PENGAMANAN DATA TEKS MELALUI PERPADUAN ALGORITMA *BEAUFORT* DAN *CAESAR CIPHER*

Muhammad Fadlan[1], S. Sinawati[2], Aida Indriani[3], Evi Dianti Bintari[4]

[1,2]Program Studi Sistem Informasi
[3]Program Studi Teknik Informatika
[4]Program Studi Manajemen Informatika
Sekolah Tinggi Manajemen Informatika dan Komputer PPKIA Tarakanita Rahmawati Tarakan
fadlan@ppkia.ac.id, sinawati@ppkia.ac.id, aida@ppkia.ac.id, evi@ppkia.ac.id

ABSTRACT

The importance of maintaining data integrity and security is one of the challenges in the current digital era. One method that can be used to face this challenge is through cryptography. In cryptography there are several algorithms that can be used, one of which is the Caesar cipher algorithm. This algorithm has several disadvantages, including a limited number of characters of 26 characters. This can make the encryption results easily recognizable by other parties. This study aims to design a proposal for maintaining data security through cryptographic techniques, while addressing the problems inherent in the Caesar cipher algorithm. The combination of Caesar and Beaufort algorithm is done to overcome the existing problems. In addition, a character list of 94 characters was determined to be used in the process of encryption and decryption of text data. The result, through the integration of these two algorithms, the text cipher becomes more difficult to solve. There are two stages of the encryption process by using two different types of Keys for each stage in securing data.

***Key*words:** *Beaufort, Caesar, Cryptography, Data Security*

ABSTRAK

Pentingnya menjaga keutuhan maupun keamanan data merupakan salah satu tantangan dalam era digital saat ini. Salah satu cara yang dapat digunakan menghadapi tantangan ini adalah melalui kriptografi. Dalam kriptografi terdapat beberapa algoritma yang dapat digunakan, salah satunya adalah algoritma *caesar cipher*. Algoritma ini memiliki beberapa kelemahan, di antaranya jumlah karakter yang terbatas sebanyak 26 karakter. Hal ini yang dapat membuat hasil enkripsi dengan mudah dapat dikenali oleh pihak lain. Penelitian ini bertujuan untuk merancang sebuah usulan dalam menjaga keamanan data melalui teknik kriptografi, sekaligus mengatasi permasalahan yang terdapat pada algoritma *caesar cipher*. Perpaduan antara algoritma *Caesar* dan algoritma *Beaufort* dilakukan untuk mengatasi permasalahan yang ada. Selain itu, dilakukan penentuan daftar karakter sebanyak 94 karakter yang dapat digunakan dalam melakukan proses enkripsi dan dekripsi terhadap data teks. Hasilnya, melalui perpaduan dua algoritma ini, maka *cipher* teks menjadi lebih sulit untuk dipecahkan. Hal ini dikarenakan terdapat dua tahapan proses enkripsi dengan menggunakan dua jenis kunci yang berbeda untuk tiap-tiap tahapan dalam mengamankan data.

Kata Kunci: *Beaufort, Caesar, Kriptografi, Keamanan Data*

DOI: http://dx.doi.org/10.15408/jti.v12i2.12262

I. PENDAHULUAN

Menjaga keutuhan maupun keamanan data merupakan salah satu tantangan terberat di era digital. Data maupun informasi merupakan hal yang penting bagi sebuah organisasi [1]. Salah satu unsur penting dari sebuah data adalah kerahasiaan data itu sendiri [2]. Menjaga kerahasiaan sebuah data dari berbagai pihak yang tidak berkepentingan merupakan salah satu hal penting untuk dilakukan [3]. Jatuhnya data maupun informasi penting milik sebuah organisasi kepada orang lain tentu saja akan membawa kerugian bagi organisasi tersebut. Untuk itu dibutuhkan sebuah cara dalam mengamankan data maupun informasi penting yang dimiliki organisasi.

Kriptografi merupakan salah satu cabang maupun disiplin ilmu yang dapat digunakan dalam mengamankan sebuah data [4]. Ilmu ini dapat digunakan pada hampir semua saluran komunikasi baik dalam jaringan maupun tanpa jaringan [4], [5]. Setelah melalui proses pengamanan data melalui kriptografi, sebuah data maupun informasi menjadi tidak dapat dipahami oleh berbagai pihak yang tidak memiliki kewenangan. Salah satu potensi yang didapat melalui teknik kriptografi adalah hanya pihak-pihak yang berwenang saja yang mampu mengetahui data sebenarnya yang telah diacak [3].

Berbagai jenis data dapat diamankan dengan menggunakan kriptografi, termasuk di antaranya adalah data berupa teks. Contoh data teks tersebut antara lain: laporan keuangan, laporan nilai mahasiswa maupun laporan inventaris kantor. Dalam kriptografi, proses yang dapat digunakan untuk mengamankan data teks adalah proses enkripsi. Melalui proses enkripsi ini data akan di acak sehingga tidak dapat diketahui oleh pihak maupun oknum yang tidak berkepentingan [6][7]. Di dalam ilmu kriptografi terdapat berbagai algoritma yang dapat digunakan. Salah satunya adalah algoritma *Caesar cipher.*

Algoritma *Caesar cipher* merupakan algoritma yang menggunakan jenis kunci simetris dalam melakukan enkripsi maupun dekripsi. Artinya, kunci yang digunakan untuk proses dekripsi sama dengan kunci yang digunakan untuk proses enkripsi [8]. Kelemahan dari algoritma *Caesar* adalah cipherteks dari hasil enkripsi yang dapat dengan mudah dikenal oleh *cryptanalysis*, hal ini dikarenakan proses enkripsi yang digunakan tergolong sederhana [9], yaitu dengan hanya menggeser karakter asli menjadi beberapa karakter setelahnya sesuai dengan kunci tunggal yang digunakan.

Beberapa penelitian terdahulu terkait dengan pemanfaatan algoritma *Caesar* dalam mengamankan data, diantaranya penelitian yang dilakukan oleh [10], penelitian ini hanya menggunakan algoritma *Caesar* dalam mengamankan data berupa teks. Penelitian yang dilakukan oleh [11], penelitian ini menggunakan algoritma *caesar cipher* dalam melakukan pengamanan terhadap aplikasi *chatting.* Kelemahan dari kedua penelitian ini adalah kurang kuatnya konsep pengamanan data yang digunakan, karena hanya mengandalkan algoritma *caesar cipher*.

Selanjutnya, penelitian yang dilakukan oleh [8], dalam penelitian ini digunakan kombinasi *Caesar cipher* dengan algoritma kriptografi modern *three pass protocol* dalam mengamankan data, sedangkan, pada penelitian yang dilakukan oleh [12] dilakukan kombinasi menggunakan *Caesar* dan *rail fence* dalam mengamankan data. Kombinasi ini dilakukan karena kedua metode tergolong rentan untuk diretas jika diterapkan secara sendiri-sendiri. Kelemahan dari penelitian ini adalah pada beberapa kondisi terdapat hasil ujicoba terhadap proses dekripsi yang tidak sesuai dengan *plaintext* aslinya.

Oleh karena itu, penelitian ini bertujuan untuk mengoptimalkan penggunaan algoritma *Caesar* dalam mengamankan sebuah data. Optimalisasi dilakukan melalui perancangan sebuah konsep perpaduan/*hybrid* antara algoritma *Caesar* dengan algoritma kriptografi lainnya. Algoritma yang diusulkan pada penelitian ini untuk dipadukan dengan algoritma *Caesar* adalah algoritma *beaufort.* Algoritma *Beaufort* Cipher adalah salah satu metode dalam kriptografi klasik yang juga merupakan turunan dari Algoritma *Viginere Cipher* [13][14][15]. Algoritma ini memiliki banyak kesamaan dengan *Vigenere Cipher*, salah satunya adalah pemanfaatan fungsi modulo dalam proses enkripsi maupun dekripsinya [14]. Perbedaan yang paling mendasar antara kedua metode ini adalah pada *Viginere Cipher* menggunakan operasi aritmatika penjumlahan. Sedangkan, algoritma *Beaufort* menggunakan operasi aritmatika pengurangan [16].

Kelebihan dari algoritma *Beaufort* adalah jumlah kunci yang digunakan memiliki panjang

yang sama dengan jumlah karakter pesan asli/*plaintext*. Hal inilah yang dapat membuat pesan hasil enkripsi menjadi sulit untuk diketahui oleh pihak lain, karena tiap-tiap karakter *plaintext* akan memiliki pasangan kunci yang berbeda dengan karakter *plaintext* lainnya.

Perbedaan mendasar antara penelitian yang dilakukan dengan beberapa penelitian terkait sebelumnya adalah terletak pada konsep dan algoritma yang digunakan. Melalui konsep *hybrid* / perpaduan dua algoritma *Beaufort* dan *Caesar* yang dirancang dalam penelitian ini diharapkan dapat digunakan dalam meningkatkan keamanan dari sebuah data teks dan dapat mengatasi kelemahan-kelemahan dari beberapa penelitian sebelumnya. Hal ini dapat terjadi karena melalui konsep perpaduan ini, proses enkripsi akan dilakukan sebanyak dua kali, yaitu melalui proses enkripsi menggunakan *Beaufort* dan enkripsi dengan *Caesar*. Sehingga, pesan hasil enkripsi menjadi sulit untuk diketahui oleh pihak-pihak lain yang tidak berkepentingan.

II. METODOLOGI

Dalam mendukung jalannya penelitian terkait dengan pengamanan data teks agar lebih terarah dan sistematis, maka dibutuhkan suatu tahapan penelitian yang disusun dengan baik. Adanya tahapan ini juga dapat memudahkan dalam melakukan penulisan artikel. Tahapan pada penelitian ini dapat dilihat pada Gambar 1.

Berdasarkan Gambar 1, tahapan penelitian diawali dengan melakukan kajian atau studi pustaka. Studi pustaka dilakukan untuk mendapatkan berbagai informasi yang relevan terkait dengan penelitian yang akan dilakukan. Informasi tersebut antara lain terkait dengan algoritma *Beaufort*, *caesar cipher*, dan penelitian-penelitian sebelumnya terkait dengan penelitian yang dilakukan. Hasil dari kajian pustaka yang telah dilakukan dapat dilihat pada artikel, khususnya pada Bagian Tinjauan Pustaka.

Setelah melakukan studi pustaka, langkah selanjutnya adalah melakukan perancangan konsep pengamanan data teks. Konsep pengamanan data teks akan menggunakan perpaduan antara algoritma *Beaufort* dan *Caesar cipher*. Konsep pengamanan tersebut dapat dilihat lebih lanjut pada Gambar 2.

Gambar 1. Tahapan penelitian

Tahapan penelitian selanjutnya adalah menerapkan konsep yang dirancang pada sebuah kasus pengamanan data teks. Tahapan selanjutnya adalah melakukan pembahasan terhadap hasil dari penerapan konsep terhadap sebuah studi kasus pengamanan data teks. Kedua tahapan ini akan dibahas lebih lanjut pada bagian Hasil dan Pembahasan dalam artikel ini. Langkah terakhir dari artikel adalah melakukan penarikan kesimpulan terhadap hasil dari penelitian yang telah dilakukan, khususnya terkait dengan pengamanan data teks menggunakan konsep yang telah dirancang.

2.1. Daftar Karakter

Untuk mengantisipasi kelemahan yang terdapat pada algoritma *Beaufort* dan *Caesar*, khususnya terkait dengan jumlah karakter yang digunakan, maka pada penelitian ini akan menggunakan karakter tertentu yang telah ditentukan. Daftar karakter tersebut dapat dilihat pada Tabel 1.

Tabel 1. Daftar karakter yang digunakan

Nilai Desimal	Karakter	Nilai Desimal	Karakter
0	1	47	l
1	2	48	m
2	3	49	n
3	4	50	o

Tabel 1. Lanjutan

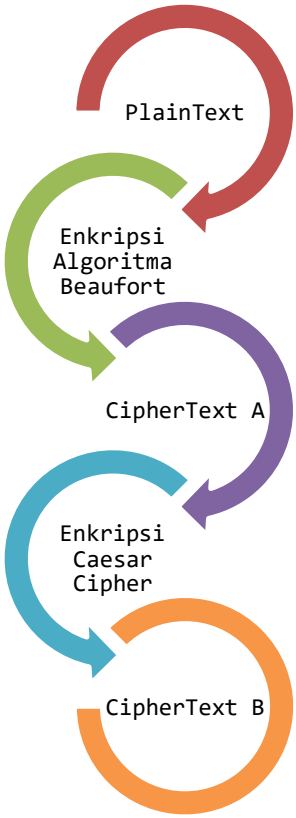
Nilai Desimal	Karakter	Nilai Desimal	Karakter
4	5	51	p
5	6	52	q
6	7	53	r
7	8	54	s
8	9	55	t
9	0	56	u
10	A	57	v
11	B	58	w
12	C	59	x
13	D	60	y
14	E	61	z
15	F	62	~
16	G	63	!
17	H	64	@
18	I	65	#
19	J	66	$
20	K	67	%
21	L	68	^
22	M	69	&
23	N	70	*
24	O	71	(
25	P	72	)
26	Q	73	-
27	R	74	+
28	S	75	=
29	T	76	{
30	U	77	}
31	V	78	|
32	W	79	[
33	X	80	]
34	Y	81	\
35	Z	82	:
36	a	83	"
37	b	84	;
38	c	85	
39	d	86	<
40	e	87	>
41	f	88	?
42	g	89	,
43	h	90	.
44	i	91	/
45	j	92	`
46	k	93	spasi

Pada Tabel 1 berisi daftar karakter yang akan digunakan dalam penelitian ini. Total karakter yang terdapat pada Tabel 1 tersebut sebanyak 94 karakter, yang terdiri atas karakter huruf, angka dan simbol-simbol tertentu yang terdapat pada *keyboard*.

Karakter-karakter yang terdapat pada Tabel 1 inilah yang nantinya akan digunakan dalam melakukan proses enkripsi maupun dekripsi pada algoritma *Beaufort* dan *Caesar cipher*.

2.2. Rancangan Proses Enkripsi

Secara keseluruhan usulan konsep untuk melakukan proses enkripsi terhadap data teks pada penelitian ini dapat dilihat pada Gambar 2.



Gambar 2. Usulan proses enkripsi

Pada Gambar 2, terlihat bahwa terdapat 5 tahapan dalam melakukan proses enkripsi dengan memanfaatkan algoritma *Beaufort* dan *Caesar*. Langkah pertama diawali dengan menentukan plain teks atau data berupa teks yang akan diacak melalui proses enkripsi. Pada tahap ini juga harus ditentukan kunci yang akan digunakan dalam proses enkripsi.

Langkah selanjutnya merupakan proses enkripsi tahap pertama dengan menggunakan algoritma *Beaufort*. Hasil dari proses enkripsi ini berupa *cipher* teks A, yang akan mengalami

proses enkripsi lagi dengan menggunakan algoritma *Caesar*.

Rumus yang digunakan dalam melakukan enkripsi pada algoritma *Beaufort* dapat dilihat pada Persamaan 1, sedangkan, untuk proses dekripsi dapat dilihat pada Persamaan 2 [14].

$$Cc = (k\text{-}Pc) \bmod 256 \quad (1)$$

$$Pc = (k\text{-}Cc) \bmod 256 \quad (2)$$

Persamaan 1, menunjukkan rumus dalam melakukan proses enkripsi algoritma *Beaufort*. Dengan ketentuan ”Cc” merupakan *ciphertext* yang akan terbentuk, ”k” merupakan kunci dan ”Pc” merupakan karakter *plaintext*.

Pada algoritma ini, kunci yang digunakan harus memiliki panjang yang sama dengan *plaintext* [13]. Artinya, setiap karakter *plaintext* memiliki pasangan kunci masing-masing. Sebagai contoh, *plaintext* ”UINJKT” terdiri atas 6 karekter, sehingga jumlah kunci yang akan digunakan dalam proses enkripsi harus sebanyak 6 karakter juga. Misalnya, ”123456” atau ”asd123”. Sebagai contoh sederhana penerapan Persamaan 1 dapat dilihat pada Tabel 2.

Tabel 2. Enkripsi pada Beaufort

Pc	k	(k-Pc) mod 256	Hasil	Cc
J	x	(120-74) mod 256	46	.
K	y	(121-75) mod 256	46	.
T	z	(122-84) mod 256	38	&

Pada Tabel 2 terlihat bahwa *plaintext* yang akan dienkripsi adalah karakter ”JKT”. Sedangkan, kunci yang digunakan adalah ”xyz”. Langkah selanjutnya adalah melakukan konversi karakter *plaintext* untuk mendapatkan nilai ASCII desimal. Sebagai contoh, karakter ”J” yang jika konversi akan mendapatkan nilai ”74”. Hal ini juga berlaku untuk melakukan konversi terhadap kunci yang digunakan. Hasil akhir dari enkripsi dapat terlihat pada kolom ”Hasil” dan ”Cc”. Berdasarkan hasil tersebut terlihat bahwa karakter ”JKT” setelah dienkripsi akan menghasilkan karakter ”..&”.

Kelemahan algoritma *Beaufort* seperti yang terdapat pada Persamaan 1 dan Persamaan 2 adalah penggunaan semua kode ASCII yaitu sebanyak 256, sedangkan, untuk karakter dengan kode ASCII 0 (char: *null*) hingga kode ASCII 31 (char: *unit separator*) tidak dapat ditampilkan dilayar komputer. Salah satu solusi untuk mengatasi kelemahan ini adalah dengan membuat /menentukan daftar karakter yang dapat digunakan dalam melakukan enkripsi.

Berdasarkan pada Gambar 2, terlihat bahwa algoritma *Caesar* digunakan untuk melakukan enkripsi tahap kedua. Hasil dari enkripsi tahap kedua ini merupakan *cipher* teks B. *Cipher* teks B inilah yang merupakan hasil akhir dari usulan konsep enkripsi pada penelitian ini.

Secara umum, persamaan yang digunakan dalam algortima *Caesar cipher* dapat dilihat pada Persamaan 3 dan Persamaan 4 [9][17][18].

$$C = E(k, p) = (p + k) \bmod 26 \quad (3)$$

$$p = D(k, C) = (C - k) \bmod 26 \quad (4)$$

Persamaan 3 merupakan persamaan untuk melakukan enkripsi terhadap *plaintext*. Dengan ketentuan, huruf ”p” merupakan *plaintext*, ”k” merupakan kunci untuk melakukan enkripsi. Melalui persamaan ini akan didapatkan hasil enkripsi dari karakter yang ingin diamankan. Sebagai contoh penerapan Persamaan 3 tersebut dapat dilihat pada Tabel 3.

Pada Tabel 3, dapat terlihat bahwa dalam melakukan enkripsi menggunakan *caesar cipher* karakter *plaintext* (p’) akan ditransformasi terlebih dahulu ke dalam nilai berupa angka (p). Dengan ketentuan huruf ”A” memiliki nilai angka ”1”, ”B” nilai angka ”2”, hingga ”Z” memiliki nilai angka ”26”. Langkah berikutnya adalah menentukan kunci (k), yang pada Tabel 3 menggunakan k=2.

Tabel 3. Enkripsi pada Caesar

p’	p	k	(p+k) mod 26	C’	C
U	21	2	(21+2) mod 26	23	W
I	9	2	(9+2) mod 26	11	K
N	14	2	(14+2) mod 26	16	P

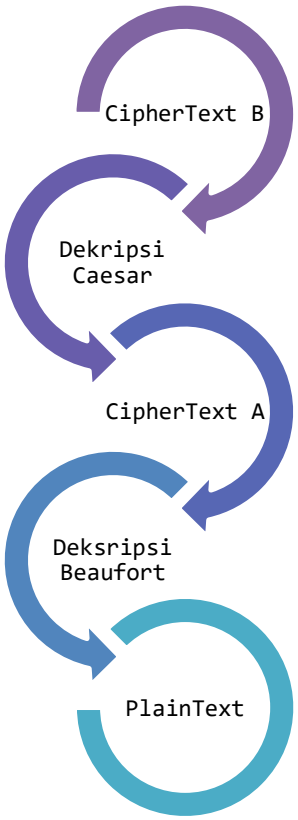
Langkah selanjutnya memasukkan tiap-tiap komponen (p dan k) ke dalam Persamaan 3. Sehingga akan dihasilkan nilai angka untuk *ciphertext* (C’). Langkah terakhir adalah mengubah kembali nilai angka *ciphertext* (C’) menjadi nilai huruf *ciphertext* (C).

Dikarenakan jumlah karakter yang terdapat pada Tabel 1 sebanyak 94 karakter, maka untuk Persamaan 1 dan 2 dilakukan penyesuaian, sehingga pada enkripsi dan dekripsi *Beaufort* tidak lagi menggunakan *“mod 256”* namun menggunakan *“mod 94”*. Begitu juga untuk melakukan enkripsi dan dekripsi pada algoritma *Caesar* yang tidak lagi

menggunakan *"mod 26"* tetapi menggunakan *"mod 94"*.

2.3. Rancangan Proses Dekripsi

Usulan konsep untuk melakukan proses dekripsi terhadap data teks pada penelitian ini dapat dilihat pada Gambar 3.



Gambar 3. Usulan proses dekripsi

Hampir sama dengan usulan proses enkripsi, pada Gambar 3 juga terlihat bahwa terdapat 5 tahapan dalam melakukan proses dekripsi dengan memanfaatkan algoritma *Beaufort* dan *Caesar*. Langkah pertama diawali dengan menentukan *cipher* teks atau data berupa teks yang telah di acak melalui proses enkripsi.

Langkah selanjutnya merupakan proses dekripsi tahap pertama dengan menggunakan algoritma *Caesar*. Hasil dari proses enkripsi ini berupa plain teks A, yang mengalami proses enkripsi lagi dengan menggunakan algoritma *Beaufort*. Jadi, untuk proses dekripsi merupakan kebalikan dari tahapan proses enkripsi yang diusulkan. Berdasarkan pada Gambar 3, terlihat bahwa algoritma *Beaufort* digunakan untuk melakukan dekripsi tahap kedua. Hasil dari dekripsi tahap kedua ini merupakan plain teks B.

Rumus dalam melakukan proses dekripsi algoritma *Beaufort* dapat dilihat pada Persamaan 2. Dengan ketentuan "Pc" merupakan *plaintext* yang akan terbentuk, "k" merupakan kunci dan "Cc" merupakan karakter *cipher* yang akan didekripsikan menjadi bentuk semula/*plaintext*. Sebagai contoh penerapan Persamaan 2 dapat dilihat pada Tabel 4.

Tabel 4. Dekripsi pada Beaufort

Cc	k	(k-Cc) mod 256	Hasil	Pc
.	x	(120-46) mod 256	74	J
.	y	(121-46) mod 256	75	K
&	z	(122-38) mod 256	84	T

Pada Tabel 4 terlihat bahwa *ciphertext* yang akan didekripsi adalah karakter "..&", sedangkan, kunci yang digunakan adalah "xyz". Langkah selanjutnya adalah melakukan konversi karakter *ciphertext* untuk mendapatkan nilai ASCII desimal. Sebagai contoh, karakter "." yang jika konversi akan mendapatkan nilai "46". Hal ini juga berlaku untuk melakukan konversi terhadap kunci yang digunakan. Hasil akhir dari enkripsi dapat terlihat pada kolom "Hasil" dan "Pc". Berdasarkan hasil tersebut terlihat bahwa karakter "..&" setelah dienkripsi akan menghasilkan karakter yang sama seperti sebelum dienkripsi yaitu "JKT".

Sedangkan, persamaan untuk melakukan dekripsi terhadap *ciphertext* menggunakan *caesar cipher* dapat dilihat pada Persamaan 4. Sebagai contoh penerapan Persamaan 4 tersebut dapat dilihat pada Tabel 5.

Tabel 5. Dekripsi pada Caesar

c'	c	k	(C-k) mod 26	p'	p
W	23	2	(23-2) mod 26	21	U
K	11	2	(11-2) mod 26	9	I
P	16	2	(16-2) mod 26	14	N

Pada Tabel 5, terlihat bahwa karakter *ciphertext* yang akan didekripsi adalah deretan karakter "WKP". Sedangkan, kunci yang digunakan adalah k=2. Langkah selanjutnya yang harus dilakukan adalah melakukan konversi terhadap karakter *ciphertext* yang akan didekripsi. Sebagai contoh karakter "W" yang setelah dikonversi menghasilkan nilai 23. Hasil akhir dari proses dekripsi dapat dilihat pada kolom "p" dalam Tabel 5. Terlihat bahwa karakter "WKP" yang telah didekripsi akan kembali ke bentuk awal (*plaintext*) yaitu karakter "UIN".

III. HASIL DAN PEMBAHASAN

Pada bagian Hasil dan Pembahasan dibagi ke dalam beberapa sub-pembahasan, yaitu implementasi kasus dan diskusi terkait dengan penerapan konsep yang diusulkan terhadap sebuah kasus.

3.1. Kasus

Pada bagian ini dijelaskan mengenai penerapan usulan proses enkripsi dan dekripsi menggunakan perpaduan *Beaufort* dan *Caesar cipher* terhadap sebuah contoh kasus.

Berdasarkan Gambar 2, tahapan pertama dalam proses enkripsi yang diusulkan adalah menentukan plain teks yang dienkripsi. Plain teks yang dijadikan contoh kasus pada penelitian ini adalah *"INFORMATIKA UINJKT"*.

Tahapan selanjutnya adalah melakukan proses enkripsi dengan menggunakan algoritma *Beaufort* seperti yang terdapat pada Persamaan 1. Adapun kunci yang digunakan dalam proses enkripsi tahap ini adalah *"stmikppkiatarakan"*. Hasil proses enkripsi ini dapat dilihat pada Tabel 6.

Tabel 6. Hasil enkripsi tahap pertama

Plain (Pc)		*Key* (k)		(k-Pc) mod 94	Hasil	Cc
I	18	s	54	(54-18) mod 94	36	Z
N	23	t	55	(55-23) mod 94	32	W
F	15	m	48	(48-15) mod 94	33	X
O	24	i	44	(44-24) mod 94	20	K
R	27	k	46	(46-27) mod 94	19	J
M	22	p	51	(51-22) mod 94	29	T
A	10	p	51	(51-10) mod 94	41	f
T	29	k	46	(46-29) mod 94	17	H
I	18	i	44	(44-18) mod 94	26	Q
K	20	a	36	(36-20) mod 94	16	G
A	10	t	55	(55-10) mod 94	45	j
U	30	a	36	(36-30) mod 94	6	7
I	18	r	53	(53-18) mod 94	35	Z
N	23	a	36	(36-23) mod 94	13	D
J	19	k	46	(46-19) mod 94	27	R
K	20	a	36	(36-20) mod 94	16	G
T	29	n	49	(49-29) mod 94	20	K

Pada Tabel 6, terlihat bahwa karakter *plaintext* yang akan dienkripsi terlebih dahulu dikonversi untuk mendapatkan nilai/indeks dari masing-masing karakter tersebut. Proses konversi ini dilakukan dengan mengacu pada Tabel 1. Sebagai contoh, karakter "I" yang memiliki nilai hasil konversi sebesar "18". Selain *plaintext*, kunci yang digunakan untuk melakukan proses enkripsi juga melalui proses konversi.

Setelah melalui proses konversi dilakukan proses enkripsi seperti yang dapat dilihat pada kolom *"(k-Pc) mod 94"* yang terdapat pada Tabel 6. Hasil dari proses enkripsi tahap pertama ini dapat dilihat pada kolom "Cc". Berdasarkan kolom "Cc" tersebut, hasil enkripsi tahap pertama adalah *"ZWXKJTfH QGj7ZDRGK"*.

Hasil enkripsi tahap pertama ini selanjutnya dijadikan sebagai *plaintext* untuk melakukan enkripsi tahap kedua. Hasil enkripsi tahap kedua dapat dilihat pada Tabel 7.

Tabel 7. Hasil enkripsi tahap kedua

Cc	Nilai (p)	*Key* (k)	(p+k) mod 94	Hasil	C
Z	36	30	(36+30) mod 94	66	$
W	32	30	(32+30) mod 94	62	~
X	33	30	(32+30) mod 94	63	!
K	20	30	(20+30) mod 94	50	o
J	19	30	(19+30) mod 94	49	n
T	29	30	(29+30) mod 94	59	x
f	41	30	(41+30) mod 94	71	(
H	17	30	(17+30) mod 94	47	l
Q	26	30	(26+30) mod 94	56	u
G	16	30	(16+30) mod 94	46	k
j	45	30	(45+30) mod 94	75	=
7	6	30	(6+30) mod 94	36	a
Z	35	30	(35+30) mod 94	65	#
D	13	30	(13+30) mod 94	43	h
R	27	30	(27+30) mod 94	57	v
G	16	30	(16+30) mod 94	46	k
K	20	30	(20+30) mod 94	50	o

Berdasarkan Tabel 7, kunci yang digunakan untuk melakukan proses enkripsi tahap kedua adalah "30". *Plaintext* pada enkripsi tahap kedua tersebut merupakan *ciphertext* dari hasil enkripsi tahap pertama. Hasil akhir dari proses enkripsi tahap kedua dapat dilihat pada kolom "C". Berdasarkan hasil akhir tersebut, dapat disimpulkan bahwa *plaintext* awal "INFORMATIKAUINJKT" setelah melalui tahapan proses enkripsi menjadi *"$~!onx(luk=a#hvko"*.

Untuk membuktikan bahwa kebenaran dari hasil enkripsi yang telah dilakukan, maka perlu diuji melalui proses dekripsi terhadap hasil enkripsi tersebut. Proses dekripsi dilakukan sebanyak 2 tahap. Hasil dekripsi tahap pertama dapat dilihat pada Tabel 8.

Tabel 8. Hasil dekripsi tahap pertama

C	Nilai (C)	*Key* (k)	(C-k) mod 94	Hasil	p
$	66	30	(66-30) mod 94	36	Z
~	62	30	(62-30) mod 94	32	W
!	63	30	(63-30) mod 94	33	X
o	50	30	(50-30) mod 94	20	K
n	49	30	(49-30) mod 94	19	J
x	59	30	(59-30) mod 94	29	T
(	71	30	(71-30) mod 94	41	f
l	47	30	(47-30) mod 94	17	H
u	56	30	(56-30) mod 94	26	Q
k	46	30	(46-30) mod 94	16	G
=	75	30	(75-30) mod 94	45	j
a	36	30	(36-30) mod 94	6	7
#	65	30	(65-30) mod 94	35	Z
h	43	30	(43-30) mod 94	13	D
v	57	30	(57-30) mod 94	27	R
k	46	30	(46-30) mod 94	16	G
o	50	30	(50-30) mod 94	20	K

Tahapan dekripsi tahap pertama dilakukan dengan menggunakan algoritma *Caesar cipher*. Pada Tabel 8, dapat dilihat bahwa kunci yang digunakan untuk melakukan proses dekripsi sama seperti proses enkripsi. Sebelum melalui proses enkripsi tahap pertama, terlebih dahulu dilakukan konversi terhadap karakter cipherteks. Secara keseluruhan, hasil akhir dari proses dekripsi tahap pertama tersebut dapat dilihat pada kolom "p" pada Tabel 8.

Selanjutnya, hasil enkripsi tahap pertama akan menjadi *ciphertext* pada proses dekripsi tahap kedua. Hasil dekripsi tahap kedua tersebut dapat dilihat pada Tabel 9.

Pada Tabel 9, dapat dilihat bahwa kunci yang digunakan untuk melakukan proses dekripsi tahap kedua sama seperti proses enkripsi tahap pertama yaitu *"stmikppkia tarakan"*. Sebelum melalui proses dekripsi, terlebih dahulu dilakukan konversi terhadap karakter yang akan di dekripsi beserta kunci yang akan digunakan. Proses konversi ini mengacu pada daftar karakter yang terdapat pada Tabel 1. Secara keseluruhan, hasil akhir dari proses dekripsi tahap kedua tersebut dapat dilihat pada kolom "Pc" di Tabel 9.

Tabel 9. Hasil dekripsi tahap kedua

Cipher (Cc)		*Key* (k)		(k-Cc) mod 94	Hasil	Pc
Z	36	s	54	(54-36) mod 94	18	I
W	32	t	55	(55-32) mod 94	23	N
X	33	m	48	(48-33) mod 94	15	F
K	20	i	44	(44-20) mod 94	24	O
J	19	k	46	(46-19) mod 94	27	R
T	29	p	51	(51-29) mod 94	22	M
f	41	p	51	(51-41) mod 94	10	A
H	17	k	46	(46-17) mod 94	29	T
Q	26	i	44	(44-26) mod 94	18	I
G	16	a	36	(36-16) mod 94	20	K
j	45	t	55	(55-45) mod 94	10	A
7	6	a	36	(36-6) mod 94	30	U
Z	35	r	53	(53-35) mod 94	18	I
D	13	a	36	(36-13) mod 94	23	N
R	27	k	46	(46-27) mod 94	19	J
G	16	a	36	(36-16) mod 94	20	K
K	20	n	49	(49-20) mod 94	29	T

Berdasarkan pada Tabel 9, dapat dilihat bahwa hasil akhir dari proses dekripsi tahap kedua adalah "INFORMATIKAUINJKT". Hasil akhir ini sama seperti *plaintext* awal sebelum dilakukan proses enkripsi.

3.2. Diskusi

Penerapan teknik pengamanan data telah dilakukan terhadap data teks *"INFORMATIKA UINJKT"*. Sedangkan, kunci yang digunakan untuk tiap-tiap tahapan proses enkripsi berbeda. Untuk tahapan pertama kunci yang digunakan adalah *"stmikppkiatarakan"*, tahap kedua pada *Caesar cipher* menggunakan kunci *"30"*. Secara keseluruhan, rangkuman hasil implementasi enkripsi yang telah dilakukan dapat dilihat pada Tabel 10.

Dengan mengacu pada Tabel 10, dapat terlihat bahwa proses enkripsi yang telah dilakukan sebanyak dua tahapan berhasil mengacak karakter awal *plaintext* menjadi lebih sulit untuk dikenal. Sebagai contoh karakter "I", yang dienkripsi menjadi simbol "$", begitu juga dengan karakter "N", setelah dienkripsi berubah menjadi simbol "~". Proses dekripsi dilakukan terhadap hasil enkripsi yang terdapat pada Tabel 10. Kunci yang digunakan dalam proses

dekripsi sama seperti kunci yang digunakan pada proses enkripsi.

Tabel 10. Rekapitulasi hasil enkripsi

Plaintext	Hasil Enkripsi Tahap Pertama	Hasil Enkripsi Tahap Kedua
I	Z	$
N	W	~
F	X	!
O	K	o
R	J	n
M	T	x
A	f	(
T	H	l
I	Q	u
K	G	k
A	j	=
U	7	a
I	Z	#
N	D	h
J	R	v
K	G	k
T	K	o

Berdasarkan implementasi yang telah dilakukan dan dibahas secara rinci di Bagian Pembahasan, dapat disimpulkan bahwa teknik pengamanan data yang diajukan telah berhasil diterapkan. Hal ini dibuktikan dengan kesesuaian antara hasil dekripsi yang dilakukan dengan *plaintext* awal yang telah ditentukan.

Sebagai bahan pertimbangan untuk menguji keberhasilan konsep pengamanan data yang telah diusulkan, juga dilakukan beberapa pengujian dengan menggunakan beberapa kata lainnya. Hasil pengujian dapat dilihat pada Tabel 11.

Tabel 11. Uji coba

Plaintext	Hasil Enkripsi	Hasil Dekripsi	Sukses
STMIK	w@]>'	STMIK	✔
PPKIA	>1EJ9	PPKIA	✔
Tarakan	""=2$"[	Tarakan	✔
Ujicoba	:+; ~:`	Ujicoba	✔
Hybrid	2x/|^]	Hybrid	✔

Keterangan: ✔ (Ya); - (Tidak)

Berdasarkan Tabel 11, dapat terlihat bahwa proses enkripsi yang dilakukan terhadap beberapa contoh *plaintext (menggunakan kunci beaufort = BIRU, kunci caesar = 7)* mampu menghasilkan *ciphertext* yang sulit untuk dikenali, yang dapat dilihat pada kolom hasil enkripsi di Tabel 11 tersebut.

Ketika dilakukan proses dekripsi terhadap beberapa cipherteks hasil enkripsi tersebut, maka hasilnya akan sama seperti *plaintext* awal sebelum dilakukan proses enkripsi. Sebagai contoh, kata *"Hybrid"* ketika dienkripsi berubah menjadi "2x/|^]" dan ketika cipherteks "2x/|^]" tersebut didekripsi, maka hasilnya kembali ke bentuk semula yaitu *plaintext "Hybrid".*

IV. PENUTUP

Berdasarkan penelitian yang telah dilakukan, maka dapat disimpulkan bahwa konsep pengamanan data yang diusulkan melalui perpaduan algoritma *Beaufort* dan *Caesar cipher* berhasil dan dapat digunakan dalam melakukan pengamanan terhadap data teks. Hal ini terlihat dari enkripsi dengan menggunakan perpaduan dua algoritma tersebut. Dengan menggunakan perpaduan dua algoritma ini, maka *ciphertext* menjadi lebih sulit untuk dipecahkan. Hal ini terjadi karena terdapat dua tahapan proses enkripsi, dengan menggunakan dua jenis kunci yang berbeda untuk tiap-tiap tahapan. Saran untuk tahapan penelitian selanjutnya dapat dilakukan dengan menerapkan konsep yang telah dirancang dalam mengamankan data berupa audio, gambar, maupun video. Selain itu dapat juga dilakukan dengan mengoptimalkan jumlah daftar karakter yang digunakan.

DAFTAR PUSTAKA

[1] K. Hedström, E. Kolkowska, F. Karlsson, and J. P. Allen, "Value conflicts for information security management," J. Strateg. Inf. Syst., vol. 20, no. 4, 2011, pp. 373–384.

[2] I. Lopes and P. Oliveira, "New Perspectives in Information Systems and Technologies, Volume 1," vol. 275, 2014.

[3] M. Fadlan and Hadriansa, "Rekayasa aplikasi kriptografi dengan penerapan kombinasi algoritma knapsack merkle hellman dan affine cipher," J. Teknol. Inf. dan Ilmu Komput., vol. 4, no. 4, 2017, pp. 268–274.

[4] Zaeniah and B. E. Purnama, "An Analysis of Encryption and Decryption

Application by using One Time Pad Algorithm," Int. J. Adv. Comput. Sci. Appl., vol. 6, no. 9, 2015, pp. 292–297.

[5] S. Sriadhi, R. Robbi, and A. S. Ahmar, "RC4 Algorithm Visualization for Cryptography Education," 2nd Int. Conf. Stat. Math. Teaching, Res., pp. 1–10, 2018.

[6] K. Goyal and S. Kinger, "Modified Caesar Cipher for Better Security Enhancement," Int. J. Comput. Appl., vol. 73, no. 3, 2013, pp. 26–31.

[7] A. Indriani and Sinawati, "Analisa Pengamanan Teks Menggunakan Teknik Character Cipher Dan Block Cipher," in Prosiding SNST ke-9, 2018, pp. 143–148.

[8] B. Oktaviana and A. P. U. Siahaan, "Three-Pass Protocol Implementation in Caesar Cipher Classic Cryptography," IOSR J. Comput. Eng., vol. 18, no. 4, 2016, pp. 26–29.

[9] B. Purnama and H. Rohayani, "A New Modified Caesar Cipher Cryptography Method With Legible Ciphertext From A Message To Be Encrypted," Procedia - Procedia Comput. Sci., vol. 59, no. Iccsci, 2015, pp. 195–204.

[10] A. Pradipta, "Implementasi Metode Caesar Chiper Alphabet Majemuk Dalam Kriptografi Untuk Pengamanan Informasi," Indones. J. Netw. Secur., vol. 5, no. 3, 2016, pp. 3–6.

[11] M. M. Amin, "Implementasi Kriptografi Klasik Pada Komunikasi Berbasis Teks," J. Pseudocode, vol. III, no. September, 2016, pp. 129–136.

[12] R. Latifah, S. N. Ambo, and S. I. Kurnia, "Modifikasi Algoritma Caesar Chiper Dan Rail Fence Untuk Peningkatan Keamanan Teks Alfanumerik Dan Karakter Khusus," in Seminar Nasional Sains dan Teknologi 2017, no. November, 2017, pp. 1–7.

[13] M. Diana and T. Zebua, "Optimalisasi Beaufort Cipher Menggunakan Pembangkit Kunci RC4 Dalam Penyandian SMS," J. Sains Komput. Inform., vol. 2, no. 1, 2018, pp. 12–22.

[14] D. R. S. M. Ignatius, C. Jatmoko, E. H. Rachmawanto, and C. A. Sari, "Kombinasi Cipher Subtitusi (Beaufort Dan Vigenere)," in Prosiding SENDI_U 2018, 2018, pp. 978–979.

[15] E. H. Rachmawanto, D. R. Ignatius, M. Setiadi, C. A. Sari, and N. Rijati, "Imperceptible and secure image watermarking using DCT and random spread technique," Telkomnika, vol. 17, no. 4, 2019, pp. 1750–1757,

[16] K. Alallayah, M. Amin, W. A. El-wahed, and A. Alhamami, "Attack and Construction of Simulator for Some of Cipher Systems Using Neuro-Identifier," Int. Arab J. Inf. Technol., vol. 7, no. 4, 2010, pp. 365–372,

[17] D. Nofriansyah and R. Rahim, "Combination Of Pixel Value Differencing Algorithm With Caesar Algorithm For Steganography," Int. J. Res. Sci. Eng., vol. 2, no. 6, 2016, pp. 153–159,

[18] I. Gunawan, "Kombinasi algoritma Caesar cipher dan algoritma rsa untuk pengamanan file dokumen dan pesan teks," InfoTekJar (Jurnal Nas. Inform. dan Teknol. Jaringan), vol. 2, no. 2, 2018, pp. 124–129,