BAB II

TINJAUAN PUSTAKA

II.1. Penelitian Terkait

Penelitian ini merujuk pada beberapa penelitian yang telah dilakukan sebelumnya terkait dengan penerapan algoritma *Hill cipher* maupun *Vigenere cipher* diantaranya berdasarkan pada penelitian yang dilakukan oleh Akim Manaor Hara Pardede pada tahun 2017 yang berjudul "Algoritma Vigenere Cipher Dan Hill Cipher Dalam Aplikasi Keamanan Data Pada File Dokumen". Penelitian tersebut menghasilkan aplikasi enkripsi file dokumen berbasis *desktop* yang dapat digunakan untuk melindungi file berekstensi TXT, DOCX, dan XLSX. Berdasarkan penelitian tersebut terdapat perbedaan dengan penelitian yang akan dilaksanakan, yaitu pada penelitian yang akan dilaksanakan akan dihasilkan aplikasi berbasis android yang digunakan untuk melindungi isi dari pesan singkat.

Penelitian yang dilakukan oleh Angga Aditya Permana pada tahun 2018 yang berjudul "Penerapan Kriptografi Pada Teks Pesan Dengan Menggunakan Metode *Vigenere Cipher* Berbasis Android". Penelitian tersebut menghasilkan aplikasi enkripsi teks pesan berbasis android yang terbatas pada 26 karakter alfabet kemudian hasilnya diteruskan sebagai pesan teks ke aplikasi pengiriman pesan seperti aplikasi *SMS (Short Message Service), Whatsapp, Line,* dan sejenisnya untuk selanjutnya dikirim. Adapun perbedaan dengan penelitian yang akan dilaksanakan adalah penggunaan karakter dalam penelitian ini berdasarkan *ASCII Printable Characters* yaitu sebanyak 96 karakter serta metode yang

digunakan tidak hanya menggunakan *Vigenere Cipher*, namun juga menambahkan *Hill Cipher* kedalam proses penyandian pesan teks.

Penelitian yang dilakukan oleh Muhammad Husnul Arif pada tahun 2016 yang berjudul "Kriptografi *Hill cipher* dan *Least significant Bit* untuk Keamanan Pesan pada Citra". Penelitian tersebut menghasilkan aplikasi berbasis windows yang mampu menyisipkan pesan kedalam citra dimana pesan yang disisip terbatas pada 26 karakter alfabet. Berdasarkan penelitian tersebut terdapat perbedaan dengan penelitian yang akan dilaksanakan, yaitu penelitian yang akan dilaksanakan terbatas pada 96 karakter *ASCII printable characters* serta mengirimkan *ciphertext* yang dihasilkan sebagai pesan *SMS* untuk kemudian dikirimkan.

II.2. Uraian Teoritis

II.2.1. Penjelasan Implementasi

Implementasi dapat didefenisikan secara umum sebagai penerapan atau tindakan yang dilakukan berdasarkan rencana yang telah disusun dengan cermat dan terperinci sebelumnya. Kamus Besar Bahasa Indonesia (KBBI) mendefenisikan implementasi secara khusus kedalam bidang ilmu komunikasi sebagai pengembangan versi kerja sistem dari desain yang diberikan.

Implementasi juga didefenisikan sebagai suatu tindakan atau bentuk aksi nyata dalam melaksanakan rencana yang telah dirancang dengan matang. Dengan kata lain, implementasi hanya dapat dilakukan jika sudah ada perencanaan dan bukan hanya tindakan semata. Oleh sebab itu, penerapan implementasi harus

sesuai dengan perencanaan yang telah dibuat agar hasil yang dicapai sesuai dengan yang diharapkan.

II.2.2. Kriptografi

Kriptografi adalah istilah umum yang digunakan untuk mendeskripsikan desain dan analisis mekanisme berdasarkan teknik matematika yang diperlukan dalam menyediakan layanan keamanan. Algoritma kriptografi pada dasarnya berupa langkah-langkah komputasi yang terperinci sehingga dapat diimplementasikan oleh programmer komputer (Martin, 2017).

Kriptografi pada dasarnya sudah dikenal sejak lama. Sejarah penulisan rahasia tertua dapat ditemukan pada peradaban Mesir kuno, yakni tahun 3000 SM. Bangsa Mesir menggunakan ukiran rahasia yang disebut dengan *hieroglyphics* untuk menyampaikan pesan kepada orang-orang yang berhak. Tahun 400 SM bangsa Spartan di Yunani memanfaatkan kriptografi di bidang militer dengan menggunakan alat yang disebut scytale, yakni pita panjang berbahan daun papyrus yang dibaca dengan cara digulungkan ke sebatang silinder. Sedangkan peradaban Cina dan Jepang menemukan kriptografi pada abad 15 M. Peradaban Islam juga menemukan kriptografi karena penguasaannya terhadap matematika, statistik, dan linguistik. Bahkan teknik kriptanalisis dipaparkan untuk pertama kalinya pada abad 9 Masehi oleh seorang ilmuwan bernama Abu Yusuf Ya'qub ibn 'Ishaq as-Shabbah al Kindi atau dikenal dengan Al-Kindi yang menulis kitab tentang seni memecahkan kode.

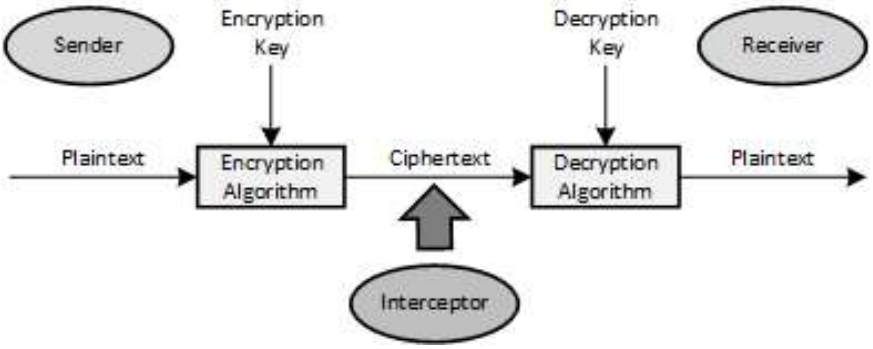
Perkembangan teknologi juga berdampak pada algoritma kriptografi yang semakin berubah menjadi algoritma yang lebih rumit dan kompleks. Kriptografi harus diakui peranannya dalam peperangan sehingga algoritma kriptografi berkembang cukup pesat pada saat perang dunia.

Algoritma kriptografi yang baik tidak ditentukan oleh kerumitan dalam mengolah data atau pesan yang akan disampaikan. Martin (2017) mendefenisikan beberapa prinsip dasar dalam kriptografi sebagai berikut.

1. Kerahasiaan *(Confidentiality)*, adalah jaminan bahwa data tidak dapat dilihat oleh pengguna yang tidak sah.
2. Integritas data *(Data Integrity)*, adalah jaminan bahwa data tidak diubah dengan cara yang tidak sah (termasuk tidak disengaja). Jaminan ini berlaku sejak saat data terakhir dibuat, dikirim, atau disimpan oleh pengguna yang berwenang.
3. Autentikasi *(Authentication)*, dikelompokan menjadi autentikasi asal data dan autentikasi pengguna. Autentikasi asal data yaitu jaminan bahwa data yang diberikan adalah sumber asli data yang diterima. Autentikasi pengguna atau disebut juga sebagai entitas adalah jaminan bahwa entitas tertentu terlibat dan saat ini aktif dalam sesi komunikasi.
4. *Non-repudiation*, adalah jaminan bahwa entitas yang terlibat tidak dapat menyangkal komitmen atau tindakan yang telah terjadi. Hal ini merupakan persyaratan yang lebih kuat dari autentikasi asal data, karena autentikasi asal data hanya memerlukan jaminan untuk diberikan kepada penerima data.

Non-repudiation adalah properti yang paling diinginkan dalam situasi dimana ada potensi timbulnya sengketa atas pertukaran data.

Martin (2017) juga menjelaskan beberapa terminologi umum yang harus diketahui dalam desain kriptografi. *Plaintext* merupakan informasi yang harus dilindungi selama transmisi dari pengirim ke penerima, sedangkan *ciphertext* adalah versi acak dari *plaintext* yang dihasilkan oleh penerapan algoritma enkripsi. Algoritma enkripsi adalah seperangkat aturan untuk mengubah *plaintext* dan kunci yang diberikan menjadi *ciphertext.* Algoritma dekripsi adalah seperangkat aturan untuk mengubah *ciphertext* dan kunci yang diberikan menjadi *plaintext*. Kunci merupakan nilai yang harus diketahui oleh pengirim atau penerima kemudian diimplementasikan kedalam algoritma enkripsi atau dekripsi. Pada beberapa kasus nilai kunci enkripsi bisa berbeda dengan kunci dekripsi. *Interceptor* adalah entitas selain pengirim atau penerima yang mencoba untuk mengetahui *plaintext.* Hubungan antara terminologi tersebut dapat digambarkan kedalam sebuah proses yang berkaitan seperti gambar berikut.



Gambar II.1. Proses Kriptografi
(Sumber : Martin ; 2017)

II.2.3. *Short Message Service (SMS)*

Telepon seluler merupakan bentuk dari perkembangan teknologi komunikasi informasi. Pesan singkat atau lebih dikenal sebagai *SMS* merupakan contoh dari layanan komunikasi yang dapat digunakan pada telepon seluler. Menurut Church dan Oliveira (2013), layanan pesan singkat adalah kemampuan yang ditanam kedalam standar GSM nirkabel yang mampu mengirim dan menerima 160 karakter pesan teks singkat dari perangkat telepon, apapun penyedia layanannya. Layanan pesan singkat sudah berkembang dari pesan teks untuk mengirim gambar, video, dan suara yang dikenal dengan layanan pesan multimedia atau dikenal sebagai MMS, yang biayanya lebih mahal dari pesan singkat biasa.

II.2.4. Algoritma

Algoritma merupakan perintah matematis dasar yang bertanggung jawab terhadap semua aplikasi komputer. Agar dapat dijalankan oleh komputer, penulisan algoritma harus dilakukan dalam notasi bahasa pemrograman. Komputer yang menjalankan sebuah algoritma berarti mengikuti langkah mekanis yang telah dirancang oleh si pemrogram.

Algoritma adalah urutan instruksi yang memberi tahu komputer apa saja yang harus dilakukan. Komputer terbuat dari milyaran sakelar kecil yang disebut transistor, dan algoritma menghidupkan dan mematikan sakelar miliaran kali per detik. Algoritma bukan sembarang kumpulan instruksi: mereka harus tepat dan tidak ambigu untuk dieksekusi oleh komputer. Misalnya, resep memasak bukan

algoritma karena tidak persis menentukan urutan melakukan apa atau tepatnya apa setiap langkahnya. Mengikutinya dapat menghasilkan sesuatu yang lezat atau berantakan. Sebaliknya, suatu algoritma selalu memperoleh hasil yang sama (Domingos, 2015).

II.2.5. *Hill Cipher*

Hill Cipher merupakan kriptografi *polyalphabetic* yang ditemukan pada tahun 1929 oleh Lester S. Hill. Teknik ini diciptakan dengan maksud untuk menciptakan *cipher* (kode) yang tidak dapat dipecahkan menggunakan analisis frekuensi. *Hill Cipher* tidak mengganti setiap abjad yang sama pada *plaintext* dengan abjad lain yang sama pada *ciphertext* karena menggunakan perkalian matriks pada dasar proses enkripsi dan dekripsinya. Sandi ini dapat dipecahkan dengan *Known-Plaintext Attacks* tetapi tahan melawan *ciphertext-only* attack. Cara kerja sandi ini berdasarkan atas perkalian matriks dengan menggunakan sebuah kunci. *Hill Cipher* dikatagorikan sebagai *block cipher* karena teks yang akan diproses dikelompokkan kedalam beberapa blok dengan ukuran tertentu (Sumandri, 2017).

Hill Cipher merupakan algoritma kriptografi yang prosesnya dilakukan dengan operasi aritmatika modulo antara matriks plaintext dengan kunci maupun matriks ciphertext dengan kunci invers. Oleh sebab itu, teks pesan dan teks kunci harus dipetakan sebagai blok matriks. *Plaintext* maupun *ciphertext* akan dipetakan kedalam matriks kolom $1 \times n$, sedangkan matriks kunci akan dipetakan kedalam matriks bujur sangkar $n \times n$ dan *invertible*.

Matriks *invertible* merupakan matriks yang berukuran $n \times n$ dan memiliki nilai determinan tidak sama dengan nol. Penggunaan matriks *invertible* bertujuan untuk memperoleh invers dari matriks kunci. Invers dari matriks kunci yang invertible akan digunakan sebagai kunci untuk proses dekripsi *ciphertext* menjadi *plaintext*. Jika nilai determinan matriks kunci sama dengan banyaknya karakter yang boleh digunakan, maka matriks dapat digunakan dalam proses enkripsi, namun akan gagal ketika proses dekripsi. Sehingga menjadi penting untuk memilih matriks kunci yang sesuai (Sumandri, 2017).

Berdasarkan pendekatan yang dilakukan oleh Stinson dan Paterson (2018) dengan memisalkan *m* sebagai bilangan bulat positif dan dientukan $P = C = (\mathbb{Z}_{26})^m$, maka diperoleh *m* kombinasi linier dari *m* karakter alfabet dalam elemen sebuah *plaintext* sehingga dapat menghasilkan *m* karakter alfabet dalam sebuah elemen *ciphertext*. Sebagai contoh, Jika $m = 2$, kita dapat menulis sebuah elemen *plaintext* sebagai $x = (x_1, x_2)$ dan sebuah elemen *ciphertext* sebagai $y = (y_1, y_2)$. Maka y_1 dan y_2 adalah kombinasi linier dari x_1 dan x_2. Jika diberikan koefisien pada x_1 dan x_2, maka dapat dituliskan menjadi:

$$y_1 = 11x_1 + 3x_2$$

$$y_2 = 8x_1 + 7x_2$$

Kombinasi linier tersebut dapat ditulis secara ringkas kedalam notasi matriks.

$$(y_1, y_2) = (x_1, x_2)\begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix}$$

Secara umum, apabila terdapat sebuah matriks berukuran $m \times m$ sebagai matriks kunci *K*, jika anggota pada baris ke-*i* dan kolom ke-*j* dari *K* adalah k_{ij},

maka ditulis $K = (k_{ij})$. Untuk $x = (x_1, \dots, x_m) \in P$ dan K $\in K$, maka kita dapat menghitung $y = e_K(x) = (y_1, \dots, y_m)$ sebagai berikut :

$$(y_1, y_2, \dots, y_m) = (x_1, x_2, \dots, x_m) \begin{pmatrix} k_{1,1} & k_{1,2} & \dots & k_{1,m} \\ k_{2,1} & k_{2,2} & \dots & k_{2,m} \\ \vdots & \vdots & & \vdots \\ k_{m,1} & k_{m,2} & \dots & k_{m,m} \end{pmatrix}$$

Secara sederhana dapat ditulis kedalam notasi $y = x\,K$ sebagai proses enkripsi, dimana *x* merupakan representasi dari blok *plaintext* dan *y* merupakan representasi dari blok *ciphertext.* Proses dekripsi dilakukan sama halnya dengan proses enkripsi, namun menggunakan invers dari matriks kunci (K^{-1}) sebagai blok kunci dekripsi. Berikut ini merupakan pembuktian apabila kedua ruas dikalikan dengan invers matriks kunci (K^{-1}).

$$\begin{aligned} y &= xK \\ K^{-1}y &= x\,KK^{-1} \\ K^{-1}y &= xI_m \\ K^{-1}y &= x \end{aligned}$$

Berdasarkan persamaan diatas, dapat kita simpulkan bahwa proses enkripsi dapat dilakukan dengan notasi $y = xK$ dan proses dekripsi dapat dilakukan dengan notasi $x = K^{-1}y$. Adapun invers matriks kunci (K^{-1}) harus menghasilkan sebuah matriks identitas apabila dikalikan dengan matriks kunci agar proses dekripsi dapat berhasil. Oleh sebab itu diperlukan sebuah teknik untuk menemukan kunci yang sesuai sebelum melakukan proses enkripsi dan dekripsi sandi *Hill cipher.*

Sebelum melakukan proses enkripsi dan dekripsi, perlu ditentukan sebuah kunci yang akan ditransformasikan kedalam matriks bujur sangkar invertible. Matriks invertible yang akan digunakan terdiri dari sebuah matriks bujur sangkar tidak *singular* serta nilai determinan matriks tersebut harus relatif prima terhadap banyaknya elemen dalam himpunan karakter yang digunakan. Kondisi ini harus terpenuhi untuk membentuk sebuah kunci enkripsi yang memiliki kunci dekripsi dalam metode *Hill cipher.*

Sebagai contoh, telah ditentukan sebuah matriks kunci sebagai "mble", maka nilai karakter kunci tersebut disusun kedalam matriks bujur sangkar sebagai $K = \begin{bmatrix} 77 & 66 \\ 76 & 69 \end{bmatrix}$ sehingga untuk memperoleh nilai K^{-1} harus menentukan nilai determinan dari matriks K terlebih dahulu yaitu sebagai berikut.

$\det(K) = (77 \times 69) - (76 \times 66) = 297\ mod\ 95 = 12\ mod\ 95.$

Kemudian tentukan apakah 12 memiliki hubungan relatif prima terhadap 95 guna mengetahui apakah terdapat nilai invers dari 12 mod 95. Untuk mengetahui dua bilangan dikatakan relatif prima adalah jika faktor persekutuan terbesar antara kedua bilangan tersebut sama dengan 1. Berikut ini adalah perhitungan nilai faktor persekutuan terbesar dari 12 dan 95 menggunakan algoritma *Euclidean* mengikuti persamaan $r_0 = q_1 \times r_1 + r_2$ dapat dilihat pada operasi berikut.

$95 = 7 \times 12 + 11$ … i

$12 = 1 \times 11 + 1$ … ii

$11 = 1 \times 11 + 0$ … iii

Adapun operasi tersebut berhenti apabila $r_2 = 0$, dan nilai faktor persekutuan terbesar dari 95 dan 12 sama dengan 1 dapat diketahui dengan melihat nilai r_2 pada baris kedua dari baris terakhir proses tersebut. Oleh sebab itu, nilai inverse yang terdapat pada 12 mod 95 memiliki penyelesaian sebagaimana proses algoritma *Extended Euclidean* berikut.

Susun (ii) menjadi :

$1 = 12 - 1 \times 11$ … iv

Susun (i) menjadi

$11 = 95 - 12 \times 7$ … v

Sulihkan (v) kedalam (iv) menjadi :

$1 = 12 - 1(95 - 12 \times 7)$ … vi

$1 = 12 - 95 + 12 \times 7$ … vii

$1 = -95 + 12 \times 8$ … viii

Berdasarkan persamaan tersebut, kita peroleh 8 sebagai nilai balikan yang memenuhi persamaan $12a \equiv 1\, mod\, 95$, sebagaimana yang kita ketahui bahwa $12 \times 8 \equiv 1\, mod\, 95$. Selanjutnya nilai balikan determinan akan dipakai untuk memperoleh matriks kunci yang digunakan untuk proses dekripsi seperti berikut.

$K^{-1} = d^{-1}.Adj(K)$

$K^{-1} = 8\begin{bmatrix} 69 & -66 \\ -76 & 77 \end{bmatrix}$

$K^{-1} = \begin{bmatrix} 552 & -528 \\ -608 & 616 \end{bmatrix} mod\ 95$

$K^{-1} = \begin{bmatrix} 77 & 42 \\ 57 & 46 \end{bmatrix}$

$K^{-1} = \begin{bmatrix} m & J \\ Y & N \end{bmatrix}$

Maka nilai dari invers kunci jika disusun kedalam bentuk teks manjadi “mJYN”, kunci inilah yang akan digunakan sebagai kunci dekripsi.

Proses enkripsi dan dekripsi pada *Hill cipher* dilakukan dengan memetakan karakter plainteks menjadi potongan blok – blok matriks 1× n sesuai dengan jumlah kolom dan baris matriks kunci. Apabila jumlah karakter plainteks pada akhirnya tidak mencukupi, maka akan ditambahkan karakter tambahan sebagai pelengkap agar operasi enkripsi dan dekripsi dapat dilakukan. Sebagai contoh telah diketahui sebuah plainteks “ridho” akan dienkripsi dengan kunci “mble”, maka dapat disusun karakter plainteks kedalam blok matriks karakter menjadi $\begin{bmatrix} r \\ i \end{bmatrix}, \begin{bmatrix} d \\ h \end{bmatrix}, \text{dan} \begin{bmatrix} o \\ * \end{bmatrix}$, dalam contoh ini ditambahkan karakter “*” sebagai pelengkap. Kemudian blok karakter diubah menjadi matriks sesuai dengan nomor karakter yang telah ditentukan menjadi $\begin{bmatrix} 82 \\ 73 \end{bmatrix}$, $\begin{bmatrix} 68 \\ 72 \end{bmatrix}, \text{dan} \begin{bmatrix} 79 \\ 10 \end{bmatrix}$. Proses enkripsi dilakukan dengan mengalikan nilai blok *plaintext* dengan matriks kunci sesuai dengan rumus *C = KP mod 95* berikut.

$$\mathrm{C} = \begin{bmatrix} 77 & 66 \\ 76 & 69 \end{bmatrix} \begin{bmatrix} 82 \\ 73 \end{bmatrix} = \begin{bmatrix} 11132 \\ 11269 \end{bmatrix} mod\ 95 = \begin{bmatrix} 17 \\ 59 \end{bmatrix} = \begin{bmatrix} 1 \\ [\end{bmatrix}$$

$$\mathrm{C} = \begin{bmatrix} 77 & 66 \\ 76 & 69 \end{bmatrix} \begin{bmatrix} 68 \\ 72 \end{bmatrix} = \begin{bmatrix} 9988 \\ 10136 \end{bmatrix} mod\ 95 = \begin{bmatrix} 13 \\ 66 \end{bmatrix} = \begin{bmatrix} - \\ \mathrm{b} \end{bmatrix}$$

$$\mathrm{C} = \begin{bmatrix} 77 & 66 \\ 76 & 69 \end{bmatrix} \begin{bmatrix} 79 \\ 10 \end{bmatrix} = \begin{bmatrix} 6743 \\ 6694 \end{bmatrix} mod\ 95 = \begin{bmatrix} 93 \\ 44 \end{bmatrix} = \begin{bmatrix} \} \\ \mathrm{L} \end{bmatrix}$$

Berdasarkan proses tersebut, diperoleh *ciphertext* dari hasil enkripsi adalah “1[-b~L”. Untuk proses dekripsi dilakukan dengan mengalikan blok *ciphertext* dengan blok kunci invers yang telah ditentukan sesuai dengan persamaan $P = K^{-1} C \bmod 95$ seperti berikut.

$$\mathrm{P} = \begin{bmatrix} 77 & 42 \\ 57 & 46 \end{bmatrix} \begin{bmatrix} 17 \\ 59 \end{bmatrix} = \begin{bmatrix} 3787 \\ 3683 \end{bmatrix} mod\ 95 = \begin{bmatrix} 82 \\ 73 \end{bmatrix} = \begin{bmatrix} \mathrm{r} \\ \mathrm{i} \end{bmatrix}$$

$$\mathrm{P} = \begin{bmatrix} 77 & 42 \\ 57 & 46 \end{bmatrix} \begin{bmatrix} 13 \\ 66 \end{bmatrix} = \begin{bmatrix} 3773 \\ 3777 \end{bmatrix} mod\ 95 = \begin{bmatrix} 68 \\ 72 \end{bmatrix} = \begin{bmatrix} \mathrm{d} \\ \mathrm{h} \end{bmatrix}$$

$$\mathrm{P} = \begin{bmatrix} 77 & 42 \\ 57 & 46 \end{bmatrix} \begin{bmatrix} 93 \\ 44 \end{bmatrix} = \begin{bmatrix} 9009 \\ 7325 \end{bmatrix} mod\ 95 = \begin{bmatrix} 79 \\ 10 \end{bmatrix} = \begin{bmatrix} o \\ * \end{bmatrix}$$

Maka, diperoleh *plainteks* dari hasil dekripsi adalah “ridho*”.

II.2.6. Aritmatika Modulo

Aritmatika modulo merupakan sisa hasil pembagian 2 (dua) bilangan. Operator yang digunakan dalam aritmatika modulo adalah mod. Misalkan *a* adalah bilangan bulat dibagi dengan *m* adalah bilangan bulat > 0 , maka akan menghasilkan sisa bagi *r* dengan *q* adalah hasil bagi. Sehingga dapat dinotasikan sebagai berikut:

$a\ mod\ m = r$ sedemikian sehingga $a = m_q + r$, dengan $0 \leq r \leq m$

Contoh: $13\ mod\ 2 = 1$, dimana $13 = (2 \times 6) + 1$.

1. *Greatest Common Divisor (GCD)*

Greatest Common Divisor (GCD) merupakan bilangan bulat terbesar yang merupakan pembagi yang sama dari dua bilangan bulat. Misalkan a dan b adalah 2 (dua) bilangan bulat yang tidak nol. *Greatest common divisor (GCD)* dari a dan b adalah bilangan bulat terbesar c sedemikian sehingga $c|a$ dan $c|b$. Greatest common divisor (GCD) dari a dan b dapat dinotasikan dengan GCD(a,b). Adapun dalam menentukan nilai dari GCD(a,b) dapat dilakukan dengan algoritma *euclid*.

2. Bilangan Prima

Bilangan prima adalah bilangan bulat positif a, dimana $a \geq 2$ hanya dapat dibagi dengan 1 dan bilangan itu sendiri. Seluruh bilangan prima adalah bilangan ganjil, kecuali 2 yang merupakan bilangan genap. Contoh bilangan prima adalah 2, 3, 5, 7, 11, 17, ….

Untuk menguji apakah *n* merupakan bilangan prima atau bukan, kita cukup membagi *n* dengan sejumlah bilangan prima, dengan syarat bilangan prima $\leq \sqrt{n}$. Jika *n* habis dibagi dengan salah satu dari bilangan prima tersebut, maka *n* bukan bilangan prima, tetapi jika *n* tidak habis dibagi oleh semua bilangan prima tersebut, maka *n* adalah bilangan prima.

3. Relatif Prima

Dua bilangan bulat a dan b dikatakan relatif prima jika $\text{GCD}(a, b) = 1$. Bilangan – bilangan $a_1, a_2, \ldots, a_n$ adalah relatif prima berpasangan (*pairwise relatively prime*) jika $\text{GCD}(a_i, a_j) = 1$ untuk $1 \leq i \leq j \leq n$. Dengan demikian, sekumpulan bilangan bisa ditunjukkan apakah relatif prima atau tidak dengan mengevaluasi GCD dari semua pasangan bilangan yang mungkin. Jika GCD pasangan-pasangan tersebut semuanya bernilai 1, maka syarat *pairwise relatively prime* dipenuhi. Sebaliknya, jika salah satu GCD dari pasangan bilangan tersebut tidak sama dengan 1, maka kumpulan bilangan tersebut bukan *pairwise relatively prime.* Dan 2 (dua) bilangan prima pasti adalah *pairwise relatively prime*.

4. Algiritma *Euclid*

Algoritma ini digunakan untuk mencari nilai pembagi persekutuan terbesar dari 2 (dua) bilangan bulat. Algoritma ini didasarkan pada dua bilangan bilangan bulat positif r_0 dan r_1, dengan $r_0 \leq r_1$, kemudian dihitung menggunakan algoritma pembagian :

$r_0 = q_1 \times r_1 + r_2$	$0 < r_2 < r_1$
$r_1 = q_2 \times r_2 + r_3$	$0 < r_3 < r_2$
$r_{n-2} = q_{n-1} \times r_{n-1} + r_n$	$0 < r_n < r_{n-1}$
$r_{n-1} = q_n \times r_n$	

Berdasarkan persamaan tersebut, maka diperoleh :

$$GCD(r_0, r_1) = GCD(r_1, r_2) = \ldots\ldots\ldots = GCD(r_{n-1}, r_n) = GCD(r_n, 0) = r_n$$

Berikut ini merupakan contoh perhitungan untuk menentukan nilai pembagi persekutuan terbesar dari 12 dan 95.

$95 = 7 \times 12 + 11$

$12 = 1 \times 11 + 1$

$11 = 11 \times 1 + 0$

5. *Extended Euclid*

Algoritma Extended Euclidean ini merupakan perluasan dari algoritma Euclidean yang berfungsi untuk menentukan nilai x dan y sedemikian sehingga $r_0 \times x + r_1 \times y = \mathrm{GCD}(r_0, r_1)$ dengan r_0, r_1 merupakan bilangan bulat positif serta x dan y merupakan bilangan bulat. Dua bilangan bulat r_0 dan r_1 yang

merupakan *pairwise relatively prime* dapat menemukan bilangan bulat x dan y sedemikian sehingga $r_0 \times x + r_1 \times y = 1$. Dengan menggunakan rumus Euclidean.

$$t_j = t_{j-2} - q_{j-1} \times t_{j-1}, j \geq 2$$

Dengan t_j adalah suatu barisan bilangan t_1 , t_2 , t_3 ,..., t_n dan q_j diperoleh dari perhitungan GCD$(r_1, r_2) = 1$ dengan ${r_1}^{-1} = t_n$. Sehingga diperoleh : $r_n = t_n \times r_1$ atau $1 = t_n \times r_1$. Sebagai contoh 13x + 8y = 1.

a. Hitung GCD(13,8) = 1 dengan mengunakan algoritma *Euclid.*

$13 = 1 \times 8 + 5$ … i

$8 = 1 \times 5 + 3$ … ii

$5 = 1 \times 3 + 2$ … iii

$3 = 1 \times 2 + 1$ … iv

$2 = 2 \times 1 + 0$ … v

b. Lakukan perhitungan substitusi dimulai dari iv.

$1 = 3 - 2 \times 1$

$1 = 3 - 1(5 - 3) = 2 \times 3 - 5$

$1 = 2 \times 3 - 5 = 2(8 - 5) - 5 = 2 \times 8 - 3 \times 5$

$1 = 2 \times 8 - 3 \times 5 = 2 \times 8 - 3(13 - 8) = 5 \times 8 - 3 \times 13$

Maka diperoleh $13(-3) + 8(5) = 1$. Sehingga $x = -3$ dan $y = 5$.

6. Balikan Modulo (*Modulo Invers)*

Dalam aritmatika bilangan riil, invers (*inverse*) dari perkalian adalah pembagian. Sebagai contoh invers dari 2 adalah ½, sebab $2 \times ½ = 1$. Jika *a* dan

m relatif prima dan *m* > 1, maka kita dapat menemukan balikan (*inverse)* dari *a* mod *m*. Balikan dari *a* mod *m* adalah bilangan bulat *x* sedemikian sehingga $xa \equiv 1 \, mod \, m$ dan dapat pula dituliskan kedalam notasi lain sebagai $x = \frac{1}{a} \, mod \, m$ sehingga $x = a^{-1} \, mod \, m$. Sebagai pembuktian, diketahui bahwa *a* dan *m* relatif prima, jadi GCD(*a,m*) = 1, dan terdapat bilangan bulat *p* dan *q* sedemikian sehingga *pa*+*qm* = 1 yang mengimplikasi bahwa *pa*+*qm* ≡ 1 mod *m.* Karena *qm* ≡ 0 mod m, maka *pa* ≡ 1 mod *m.* Kekongruenan yang terakhir ini berarti bahwa *p* adalah balikan dari *a* modulo *m*. Sebagai contoh, untuk menentukan balikan dari 4 mod 9 dapat dibuktikan dengan menghitung GCD(4,9) = 1, yang artinya balikan 4 mod 9 ada. Berdasarkan Algoritma *Euclid* diperoleh bahwa $9 = 2 \times 4 + 1$. Susun persamaan tersebut menjadi $-2 \times 4 + 1 \times 9 = 1$. Berdasarkan persamaan tersebut diperoleh bahwa -2 adalah nilai balikan dari 4 mod 9.

II.2.7. *Vigenere Cipher*

Vigenere Cipher atau sandi *Vigenere* merupakan jenis cipher abjad majemuk yang paling sederhana. *Vigenere cipher* menerapkan metode substitusi polialfabetik dan termasuk ke dalam kategori kunci simetris dimana kunci yang digunakan untuk proses enkripsi adalah sama dengan kunci yang digunakan untuk proses dekripsi (Benni, 2014).

Untuk menyandikan suatu pesan, digunakan sebuah tabel alfabet yang disebut tabel *Vigenere* atau *tabula recta*. Tabel *Vigenere* berisi alfabet yang dituliskan dalam 26 baris, masing-masing baris digeser satu urutan ke kiri dari baris sebelumnya, membentuk ke-26 kemungkinan sandi Caesar. Setiap huruf

disandikan dengan menggunakan baris yang berbeda-beda, sesuai kata kunci yang diulang.

	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
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 II.2. Tabel *Vigenere* (*Tabula Recta*)

Misalnya, diketahui sebuah *plaintext* yang akan dienkripsi adalah "POTENSI" dan kunci yang akan digunakan pengirim dan penerima adalah "UTAMA" sehingga jumlah karakter kunci perlu disamakan dengan karakter *plaintext* yang membentuk kunci "UTAMAUT". Berdasarkan tabel *Vigenere,* maka huruf pertama plainteks yaitu P akan disandikan dengan menggunakan baris U, huruf pertama kunci. Pada baris U dan kolom P di tabel *Vigenere.* Demikian pula untuk huruf selanjutnya dan proses ini dijalankan terus sehingga :

Plaintext : POTENSI

Kunci : UTAMAUT

Ciphertext : JHTQNMB

Proses dekripsi dilakukan dengan mencari huruf *ciphertext* pada baris judul huruf dari kata kunci. Misalnya, cari huruf J (huruf pertama pada *ciphertext*) pada baris U (huruf pertama kata kunci) dan akan ditemukan pada kolom P, maka P menjadi huruf pertama *plaintext*. Lanjutkan hingga diperoleh *plainteks* "UTAMA".

Enkripsi dengan *Vigenere cipher* dapat dituliskan secara matematis dengan menggunakan operasi aritmatika modulus mengikuti persamaan $C = (P + K)\ mod\ 26$. Sebaliknya, pada proses dekripsi dengan *Vigenere cipher* dapat dituliskan dengan persamaan $C = (P - K)\ mod\ 26$ dimana *C,P* dan *K* merupakan representasi dari karakter *ciphertext, plaintext,* dan kunci.

II.2.8. UML

UML (*Unified Modelling Language*) merupakan standar pemodelan yang memungkinkan untuk pembuatan diagram terstandarisasi guna meningkatkan komunikasi serta partisipasi dari semua pemangku kepentingan proyek dalam waktu yang panjang. UML juga dapat membantu memodelkan persyaratan sistem baru dan juga membantu memahami proses dan tugas aplikasi bisnis yang ada. Nilai model berbasis UML berasal dari kemampuan untuk memfasilitasi komunikasi, diskusi, dokumentasi, dan pertimbangan sejumlah skenario "bagaimana-jika" untuk sistem perangkat lunak yang besar dan kompleks (Unhelkar, 2018).

UML pertama kali diusulkan sekitar tahun 1995 sebagai kombinasi dari tiga metode yang paling populer pada waktu itu yaitu *Booch, Object Oriented Technique,* dan *Object Oriented Software Engineering.* Kemudian, beberapa metode lain digabungkan ke dalam UML, akhirnya menghasilkan UML versi 1.4 yang populer. Sekitar 2004, UML sebagian diformalkan versi 2.0 dirilis. Versi UML terdiri dari 13 diagram resmi dan perubahan yang sesuai dengan metamodel, yang mengarah keinisiatif seperti *Model-Driven Architecture* (*MDA*). Satu dekade kemudian, muncul UML versi 2.5 yang terdiri dari 14 diagram, dianggap sebagai bahasa pemodelan *de-facto* untuk *Software Engineering*.

II.2.9. *Use Case* Diagram

Use case diagram adalah deskripsi fungsi dari sebuah sistem dari perspektif pengguna. Use case diagram bekerja dengan cara mendeskripsikan tipikal interaksi antara user (pengguna) sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem dipakai.

Dalam menggambarkan *use case* diagram dikenal beberapa notasi yang mewakili *actor*, *use case, relationship, boundary,* dan *notes. Actor* adalah peran yang dimainkan oleh seseorang atau sesuatu yang berada di luar sistem perangkat lunak. Seorang *actor* (pengguna suatu sistem) berinteraksi dengan sistem untuk mencapai tujuan bisnis. *Use case* mendokumentasikan serangkaian interaksi *actor* dengan sistem. *Use case* menggambarkan apa yang dilakukan suatu sistem, tetapi mereka tidak menentukan bagaimana sistem melakukannya. *Relationship* pada *use case* diagram dapat terjadi antara sesama *actor, actor* dengan *use case,* dan antara

dua *use case*. *Boundary* merupakan representasi dari sistem . *Notes* dapat digunakan untuk menjelaskan hubungan antar-*usecase*, melengkapi informasi tambahan pada *actor,* dan menandai kepentingan *usecase* tertentu (Unhelkar, 2018).

II.2.10. Java

Java adalah salah satu bahasa pemrograman paling populer. Ini adalah turunan dari bahasa pemrograman C dan banyak terkait dengan C ++. Java, seperti C ++, mewujudkan konsep pemrograman berorientasi objek, yang memungkinkan seorang programmer untuk menentukan tipe data dengan serangkaian operasi yang diizinkan. Untuk menjalankan program Java pada mesin, mesin membutuhkan instalasi Java Running Environment (JRE). Bagian utama dari JRE adalah Java Virtual Machine (JVM). JVM menciptakan lingkungan dimana program Java berinteraksi dengan perangkat keras.

Seorang programmer membuat program Java dengan menulis kode sumber. Kode sumber adalah file teks yang menjelaskan program sesuai dengan sintaksis Java dan memiliki ekstensi nama *file*.java. Bahasa Java yang datang dengan JDK terdiri dari dua bagian. Bagian pertama, disebut java.lang, adalah komponen penting dari Java. Bagian kedua adalah kumpulan kode sumber yang dapat dipilih dan ditambahkan ke program Java yang sedang ditulis .

II.2.11. Android Studio

Android merupakan sistem operasi *smartphone* yang paling umum digunakan diseluruh dunia. Android Studio adalah sebuah *Integrated Development Enviroment* resmi Android yang bertujuan untuk mempercepat pengembangan dan membantu pengembang perangkat lunak android membuat aplikasi berkualitas untuk perangkat android. Android Studio dipilih karena memiliki banyak fitur yang mempermudah programmer yang ingin membangun aplikasi android.