkan nilai yang baik, yang artinya algoritma ini cukup baik dalam mengamankan sebuah pesan atau data. Kekurangan dari algoritma ini dapat terlihat dalam hasil entropy, memang algoritma ini cukup baik dalam mengamankan data, namun masih ada banyak algoritma modern yang lain yang lebih aman dalam mengenkripsikan data daripada algoritma ini.

REFERENCES

[1] Khairun Nahar, Partha Chakraborty. “A Modified Version of Vigenere Cipher using 95×95 Table”. *International Journal of Engineering and Advanced Technology (IJEAT)*. , vol. 9, no. 5, pp. 1144 – 1148, 2020.

[2] Aized Amin Soofi, Irfan Riaz, Umair Rasheed. “An Enhanced Vigenere Cipher For Data Security”. *INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH*. Vol. 5 no. 3, pp. 141-145, 2016.

[3] Imam Saputra, Mesran, Nelly Astuti Hasibuan, Robbi Rahim. “Vigenere Cipher Algorithm with Grayscale Image Key Generator for Secure Text File”. *International Journal of Engineering Research & Technology (IJERT).* Vol. 6, no. 1, pp. 266-269, 2017.

[4] Amer Nadeem, Dr M. Younus Javed. “A Performance Comparison of Data Encryption Algorithms”. IEEE Xplore. pp. 84-89, 2022.

[5] G. C. Kessler. “An overview of cryptography”. ed:Gary C. Kessler, 2023.

[6] Surya Darma Nasution, Muhammad Syahrizal, Guidio Leonarde Ginting, Robbi Rahim. “Data Security Using Vigenere Cipher and Goldbach Codes Algorithm”. *International Journal of Engineering Research & Technology (IJERT).* Vol. 6, no. 1, pp. 360-363, 2017.

[7] April Lia Hananto, Arip Solehudin, Agung Susilo Yuda Irawan, Bayu Priyatna. “Analyzing the Kasiski Method Against Vigenere Cipher”. *International Journal of Computer Techniques*. Vol. 6, no. 6, pp. 1-8, 2019.

[8] Fairouz Mushtaq Sher Ali, Falah Hassan Sarhan. "Enhancing Security of Vigenere Cipher by Stream Cipher". *International Journal of Computer Applications.* Vol. 100, no. 1, pp. 1-4, 2014.
[9] Quist-Aphetsi Kester. "A cryptosystem based on Vigenère cipher with varying key". *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*. Vol. 1, no. 10, pp. 108-113, 2012.
[10] T.Gunasundari, Dr. K.Elangovan. "A Comparative Survey on Symmetric Key Encryption Algorithms". International Journal of Computer Science and Mobile Applications. vol. 2, no. 2, pp. 78-83, 2014.
[11] Monika Agrawal, Pradeep Mishra. "A Comparative Survey on Symmetric Key Encryption Techniques". International Journal on Computer Science and Engineering (IJCSE). vol. 4, no. 5), pp. 877-882, 2012.
[12] William Stallings. "Cryptography and Network Security: Principles and Practice. Sixth Edition". New Jersey: Pearson Education, Inc. 2018: 28.
[13] Shyam Nandan Kumar. "Review on Network Security and Cryptography". International Transaction of Electrical and Computer Engineers System. Vol. 3, no. 1, pp. 1-11, 2015.
[14] Sriram Ramanujam, Marimuthu Karuppiah. "Designing an algorithm with high Avalanche Effect". IJCSNS International Journal of Computer Science and Network Security. Vol. 11, no. 1, pp. 106-111, 2011.
[15] E. Surya, C.Diviya . "A Survey on Symmetric Key Encryption Algorithms". International Journal of Computer Science & Communication Network. Vol. 2, no. 4, pp. 475-477.
[16] Aiden A. Bruen, Mario A. Forcinito. "Cryptography, Information Theory, and Error-Correction: A Handbook for the 21st Century". New Jersey: John Wiley & Sons, Inc. 2011:21
[17] Al-Amin Mohammed Aliyu, Abdulrahman Olaniyan. "Vigenere Cipher: Trends, Review and Possible Modifications". International Journal of Computer Applications. Vol. 135, no. 11, pp. 46-50, 2016.
[18] Amish Kumar, Mrs. Namita Tiwari. "EFFECTIVE IMPLEMENTATION AND AVALANCHE EFFECT OF AES". International Journal of Security, Privacy and Trust Management (IJSPTM). Vol. 1, no. 3/4, pp. 31-35, 2012.
[19] Thamer Hassan Hameed, Haval Tariq Sadeeq. "Modified Vigenère cipher algorithm based on new key generation method". Indonesian Journal of Electrical Engineering and Computer Science. vol. 28, no. 2, pp. 954-961, 2022.
[20] Muslih, Lekso Budi Handoko. "PENGUJIAN AVALANCHE EFFECT PADA KRIPTOGRAFI TEKS MENGGUNAKAN AUTOKEY CIPHER". 2st Proceeding STEKOM. Vol. 2, no. 1, pp. 127-134, 2022.
[21] Irfan Ali. "Bit-Error-Rate (BER) Simulation Using MATLAB". International Journal of Engineering Research and Applications. Vol. 3, no. 1, pp. 706-711, 2013.
[22] Candra Irawan, Eko Hari Rachmawanto, Christy Atika Sari, Castaka Agus Sugianto. "SUPER ENKRIPSI FILE DOKUMEN MENGGUNAKAN BEAUFORT CIPHER DAN TRANSPOSISI KOLOM". Seminar Nasional LPPM. 2020.
[23] Md. Golam Sadeque. "Bit Error Rate (BER) Comparison of AWGN Channels for Different Type's Digital Modulation Using MATLAB Simulink". American Scientific Research Journal for Engineering, Technology, and Sciences (ASRJETS). Vol. 13, no. 1, pp. 61-71, 2015.
[24] EMIL SIMION. "Entropy and Randomness: From Analogic to Quantum World". IEEE Access. vol. 8, pp. 74553-74561, 2020.
[25] Andrew L. Rukhin. "Approximate Entropy for Testing Randomness". Cambridge University Press.; vol. 37, no. 1, pp. 1-16, 2016.