UJM 11(1) 2022
UNNES Journal of Mathematics
http://journal.unnes.ac.id/sju/index.php/ujm

Desain Algoritma Kriptografi Klasik dengan Konsep Koset Grup dalam Teori Aljabar

Nia Yulianti✉, **Angga Wijaya**

Politeknik Siber dan Sandi Negara, Indonesia
Jl. Haji Usa, Ciseeng, Bogor, 16120
Institut Teknologi Sumatera, Indonesia
Jl. Terusan Ryacudu, Way Huwi, Lampung Selatan, 35365

Info Artikel

Sejarah Artikel:
Diterima Februari 2022
Disetujui Mei 2022
Dipublikasikan Mei 2022

Keywords:
Kriptografi klasik, koset grup, pemetaan one to many

Abstrak

Penelitian ini mengaji konsep koset grup apabila diterapkan pada kriptografi dalam mengodekan karakter yang ada pada *plaintext* ke beberapa kemungkinan *ciphertext* yang berada di koset yang sama. Kelebihan menggunakan koset untuk partisi yaitu dapat mengetahui apakah dua anggota dalam suatu grup terletak pada suatu bagian partisi yang sama atau tidak. Penelitian ini menggunakan metode kuantitatif dengan dua pendekatan, yaitu studi literatur dan menyajikan contoh enkripsi deskripsi dilengkapi dengan simulasi program bahasa C++. Perbandingan karakter *plaintext* dan *ciphertext* pada pengujian recovery menggunakan percobaan satu sampel dengan *plaintext* dari teks dalam bahasa Inggris berukuran 48156 huruf dengan kunci *k=3* diperoleh kesamaan 100 persen atau identik. Dengan demikian algoritma enkripsi dan dekripsi dapat dikatakan berhasil memenuhi syarat recovery *plaintext.* Berdasarkan penelitian dan pengujian yang telah dilakukan, dapat disimpulkan bahwa hasil pengujian analisis frekuensi pada enkripsi kriptografi klasik dengan penerapan konsep koset grup tidak terlihat hubungan antara sebaran kemunculan huruf di *plaintext* dengan kemunculan huruf di *ciphertext,* sehingga kriptanalis tidak dapat menerka aturan enkripsi yang digunakan.

Abstract

This study examines the concept of group coset when applied to cryptography in encoding characters that exist in plaintext into several possible ciphertexts that are in the same coset. The advantage of using a coset for partitioning is that it can tell whether two members of a group are in the same partition or not. This research uses quantitative methods with two approaches, namely literature study and presenting examples of description encryption equipped with a simulation of the C++ language program. Comparison of plaintext and ciphertext characters in recovery testing using one sample experiment with plaintext of 48156 letters in English with key $k = 3$ *obtained 100 percent similarity or identical. Thus the encryption and decryption algorithms can be said to be successful in fulfilling the plaintext recovery requirements. Based on the research and testing that has been done, it can be concluded that the results of the frequency analysis test on classical cryptographic encryption with the application of the group coset concept do not show a relationship between the distribution of letters in plaintext and letters in ciphertext, so cryptanalysts cannot guess the encryption rules used.*

How to cite:

Yulianti, N. & Wijaya, A. (2022). Desain Algoritma Kriptografi Klasik dengan Konsep Koset Grup dalam teori Aljabar. *UNNES Journal of Mathematics*, 11(1), 16-26.

✉Alamat korespondensi:
E-mail: nia.yulianti@poltekssn.ac.id

p-ISSN 2252-6943
e-ISSN 2460-5859

PENDAHULUAN

Pada komunikasi digital dan pertukaran data di internet, jejak digital ataupun data yang dikirimkan dapat dengan mudah didapatkan, sehingga data yang dikirimkan melalui internet perlu diamankan. Dalam pengamanan data tersebut, kriptografi merupakan salah satu instrumen penting (Gencoglu, 2019).

Kriptografi adalah ilmu sekaligus seni untuk menjaga keamanan pesan (*message*) (Scheneier, 1996). Pada tahun yang sama, menurut Menezes Kriptografi merupakan studi tentang teknik Matematika yang berkaitan dengan aspek keamanan informasi seperti kerahasian, integritas data, otentikasi entitas, dan otentikasi asal data. Soemarkidjo juga menyebutkan pada tahun 2007 bahwa Kriptografi merupakan ilmu untuk menjaga kerahasiaan informasi dengan metode dan teknik Matematika yang mencakup kerahasiaan, integritas data, otentikasi entitas, dan otentikasi asal data. Sesuai dengan prinsip dan asumsi yang dijelaskan oleh Kerckoff pada suatu sistem Kriptografi diharuskan bersifat aman walaupun informasi mengenai sistem diketahui terkecuali untuk nilai kuncinya (Henk, 2011).

Untuk mempersulit kriptanalis dalam memecahkan sandi maka didisain aturan enkripsi yang *one to many*, sehingga satu karakter yang ada di pesan asli atau teks yang dapat dibaca (*plaintext*) dapat dienkripsi menjadi lebih dari satu kemungkinan karakter yang tersandikan (*ciphertext*) (Munir, 2019). Sebaliknya, *chipertext* dapat dikembalikan menjadi *plaintext* disebut dengan proses dekripsi, seperti yang ditulis Angga, 2020.

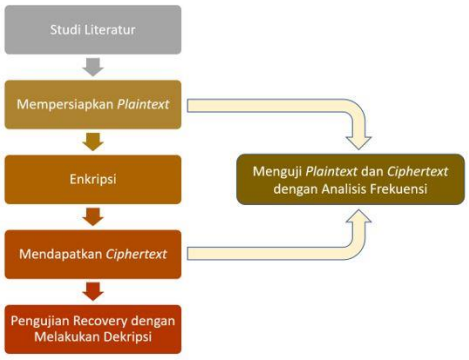
Salah satu cabang ilmu yang ada di Matematika adalah Aljabar. Aljabar merupakan cabang ilmu yang sangat fundamental. Salah satu konsep yang ada dalam Aljabar yaitu Teori Grup. Suatu himpunan tak kosong G disebut sebagai grup jika pada G didefinisikan operasi $*$ sehingga G tertutup terhadap operasi $*$, berlaku sifat asosiatif, terdapat identitas dari setiap elemen di G. Selanjutnya jika diberikan himpunan tak kosong H dari grup $(G,*)$, himpunan H disebut subgrup dari G jika H merupakan grup terhadap operasi yang sama pada G, yaitu operasi $*$ (Durbin, 2009).

Suatu Grup dapat dipartisi menjadi beberapa bagian. Salah satu cara mempartisinya adalah dengan konsep koset suatu grup. Didefinisikan $aH = \{ah | h \in H\}, Ha = \{ha | h \in H\}$ dan $aHa^{-1} = \{aha^{-1} | h \in H\}$. Jika H adalah suatu suatu subgrup dari G, maka himpunan aH disebut koset kiri dari H di G yang memuat a, sementara Ha disebut koset kanan dari H di G yang memuat a (Herstein, 1996).

Pada penelitian ini dikaji terkait konsep koset grup apabila diterapkan pada kriptografi, khususnya ingin mengkodekan karakter yang ada pada *plaintext* ke beberapa kemungkinan *ciphertext* yang berada di koset yang sama. Hal ini bertujuan supaya dapat membentuk algoritma kriptografi yang pemetaannya *one to many*. Kelebihan menggunakan koset untuk partisi yaitu dapat mengetahui apakah dua anggota dalam suatu grup terletak pada suatu bagian partisi yang sama atau tidak.

METODE

Metode penelitian yang digunakan pada penelitian ini adalah metode kuantitatif dengan menggunakan dua pendekatan, yaitu studi literatur dan menyajikan contoh enkripsi deskripsi dilengkapi dengan simulasi menggunakan program bahasa C++. Langkah yang dilakukan dimulai dengan memberikan definisi formal terkait kriptografi, grup, sub grup, koset dan sifat-sifat yang berlaku, selanjutnya mendesain algoritma enkripsi dan menentukan *plaintext* yang akan diuji coba, menerapkan algoritma, memperoleh *ciphertext* hasil enkripsi dan terakhir melakukan pengujian analisis frekuensi karakter yang ada di *chipertext*. Adapun skema alur penelitian dapat dilihat pada Gambar 1.



Gambar 1. Alur Penelitian

HASIL DAN PEMBAHASAN

Landasan Teori

Sebelum komputer ditemukan, kriptografi dilakukan dengan menggunakan pensil dan kertas. Algoritma kriptografi *(cipher)* yang digunakan saat itu, dinamakan algoritma klasik yang berbasiskan karakter dimana proses

persandian dilakukan pada setiap karakter pesan. Semua algoritma klasik termasuk ke dalam sistem kriptografi simetris dan digunakan jauh sebelum sistem kriptografi kunci publik ditemukan (Munir, 2019). Menurut Munir, terdapat tiga alasan untuk mempelajari algoritma klasik, yaitu:

1. Untuk memberikan pemahaman konsep dasar kriptografi.
2. Sebagai dasar dari algoritma kriptografi modern, dan
3. Agar dapat memahami potensi-potensi kelemahan sistem *cipher.*

Secara umum, kriptografi klasik dibagi menjadi dua kategori yaitu cipher substitusi (mengganti setiap huruf atau kelompok huruf dengan sebuah huruf atau kelompok huruf lain) dan cipher transposisi (mengubah susunan pada pesan). Salah satu contoh algoritma kriptografi klasik adalah algoritma caesar cipher. Algorima ini melakukan pergeseran ke kanan sejumlah nilai $\boldsymbol{k}$ tertentu sebagai kuncinya. Apabila pergeserannya melebihi huruf $\boldsymbol{Z}$ maka akan Kembali lagi ke huruf $\boldsymbol{A}$ (untuk 26 huruf alfabet) seperti yang ditulis M. M. Amin, 2017 dan T. Limbong, 2015.

Kriptografi pada dasarnya dimanfaatkan untuk menjaga kerahasiaan pesan yang dikirimkan. Lebih jauh, Kriptografi juga digunakan untuk menyelesaikan masalah keamanan yang mencakup keabsahan pengirim (*user authentication*), keaslian pesan (*message authentication*), anti penyangkalan (*nonrepudiation*). Keabsahan pengirim berkaitan dengan keaslian pengirim, apakah pesan yang diterima betul-betul berasal dari pengirim yang sebenarnya.Keaslian pesan berkaitan dengan keutuhan pesan, apakah pesan yang diterima mengalami perubahan atau tidak. Anti-penyangkalan artinya pengirim tidak dapat menyangkal bahwa memang dialah yang mengirim pesan.

Sifat-sifat dasar dan karakteristik grup dibahas oleh Herstein (1996), Fraileigh (2000), Gallian (2017), Rotman (2003), dan Dummit (2004).

Definisi 1. (Fraleigh, 2000) *Suatu himpunan tak kosong* $\boldsymbol{G}$ *disebut sebagai grup jika pada* $\boldsymbol{G}$ *didefinisikan suatu operasi* $\star$ *sehingga:*

1. $\boldsymbol{a, b} \in \boldsymbol{G}$ *menyatakan bahwa* $\boldsymbol{a \star b} \in \boldsymbol{G}$ *(*$\boldsymbol{G}$ *tertutup terhadap* $\star$*);*
2. *Diberikan* $\boldsymbol{a, b, c} \in \boldsymbol{G}$*, maka* $\boldsymbol{a \star (b \star c) = (a \star b) \star c}$ *(sifat asosiatif);*
3. *Terdapat* $\boldsymbol{e} \in \boldsymbol{G}$ *sehingga* $\boldsymbol{a \star e = e \star a = a}$ *untuk seluruh* $\boldsymbol{a} \in \boldsymbol{G}$ *(*$\boldsymbol{e}$ *disebut unsur identitas dari* $\boldsymbol{G}$*);*
4. *Untuk setiap* $\boldsymbol{a} \in \boldsymbol{G}$ *terdapat suatu unsur* $\boldsymbol{b} \in \boldsymbol{G}$ *(yang bergantung kepada* $\boldsymbol{a}$*) sehingga* $\boldsymbol{a \star b = b \star a = e}$ *(unsur* $\boldsymbol{b}$ *disebut sebagai invers dari* $\boldsymbol{a}$ *dan sering dituliskan sebagai* $\boldsymbol{a^{-1}}$*).*

Keempat postulat tersebut sering disebut sebagai aksioma grup. Jika sebuah grup $(\boldsymbol{G},\star)$ operasinya juga memenuhi sifat komutatif, maka $\boldsymbol{G}$ disebut sebagai grup komutatif atau grup Abel.

Contoh 1. Misalkan $\mathbb{Z}$ adalah himpunan bilangan bulat dan $\star$ adalah operasi penjumpahan biasa, $+$ di $\mathbb{Z}$. Himpunan $\mathbb{Z}$ bersifat tertutup dan asosiatif terhadap $\star$ merupakan sifat dasar dari bilangan bulat. Apakah yang berfungsi sebagai unsur identidas $\boldsymbol{e}$, dari $\mathbb{Z}$ terhadap $\star$? Jelas, karena $\boldsymbol{a = a \star e = a + e}$, maka $\boldsymbol{e = 0}$ dan $\boldsymbol{0}$ merupakan unsur identitas terhadap penjumlahan. Bagaimana dengan $\boldsymbol{a^{-1}}$? Karena $\boldsymbol{e = 0 = a \star a^{-1} = a + a^{-1}}$, maka $\boldsymbol{a^{-1}}$ adalah $\boldsymbol{-a}$ dan jelas bahwa $\boldsymbol{a \star (-a) = a + (-a) = 0}$.

Contoh 2. $\mathbb{Z}_{\boldsymbol{n}} = \{\overline{\boldsymbol{0}}, \overline{\boldsymbol{1}}, \overline{\boldsymbol{2}}, \dots, \overline{\boldsymbol{n-1}}\}$, merupakan himpunan yang memuat koleksi semua kelas yang diperoleh dari relasi ekivalensi kongruen modulo $\boldsymbol{n}$. Apabila didefinisikan operasi penjumlahan, yaitu untuk setiap $\overline{\boldsymbol{a}}, \overline{\boldsymbol{b}} \in \mathbb{Z}_{\boldsymbol{n}}$, $\overline{\boldsymbol{a}} + \overline{\boldsymbol{b}} = \overline{\boldsymbol{a+b}}$, maka $(\mathbb{Z}_{\boldsymbol{n}}, +)$ adalah grup komutatif dengan $\overline{\boldsymbol{0}} \in \mathbb{Z}_{\boldsymbol{n}}$ sebagai elemen identitasnya.

Definisi 2. (Gallian, 2017) *Misalkan* $(\boldsymbol{G},\star)$ *suatu grup. Suatu himpunan bagian* $\boldsymbol{H} \subseteq \boldsymbol{G}$ *yang tak kosong dikatakan merupakan subgrup jika* $(\boldsymbol{H},\star)$ *juga merupakan grup. Jika* $\boldsymbol{H}$ *merupakan subgroup dari* $\boldsymbol{G}$*, dituliskan dengan* $\boldsymbol{H} \leqq \boldsymbol{G}$*.*

Setiap grup $\boldsymbol{G}$ secara otomatis mempunyai dua subgrup, yaitu $\boldsymbol{G}$ itu sendiri dan subgrup yang terdiri atas suatu untus identitas $\boldsymbol{e}$. Kedua subgrup ini disebut subgrup trivial.

Contoh 3. Misalkan $\boldsymbol{G}$ adalah grup $\mathbb{Z}$ dari bilangan bulat terhadap operasi $+$ dan misalkan $\boldsymbol{H}$ adalah himpunan bilangan bulat genap. Klaim bahwa $\boldsymbol{H}$ adalah subgrup dari $\mathbb{Z}$. Mengapa? Karena jika diberikan $\boldsymbol{a, b} \in \boldsymbol{H}$, mengakibatkan $\boldsymbol{a + b} \in \boldsymbol{H}$. Dengan kata lain, jika $\boldsymbol{a, b}$ adalah bilangan bulat genap, maka $\boldsymbol{a + b}$ bilangan bulat genap juga. Sehingga $\boldsymbol{H}$ tertutup terhadap operasi $+$. Selanjutnya, dapat

dibuktikan terdapat invers. Oleh karena operasi di $\mathbb{Z}$ adalah $+$, maka invers dari $\boldsymbol{a} \in \mathbb{Z}$ terhadap operasi tersebut adalah $-\boldsymbol{a}$. Jika $\boldsymbol{a} \in \boldsymbol{H}$, karena $\boldsymbol{a}$ bilangan genap, maka $-\boldsymbol{a}$ juga bilangan genap, sehingga $-\boldsymbol{a} \in \boldsymbol{H}$. Jadi, $\boldsymbol{H}$ adalah subgrup dari $\mathbb{Z}$ terhadap operasi $+$.

Definisi 3. (Rotman, 2003) *Misalkan $\boldsymbol{H}$ adalah subgrup dari $\boldsymbol{G}$. Untuk setiap $\boldsymbol{a} \in \boldsymbol{G}$ berturut-turut notasikan $\boldsymbol{aH}$ dan $\boldsymbol{Ha}$ himpunan $\boldsymbol{aH} = \{\boldsymbol{ah} : \boldsymbol{h} \in \boldsymbol{H}\}$ dan $\boldsymbol{Ha} = \{\boldsymbol{ha} : \boldsymbol{h} \in \boldsymbol{H}\}$. Kedua himpunan $\boldsymbol{aH}$ dan $\boldsymbol{Ha}$ berturut-turut disebut sebagai koset kiri dan koset kanan dari $\boldsymbol{H}$.*

Contoh 4. $3\mathbb{Z}$ adalah subgrup dari grup $(\mathbb{Z}, +)$. Koset-koset kiri yang terbentuk dari $3\mathbb{Z}$ yaitu $\boldsymbol{0} + \boldsymbol{3}\mathbb{Z}, \boldsymbol{1} + \boldsymbol{3}\mathbb{Z}$, dan $\boldsymbol{2} + \boldsymbol{3}\mathbb{Z}$ sehingga $\mathbb{Z}$ terbagi atas tiga kelas ekivalensi (koset kiri).

Teorema 1. (Dummit, 2004) *Misalkan $\boldsymbol{G}$ suatu grup berhingga dan $\boldsymbol{H}$ subgrup berhingga dan $\boldsymbol{H}$ subgrup dari $\boldsymbol{G}$. Maka $|\boldsymbol{H}|$ membagi $|\boldsymbol{G}|$.*

Bukti: Misalkan $\boldsymbol{a_1H}, \boldsymbol{a_2H}, \dots, \boldsymbol{a_kH}$ adalah semua koset kiri yang berbeda di $\boldsymbol{G/H}$. Koset-koset ini mempartisi $\boldsymbol{G}$, yaitu $\boldsymbol{G} = \cup_{\boldsymbol{j=1}}^{\boldsymbol{k}} \boldsymbol{a_jH}$ dan $\boldsymbol{a_iH} \cap \boldsymbol{a_jH} = \emptyset$ untuk $\boldsymbol{i} \neq \boldsymbol{j}$. Karena untuk setiap $\boldsymbol{a} \in \boldsymbol{G}$ berlaku $|\boldsymbol{aH}| = |\boldsymbol{H}|$ maka $|\boldsymbol{G}| = |\boldsymbol{a_1H}| + |\boldsymbol{a_2H}| + \cdots + |\boldsymbol{a_kH}| = |\boldsymbol{H}| + |\boldsymbol{H}| + \cdots + |\boldsymbol{H}| = \boldsymbol{k}|\boldsymbol{H}|$. Jadi, $|\boldsymbol{H}|$ membagi $|\boldsymbol{G}|$.

Rancangan Algoritma

Grup yang akan dipakai pada rancangan algoritma ini adalah $\mathbb{Z}_{\boldsymbol{26}}$, sedangkan subgrup yang dipilih adalah $\boldsymbol{26}.\mathbb{Z}_{\boldsymbol{26}}$.

Berikut adalah tahapan enkripsi dan dekripsi dari rancangan algoritma yang telah dibuat.

a. Tahapan Enkripsi

Alur proses enkripsi pesan adalah sebagai berikut:

1. Mempersiapkan *plaintext.*
2. Mengkodekan karakter *plaintext* ke dalam bilangan desimal.
3. Input kunci enkripsi.
4. Mendapatkan Grup dan Subgrup sehingga setiap karakter di *plaintext* terdapat pada koset yang ada di Grup dan Subgrup itu.
5. Mengganti setiap karakter pesan dengan sebarang karakter yang ada di koset yang sama.
6. mengubah nilai desimal ke karakter huruf dengan cara hitung ada berapa digit bilangan $\boldsymbol{26} \times \boldsymbol{k}$, jika bilangan desimal yang ada di plain teks jumlah digitnya kurang dari jumlah digit $\boldsymbol{26} \times \boldsymbol{k}$ maka di depan bilangan tersebut ditambahkan bilangan 0.
7. Jika digitnya 0 maka diubah jadi karakter $\boldsymbol{A}$. Jika digitnya bukan nol, gabungkan dengan bilangan setelahnya, Ketika nilai gabungan lebih dari 25 maka ambil bilangan pertama dan diubah ke karakter huruf sesuai aturan pemetaan. Jika nilai gabungannya ≤ 25 maka gabungannya yang diubah ke karakter huruf sesuai aturan pemetaan.

Contoh 5.

1. Pilih *plaintext* yaitu "NIAN".
2. Karakter *plaintext* dalam bilangan desimal adalah:

	N	I	A	N
Nilai	13	8	0	13

3. Pilih kunci $\boldsymbol{k} = \boldsymbol{7}$ sehingga dibentuk koset $\{\overline{\boldsymbol{0 \cdot 26}}, \overline{\boldsymbol{1 \cdot 26}}, \overline{\boldsymbol{2 \cdot 26}}, \dots, \overline{\boldsymbol{(7-1) \cdot 26}}\} = \{\overline{\boldsymbol{0}}, \overline{\boldsymbol{26}}, \overline{\boldsymbol{52}}, \dots, \overline{\boldsymbol{156}}\}$.
4. Mendapatkan Grup dan Subgrup sehingga setiap karakter di *plaintext* terdapat pada koset yang ada di Grup dan Subgrup itu.

	Koset
13	$\overline{\boldsymbol{13}} + \{\overline{\boldsymbol{0}}, \overline{\boldsymbol{26}}, \overline{\boldsymbol{52}}, \dots, \overline{\boldsymbol{156}}\}$, $\{\overline{\boldsymbol{13}}, \overline{\boldsymbol{39}}, \overline{\boldsymbol{65}}, \dots, \overline{\boldsymbol{169}}\}$
8	$\overline{\boldsymbol{8}} + \{\overline{\boldsymbol{0}}, \overline{\boldsymbol{26}}, \overline{\boldsymbol{52}}, \dots, \overline{\boldsymbol{156}}\}$, $\{\overline{\boldsymbol{13}}, \overline{\boldsymbol{39}}, \overline{\boldsymbol{65}}, \dots, \overline{\boldsymbol{169}}\}$
0	$\overline{\boldsymbol{0}} + \{\overline{\boldsymbol{0}}, \overline{\boldsymbol{26}}, \overline{\boldsymbol{52}}, \dots, \overline{\boldsymbol{156}}\}$, $\{\overline{\boldsymbol{0}}, \overline{\boldsymbol{26}}, \overline{\boldsymbol{52}}, \dots, \overline{\boldsymbol{156}}\}$
1	$\overline{\boldsymbol{13}} + \{\overline{\boldsymbol{0}}, \overline{\boldsymbol{26}}, \overline{\boldsymbol{52}}, \dots, \overline{\boldsymbol{156}}\}$, $\{\overline{\boldsymbol{13}}, \overline{\boldsymbol{39}}, \overline{\boldsymbol{65}}, \dots, \overline{\boldsymbol{169}}\}$

5. Setiap karakter pesan diganti dengan sebarang karakter yang ada di koset yang sama, diperoleh:

	Koset	
13	$\overline{\mathbf{13}} + \{\overline{\mathbf{0}}, \overline{\mathbf{26}}, \overline{\mathbf{52}}, \dots, \overline{\mathbf{156}}\}$, $\{\overline{\mathbf{13}}, \overline{\mathbf{39}}, \overline{\mathbf{65}}, \dots, \overline{\mathbf{169}}\}$	65
8	$\overline{\mathbf{8}} + \{\overline{\mathbf{0}}, \overline{\mathbf{26}}, \overline{\mathbf{52}}, \dots, \overline{\mathbf{156}}\}$, $\{\overline{\mathbf{13}}, \overline{\mathbf{39}}, \overline{\mathbf{65}}, \dots, \overline{\mathbf{169}}\}$	34
0	$\overline{\mathbf{0}} + \{\overline{\mathbf{0}}, \overline{\mathbf{26}}, \overline{\mathbf{52}}, \dots, \overline{\mathbf{156}}\}$, $\{\overline{\mathbf{0}}, \overline{\mathbf{26}}, \overline{\mathbf{52}}, \dots, \overline{\mathbf{156}}\}$	156
1	$\overline{\mathbf{13}} + \{\overline{\mathbf{0}}, \overline{\mathbf{26}}, \overline{\mathbf{52}}, \dots, \overline{\mathbf{156}}\}$, $\{\overline{\mathbf{13}}, \overline{\mathbf{39}}, \overline{\mathbf{65}}, \dots, \overline{\mathbf{169}}\}$	91

6. mengubah nilai desimal ke karakter huruf dengan cara hitung ada berapa digit bilangan $\mathbf{26} \times \mathbf{7}$, jika bilangan desimal yang ada di plain teks jumlah digitnya kurang dari jumlah digit $\mathbf{26} \times \mathbf{7}$ maka di depan bilangan tersebut ditambahkan bilangan 0.

	65	34	156	91
Perubahan	065	034	156	091

Sehingga diperoleh: 065034156091.

7. Jika digitnya 0 maka diubah jadi karakter $\boldsymbol{A}$. Jika digitnya bukan nol, gabungkan dengan bilangan setelahnya. Ketika nilai gabungan lebih dari 25 maka ambil bilangan pertama dan diubah ke karakter huruf sesuai aturan pemetaan. Jika nilai gabungannya ≤ 25 maka gabungannya yang diubah ke karakter huruf sesuai aturan pemetaan.
   Sehingga 065034156091 menjadi:

	0	6	5	0	3	4	15	6	0	9	1
karakter	A	G	F	A	D	E	P	G	A	J	B

b. Tahapan Dekripsi

Alur proses dekripsi pesan adalah sebagai berikut:

1. Mempersiapkan *ciphertext* yang karakternya huruf.
2. Ubah setiap huruf yang ada di *ciphertext* ke dalam bilangan desimal.
3. Input nilai $\boldsymbol{k}$ untuk kunci dekripsi.
4. Partisi karakter *ciphertext* yang berbentuk desimal ke dalam blok pesan berukuran digit dari $\mathbf{26} \times \boldsymbol{k}$.
5. Dalam 1 blok, jika ada rangkaian digit 0 di sebelah kiri maka digit yang diambil adalah digit bilangan sisanya. Akan tetapi, apabila dalam satu blok isinya digit 0 semua maka ambil digit bilangan 0 tersebut.
6. Lakukan pengecekan setiap bilangan desimal apakah terletak pada suatu koset tertentu. Selanjutnya konversi ke dalam bilangan yang $\leq \mathbf{25}$ yang ada di koset tersebut.
7. Mengubah bilangan desimal yang diperoleh pada tahapan sebelumnya ke huruf sesuai aturan pemetaan.

Contoh 6.

1. *Ciphertext* nya yaitu "AGFADEPGAJB".
2. Ubah setiap huruf yang ada di *ciphertext* ke dalam bilangan desimal.

	A	G	F	A	D	E	P	G	A	J	B
Nilai	0	6	5	0	3	4	15	6	0	9	1

3. Input nilai $\boldsymbol{k} = \mathbf{7}$ untuk kunci dekripsi.
4. Partisi karakter *ciphertext* yang berbentuk desimal ke dalam blok pesan berukuran digit dari $\mathbf{26} \times \boldsymbol{k}$.

Nilai	065	034	156	091

5. Dalam 1 blok, jika ada rangkaian digit 0 di sebelah kiri maka digit yang diambil adalah digit bilangan sisanya. Akan tetapi, apabila dalam satu blok isinya digit 0 semua maka ambil digit bilangan 0 tersebut.

Nilai	65	34	156	91

6. Lakukan pengecekan setiap bilangan desimal apakah terletak pada suatu koset tertentu. Selanjutnya konversi ke dalam bilangan yang $\leq \mathbf{25}$ yang ada di koset tersebut.

   Misalkan barisan bilangan dituliskan dalam array $\boldsymbol{B}[\boldsymbol{i}]$ dengan $\boldsymbol{i}$ adalah indeks barisan bilangan. Sehingga algoritma transformasinya sebagai berikut:

   for $\boldsymbol{i}$ from $\mathbf{1} \rightarrow \boldsymbol{length}(\boldsymbol{B})$
   for $\boldsymbol{j}$ from $\mathbf{0} \rightarrow \mathbf{25}$

if $(\boldsymbol{B}[\boldsymbol{i}] - \boldsymbol{j})\ \boldsymbol{mod}\ \boldsymbol{26} = \boldsymbol{0}$
then $\boldsymbol{C}[\boldsymbol{i}] = \boldsymbol{j}$

$\boldsymbol{C}[\boldsymbol{i}]$ adalah hasil transformasi dari $\boldsymbol{B}[\boldsymbol{i}]$. Berdasarkan contoh diperoleh $\boldsymbol{C}[\boldsymbol{i}]$ adalah

Nilai	13	8	0	13

7. Mengubah bilangan desimal yang diperoleh pada tahapan sebelumnya ke huruf sesuai aturan pemetaan.

	13	8	0	13
Karakter	N	I	A	N

Sehingga diperoleh "NIAN".

Implementasi Program

Dalam penelitian ini, implementasi program menggunakan Bahasa C++.

Kode program untuk enkripsi sebagai berikut:

```
#include <iostream>
#include <fstream>
#include <string>
#include <cstdlib>
#include <ctime>

using namespace std;
int main()
{
      cout<<"Plaintext : \n\n";
      ifstream myfile;
      string word;
      myfile.open("plainteks.txt"
);
      getline(myfile,word);
      //print plaintext to screen
      cout<<word<<endl<<endl;

      int k;
      cout<<"Input key (k) = ";
    cin>>k;

      int c[word.length()];
    for(int                  i=0;
i<word.length();i++)
    {
            srand(time(NULL));
            c[i]=((word[i]-
65)+(rand()%k)*26) % (26*k);
    }

      //ciphertext in number write
to file
      ofstream myfile2;
      myfile2.open
("cipherteks1.txt");
      for(int
j=0;j<=word.length();j++){
      myfile2 <<c[j];
    }
      myfile2.close();

      //ciphertext in number same
digit length write to file
      ofstream myfile3;
      myfile3.open
("cipherteks2.txt");

      int len_26k=1,len_cj=1;
      int input=26*k,input2;
      while (input/10 >= 1){
    input/=10;
    len_26k++;}

      for(int
j=0;j<=word.length();j++){

            input2=c[j];
            while (input2/10 > 0){
      input2/=10;
            len_cj++;}

      for(int     k=1;k<=len_26k-
len_cj;k++){
            myfile3 <<0;
            }myfile3 <<c[j];
            len_cj=1;
      }
      myfile3.close();

      //ciphertext in letter write
to file
      ifstream myfile4;
      string cip;
      myfile4.open("cipherteks2.t
xt");
      getline(myfile4,cip);
      char     num2alp[26]     =
{'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'};

      ofstream myfile5;
      myfile5.open("cipherteks3.t
xt");
      for(int
i=0;i<cip.length();i++){
```

```
            if(cip[i]==48       ||
cip[i]>=51                      ||
(cip[i]>=50&&cip[i+1]>=54)){

      myfile5<<num2alp[cip[i]-
48];
            }else{
                  if(cip[i]==49
|| (cip[i]>=50&&cip[i+1]<=53)){

      myfile5<<num2alp[10*(cip[i]
-48)+(cip[i+1]-48)];
                        i++;
                        }
                  }
      }
      myfile5.close();
      cout<<"\nciphertext    files
created";
    return 0;
}
```

Kode program untuk dekripsi:

```
#include <iostream>
#include <fstream>
#include <string>
#include <cstdlib>
#include <cmath>
#include <ctime>

using namespace std;
int main(){
      cout<<"Ciphertext : \n\n";
      ifstream myfile;
      string word;
      myfile.open("cipherteks3.tx
t");
      getline(myfile,word);
      //print ciphertext to screen
      cout<<word<<endl<<endl;

      int k;
      cout<<"Input key (k) = ";
    cin>>k;

    int len_26k=1;
      int input=26*k;
      while (input/10 >= 1){
    input/=10;
    len_26k++;}

      //ciphertext number write to
file
      int c[word.length()];
      for(int
i=0;i<word.length();i++){
            c[i]=word[i]-65;
            }
      ofstream myfile2;
      myfile2.open("dekripsi1.txt
");
      for(int
i=0;i<word.length();i++){
            myfile2<<c[i];
            }
      myfile2.close();

      //partition  to  block  size
digit of 26*k write to file
      ifstream myfile3;
      string cip;
      myfile3.open("dekripsi1.txt
");
      getline(myfile3,cip);

      ofstream myfile4;
      myfile4.open("dekripsi2.txt
");
      int
d[cip.length()/len_26k];
      for(int
i=0;i<cip.length()/len_26k;i++){
            d[i]=0;
            for(int
j=0;j<len_26k;j++){

      d[i]+=pow(10,len_26k-j-
1)*(cip[len_26k*i+j]-48);
            }
            myfile4<<d[i]<<" ";
      }
      myfile4.close();

      //transform  decription  to
mod 26
      ofstream myfile5;
      myfile5.open("dekripsi3.txt
");
      int
e[cip.length()/len_26k];
      for(int
i=0;i<=(cip.length()/len_26k);i++
){
            e[i]=d[i]%26;
            myfile5<<e[i]<<" ";
      }
      myfile5.close();

      //transform to plainteks
      ofstream myfile6;
      myfile6.open("hasil_akhir_d
ekripsi.txt"    );
      char      num2alp[26]      =
{'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'};
```

```
      for(int
i=0;i<cip.length()/len_26k-
1;i++){

      myfile6<<num2alp[e[i]];
      }
      myfile6.close();
      cout<<"\nplaintext     files
created";
      return 0;
}
```

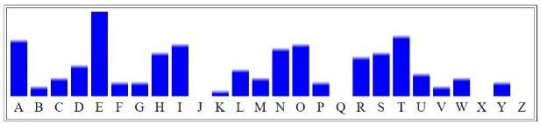
Analisis Frekuensi

Evaluasi dilakukan dengan pengujian analisis frekuensi. Analisis frekuensi menggunakan tools online yang dapat diakses pada website Frequency Counter (mtholyoke.edu). *Plaintext* diperoleh dari teks dengan judul "The Adventure of Wisteria Lodge" berukuran 48156 huruf yang ada di website www.textfiles.com/stories/. Teks diproses sehingga hanya berupa huruf kapital saja dan selanjtnya diinputkan pada tools *frequency counter*, kemudian dilakukan perhitungan frekuensi kemunculan tiap huruf. Hasilnya diperoleh seperti pada Tabel 1. Berikut:

Tabel 1. Frekuensi Kemunculan Huruf *Plaintext*

	N	Persentase (%)
E	6010	12.48
T	4347	9.03
A	3987	8.28
O	3565	7.40
I	3459	7.18
N	3228	6.70
S	3152	6.55
H	3073	6.38
R	2716	5.64
D	2089	4.34
L	1812	3.76
U	1438	2.99
M	1333	2.77
C	1254	2.60
W	1230	2.55
Y	994	2.06
F	956	1.99
G	924	1.92
P	754	1.57
B	741	1.54
V	524	1.09
K	361	0.75
X	86	0.18
Q	52	0.11
J	44	0.09
Z	27	0.06

Pada Tabel 1 terlihat persentase kemunculan setiap huruf pada *plaintext*. Kemunculan huruf tertinggi adalah huruf E sebanyak 12,48% dan yang terendah adalah huruf Z sebanyak 0,06%. Secara statistik kemunculan huruf pada *plaintext* dapat dilihat pada Gambar 2.



Gambar 2. Statistik Frekuensi Huruf pada *Plaintext*

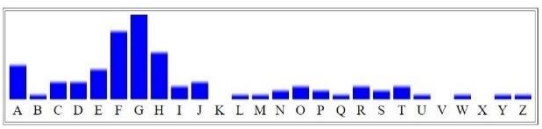
Selanjutnya *plaintext* dienkripsi sehingga diperoleh *ciphertext* menggunakan program yang sudah dirancang. Dilakukan dua kali percobaan menggunakan kunci yang sama yaitu $\boldsymbol{k = 3}$ dan satu kali percobaan dengan kunci $\boldsymbol{k = 10}$. Setelah diperoleh hasil *ciphertext* untuk $\boldsymbol{k = 3}$ pada percobaan pertama dilakukan analisa

frekuensi kemunculan huruf dan diperoleh hasil sebagai berikut.

Tabel 2. Frekuensi Kemunculan Huruf *Chipertext* untuk $k = 3$ pada percobaan pertama

	N	Persentase (%)
G	27618	19.89
F	21826	15.72
H	15784	11.37
A	10598	7.63
E	9827	7.08
C	5840	4.21
J	5833	4.20
D	5166	3.72
I	4435	3.19
T	4347	3.13
O	3565	2.57
R	3541	2.55
N	3228	2.32
S	3152	2.27
P	3121	2.25
L	1812	1.30
U	1438	1.04
M	1334	0.96
Q	1251	0.90
W	1230	0.89
Z	1227	0.88
Y	994	0.72
B	741	0.53
V	524	0.38
K	361	0.26
X	86	0.06

Pada Tabel 2 terlihat persentase kemunculan setiap huruf *ciphertext* untuk $\boldsymbol{k = 3}$ pada percobaan pertama. Kemunculan huruf tertinggi adalah huruf G sebanyak 19,89% dan yang terendah adalah huruf X sebanyak 0,06%. Secara statistik kemunculan huruf pada *ciphertext* untuk $\boldsymbol{k = 3}$ percobaan pertama dapat dilihat pada Gambar 3.



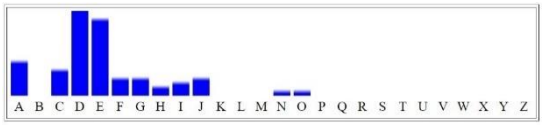
Gambar 3. Statistik Frekuensi Huruf *Ciphertext* Untuk $\boldsymbol{k = 3}$ pada Percobaan Pertama

Percobaan ke dua untuk kunci $\boldsymbol{k = 3}$ diperoleh hasil *ciphertext* dan dilakukan analisa frekuensi kemunculan huruf menggunakan tools serupa, diperoleh hasil sebagai berikut:

Tabel 3. Frekuensi Kemunculan Huruf *Chipertext* untuk $k = 3$ pada Percobaan Ke Dua

	N	Persentase (%)
A	10569	11.29
B	0	0.00
C	7702	8.23
D	25826	27.59
E	23311	24.90
F	5395	5.76
G	5786	6.18
H	3078	3.29
I	3817	4.08
J	5403	5.77
K	0	0.00
L	0	0.00
M	238	0.25
N	681	0.73
O	809	0.86
P	10	0.01
Q	0	0.00
R	0	0.00
S	0	0.00
T	0	0.00
U	0	0.00
V	0	0.00
W	132	0.14
X	482	0.51
Y	355	0.38
Z	7	0.01

Pada Tabel 3 terlihat persentase kemunculan setiap huruf *ciphertext* untuk $\boldsymbol{k = 3}$ pada percobaan kedua. Kemunculan huruf tertinggi adalah huruf A sebanyak 11,29% dan yang terendah adalah huruf Z sebanyak 0,01%. Secara statistik kemunculan huruf *ciphertext* untuk $\boldsymbol{k = 3}$ pada percobaan kedua dapat dilihat pada Gambar 4.



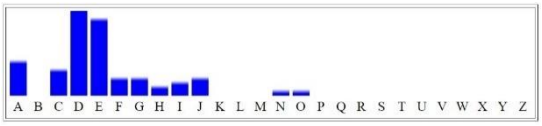
Gambar 4. Statistik Frekuensi Huruf *Ciphertext* Untuk $\boldsymbol{k = 3}$ pada Percobaan Ke Dua

Dapat dilihat bahwa untuk $\boldsymbol{k}$ yang sama, hasil *ciphertext* juga bisa berbeda.
Setelah diperoleh hasil *ciphertext* untuk kunci $\boldsymbol{k = 10}$, dilakukan analisa frekuensi kemunculan huruf menggunakan tools serupa, diperoleh hasil sebagai berikut:

Tabel 3. Frekuensi Kemunculan Huruf *Chipertext* untuk $k = 10$

	N	Persentase (%)
A	44590	33.20
B	0	0.00
C	0	0.00
D	3077	2.29
E	3817	2.84
F	24437	18.20
G	27893	20.77
H	13536	10.08
I	976	0.73
J	5789	4.31
K	4391	3.27
L	0	0.00
M	1	0.00
N	0	0.00
O	0	0.00
P	0	0.00
Q	0	0.00
R	0	0.00
S	0	0.00
T	0	0.00
U	5786	4.31
V	0	0.00
W	0	0.00
X	0	0.00
Y	0	0.00
Z	0	0.00

Pada Tabel 4 terlihat persentase kemunculan setiap huruf *ciphertext*. Kemunculan huruf tertinggi adalah huruf A sebanyak 33,2% dan Sebagian besar tidak muncul lagi yaitu huruf P, Q, R, S, T, V, W, X, Y, Z. Secara statistik kemunculan huruf *ciphertext* untuk $\boldsymbol{k = 10}$ dapat dilihat pada Gambar 4.



Gambar 4. Statistik Frekuensi Huruf *Ciphertext* Untuk $\boldsymbol{k = 10}$

Perhatikan bahwa sebaran persentase kemunculan huruf pada *plaintext* dan *ciphertext* memiliki proporsi yang berbeda. Dengan demikian, analisis frekuensi tidak dapat dijadikan dasar pertimbangan untuk menerka aturan enkripsi.

Recovery

Pengujian recovery menggunakan sampel satu percobaan dengan *plaintext* dari teks dengan judul "The Adventure of Wisteria Lodge" berukuran 48156 huruf yang ada di website www.textfiles.com/stories/ dengan *k=3.* Berdasarkan perbandingan karakter *plaintext* dan *ciphertext* menggunakan tools online yang dapat diakses pada website https://text-compare.com diperoleh kesamaan 100 persen atau identik. Dengan demikian algoritma enkripsi dan dekripsi dapat dikatakan berhasil memenuhi syarat recovery *plaintext.*

PENUTUP

Berdasarkan penelitian dan pengujian yang telah dilakukan, dapat disimpulkan bahwa hasil pengujian analisis frekuensi pada enkripsi kriptografi klasik dengan penerapan konsep koset grup tidak terlihat hubungan antara sebaran kemunculan huruf di *plaintext* dengan kemunculan huruf di *ciphertext,* **sehingga kriptanalis** tidak dapat menerka aturan enkripsi yang digunakan. Walaupun algoritma kriptografi klasik pada saat ini sudah jarang digunakan karena informasi dalam bentuk digital yang menyebabkan representasi pesan asli dalam kode bit biner, akan tetapi ide algoritma dari penelitian ini yang merupakan modifikasi algoritma kriptografi klasik dapat diterapkan pada algoritma kriptografi modern.

DAFTAR PUSTAKA

Gencoglu, M. T. (2019). *Importance of Cryptography in Information Security. IOSR J. Comput. Eng*, 21(1), 65-68

Scheneier, B. (1996). *Applied Cryptography second edition.* USA: Jhon Wiley & Sons.

Menezes, A. J., Oorschot, P. C., & Vanstone, S. A. (1996). *Handbook of applied crypthography.* Boca Raton: CRC Press.

Soemarkidjo. (2007). *Jelajah Kriptologi*. Lembaga Sandi Negara.

Tilborg, H. C. V., & Jajodia, S.. (2011). *Encyclopedia of Cryptography and Security.* USA: Springer.

Munir, R. (2019). *Kriptografi.* Bandung: Informatika.

Wijaya, A. (2020). Modifikasi Algoritma Kriptografi Klasik dengan Implementasi Finite Automata melalui Partisi Pesan Asli berdasarkan Kriteria Pesan Bagian. *Journal of Science and Applicative Technology*. 4(2), 133-139.

Amin, M.M. (2017). *Implementasi Kriptografi Klasik pada Komunikasi Berbasis Teks*. DOI: 10.33369/pseudo.3.3.129-136.

Limbong, T. (2015). Pengujian Kriptografi Klasik Caesar Cipper Menggunakan Matlab. *Semin. Nas. Inov dan Teknol. Inf. Sept., mo. September 2015, pp. 77-80.*

Durbin, J.R. (2009). *Modern Algebra sixth edition*. USA: Jhon Wiley & Sons.

Herstein, I.M. (1996). *Abstract Algebra*, 3^{nd} *edition*. New York: Prentice Hall.

Fraleigh, J.B. (2000). *A First Course in Abstract Algebra,* 6^{th} *Edition.* New York: Addison-Wesley.

Gallian, J.A. (2017). *Contemporary Abstract Algebra,* 9^{nd} *edition.* Boston: Cengage-Learning.

Rotman, J.J. (2003). *Advance Modern Algebra.* New York: Prentice Hall.

Dummit, D.S. (2004). *Abstract Algebra,* 3^{rd} *Edition.* New York: John Wiley and Sons, Inc.