BAB II

TINJAUAN PUSTAKA

II.1. Penelitian Terkait

Untuk mendukung keberhasilan penelitian ini, penulis melakukan pendekatan teoritis melalui beberapa literatur yang berhubungan dengan penelitian yang dilakukan. Beberapa uraian penelitian terdahulu yang menjadi acuan dalam penelitian ini yaitu:

Penelitian yang dilakukan oleh Raja Sakti Arief Daulay, dkk (2019) dengan judul "Meningkatkan Keamanan *Vigenere Cipher* dengan *Stream Cipher*". Penelitian ini menghasilkan penggabungan dari kedua *Cipher* ini sangat cocok karena penggunaannya yang tidak sulit dan penggabungan dua *Cipher* ini menghasilkan *Ciphertext* yang sulit dide*tek*si dari pola maupun dari kunci. Penambahan ASCII karakter dibuat agar jangkauan dari karakter semakin luas. Jika hanya dipusatkan dengan huruf alphabet yang berjumlah 26 maka akan mudah untuk diketahui dari pola sedangkan karakter ASCII yang berjumlah 127 akan membuat kriptanalis akan lebih sulit untuk mengetahui. Pola yang dimunculkan pada algoritma ini sangat acak karena *Ciphertext* mengandung unsur karakter symbol maupun karakter *alphabet*.

Pada penelitian yang dilakukan oleh Syamsurizal Angga Agusta dan Triyani Arita Fitri A (2016) dengan judul "Implementasi Algorithma *Stream Cipher* RC4 dalam Aplikasi Pendataan Alumni STMIK Amik Riau". Penelitian ini menghasilkan sebuah Algoritma *Stream Cipher* RC4 yang dapat mencegah

terjadinya serangan SQL Injection. Data yang dienkripsi dengan Algoritma *Stream Cipher* RC4 berada dalam kondisi aman saat tersimpan pada *database.* Algoritma *Stream Cipher* RC4 memiliki tingkat efisiensi yang baik dalam penyimpanan data pada *database*, karena hasil enkripsi yang dihasilkan sama jumlahnya dengan karakter aslinya.

Pada penelitian yang dilakukan oleh Galih Agustian Perdana, dkk (2021) dengan judul "Implementasi Algoritma Kriptografi *Playfair Cipher* untuk mengamankan data aset (Studi kasus: PT. Adywinsa Stamping Industries)". Penelitian ini menghasilkan pengamanan data aset degan algoritma kriptografi *Playfair Cipher* dilakukan untuk mengamankan *field* data aset yaitu nama *user* dan nama aset yang bertujuan aplikasi mampu melakukan enkripsi data dengan menggunakan algoritma kriptografi *Playfair Cipher* kemudian memasukkan data yang telah di enskripsi ke dalam *database* dan mampu melakukan dekripsi dari data di dalam *database* telah di enskripsi.

Pada penelitian yang dilakukan oleh Pandu Ridho Ekananda, dkk (2019) dengan judul "Keamanan *database* pada sistem informasi layanan dan pengaduan Dinas Pendidikan Provinsi Sumatera Selatan". Penelitian ini menghasilkan Data yang di input akan di simpan pada *database* dalam keadaan terenkripsi sehingga keamanan dan kerahasiaan datanya dapat terjaga.

Penelitian yang dilakukan oleh Andri Syahputra (2018) dengan judul "Analisa Implementasi Pengamanan *File audio* menggunakan *Algoritma playfair*. Penelitian ini menghasilkan sebuah algoritma kriptografi pada file audio membuat data terjaga dari modifikasi data. *Algoritma playfair Cipher* pada *file audio* telah

selesai diimplementasikan dan dapat digunakan sebagai media pengamanan *file audio.*

Penelitian yang dilakukan oleh Satriya Tri Cahya Kurniawan, dkk (2017) dengan judul "Implementasi kriptografi algoritma *rivest shamir adleman* dengan *playfair Cipher* pada pesan *teks* berbasis *android".* Penelitian ini menghasilkan yang dapat mengimplementasikan kriptografi *algoritma rivest shamir adleman* (RSA) yang dipadukan dengan *playfair Cipher* berbasis *android* pada media file *teks*. Dapat mengimplementasikan ilmu kriptografi dalam bidang teknik informatika sebagai menyamarkan pesan aslinya untuk melindungi keamananya.

Penelitian yang dilakukan oleh Dedi Putra Oloan Simamora (2017) dengan judul "Implementasi algoritma RC4 dan *Playfair Cipher* untuk mengamankan data *teks*". Penelitian ini menghasilkan Teknik penyadian *teks* kriptografi terdapat dua cara yaitu dengan subsitusi dan transposisi yang memilki kunci sebagai jembatan untuk enkripsi dan dekripsi. Penerapan teknik RC4 dan *algoritma playfair Cipher* hanya dapat diterapkan pada data *teks*.

Penelitian yang dilakukan oleh Rahmat Suriadi, dkk. (2020) dengan judul "Peningkatan Keamanan Data Dengan Menggunakan *Equation* Pada Metode *Playfair Cipher*". Penelitian ini menghasilkan proses kriptografi menggunakan metode playfair cipher efektif dalam melakukan proses enkripsi dan dekripsi data teks. Proses enkripsi dan dekripsi data menggunakan simbol ASCII terhadap teks berupa equation atau rumus matematika tidak tepat digunakan dikarenakan ada banyak simbol matematika yang tidak ada dalam simbol ASCII seperti simbol akar (√), zigma (∑), integral (∫), dll. Proses enkripsi dam dekripsi masih

berpeluang tidak berhasil dalam proses enkripsi dan dekripsinya jika dalam proses enkripsi melibatkan karakter “RS”, “US” dan spasi. Ini dikarenakan implementasi enkripsi dan dekripsi dilakukan per karakter.

II.2. Landasan Teori

II.2.1. Rancang Bangun

Menurut Teisnajaya (2015) Rancang bangun adalah program yang menentukan aktifitas pemrosesan informasi yang dibutuhkan untuk penyelesaian tugastugas khusus dari pemakai atau pengguna komputer. (Panglipur & Pratiwi, 2021)

Menurut Maulani (2018) Rancang bangun adalah menciptakan dan membuat suatu aplikasi ataupun sistem yang belum ada pada suatu instansi atau objek tersebut. (Panglipur & Pratiwi, 2021)

Berdasarkan pengertian para ahli di atas maka dapat di simpulkan bahwa rancang bangun adalah program yang menentukan aktifitas pemrosesan informasi yang dibutuhkan untuk penyelesaian tugas-tugas khusus dari pemakai atau pengguna komputer dan membuat suatu aplikasi atau pun sistem yang belum ada pada suatu instansi atau objek. (Panglipur & Pratiwi, 2021)

II.2.2. Aplikasi

Menurut Jogiyanto (dalam Santoso dan Rahman, 2015:79) aplikasi adalah suatu program yang memiliki perintah untuk dapat mengolah suatu data. Aplikasi memiliki berbagai atribut yang terdiri dari beberapa kolom-kolom form yang

dibangun dengan baik agar membentuk suatu tampilan yang menarik sehingga dapat membuat pengguna mudah dalam pengopersaiannya. (Kinaswara, 2019)

Aplikasi merupakan suatu perangkat lunak yang ditanamkan ke dalam komputer yang memiliki berbagai perintah untuk dapat melakukan bentuk pekerjaan sesuai dengan instruksi yang dilakukan oleh pengguna (Santoso dan Rahman, 2015:79).

Dari uraian di atas maka dapat disimpulkan bahwa aplikasi adalah perangkat lunak yang diciptakan dengan berbagai komponen atribut yang sesuai dengan pengguna agar dapat membantu pengguna dalam mengolah setiap data agar menghasilkan input dan *output*. (Kinaswara, 2019)

II.2.3. Keamanan Komputer

Keamanan komputer adalah berhubungan dengan pencegahan diri dan de*teks*i terhadap tindakan pengganggu yang tidak dikenali dalam sistem komputer. Dalam keamanan sistem komputer yang perlu kita lakukan adalah untuk mempersulit orang lain untuk mengganggu sistem yang kita pakai, baik itu kita menggunakan komputer yang sifatnya stand alone, jaringan local maupun global. Kita harus memastikan sistem bisa berjalan dengan baik dan kondusif, selain itu program aplikasinya masih bisa dipakai tanpa masalah. (RAHANTA, 2021)

Sistem keamanan komputer merupakan subsistem organisasi yang mengendalikan risiko-risiko khusus yang berkaitan dengan sistem informasi berdasar komputer. Sistem keamanan komputer memiliki elemen-elemen dasar sistem informasi, seperti perangkat keras, basis data, prosedur, dan laporan. Komputer merupakan sebuah contoh dari bagian perangkat keras. Sistem

keamanan komputer tidak hanya meliputi keamanan fisik tetapi juga menyangkut dengan keamanan non fisik. (Sayuthi, 2021).

Keamanan komputer merupakan suatu cabang teknologi yang dikenal dengan namakeamanan informasi yang diterapkan pada computer. Sasaran keamanan komputer antara lainadalah sebagai perlindungan informasi terhadap pencurian atau pemeliharaan ketersediaan, sepertidijabarkan dalam kebijakan keamanan. Citra atau gambar merupakan salah satu bentuk multimedia yang penting. (Prayoga & Yusfrizal, 2021)

II.2.4. Keamanan *Database*

Keamanan merupakan suatu pro*teks*i terhadap pengrusakan data dan pemakaian data oleh pemakai yang tidak memiliki kewenangan. Keamanan *database* adalah pemberian perlindungan pada *database* terhadap berbagai bentuk ancaman dan gangguan, baik yang bersifat teknis maupun administrasi. Jika *database* perpustakaan tidak dilengkapi dengan sistem keamanan yang baik maka kelemahan ini akan menjadi peluang terjadinya *cybercrime* di dalam perpustakaan. Jadi berdasarkan pengertian di atas dapat disimpulkan bahwa sistem keamanan *database* merupakan kumpulan elemen-elemen yang dirancang untuk melindungi aset dari sebuah ancaman atau gangguan yang bersifat teknis maupun administratif.(Kifli et al., 2021)

Keamanan sistem komputer dalam *database* dikategorikan dalam beberapa aspek yaitu:

1. *Privacy / Confidentiality*

   Inti utama aspek *privacy* atau *confidentiality* adalah usaha untuk menjaga

informasi dari orang yang tidak berhak mengakses. *Privacy* lebih kearah data-data yang sifatnya privat sedangkan confidentiality biasanya berhubungan dengan data yang diberikan ke pihak lain untuk keperluan tertentu (misalnya sebagai bagian dari pendaftaran sebuah servis) dan hanya diperbolehkan untuk keperluan tertentu tersebut.

2. *Integrity*

   Aspek ini menekankan bahwa informasi tidak boleh diubah tanpa seizin pemilik informasi. Adanya virus, *trojan horse*, atau pemakai lain yang mengubah informasi tanpa izin merupakan contoh masalah yang harus dihadapi. Sebuah *e-mail* dapat saja "ditangkap" (*intercept*) di tengah jalan, diubah isinya (*altered,tampered, modified*), kemudian diteruskan ke alamat yang dituju. Dengan kata lain, integritas dari informasi sudah tidak terjaga. Penggunaan *encryption* dan *digital signature*, misalnya, dapat mengatasi masalah ini.

3. *Authentication*

   Aspek ini berhubungan dengan metode untuk menyatakan bahwa informasi betul- betul asli, orang yang mengakses atau memberikan informasi adalah betul-betul orang yang dimaksud, atau server yang kita hubungi adalah betul-betul server yang asli.

4. *Availability*

   Aspek *availability* atau ketersediaan berhubungan dengan ketersediaan informasi ketika dibutuhkan. Sistem informasi yang diserang atau dijebol dapat menghambat atau meniadakan akses ke informasi.

5. *Nonrepudiation*

   Merupakan hal yang bersangkutan dengan sipengirim, sipengirim tidak dapat mengelak bahwa dialah yang mengirim pesan/informasi itu.

6. *Access control*

   Pengaturan (user ID) Aspek ini berhubungan dengan cara pengaturan akses kepada informasi. *Acces control* seringkali dilakukan menggunakan kombinasi user id dan password atau dengan menggunakan mekanisme lainnya. (WINATA, n.d.)

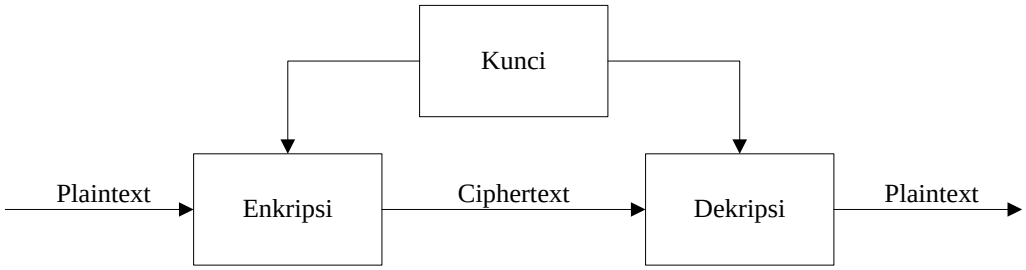
II.2.5. Kriptografi

Kriptografi dapat didefinisikan sebagai seni maupun ilmu yang menghasilkan pesan yang rahasia. Sebuah pesan asli yang disebut sebagai *plaintext* disandikan menjadi pesan yang tersandi yang disebut sebagai *Ciphertext* melalui proses enkripsi dan *Ciphertext* dipulihkan menjadi *plaintext* kembali melalui proses dekripsi. Kriptografi memiliki beragam algoritma yang telah banyak digunakan sebagai keamanan untuk informasi. Algoritma kriptografi dikelompokkan ke dalam dua jenis yaitu algoritma kriptografi klasik dan algoritma kriptografi modern. Dalam pengoperasiannya, algoritma kriptografi klasik bekerja menggunakan mode karakter sedangkan algoritma kriptografi modern bekerja menggunakan mode bit.

Masalah keamanan merupakan salah satu aspek penting dari sebuah sistem informasi. Salah satu hal penting dalam komunikasi menggunakan komputer dan dalam jaringan komputer untuk menjamin keamanan pesan, data ataupun informasi adalah enkripsi. Enkripsi dapat diartikan sebagai sebuah proses yang

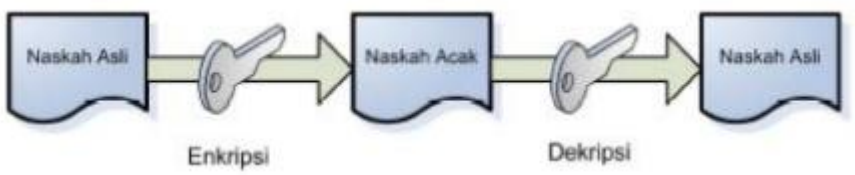
dilakukan untuk mengubah pesan asli menjadi pesan yang tersandikan. Informasi yang asli disebut sebagai *plaintext*, dan bentuk yang sudah dienkripsi disebut sebagai *Ciphertext*. Pesan *Ciphertext* berisi seluruh informasi dari pesan *plaintext*, tetapi tidak dalam format yang dapat dibaca oleh manusia ataupun komputer tanpa menggunakan mekanisme yang tepat untuk melakukan dekripsi.

Kriptografi memiliki dua konsep utama, yaitu enkripsi (*encryption*) dan dekripsi (*decryption*). Enkripsi adalah proses penyandian plain*teks* menjadi *Cipherteks*, sedangkan dekripsi adalah proses mengembalikan *Cipherteks* menjadi plain*teks* semula. Enkripsi dan dekripsi membutuhkan kunci sebagai parameter yang digunakan untuk transformasi.



Gambar II.1. Skema Enkripsi dan Dekripsi Kriptografi *Type Symmentric Key* *(Sumber: Yusfrizal, 2019)*

Algoritma kriptografi klasik memiliki ciri di antaranya berbasis karakter dan menggunakan kunci simetri. Dalam kriptografi klasik, teknik enkripsi yang digunakan adalah enkripsi simetris dimana kunci dekripsi sama dengan kunci enkripsi seperti dapat dilihat pada Gambar 2.



Gambar II.2. Proses Enkripsi dan Dekrispi
(Sumber: Yusfrizal, 2019)

Salah satu algoritma klasik adalah *Caesar Cipher*. Dalam kriptografi klasik, secara umum dapat dikelompokkan dalam dua model yaitu menggunakan tekik substitusi dan transposisi. Teknik substitusi dilakukan dengan mengganti salah satu karakter yang ada dalam sebuah *teks* menggunakan karakter yang lain. Teknik yang termasuk dalam kategori substitusi adalah kriptografi *Caesar*.

Algoritma kriptografi modern merupakan suatu perbaikan yang mengacu pada kriptografi klasik. Algoritma ini menggunakan pengolahan simbol biner yang dibentuk dari kode ASCII (American Standard Code for Information Interchange) karena berjalan mengikuti operasi komputer digital, sehingga membutuhkan pengetahuan dasar matematika untuk menguasainya. Algoritma ini memiliki tingkat kesulitan yang kompleks yang menyebabkan kriptanalis sangat sulit memecahkan *ciphertext* tanpa mengetahui kuncinya. Adapun jenis kunci dalam kriptografi modern terdiri dari 3 yaitu: simetri, asimetri, dan hibrida. Pada kriptografi modern terdapat berbagai macam algoritma yang dimaksudkan untuk mengamankan informasi yang dikirim melalui jaringan komputer. Contoh kriptografi modern yaitu MD5, RC4, AES dan lain-lain. (Yusfrizal, 2019)

Dalam bidang kriptografi terdapat dua konsep yang sangat penting atau utama yaitu enkripsi dan dekripsi. Enkripsi adalah proses dimana informasi atau data yang hendak dikirim diubah menjadi bentuk yang hampir tidak dikenali sebagai informasi awalnya dengan menggunakan algoritma tertentu. Dekripsi adalah kebalikan dari enkripsi yaitu mengubah kembali bentuk tersamar tersebut menjadi informasi awal. Sebuah pesan atau data yang masih asli dan belum mengalami penyandian dikenal dengan istilah plaintext. Kemudian setelah

disamarkan dengan suatu cara penyandian, maka plaintext ini disebut sebagai chipertext. Proses penyamaran dari plaintext ke ciphertext disebut sebagai enkripsi (encryption), dan proses pengembalian dari ciphertext menjadi plaintext kembali disebut dekripsi. (Simamora et al., 2022).

II.2.6. *Stream Cipher*

Stream Cipher ditemukan oleh Gilbert Vernam pada tahun 1917, meskipun algoritma saat itu belum bisa disebut dengan demikian pada saat itu. Ia membangun sebuah mesin elektromagnetik yang mana secara otomatis mengenkripsi komunikasi dari mesin ketik teletip. *Plaintext* akan masuk kedalam mesin menjadi sebuah kertas tape, dan key *Stream* sebagai tape kedua. Proses tersebut adalah pertama kalinya antara enkripsi dan transmisi dikerjakan secara otomatis dalam satu mesin.

Perkembangan algoritma kriptografi modern berbasis bit didorong oleh penggunaan komputer digital yang merepresentasikan data dalam bentuk biner. Algoritma kriptografi yang beroperasi dalam mode bit dapat dikelompokkan menjadi dua kategori yaitu block *Cipher* dan *Stream Cipher*. *Stream Cipher* (aliran *Cipher*) merupakan suatu *Cipher* yang berasal dari hasil XOR. Setiap bit *plaintext* dengan dengan setiap bit kunci. Kunci merupakan kunci utama (kunci induk) yang digunakan untuk membangkitkan kunci acak semu yang dibangkitkan dengan Pseudo-Random *Squence* Generator yang merupakan suatu nilai yang nampak seperti di acak, tetapi sesungguhnya nilai tersebut merupakan suatu urutan. Secara khusus urutan dari nilai yang dihasilkan oleh RNG (random

number generator), computational mekanisme *deterministic* atau FSM (finite state machine) merupakan kebalikan dari really random.

Random Number Generato (RNG) secara umum adalah Pseudorandom. Yang memberikan initial state atau seed (nilai yang diinputkan kedalam state), seluruh urutan tersebut ditentukan secara keseluruhan, tetapi meskipun demikian banyaknya karakteristik yang ditampilkan dari suatu urutan yang acak tersebut. Pseudorandomness menghasilkan yang sama secara berulang-ulang pada penempatan yang berbeda. Kemudian kunci acak semu tersebut diberikan operasi XOR dengan *plaintext* untuk mendapatkan *Ciphertext*. *Stream Cipher* rawan terhadap serangan pembalikan bit. Jika penyerang mengetahui bahwa *Stream Cipher* yang digunakan, penyerang mencoba untuk mengedintifikasi posisi bit yang telah dirubah dan mengembalikannya kebentuk aslinya, mereka terlebih dahulu mencari polda dari perubahan bit tersebut untuk mengedintifikasi bentuk asli dari bit tersebut. (Rizky, 2021)

One-time pad (OTP) adalah *stream cipher* yang melakukan enkripsi dan ekripsi satu karakter setiap kali. Algoritma ini ditemukan pada tahun 1917 oleh Major Joseph Mauborgne sebagai perbaikan dari Vernam Cipher untuk menghasilkan keamanan yang sempurna. Mauborgne mengusulkan penggunaan One-Time Pad (pad = kertas bloknot) yang berisi deretan karakterkarakter kunci yang dibangkitkan secara acak. (Sianturi & Yusfrizal, 2020)

Berikut contoh kasus, misalkan panjang *register* = 4 bit dan nilai awal dari masing-masing *register* adalah sebagai berikut:

1. b0 = 1001

2. b1 = 0011
3. b2 = 0001
4. b3 = 0110
5. b4 = 0000
6. b5 = 1000
7. b6 = 1110
8. b7 = 1111

Proses kerja yang dilakukan metode *Stream chiper* adalah sebagai berikut:

1. Untuk menghasilkan output-1, lakukan proses output function berikut.
   a. Gabungkan isi register antara b0 dan b2 dan antara b4 dan b7. Hasil penggabungan register b0 dan b2 = 10010011. Hasil penggabungan register b4 dan b7 = 00010110.
   b. Kalikan hasil penggabungan register. 10010011 x 00010110 = 0000100001111111. c. Didapat Output bit kunci ke-1 adalah byte ketiga dari kiri, yaitu: 00001000
2. Sticky Right Shift 1 bit pada register b1.
   A = SSR(b1)
   = SSR(0011)
   A = 0001
3. Left Shift 1 bit pada register b7.
   B = LS(b7)
   = LS(1111)
   B = 1110

4. Lakukan operasi XOR antara register b0, nilai A dan nilai B.

   C = register b0 XOR A XOR B

   = 1001 XOR 0001 XOR 1110

   C = 0110

5. Geser register ke kanan dengan melakukan langkah-langkah berikut,

   a. Buang isi register b7.

   b. Geser isi register ke kanan.

      b7 = b6 = 1110.

      b6 = b5 = 1000.

      b5 = b4 = 0000.

      b4 = b3 = 0110.

      b3 = b2 = 0001.

      b2 = b1 = 0011.

      b1 = b0 = 1001.

   c. Isi register b0 dengan nilai C.

      b0 = 0110.

   Untuk bit kunci pada proses berikutnya, lakukan proses yang sama. Pada proses pertama, didapatkan bit kunci ke-1 = 00001000 Proses enkripsi pada metode *Stream* chiper Gifford akan menggunakan bit kunci untuk mengubah *plaintext* menjadi *ciphertext*, dan sebaliknya pada proses dekripsi. Proses enkripsi dan dekripsi adalah berupa operasi XOR dari *plaintext* dan kunci untuk menghasilkan *ciphertext* atau operasi XOR *ciphertext* dan kunci untuk menghasilkan *plaintext*

$P = C \oplus K$

$C = P \oplus K$

dengan :

P = *Plaintext*

K = Key

C = *Ciphertext* (Harahap et al., 2021)

II.2.7. *Playfair Cipher*

Playfair Cipher merupakan metoda enkripsi klasik yang sangat sulit untuk dikriptanalisis secara manual. Meskipun demikian *Playfair* dapat dipecahkan dengan menggunakan informasi frekuensi kemunculan bigram. Komponen yang penting pada *algoritma playfair* adalah tabel *Cipher* yang digunakan untuk melakukan enkripsi dan dekripsi tabel bawaan yang diperkenalkan oleh *playfair* adalah tabel yang berbentuk matrik berukuran (5x5) yang berisi huruf kapital dari A-Z dengan menghilangkan J. (Amrulloh, 2021)

Berikut contoh kasus, kunci enkripsinya adalah 25 buah huruf yang disusun dalam bujursangkar 5 x 5 dengan menghilangkan huruf J dari alfabet (dalam beberapa versi lain huruf I dan J ditulis sebagai I/J). Setiap elemen bujursangkar berisi huruf yang berbeda satu sama lain. Contoh sebuah bujursangkar (persegi) *playfair*:

Tabel II.1. Contoh Kunci *Playfair Chiper*

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

***(Sumber:** Amrulloh, 2021)*

Huruf-huruf di dalam bujur sangkar biasanya hasil permutasi huruf-huruf alfabet. Jumlah kemungkinan bujur sangkar yang dapat dibuat adalah sebanyak permutasi dari 25 alfabet, yaitu

25! = 25! = 15.511.210.043.330.985.984.000.000

Susunan huruf didalam bujursangkar juga dapat dipilih dari sebuah kalimat kunci yang mudah diingat, misalnya:

JALAN GANESHA SEPULUH

Buang huruf yang berulang dan huruf J jika ada:

ALNGESHPU

Kemudian tambahkan huruf-huruf alfabet lain yang belum ada (kecuali J):

ALNGESHPUBCDFIKMOQRTVWXYZ

Masukkan huruf-huruf di atas kedalam bujursangkar dari atas kebawah dan dari kiri kekanan sehingga membentuk bujursangkar *playfair* sebagai berikut:

Tabel II.2. Kunci *Playfair Cipher*

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

***(Sumber:** Amrulloh, 2021)*

Sebelum melakukan enkripsi, pesan yang akan dienkripsi diatur terlebih dahulu dengan aturan berikut:

1. Ganti huruf J (bila ada) dengan huruf I.
2. Tulis pesan dalam pasangan huruf (bigram).
3. Tidak boleh ada pasangan huruf yang sama. Jika ada, sisipkan X ditengahnya (atau huruf lain, misalnya Z).
4. Jika jumlah huruf ganjil, tambahkan huruf X pada bigram terakhir. (Munir, 2019:155)

   Misalnya pada plain*teks* temui ibu nanti malam tidak ada huruf J, maka pesan langsung ditulis dalam pasangan huruf (bigram):

   TE MU II BU NA NT IM AL AM Karena ada pasangan huruf yang sama, yaitu II, maka tambahkan huruf X ditengahnya sehingga bigram menjadi:

   TE MU IX IB UN AN TI MA LA MX

Menurut (Nurkifli, 2014:367) terdapat beberapa aturan dalam proses enkripsi maupun dekripsi pada *Playfair* Cipher. Berikut merupakan Langkahlangkah emkripsi dan dekripsi *Playfair* Cipher.

Algoritma enkripsi untuk *Playfair* Cipher adalah sebagai berikut:

1. Jika ada dua huruf terdapat pada baris kunci yang sama maka tiap huruf diganti dengan huruf di kanannya.
2. Jika ada dua huruf terdapat pada kolom yang sama maka tiap huruf diganti dengan huruf di bawahnya.
3. Jika dua huruf tidak pada baris yang sama atau kolom yang sama, maka huruf pertama diganti dengan huruf pada perpotongan baris huruf pertama

dengan kolom huruf kedua.

4. Huruf kedua diganti dengan huruf pada titik sudut keempat dari persegi panjang yang dibentuk dari huruf yang digunakan

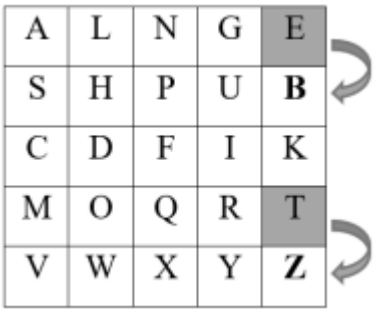
   Contoh: Plain*teks* temui ibu nanti malam telah ditulis dalam bigram kapital sebagai berikut:

   TE MU IX IB UN AN TI MA LA MX

   Dan bujur sangkar *playfair* yang digunakan sama seperti tabel 2.2 Maka *cipherteks* yang dihasilkan adalah:

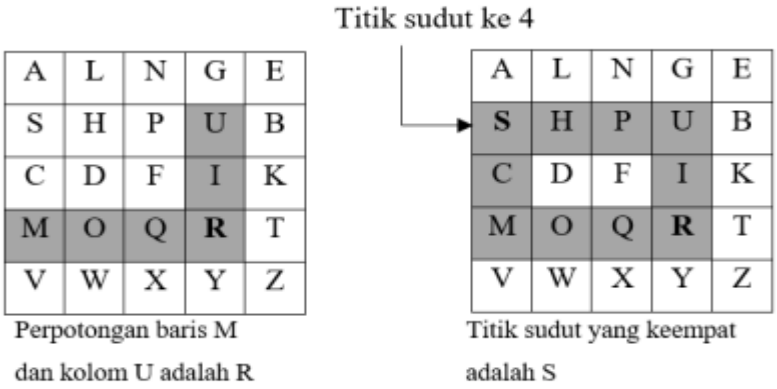
   ZB RS FY KU PG LG RK VS NL QV

   Penjelasannya adalah sebagai berikut: untuk bigram yang pertama, TE, huruf T dan E terletak pada satu kolom.maka huruf T diganti dengan huruf dibawahnya, Z, begitu juga huruf E diganti dengan huruf dibawahnya, B.



Gambar II.3. Contoh Enkripsi Bigram Pada Yang Kolom Sama
***(Sumber:** Amrulloh, 2021)*

Untuk bigram MU, karena tidak terletak pada satu baris atau satu kolom, maka cara enkripsinya ditunjukkan pada bujur sangkar berikut ini:



Gambar II.4. Contoh Enkripsi bigram yang tidak pada baris dan kolom yang sama

***(Sumber:** Amrulloh, 2021)*

Perpotongan baris yang berisi M dengan kolom yang berisi U bertemu pada huruf R. sampai sini kita sidah mempunyai tiga titik sudut yang sudah terbentuk, yaitu M, U, dan R. Titik sudut keempat agar terbentuk persegi Panjang adalah S. Jadi, enkripsi terhadap bigram MU adalah RS. Untuk bigram yang lainnya caranya sama.

Algoritma dekripsi merupakan kebalikan dari algoritma enkripsi. Langkahlangkahnya adalah sebagai berikut:

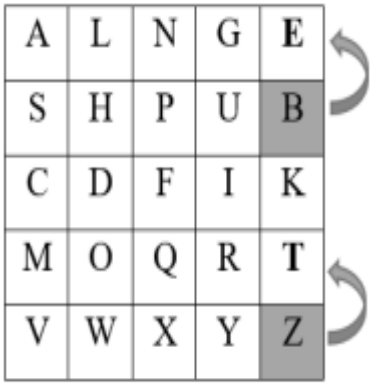
1. Jika ada dua huruf terdapat pada baris kunci yang sama maka tiap huruf diganti dengan huruf di kirinya.
2. Jika ada dua huruf terdapat pada kolom yang sama maka tiap huruf diganti dengan huruf di atasnya.
3. Jika dua huruf tidak pada baris yang sama atau kolom yang sama, maka huruf pertama diganti dengan huruf pada perpotongan baris huruf pertama dengan kolom huruf kedua.
4. Huruf kedua diganti dengan huruf pada titik sudut keempat dari persegi panjang yang dibentuk dari huruf yang digunakan.

Contoh : *Ciphertеks* zbrsfykupglgrkvsnlqv ditulis dalam bigraf kapital sebagai berikut : ZB RS FY KU PG LG RK VS NL QY

Dan bujursangkar *playfair* yang digunakan sama seperti pada Tabel 2.2 Maka *teks* yang dihasilkan adalah:

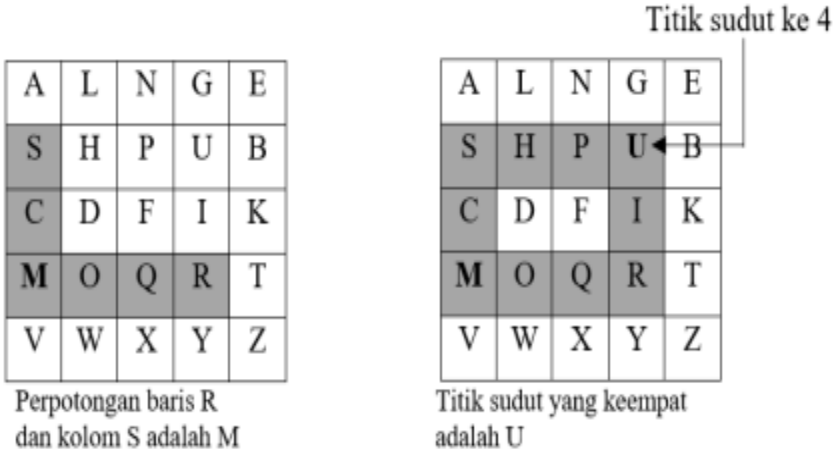
TE MU IX IB UN AN TI MA LA MX

Penjelasannya adalah sebagai berikut: untuk bigram yang pertama, ZB, huruf Z dan B terletak pada satu kolom.maka huruf Z diganti dengan huruf diatasnya, T, begitu juga huruf B diganti dengan huruf diatasnya, E.



Gambar II.5. Contoh Dekripsi Bigram Pada Kolom Yang Sama
***(Sumber:** Amrulloh, 2021)*

Untuk bigram MU, karena tidak terletak pada satu baris atau satu kolom, maka cara enkripsinya ditunjukkan pada bujur sangkar berikut ini:



Gambar II.6. Contoh Dekripsi Bigram yang tidak pada baris dan kolom yang sama
***(Sumber:** Amrulloh, 2021)*

Perpotongan baris yang berisi R dengan kolom yang berisi S bertemu pada huruf M. sampai sini kita sidah mempunyai tiga titik sudut yang sudah terbentuk, yaitu R, S, dan M. Titik sudut keempat agar terbentuk persegi Panjang adalah U. Jadi, enkripsi terhadap bigram RS adalah MU. Untuk bigram yang lainnya caranya sama.

Setelah itu buang huruf X yang tidak mengandung makna. Plain*teks* yang dihasilkan TEMUI IBU NANTI MALAM. (Amrulloh, 2021)

II.2.8. Aplikasi Berbasis *Hybrid*

Menurut Harmadya dkk. (2015:110), "*Hybrid* merupakan kombinasi dari kelebihan yang dimiliki antara aplikasi web dan aplikasi native. Aplikasi *Hybrid* mengkonversi aplikasi web *mobile* HTML5 ke aplikasi native *smartphone* yang ditargetkan."

Hybrid memungkinkan pengembang aplikasi untuk membuat aplikasi *mobile* lintas platform berbasis web dengan menyediakan fitur-fitur native dan menjembatani semua service request dari kode berbasis web ke platform API yang

sesuai (Ilhami, 2017:17). *Hybrid* merupakan gabungan dari aplikasi native dan web. Dibuat dengan menggunakan bahasa pemrograman web (HTML, CSS, dan JavaScript) yang didesain sedemikian rupa agar dapat dijalankan di berbagai platform perangkat seluler. (Melani, 2020)

Dari pendapat di atas, dapat disimpulkan bahwa *Hybrid* adalah kombinasi dari kelebihan yang dimiliki antara aplikasi web dan aplikasi native yang menggunakan bahasa pemograman web (HTML, CSS, JavaScript) memungkinan pengembang membuat aplikasi *mobile* lintas platform ke aplikasi native *smartpho]ne*. (Melani, 2020)

II.2.9. *Hypertext Preprocessor* (PHP)

PHP adalah bahasa pelengkap HTML yang memungkinkan dibuatnya aplikasi dinamis yang memungkinkan adanya pengolahan data dan pemrosesan data. Semua sintax yang diberikan akan sepenuhnya dijalankan pada server sedangkan yang dikirimkan ke *playfair* hanya hasilnya saja. Kemudian merupakan bahasa berbentuk script yang ditempatkan dalam server dan diproses di *server*. Hasilnya akan dikirimkan ke client, tempat pemakai menggunakan *playfair*. PHP dikenal sebagai sebuah bahasa scripting, yang menyatu dengan tag-tag HTML, dieksekusi di server, dan digunakan untuk membuat halaman web yang dinamis seperti halnya *Active Server Pages* (ASP) atau *Java Server Pages* (JSP). PHP merupakan sebuah *software Open Source*. (Hermiati et al., 2021)

II.2.10. *Android*

Android adalah sebuah sistem operasi perangkat *mobile* berbasis *linux*. *Android* merupakan sebuah sistem operasi berbasis java yang beroperasi pada

kernel Linux 2.6. *Android* bukanlah sebuah bahasa pemrograman tetapi *android* merupakan sebuah lingkungan untuk menjalankan aplikasi. (DiMarzio, 2008). *Android* menyediakan platform terbuka/open source bagi para pengembang sehingga menjadikan sistem operasi ini sangat digemari di pasaran. Sebagian besar vendor *smartphone* yang diproduksi adalah berbasis *android*. Hal ini juga yang menjadikan banyak pengembang mulai mengembangkan aplikasi berbasis *android*. Berdasarkan pengertian *android* yang telah di uraikan diatas maka peneliti menyimpulkan *android* adalah perangkat *mobile* berbasis java dimana *android* adalah tempat untuk menjalankan aplikasi dan menyediakan platform terbuka/open source bagi para pengembang. (Aulianti et al., 2021)

II.2.11. *Android Studio*

Android Studio adalah sebuah IDE, atau lingkungan pengembangan aplikasi *android*, berdasarkan intel IDEA. *Android Studio* (AS) menyediakan fitur canggih dan perbaikan serta peningkatan kecepatan IDE dari IDE sebelumnya yang bernama *Eclipse Android Development Tool* (ADT) dan *Android Studio* menjadi IDE *Android* yang sudah resmi dari Google. *Android studio* ini gratis, dan para developer bebas menggunakannya, AS dirancang untuk pengembangan aplikasi *android*. Bahasa yang dipakai di *Android Studio* ialah bahasa pemograman *java.* (Primatama et al., 2021)

II.2.12. *Sublime Text*

Sublime Text merupakan perangkat lunak text editor yang di gunakan untuk membuat atau mengedit suatu aplikasi. Sublime Text memiliki *plugin* tambahan yang memudahkan programmer. Selain itu sublime text juga memiliki

desain yang simple dan keren sehingga terlihat elegan untuk sebuah *syntax editor*. (Fitria, 2021)

II.2.13. *Framework*

Framework yaitu kumpulan fungsi-fungsi dasar atau perintah yang biasa digunakan dalam mengembangkan suatu *sotfware*, dengan harapan agar *software* yang dibangun menjadi lebih cepat dan terstruktur. (Devianty et al., 2021)

II.2.14. MySQL

MySQL adalah sebuah *database* atau media penyimpanan data yang mendukung script PHP. MySQL juga mempunyai query atau bahasa SQL (*Structured Query Language*) yang simpel dan menggunakan escape character yang sama dengan PHP, selain itu MySQL adalah *database* tercepat saat ini**.** (Fitria, 2021)

II.2.15. *Unified Modeling Language* (UML)

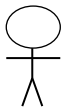
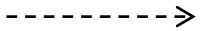
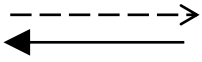
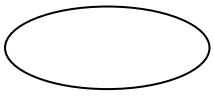
UML yaitu satu kumpulan konvensi pemodelan yang digunakan untuk menentukan atau menggambarkan sebuah sistem perangkat lunak yang terkait dengan obbjek. UML merupakan suatu kumpulan teknik terbaik yang telah terbukti sukses dalam memodelkan system yang besar dan kompleks. UML tidak hanya digunakan dalam proses pemodelan perangkat lunak, namun hampir dalam semua bidang yang membutuhkan pemodelan. (Andikos, 2019)

Alat bantu yang digunakan untuk perancangan berorentasi objek berbasiskan UML adalah sebagai berikut:

1. *Use Case* Diagram

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah apa yang diperbuat sistem, dan bukan "bagaimana". Sebuah use case merepresentasikan sebuah interaksi antara aktor dengan sistem. use case diagram dapat digambarkan dengan sumber pada Tabel II.3.

Table II.3. Simbol *Use Case*

Gambar	NAMA	Keterangan
	Actor	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan use case.
	*Depedency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (independent) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (*independent*).
	Generalization	Hubungan dimana objek anak (descendent) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (ancestor).
	Include	Menspesifikasikan bahwa use case sumber secara eksplisit.
	*Extend*	Menspesifikasikan bahwa use case target memperluas perilaku dari use case sumber pada suatu titik yang diberikan .
	Association	Apa yang mengubungkan antara objek satu dengan objek lainnya.
	*System*	Menspesifikan paket yang menampilkan sistem secara terbatas.
	*Use Case*	Deskripsi dari urutan aksi-aksi ynag ditampilkan sistem yang menghasilkan suatu hasil yang terukur agi suatu actor.

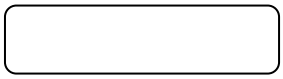
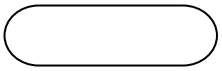
	*Collaboration*	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan prilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
	*Note*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi.

(*Sumber* : *Andikos, 2019 : 39*)

2. Diagram Aktivitas (*Activity Diagram*)

Activity diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir, *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. *Activity diagram* dapat digambarkan dengan simbol-simbol pada tabel II.4.

Tabel II.4. Simbol *Activity Diagram*

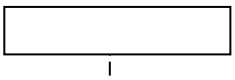
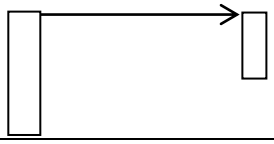
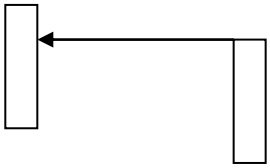
Gambar	Nama	Keterangan
	*Activity*	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain.
	*Action*	State dari sistem yang mencerminkan eksekusi dari suatu aksi.
	*Initial Node*	Bagaimana objek dibentuk atau diawali.
	*Activity Final*	Bagaimana objek dibentuk dan dihancurkan
	*Fork Node*	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran

(*Sumber: Andikos, 2019: 39*)

3. Diagram Urutan (*Sequence Diagram*)

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, display, dan sebagainya) berupa message yang diambarkan terhadap waktu. Sequence Diagram dapat digambarkan dengan simbol-simbol seperti pada Tabel II.5.

Tabel II.5. Simbol *Sequence Diagram*

Gambar	Nama	Keterangan
	*Lifeline*	Objek entity, antarmuka yang saling berinteraksi.
	*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi.
	*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi.

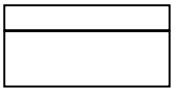
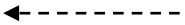
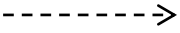
(*Sumber : Andikos, 2019: 39*)

4. *Class Diagram* (Diagram Kelas)

Class adalah sebuah V yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desai berorientasi objek. Class diagram menggamarkan struktur dan deskripsi class, package dab objek beserta hubungan satu sama lain seperti containment, pewarisan, asosiasi, dan lain-lain. Class diagram dapat digambarkan dengan simbol-simbol seperti pada Tabel II.6.

Tabel II.6. *Class Diagram*

Gambar	Nama	Keterangan
	Generalization	Hubungan dimana objek anak (descendent) erbagai perilaku dan struktur data dari objek yang ada di atasnya objek induk (ancestor).

	*Nary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
	*Class*	Himpunan dari objek-objek yang berbagai atribut serta operasi yang sama.
	*Collaboration*	Deskripsi dari urutan aksi-aksi yang ditampilkan system yang menghasilkan suatu hasil yang terukur bagi suatu actor
	*Realization*	Operasi yang benar-benar dilakukan oleh suatu objek.
	*Depedency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (independent) akan mempengarui elemen yang bergantung padanya elemen yang tidak mandiri
	Assocation	Apa yang menghubungkan antara objek satu dengan objek lainnya.

(*Sumber: Andikos*, 2019: 39)