BAB II

TINJAUAN PUSTAKA

II.1. Penelitian Terkait

Berdasarkan penelitian yang dilakukan oleh Sinaga dan Mesran (2017) mengenai Implementasi Algoritma ROT13 Dan Algoritma *Caesar Cipher* Dalam Penyandian Teks, Sinaga dan Mesran menyimpulkan bahwa Proses penyandian teks pada *file* teks ini merupakan salah satu cara untuk menjaga keaslian data teks dari para kriptanalis, sehingga para kriptanalis akan sulit dan membutuhkan waktu yang lama untuk mengetahui teks asli/*plaintext*.

Berdasarkan penelitian yang dilakukan oleh Akhyar, dkk (2017) mengenai *Analysis Stego-Image Extraction Using* ROT13 *and Least Significant Bit* (LSB) *Algorithm Method on Text Security*, Akhyar, dkk menyimpulkan bahwa penggunaan teknk kriptografi menggunakan algoritma ROT13 memperkuat penyembunyian pesan teks ke dalam gambar.

Berdasarkan penelitian yang dilakukan oleh Siregar (2016) mengenai Implementasi Keamanan Data Teks Dengan Algoritma GOST Dan ROT13, Siregar menyimpulkan bahwa kombinasi algoritma GOST dan ROT13 menjadikan keamanan data teks memilliki keamanan yang sangat kuat.

Berdasarkan penelitian yang dilakukan oleh Aulia, dkk (2019) mengenai Penerapan Algoritma *One Time Pad* & *Linear Congruential Generator* Untuk Keamanan Pesan Teks, Aulia, dkk menyimpulkan bahwa tingkat kemanan pesan dapat dikatakan cukup karena menggunakan dua algoritma.

Berdasarkan penelitian yang dilakukan oleh Jumeidi, dkk (2016), mengenai Implementasi Algoritma Kriptografi *Vernam Cipher* Berbasais FPGA, Jumeidi, dkk menyimpulkan bahwa proses enkripsi dan dekripsi dilakukan dengan menggunakan implementasi sistem dari rancangan sistem yang sama.

Berdasarkan penelitian yang dilakukan oleh Sari, dkk (2016) mengenai Penyembunyian Data Untuk Seluruh Ekstensi *File* Menggunakan Kriptografi *Vernam Cipher* Dan *Bit Shiffting*, Sari, dkk menyimpulkan bahwa gabungan algoritma ini berhasil melakukan proses enkripsi dan dekripsi dengan baik.

Berdasarkan penelitian yang dilakukan oleh Sihombing (2019) mengenai Penerapan Algoritma *Vernam Cipher* (*One Time*) Untuk Pengamanan *Login*, Sihombing menyimpulkan bahwa aplikasi dapat mengacak dan menyembunyikan *file* dengan aman dan tidak menimbulkan kecurigaan pada pihak lain.

Berdasarkan penelitian yang dilakukan oleh Farid, dkk (2016) mengenai Implementasi algoritma Steganografi *Least Significant Bit* Dengan *Algoritma Hill Cipher* Pada Citra *Bit*map, Farid, dkk menyimpulkan bahwa Penggabungan metode kriptografi *Hill Cipher* dan steganografi *Least Significant Bit* (LSB) dapat meningkatkan keamanan pada pesan dalam bertukar informasi.

Berdasarkan penelitian yang dilakukan oleh Harjo, dkk (2016) mengenai Aplikasi Steganografi Menggunakan *Least Significant Bit* dan Enkripsi *Triple* DES Menggunakan Bahasa Pemrograman C#, Harjo, dkk menyimpulkan bahwa Hasil dari penerapan untuk penyisipan pesan rahasia pada gambar berjalan dengan baik. Pesan atau dokumen yang disisipkan pada *file* gambar dapat diperoleh kembali secara utuh atau dengan kata lain pesan yang disisipkan sebelum proses

enkripsi dan setelah proses dekripsi mempunyai hasil yang sama tanpa ada perubahan pesan atau gangguan.

II.2. Landasan Teori

Berikut ini adalah beberapa landasan teori yang berkaitan dengan penelitian yang peneli buat :

II.2.1. Kriptografi

Kriptografi adalah merupakan ilmu yang mempelajari teknik-teknik matematika yang berhubungan dengan aspek keamanan informasi seperti kerahasiaan, integritas data serta otentikasi. Kriptografi adalah proses penggunaan berbagai teknik dan atau ilmu dan seni untuk menjaga keamanan pesan. *Cryptographic algorithm* adalah fungsi matematika yang digunakan untuk enkripsi dan dekripsi.Terdapat dua fungsi yang saling berhubungan yaitu satu untuk enkripsi dan satu lagi untuk dekripsi. Enkripsi merupakan proses pengkodean sebuah pesan sehingga isi dari pesan tersebut tidak diketahui. Dekripsi adalah proses kebalikan dari enkripsi yaitu mentransformasi pesan yang dienkripsi kembali menjadi bentuk semula. Sebuah sistem enkripsi dan dekripsi disebut *cryptosystem*. Bentuk asli dari sebuah pesan disebut *plaintext* dan bentuk asli yang dienkripsi disebut *ciphertext*. (Fauzi, dkk, 2017 : 2).

II.2.1.1. Algoritma Kriptografi

Algoritma kriptografi merupakan langkah-langkah logis bagaimana menyembunyikan pesan dari orang-orang yang tidak berhak atas pesan tersebut. Algoritma kriptografi terdiri dari tiga fungsi dasar yaitu :

1. Enkripsi merupakan hal yang sangat penting dalam kriptografi, merupakan pengamanan data yang dikirimkan agar terjaga kerahasiannya. Pesan asli disebut *plaintext,* yang diubah menjadi kode-kode yang tidak dimengerti. Enkripsi bisa diartikan dengan cipher atau kode. Untuk mengubah teks asli ke bentuk teks kode digunakan algoritma yang dapat mengkodekan data.
2. Dekripsi merupakan kebalikan dari enkripsi. Pesan yang telah dienkripsi dikembalikan ke bentuk asalnya (teks asli/*plaintext*) disebut dengan dekripsi.
3. Kunci yang dipakai untuk melakukan enkripsi dan dekripsi. Kunci terbagi menjadi dua bagian yaitu kunci rahasia (*private key*) dan kunci umum (*public key*). (Fauzi, dkk, 2017 : 2).

II.2.1.2. Notasi Algoritma Kriptografi

Biasanya algoritma kriptografi dapat dinotasikan sebagai berikut :

1. *Plaintext*(M)
2. *Ciphertext*(C)
3. Enkripsi (fungsi E)
4. Dekripsi (fungsi D)

Kriptografi itu sendiri terdiri dari dua proses utama yakni proses enkripsi dan proses dekripsi. Seperti yang telah dijelaskan di atas, proses enkripsi mengubah

plaintext menjadi *ciphertext* (dengan menggunakan kunci tertentu) sehingga isi informasi pada pesan tersebut sukar dimengerti. (Fauzi, dkk, 2017 : 3).

II.2.2. Steganografi

Steganografi adalah teknik penyembunyian data dalam sebuah medium yang dapat berupa jenis data apapun seperti *file* citra gambar, *audio*, *video,* maupun jenis data yang lainnya. (Farid, dkk, 2016 : 110).

Steganografi adalah ilmu seni menulis atau menyembunyikan pesan ke dalam sebuah media sehingga keberadaan pesan tidak dapat diketahui atau tidak disadari oleh indera manusia. Teknik steganografi berfungsi menyembunyikan pesan rahasia di dalam media digital sehingga keberadaan data rahasia tersebut tidak diketahui oleh orang lain. Steganografi membutuhkan dua bagian yang sangat penting yaitu berkas atau media penampung dan pesan rahasia yang akan disembunyikan. Penggunaan steganografi adalah untuk menyamarkan keberadaan data rahasia sehingga sulit dideteksi, dan juga dapat melindungi hak cipta dari suatu produk. Pesan rahasia yang disembunyikan dapat diungkapkan kembali sama seperti aslinya. (Wahab, 2017 : 203).

Pada dasarnya, terdapat tujuh teknik yang digunakan dalam steganografi :

1. *Injection,* Merupakan suatu teknik menanamkan pesan rahasia secara langsung ke suatu media.Salah satu masalah dari teknik ini adalah ukuran media yang diinjeksi menjadi lebih besar dari ukuran normalnya sehingga mudah dideteksi.Teknik itu sering juga disebut *Embedding*.
2. *Substitusi,* Data normal digantikan dengan data rahasia, Biasanya, hasil teknik itu tidak terlalu mengubah ukuran data asli, tetapi tergantung pada

file media dan data yang akan disembunyikan. Teknik *subsitusi* bias menurunkan kualitas media yang ditumpangi.

3. *Transform Domain,* Teknik ini sangat efektif. Pada dasarnya, *transformasi domain* menyembunyikan data pada *"transform space*". Akan sangat lebih efektif bila teknik ini diterapkan pada *file* berekstensi jpeg (gambar).
4. *Spread Spectrum,* Sebuah teknik pentransmisian menggunakan *pseudo-noise code*, yang independen terhadap data informasi sebagai modulator bentuk gelombang untuk menyebarkan energy sinyal dalam sebuah jalur komunikasi informasi. Oleh penerima, sinyal dikumpulkan kembali menggunakan *relika pseudo-noise code* tersinkronisasi.
5. *Statistic Method*, Teknik ini disebut juga skema steganografi 1 *bit*. Skema tersebut menanamkan satu *bit* informasi pada media tumpangan dan mengubah statistic walaupun hanya 1 *bit*. Perubahan statistic ditunjukan dengan indikasi 1 dan jika tidak ada perubahan. Terlihat indikasi 0. Sistem ini bekerja berdasarkan kemampuan penerima dalam membedakan antara informasi uang dimodifikasi dan yang belum.
6. *Distortion,* Metode ini menciptakan perubahan atas benda yang ditumpangi oleh data rahasia.
7. *Cover Generation,* Metode ini lebih unik daripada metode lainnya Karena*cover object* dipilih untuk menyembunyikan pesan. Contoh dari metode ini adalah *spam mimic*.(Wahab, 2017 : 203).

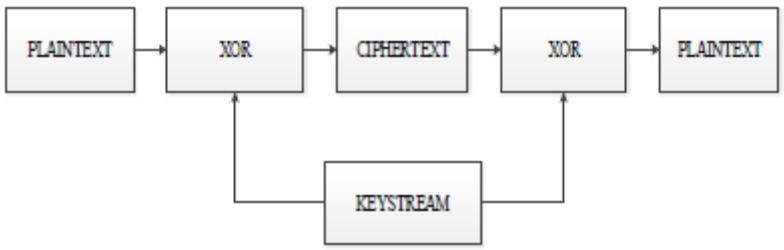
II.2.3. Algoritma *Vernam Cipher*

One Time Pad termasuk salah satu algoritma yang memiliki kesempurnaan

saat enkripsi dan dekripsinya. *One Time Pad* yang dikenal dengan algoritma *Vernam*, ditemukan oleh Gillbert Vernam di Major Joseph Mauborge and AT & T's. Konsep dasar algoritma *One Time Pad* hampir sama dengan algoritma *Vigenere*, hanya saja pada algoritma ini, kunci dibangkitkan secara acak, dan kecil kemungkinan kunci akan saling sama satu dengan lainnya. (Harahap, 2016 : 63).

Vernam cipher merupakan algoritma kriptografi yang ditemukan oleh Mayor J. Maugborne dan G. Vernam. Algoritma *vernam cipher* diadopsi dari *one time pad cipher*, dimana dalam hal ini karakter diganti dengan *bit* (0 atau 1). Dengan kata lain, vernam *cipher* merupakan versi lain dari *one-time pad cipher*. Algoritma kriptografi vernam *cipher* merupakan algoritma kriptografi berjenis *symmetric key*. Kunci yang digunakan untuk melakukan enkripsi dan dekripsi menggunakan kunci yang sama. Dalam melakukan proses enkripsi, algoritma vernam *cipher* menggunakan cara*stream cipher* dimana *cipher* berasal dari hasil operasi XOR antara *bitplainteks* dan *bit key*.

Pada *cipher* aliran, *bit* hanya mempunyai dua buah nilai, sehingga proses enkripsi hanya menyebabkan dua keadaan pada *bit* tersebut, yaitu berubah atau tidak berubah. Dua keadaan tersebut ditentukan oleh kunci enkripsi yang disebut dengan aliran-*bit*-kunci (*keystream*). Secara sederhana proses enkripsi dan dekripsi algoritma vernam *cipher* dapat adalah pada Gambar II.2.



Gambar II.2. Proses Enkripsi Dan Dekripsi Algoritma Kriptografi *Vernam Cipher*
(Juneidi, dkk, 2016 : 22)

II.2.4. Algoritma ROT13

ROT13 (*Rotate* 13) adalah enkripsi *substitution cipher* yang umum digunakan di sistem operasi UNIX. Pada sistem enkripsi ROT13 sebuah huruf digantikan dengan huruf yang letaknya di atas 13 posisi darinya. Kelebihan dari algoritma ROT13 terletak pada enkripsi yang sederhana, yaitu huruf digantikan dengan huruf yang letaknya di atas 13 posisi darinya, sehingga tidak perlu dibentuknya sebuah kunci. Kekurangan dari algoritma ROT13 berdasarkan kelebihannya sendiri. Huruf digantikan dengan huruf yang letaknya di atas 13 posisi darinya, sehingga untuk mengembalikan pesan asli, maka hanya tinggal menggantikan ulang di bawah posisi 13 darinya. (Sinaga dan Mesran, 2017 : 38).

Enkrip algoritma ROT13 menggunakan penjumlahan *ascii plaintext* dengan *ascii* kunci 13 sebagai berikut :

$C_i = P_i + 13$

Keterangan :

C_i = *Ciphertext* hasil enkrip

P_i = *Plaintext* yang akan dienkrip

Dekrip algoritma ROT13 menggunakan pengurangan *ascii ciphertext* dengan *ascii* kunci 13 sebagai berikut :

$P_i = C_i - 13$

Keterangan :

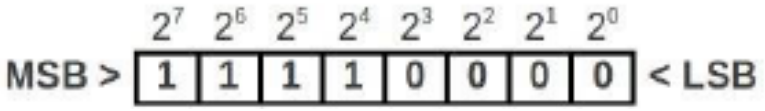
C_i = *Ciphertext* yang akan didekrip

P_i = *Plaintext* hasil dekrip. (Sinaga dan Mesran, 2017 : 39).

II.2.5. *Least Signofocant Bit* (LSB)

Least Significant Bit (LSB) merupakan metode stegnografi yang paling sederhana dan mudah untuk diimplementasikan ke sebuah aplikasi. Metode ini menggunakan citra digital sebagai *convertext*. Pada susunan *bit* di dalam sebuah byte (1 byte = 8 *bit*), ada *bit* yang paling berarti (*most significant bit atau MSB*) dan *bit* yang paling kurang berarti (*least significant bit atau* LSB). (Harjo, dkk, 2016 : 14).

Bit atau *binary digit* adalah unit dasar penyimpanan data di dalam komputer, nilai *bit* suatu data adalah 0 atau 1. Semua data yang ada pada komputer disimpan ke dalam suatu *bit* ini, termasuk gambar, suara ataupun video. Format pewarnaan didalam media gambar, seperti *grayscale*, RGB dan CMYK, juga menggunakan satuan *bit* ini dalam penyimpanannya. Sebagai contoh pewarnaan *monochromebitmap* (menggunakan 1 *bit* untuk tiap pikselnya), RGB – 24 *bit* (8 *bit* untuk Red, 8 *bit* untuk *Green*, dan 8 *bit* untuk *Blue*), *Grayscale*-8 *bit* (menentukan tingkat kehitaman suatu *pixel* berdasarkan nilai *bit*nya). (Wahab, 2017 : 205).



Gambar II.1. Daerah Penyisipan LSB
(Sumber : Wahab, 2017 : 205)

II.2.6. Aplikasi

Aplikasi adalah penerapan dari rancang sistem untuk mengolah data yang menggunakan aturan atau ketentuan bahasa pemrograman tertentu.Aplikasi adalah Program yang dibuat oleh manusia yang berfungsi untuk menyelesaikan permasalahan-permasalahan masalah yang akan dihadapi. (Zulfauzi, 2015 : 57).

II.2.7. *Visual Basic* 2010

Visual Basic 2010 merupakan versi perbaikan dan pengembangan dari versi pendahulunya, yaitu *Visual Basic 2008*. Beberapa pengembangan yang terdapat didalamnya antara lain dukungan terhadap *library* terbaru dari *Microsoft*, yaitu .*Net FrameWork* 4.0, dukungan terhadap aplikasi berbasis *Cloud Computing*, serta perluasan dukungan terhadap *database-database*, baik *standalone* maupun *database server*. (Sari, dkk, 2015 : 2).

II.2.8. *Unitified Modeling Language* (UML)

Unitified Modeling Language (UML) adalah bahasa spesifik standar yang dipergunakan untuk mendokumentasikan, menspesifikan dan membangun perangkat lunak. *Unitified Modeling Language* (UML) merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem. Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan *Unitified Modeling Language* (UML)

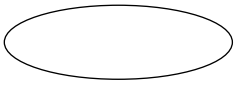
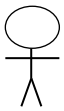
yaitu *Use Case Diagram*, *Activity Diagram*, *Class Diagram* dan *Sequence Diagram*. (Urva dan Siregar, 2015 : 95).

Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan UML adalah sebagai berikut :

1. *Use Case* Diagram

Use case diagram merupakan pemodelan untuk kelakukan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Dapat dikatakan *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Simbol-simbol yang digunakan dalam *use case* diagram dapat dilihat pada tabel II.1 dibawah ini:

Tabel II.1. Simbol *Use Case*

Gambar	Keterangan
	*Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukar pesan antar unit dengan aktor, dan dinyatakan dengan menggunakan kata kerja di awal nama *use case*.
	Aktor adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan aktor, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. Orang atau sistem bisa muncul dalam beberapa peran. Perlu dicatat bahwa aktor berinteraksi dengan *use case*, tetapi tidak memiliki *control* terhadap *use case*.
	Asosiasi antara aktor dan *use case*, digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengidikasikan aliran data.
	Asosiasi antara aktor dan *use case* yang menggunakan panah terbuka untuk mengidinkasikan

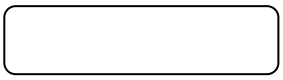
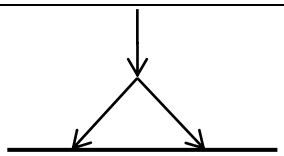
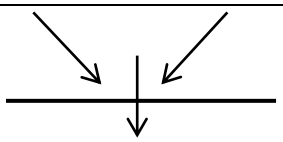
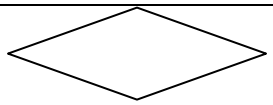
	bila aktor berinteraksi secara pasif dengan sistem.
	Include, merupakan di dalam *use case* lain (*required*) atau pemanggilan *use case* oleh *use case* lain, contohnya adalah pemanggilan sebuah fungsi program.
	Extend, merupakan perluasan dari *use case* lain jika kondisi atau syarat terpenuhi.

(Sumber: Urva dan Siregar, 2015 : 94)

2. Diagram Aktivitas (*Activity Diagram*)

Activity Diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Simbol-simbol yang digunakan dalam *activity diagram* dapat dilihat pada tabel II.2 dibawah ini:

Tabel II.2. Simbol *Activity Diagram*

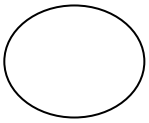
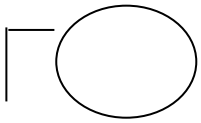
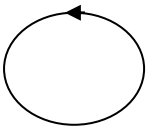
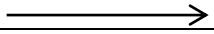
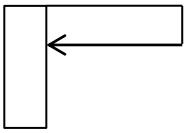
Gambar	Keterangan
	*Start point*, diletakkan pada pojok kiri atas dan merupakan awal aktifitas.
	*End point*, akhir aktifitas.
	*Activites*, menggambarkan suatu proses/kegiatan bisnis.
	*Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan pararel menjadi satu.
	*Join* (penggabungan) atau rake, digunakan untuk menunjukkan adanya dekomposisi.
	*Decision Points*, menggambarkan pilihan untuk pengambilan keputusan, *true*, *false.*
	*Swimlane*, untuk menunjukkan siapa melakukan apa.

(Sumber : Urva dan Siregar, 2015 : 94)

3. Diagram Urutan (*Sequence Diagram*)

Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. Simbol-simbol yang digunakan dalam *sequence diagram* dapat dilihat pada tabel II.3 dibawah ini :

Tabel II.3. Simbol *Sequence Diagram*

Gambar	Keterangan
	*EntityClass*, merupakan bagian dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran awal sistem dan menjadi landasan untuk menyusun basis data.
	*Boundary Class*, berisi kumpulan kelas yang menjadi *interface* atau interaksi antara satu atau lebih aktor dengan sistem, seperti tampilan formentry dan *form* cetak.
	*Control class*, suatu objek yang berisi logika aplikasi yang tidak memiliki tanggung jawab kepada entitas, contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai objek.
	*Message*, simbol mengirim pesan antar *class*.
	*Recursive*, menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri.
	*Activation*, *activation* mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivitas sebuah operasi.
	Lifeline, garis titik-titik yang terhubung dengan objek, sepanjang *lifeline* terdapat *activation*.

(Sumber : Urva dan Siregar, 2015 : 95)

4. *Class Diagram* (Diagram Kelas)

Merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggng jawab entitas yang menentukan perilaku sistem.*Class diagram* juga menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan *constraint* yang berhubungan dengan objek yang dikoneksikan. *Class diagram* secara khas meliputi: Kelas (*Class*), Relasi, *Associations*, *Generalization* dan *Aggregation*, Atribut (*Attributes*), Operasi (*Operations/Method*), *Visibility*, tingkat akses objek eksternal kepada suatu operasi atau atribut. Hubungan antar kelas mempunyai keterangan yang disebut dengan *multiplicity* atau kardinaliti yang dapat dilihat pada tabel II.4 dibawah ini:

Tabel II.4. *Multiplicity Class Diagram*

Multiplicity	Penjelasan
1	Satu dan hanya satu
0..*	Boleh tidak ada atau 1 atau lebih
1..*	1 atau lebih
0..1	Boleh tidak ada, maksimal 1
n..n	Batasan antara. Contoh 2..4 mempunyai arti minimal 2 maksimum 4

(Sumber : Urva dan Siregar, 2015 : 95)