BAB II
TINJAUAN PUSTAKA

II.1. Penelitian Terkait

Dari penelitian berkaitan dengan penerapan enkripsi yang diteliti oleh Irma Wahyudi pada tahun 2021 dengan judul "Implementasi Algoritma Massey-Omura Cryptosystem Dalam Pengamanan Dokumen" dengan kesimpulan dari penelitian ini adalah sistem ini mampu menampilkan hasil yang diinginkan sesuai dengan keamanan data atau bidang kriptografi bagian algoritma simetris. Sistem ini dapat memberikan laporan hasil dengan tingkatan dari proses enkripsi dan dekripsi sesuai dengan keamanan dokumen yang dirancang sesuai dengan sistem yang digunakan.

Penelitian lain juga dilakukan oleh Eko Aribowo pada tahun 2018 dengan judul "Aplikasi Pengamanan Dokumen Office Dengan Algoritma Kriptografi Kunci Asimetris Elgamal" Implementasi program ini menghasilkan suatu aplikasi yang dapat mengubah isi suatu dokumen (plainteks) yang berupa teks, tabel dan gambar menjadi kode-kodeyang tidak dikenal (cipherteks). Mengembalikan isi dari dokumen dari kode-kode (cipherteks) menjadi dokumen aslinya (plainteks).

Ada penelitian lain yang judul "Penerapan Algoritma Massey-Omura Cryptosystem Dalam Pengkodean Sandi" penelitinya adalah Roziman pada tahun 2019. Implementasi program ini menghasilkan suatu sistem yang dapat memberikan laporan hasil dengan tingkatan dari proses enkripsi dan dekripsi sesuai dengan keamanan dokumen yang dirancang sesuai dengan sistem yang digunakan.

II.2. Landasan Teori

II.2.1. Aplikasi

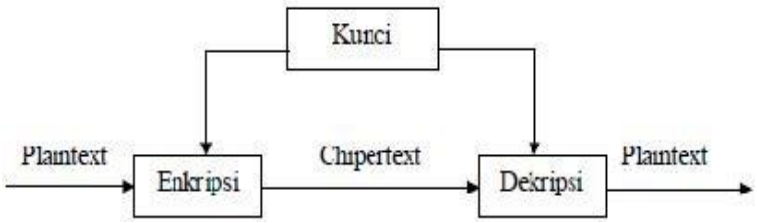
Secara istilah pengertian aplikasi adalah suatu program yang siap untuk digunakan yang dibuat untuk melaksanakan suatu fungsi bagi pengguna jasa aplikasi serta penggunaan aplikasi lain yang dapat digunakan oleh suatu sasaran yang akan dituju. Menurut kamus Komputer Eksekutif, aplikasi mempunyai arti yaitu pemecahan masalah yang menggunakan salah satu tehnik pemrosesan data aplikasi yang biasanya berpacu pada sebuah komputansi yang diinginkan atau diharapkan maupun pemrosesan data yang di harapkan. Pengertian aplikasi menurut Kamus Besar Bahasa Indonesia, "Aplikasi adalah penerapan dari rancang sistem untuk mengolah data yang menggunakan aturan atau ketentuan bahasa pemrograman tertentu". (Andi Juansyah, 2015)

II.2.2. 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.*

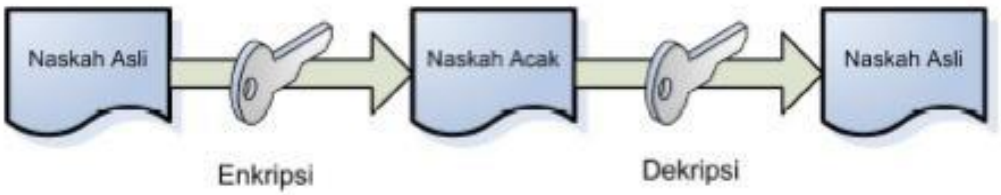
Masalah keamanan merupakan salah satu aspek penting dari sebuah sistem informasi. Salah satu hal penting dalam komunikasi menggunakan komputer dan dalam jaringan komputer untuk menjamin keamanan pesan, data ataupun informasi adalah enkripsi. Enkripsi dapat diartikan sebagai sebuah proses yang dilakukan 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 plainteks menjadi cipherteks, sedangkan dekripsi adalah proses mengembalikan cipherteks menjadi plainteks semula. Enkripsi dan dekripsi membutuhkan kunci sebagai parameter yang digunakan untuk transformasi (Yusfrizal, 2019).



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 II.2.



Gambar II.2. Proses Enkripsi Dan Deskripsi
(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).

Berikut akan dijelaskan jenis – jenis kunci dalam kriptografi modern adalah sebagai berikut:

1. Algoritma Simetris, adalah algoritma yang menggunakan kunci yang sama untuk enkripsi dan dekripsinya. Algoritma kriprografi simetris sering disebut algoritma kunci rahasia, algoritma kunci tunggal, atau algoritma satu kunci, dan mengharuskan pengirim dan penerima menyetujui suatu kunci tertentu. Kelebihan dari algoritma kriprografi simetris adalah waktu proses untuk enkripsi dan dekripsi relatif cepat. Hal ini disebabkan efesiensi yang terjadi pada pembangkit kunci. Karena prosesnya relatif cepat maka algoritma ini tepat untuk digunakan pada sistem komunikasi digital secara *real time* seperti GSM. Aplikasi dari algoritma simetris digunakan oleh beberapa algoritma seperti Data Encryption Standard (DES), Advance Encryption Standard (AES), International Data Encryption Algoritma (IDEA), A5, dan lain – lain.
2. Algoritma Asimetris, adalah pasangan kunci kriptografi yang salah satunya digunakan untuk proses enkripsi dan satu lagi lagi deskripsi. Semua orang yang mendapatkan kunci publik dapat menggunakannya untuk mengenkripsi suatu pesan, sedangkan hanya satu orang saja yang memiliki rahasia itu, yang dalam hal ini kunci rahasia, untuk melakukan pembongkaran terhadap kode yang dikirim untuknya. Contoh algoritma terkenal yang menggunakan kunci asimetris adalah RSA (merupakan singkatan dari nama penemunya, yakni Rivest, Shamir dan Adleman).

3. Algoritma Hibrida, adalah algoritma yang memanfaatkan dua tingkatan kunci, yaitu kunci rahasia (simetri) – yang disebut juga *session key* (kunci sesi) untuk enkripsi data dan pasangan kunci rahasia – kunci publik untuk pemberian tanda tangan digital serta melindungi kunci simetri.

II.2.3. *Microsoft Word*

Menurut Agung pada tahun 2010, Microsoft word adalah sebuah program atau aplikasi pengolah kata yang digunakan untuk membantu pekerjaan manusia di perangkat komputer. *Software* keluaran perusahaan teknologi ternama dunia itu ada sejak lama dan masih eksis hingga sekarang. Di dalamnya user aka menemukan berbagai menu yang masing-masing mempunyai fungsi dalam proses pengolahan kata.

II.2.4. *Massey-Omura*

Algoritma *Massey-Omura* adalah algoritma kriptografi modern yang menggunakan sistem kriptografi asimetri (kunci privat). Algoritma *Massey-Omura* diusulkan oleh *James Massey* dan *Jim K. Omura* pada tahun 1982 ini, melakukan pemfaktoran bilangan yang sangat besar untuk mendapatkan kunci privat. Oleh karena alasan tersebut, *Massey-Omura* dianggap aman. Selain itu, *Massey-Omura* adalah salah satu algoritma yang pengimplementasiannya menggunakan *three pass protocol* (Winton, 2007:10).

Cara kerja dari algoritma *Massey-Omura* yaitu semua pengguna telah mensepakati kelompok batasan atas bidang tetap batasan *Fp* dengan *p* sebagai

kekuatan utama. Setiap pengguna secara rahasia memiliki acak bilangan bulat seperti :

$0 \leq e < p - 1$ Menghitung

GCD (e, p – 1) = 1

Dan menghitung

$d = e^{-1} \bmod (p -1)$

Keterangan algoritma Massey-Omura adalah:

1.	p adalah bilangan prima	(tidak
2.	eA, eB (kunci enkripsi)	(rahasia)
3.	$dA \equiv eA^{-1} \pmod{p - 1}$ (kunci dekripsi sender)	(rahasia)
4.	$dB \equiv eB^{-1} \pmod{p - 1}$ (kunci dekripsi receiver)	(rahasia)
5.	m (plainteks)	(rahasia)
6.	C1, C2, C3 (cipherteks)	(tidak rahasia)

Berikut ini cara kerja dari Algoritma *Massey- Omura* :

1. Semua pengguna telah mensepakati kelompok batasan atas bidang tetap batasan *Fp* dengan *p*sebagai kekuatan utama.
2. Setiap pengguna secara rahasia memilih acak bilangan bulat *e* antara *p-1* seperti GCD (*e,p-1)=1,* dan menghitung $d=e^{-1}$ mod *(p-1)* dengan menggunakan Algoritma *euclidean.*
3. Sekarang anggaplah bahwa *Alice* ingin mengirim pesan *M* yang aman untuk Bob, kemudian mereka mengikuti prosedur berikut :
   a. *Alice* pertama mengirimkan M^{eA} kepada Bob,
   b. Pada saat menerima pesan, Bob mengirimkan M^{eAeB} kembali ke *Alice* (perhatikan bahwa saat ini, Bob tidak membaca pesan *Alice M*).

c. *Alice* mengirim $M^{eAeBdA}=M^{eB}$ kepada Bob, Bob kemudian menghitung $M^{dBeB}=M$, dan terbukalah pesan asli *Alice M*.

II.2.5. Visual Studio 2010

Visual Studio 2010 pada dasarnya adalah sebuah bahasa pemrograman komputer. Dimana pengertian dari bahasa pemrograman itu adalah perintah-perintah atau instruksi yang dimengerti oleh komputer untuk melakukan tugas-tugas tertentu. Visual Studio 2010 (yang sering juga disebut dengan VB .Net 2010) selain disebut dengan bahasa pemrograman, juga sering disebut sebagai sarana *(tool)* untuk menghasilkan program-progam aplikasi berbasiskan windows. *Visual basic* adalah sebuah bahasa pemrograman yang berpusat pada *object* (*Object Oriented Programming*) digunakan dalam pembuatan aplikasi *Windows* yang berbasis *Graphical User Interface*, hal ini menjadikan *Visual Basic* menjadi bahasa pemrograman yang wajib diketahui dan dikuasai oleh setiap programmer. Beberapa karakteristik obyek tidak dapat dilakukan oleh *Visual Basic* misalnya seperti *Inheritance* tidak bisa module dan *Polymorphism* secara terbatas bisa dilakukan dengan deklarasi *class module* yang mempunyai *Interface* tertentu (Ninuk Wiliani, 2017).

Beberapa kemampuan atau manfaat dari Visual Studio 2010 diantaranya seperti :

a. Untuk membuat program aplikasi berbasiskan windows.

b. Untuk membuat objek-objek pembantu program seperti, misalnya : kontrol ActiveX, file Help, aplikasi Internet dan sebagainya.

c. Menguji program (debugging) dan menghasilkan program berakhiran EXE yang bersifat executable atau dapat langsung dijalankan.

Visual Studio 2010 adalah bahasa yang cukup mudah untuk dipelajari. Bagi programmer pemula yang baru ingin belajar program, lingkungan Visual Studio dapat membantu membuat program dalam sekejap mata. Sedang bagi programer tingkat lanjut, kemampuan yang besar dapat digunakan untuk membuat program-program yang kompleks, misalnya lingkungan net-working atau client server.. Bahasa Visual Studio cukup sederhana dan menggunakan kata-kata bahasa Inggris yang umum digunakan. Kita tidak perlu lagi menghafalkan sintaks-sintaks maupun format-format bahasa yang bermacam-macam, di dalam Visual Basic semuanya sudah disediakan dalam pilihan-pilihan yang tinggal diambil sesuai dengan kebutuhan. Selain itu, sarana pengembangannya yang bersifat visual memudahkan kita untuk mengembangkan aplikasi berbasiskan Windows, bersifat mouse-driven (digerakkan dengan mouse) dan berdaya guna tinggi. (Ninuk Wiliani, 2017).

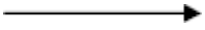
II.2.6. *Unified Modeling Language* (UML)

Unified Modeling Language (UML) adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membanngun perangkat lunak. UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem (Ade Hendini, 2016).

II.2.6.1. *Use Case Diagram*

Use case diagram merupakan pemodelan untuk kelakuakn (*behavior*) sistem informasi yang akan dibuat. *Use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut (Ade Hendini, 2016).

Tabel II.1. *Use Case Diagram*

Gambar	Keterangan
	*UseCase* menggambarkan fungsionalitas yang disediakan system sebagai unit-unit yang bertukar pesan antar unit dengan aktir, yang dinyatakan dengan menggunakan kata kerja.
	*Actor* atau Aktor adalah *Abstraction* dari orang atau system 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 actor berinteraksi dengan *UseCase*, tetapi tidak memiliki control terhadap *usecase*.
	Asosiasi antara actor dan *usecase*, di gambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengindikasikan data.
	Asosiasi antara aktor dan *usecase* yang menggunakan panah terbuka untuk mengindikasikan bila aktor berinteraksi secara pasif dengan sistem.
- - - - - - - - - - - - - - - - <<include>>	*Include*, merupakan didalam *usecase* lain (*required*) atau pemanggilan *usecase* oleh *usecase* lain, contohnya adalah pemanggilan sebuah fungsi

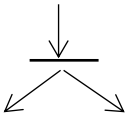
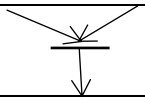
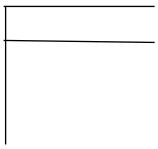
	program.
<- - - - - - - - - - - - - - - <<extends>>	*Extend*, merupakan perluasan dari *usecase* lain jika kondisi atau syarat.

(Sumber: Ade Hendini, 2016)

II.2.6.2. *Activity Diagram*

Activity Diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis (Ade Hendini, 2016).

Tabel II.2. *Activity Diagram*

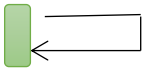
Gambar	Keterangan
	*Start Point*, diletakkan pada pojok kiri atas dan merupakan awal aktivitas.
	*End Point*, akhir aktivitas.
	*Activities*, menggambarkan suatu proses /kegiatan bisnis.
	*Fork*/percaban gan, digunakan untuk menunjukkan kegiatan yang dilakukan secara paralel atau untuk menggabungkan dua kegiatan parallel menjadi satu.
	*Join* (penggabung an) atau *rake*, digunakan untuk menunjukkan adanya dekomposisi.
	*Decision Points*, menggambar kan pilihan untuk pengambilan keputusan, *tru* atau *false.*
	*Swimlane*, pembagian *activity diagram* untuk menunjukkan siapa melakukan apa

(Sumber: Ade Hendini, 2016)

II.2.6.3. *Sequence Diagram*

Sequence Diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek (Ade Hendini, 2016).

Tabel II.3. *Sequence Diagram*

Gambar	Keterangan
	*Entity Class*, 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 *interfaces* atau interaksi antara satu atau lebih actor dengan sistem, seperti tampilan form entry 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*,mewa kili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivasi sebuah operasi.
	Lifeline, garis titik-titik yang terhubung dengan objek, sepanjang *lifeline* terdapat *Activation.*

(Sumber: Ade Hendini, 2016)

II.2.6.4. *Class Diagram*

Merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggung 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 *Assosiations*, *Generalitation* dan *Aggregation*, attribut (*Attributes*), operasi (*operation/method*) dan *visibility*, tingkat akses objek eksternal kepada suatu operasi atau attribut. Hubungan antar kelas mempunyai keterangan yang disebut dengan *Multiplicity* atau *Cardinality* (Ade Hendini, 2016).

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 maksimal 4

(Sumber: Ade Hendini, 2016)