BAB II

TINJAUAN PUSTAKA

II.1. Penelitian Terkait

Penelitian lain juga dilakukan oleh Diana dan Zebua pada tahun 2018 dengan judul "Optimalisasi Beaufort Cipher Menggunakan Pembangkit Kunci RC4 Dalam Penyandian SMS" menghasilkan sebuah aplikasi yang dapat melakukan enkripsi dan dekripsi terhadap teks pada SMS. Dari penelitian tersebut yang menjadi perbedaan dengan penelitian yang akan dibuat adalah pada penelitian yang akan dibuat algoritma beaufort akan diterapkan untuk keamanan pesan pada aplikasi *chatting* menggunakan koneksi *wifi direct*.

Penelitian lain juga dilakukan oleh Gratia Vintana dan Mardi Hardjianto pada tahun 2016 dengan judul "*Security Chatting* Berbasis *Desktop* dengan Enkripsi Caesar *Cipher Key Random*". Dari penelitian tersebut telah dihasilkan aplikasi *chatting* untuk digunakan pada komputer yang terdapat pada satu area jaringan dengan kemampuan untuk melakukan enkripsi dan dekripsi pesan yang dikirim maupun diterima. Dari penelitian tersebut terdapat perbedaan dengan penelitian yang akan dilaksanakan, yaitu pada penelitian yang akan dilaksanakan aplikasi *chatting* yang akan dibuat akan digunakan pada perangkat *mobile*. Perbedaan lainnya adalah pada penelitian sebelumnya menggunakan algoritma caesar sedangkan pada penelitian yang akan dilaksanakan menggunakan algoritma beaufort.

II.2. Uraian Teoritis

II.2.1. Penjelasan Algoritma

Algoritma adalah sistim kerja komputer memiliki *brainware*, *hardware*, dan *software*. Tanpa salah satu dari ketiga sistim tersebut, komputer tidak akan berguna. Kita akan lebih fokus pada *software* komputer. *Software* terbangun atas susunan program) dan *syntax* (cara penulisan/pembuatan program). Untuk menyusun program atau *syntax*, diperlukannya langkah-langkah yang sistematis dan logis untuk dapat menyelesaikan masalah atau tujuan dalam proses pembuatan suatu *software*. Maka, algoritma berperan penting dalam penyusunan program atau *syntax* tersebut.

Pengertian algoritma adalah susunan yang logis dan sistematis untuk memecahkan suatu masalah atau untuk mencapai tujuan tertentu. Dalam dunia komputer, algoritma sangat berperan penting dalam pembangunan suatu *software*. Dalam dunia sehari-hari, mungkin tanpa kita sadari algoritma telah masuk dalam kehidupan kita.

Algoritma berbeda dengan logaritma. Logaritma merupakan operasi matematika yang merupakan kebalikan dari eksponen atau pemangkatan. Contoh logaritma seperti bc= a ditulis sebagai blog a = c (b disebut basis). (Maulana, Gun Gun ; 2017 : 70).

II.2.2. Kriptografi

Kriptografi (*Cryptography*) berasal dari bahasa Yunani, terdiri dari dua suku kata yaitu kripto dan graphia. Kripto artinya menyembunyikan, sedangkan graphia artinya tulisan. Kriptografi adalah ilmu yang mempelajari teknik-teknik matematika yang berhubungan dengan aspek keamanan informasi, seperti kerahasian data, keabsahan data, integritas data, serta autentikasi data. Tetapi tidak semua aspek keamanan informasi dapat diselesaikan dengan kriptografi.

Kriptografi dapat pula diartikan sebagai ilmu atau seni untuk menjaga keamanan pesan. Ketika suatu pesan dikirim dari suatu tempat ke tempat lain, isi pesan tersebut mungkin dapat disadap oleh pihak lain yang tidak berhak untuk mengetahui isi pesan tersebut. Untuk menjaga pesan, maka pesan tersebut dapat diubah menjadi sebuah kode yang tidak dapat dimengerti pihak lain.

Enkripsi adalah sebuah proses penyandian yang melakukan perubahan sebuah kode (pesan) dari yang bisa dimengerti (*plaintext*) menjadi sebuah kode yang tidak bisa dimengerti (*chipertext*). Sedangkan proses kebalikannya untuk mengubah *cipertext* menjadi *plaintext* disebut dekripsi. Proses enkripsi dan deskripsi memerlukan suatu mekanisme dan kunci tertentu. Kriptografi adalah ilmu mengenai teknik enkripsi dimana data diacak menggunakan suatu kunci enkripsi menjadi sesuatu yang sulit dibaca oleh seseorang yang tidak memiliki kunci dekripsi. Dekripsi menggunakan kunci dekripsi mendapatkan kembali data asli. Proses enkripsi dilakukan menggunakan suatu algoritma dengan beberapa parameter. Biasanya algoritma tidak dirahasiakan, bahkan enkripsi yang mengandalkan kerahasiaan algoritma dianggap sesuatu yang tidak baik. Rahasia

terletak di beberapa parameter yang digunakan, jadi kunci ditentukan oleh parameter. (Amin, M. Miftakul ; 2016 : 130-131)

II.2.3. Algoritma Beaufort

Beaufort cipher merupakan salah satu algoritma dalam teknik keamanan kriptografi klasik. Kunci (K) pada beaufort cipher adalah urutan karakter-karakter K = k1 ... kd dimana k1 didapat dari banyaknya pergeseran dari alfabet ke-i sama seperti viginere cipher. Artinya bahwa jumlah kunci yang dibangkitkan harus sama dengan jumlah karakter plaintext yang diamankan. Algoritma ini melakukan proses enkripsi dan dekripsi secara stream (masing-masing karakter plaintext harus memiliki pasangan kunci). Hal ini yang menyebabkan algoritma ini sama hampir sama dengan algoritma vigeneere cipher. Adapun formulasi yang digunakan dalam proses enkripsi dan dekripsi adalah :

Formula proses enkripsi :

$Ci = Ek(Mi) = (Ki - Mi) \text{ Mod } 26$ .....(II.1)

Formula proses dekripsi :

$Mi = Dk(Ci) = (Ki - Ci) \text{ Mod } 26$ .....(II.2)

Keterangan :

Mi = Pesan yang akan dienkripsi (plain)

Ci = Sandi (cipher) Ki = Kunci

Ek = Fungsi Enkripsi Dk = Fungsi Dekripsi

Nilai mod 26 di atas tergantung dari jumlah kebutuhan karakter yang digunakan, pada awalnya beaufort cipher hanya menggunakan 26 karakter, namun

seiring dengan perkembangan teknologi komputer saat ini, maka dapat menggunakan mod 256 (mengguakan seluruh tabel ASCII). (Mia Diana dan Taronisokhi Zebua ; 2018 : 15-16).

II.2.4. Aplikasi

Secara istilah pengertian aplikasi adalah suatu program yang siap untuk digunakan yang dibuat untuk melaksanankan 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". (Juansyah, Andi ; 2015 : 2)

II.2.5. Aplikasi *Chat Messanger (Chatting)*

Aplikasi adalah sebuah media penunjang dalam sebuah objek yang memiliki beberapa instruksi yang disusun sedemikian rupa sehingga dapat menghasilkan *input* dan *output*. Pengertian *Chat Messanger* atau *Chatting* adalah mengobrol jika diterjemahkan langsung dari bahasa inggris. Dalam dunia komputer dan *internet*, pengertian *Chat Messanger* adalah suatu fasilitas dalam

internet untuk berkomunikasi sesama pengguna *internet* yang sedang *on-line*. Komunikasi dapat berupa teks. (Sutikno, et al. ; 2018 : 2)

II.2.6. Android

Android adalah sebuah *platform* pertama yang betul-betul terbuka dalam pengembangannya dan komperehensif untuk perangkat *mobile,* semua perangkat lunak yang ada difungsikan menjalankan sebuah *device mobile* tanpa memikirkan kendala kepemilikan yang menghambat inovasi pada teknologi *mobile*. Dalam definisi lain, Android merupakan subset perangkat lunak untuk perangkat *mobile* yang meliputi sistem operasi, *middleware,* dan aplikasi inti yang dirilis oleh Google. Sedangkan Android *SDK (Software Development Kit)* menyediakan *tools* dan API yang diperlukan untuk mengembangkan aplikasi pada *platform* Android dengan menggunakan bahasa pemrograman Java. (Ahmad : 2015 : 190-200)

Aplikasi *Android* ditulis dalam bahasa pemrograman *java*, yaitu kode *java* yang terkompilasi bersama-sama dengan data dan *file-file* sumber yang dibutuhkan oleh aplikasi yang digabungkan oleh *app tools* menjadi paket aplikasi Android, sebuah file yang ditandai dengan akhiran *.apk.* file inilah yang didistribusikan sebagai aplikasi dan diinstal pada *handset Android*. File ini diunduh oleh pengguna ke perangkat *mobile* mereka. Semua kode dijadikan satu file *.apk,* dan kemudian kita sebut sebagai sebuah aplikasi. (Ahmad : 2015 : 190-200)

II.2.7. Android Studio

Android Studio adalah IDE (*Integrated Development Environment*) resmi untuk pengembangan aplikasi Android dan bersifat *open source* atau gratis. Peluncuran Android Studio ini diumumkan oleh Google pada 16 mei 2013 pada *event* Google I/O *Conference* untuk tahun 2013. Sejak saat itu, Android Studio mengantikan *Eclipse* sebagai IDE resmi untuk mengembangkan aplikasi Android.

Android Studio sendiri dikembangkan berdasarkan IntelliJ IDEA yang mirip dengan *Eclipse* disertai dengan ADT plugin (*Android Development Tools*). Android Studio memiliki fitur :

a. Projek berbasis pada *Gradle Build*
b. *Refactory* dan pembenahan *bug* yang cepat
c. *Tools* baru yang bernama "*Lint*" dikalim dapat memonitor kecepatan, kegunaan, serta kompetibelitas aplikasi dengan cepat.
d. Mendukung *Proguard And App-signing* untuk keamanan.
e. Memiliki GUI aplikasi android lebih mudah
f. Didukung oleh *Google Cloud Platfrom* untuk setiap aplikasi yang dikembangkan. (Juansyah, Andi ; 2015)

II.2.8. Android SDK (*Software Development Kit*)

Android SDK mencakup perangkat *tools* pengembangan yang komprehensif. Android SDK terdiri dari *debugger*, *libraries*, *handset emulator*, dokumentasi, contoh kode program dan tutorial. Saat ini Android sudah mendukung arsitektur x86 pada Linux (distribusi Linux apapun untuk *desktop*

modern), Mac OS X 10.4.8 atau lebih, Windows XP atau Vista. Persyaratan mencakup JDK, Apache Ant dan Python 2.2 atau lebih. IDE yang didukung secara resmi adalah Eclipse 3.2 atau lebih dengan menggunakan plugin *Android Development Tools* (ADT), dengan ini pengembang dapat menggunakan IDE untuk mengedit dokumen Java dan XML serta menggunakan peralatan *command line* untuk menciptakan, membangun, melakukan *debug* aplikasi Android dan pengendalian perangkat Android (misalnya *reboot*, menginstal paket perangkat lunak). (Sulihati dan Andriyani ; 2016 : 20)

II.2.9. Bahasa Pemrograman Java

Java dikembangkan oleh *Sun Microsystems* pada Agustus 1991. Java disebut juga merupakan hasil perpaduan sifat dari sejumlah bahasa pemrograman, yaitu C dan C++. Pemrograman Java bersifat tidak bergantung pada *platform*, yang artinya, java dapt dijalankan pada sembarang komputer dan bahkan pada sembarang sistem operasi. Sebagaimana halnya C++, salah satu bahasa yang mengilhami Java, Java juga merupakan bahasa pemrograman berorientasi objek. Sebagai bahasa pemrograman berorientasi objek, Java menggunakan kelas untuk membentuk suatu objek. Karakteristik Java antara lain adalah berorientasi objek (*object-oriented*), terdistribusi (*distributed*), sederhana (*simple*), aman (*secure*), *interpreted, robust, multithreaded*, dan dinamis. (Annisa Rahmawati, et al. ; 2015 : 336)

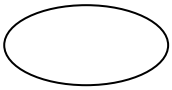
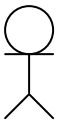
II.2.10. Pengertian UML

UML adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membangun perangkat lunak. UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem. UML saat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa pemodalan umum dalam industri perangkat lunak dan pengembangan sistem (Windu dan Grace ; 2013 : 81). Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan UML adalah sebagai berikut :

II.2.10.1. *Use Case Diagram*

Use case Diagram merupakan pemodelan untuk kelakuan (*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 didalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. (Windu dan Grace ; 2013)

Tabel II.1. Use Case Diagram

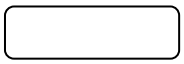
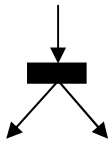
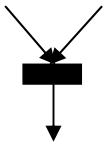
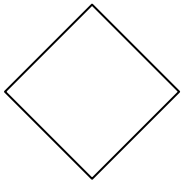
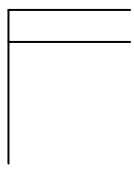
Gambar	Keterangan
	*Use Case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukar pesan antar unit dengan aktir, yang dinyatakan dengan menggunakan kata kerja
	*Actor* atau Aktor adalah *Abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan aktir, 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 kontrol 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 mengindikasikan data.
	Asosiasi antara aktor dan *use case* yang menggunakan panah terbuka untuk mengindikasikan 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

(Sumber : Ade Hendini ; 2016)

II.2.10.2. *Activity Diagram*

Activity diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis (Windu dan Grace ; 2013 : BIT. 10. 80-87).

Tabel II.2. Activity Diagram

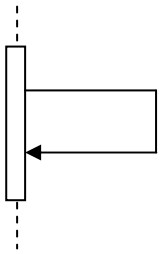
Gambar	Keterangan
	*Start Point*, diletakkan pada pojok kiri atas dan merupakan awal aktivitas
	*End Point*, akhir aktivitas
	*Activities*, menggambarkan suatu proses/kegiatan bisnis
	*Fork*/percabangan, digunakan untuk menunjukkan kegiatan yang dilakukan secara paralel atau untuk menggabung kan dua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

(Sumber : Ade Hendini ; 2016)

II.2.10.3. *Sequence Diagram*

Sequence diagram menggambarkan kelakuan obyek pada *use case* dengan mendeskripsikan waktu hidup obyek dan pesan yang dikirimkan dan diterima antar obyek (Windu dan Grace ; 2013).

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 aktor 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*, mewakili 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)