# BAB II

# TINJAUAN PUSTAKA

## II.1. Penelitian Terkait

Penelitian yang dilakukan oleh Murdani (2017) dengan judul Perancangan Aplikasi Penyandian Data Teks Menggunakan Algoritma Merkle Hellman Knapsack. *Cryptosystem* terkenal pertama kali dideskripsikan oleh *Merkle* dan *Hellman* ide dasar di balik skema *enkripsi Merkle Hellman* adalah untuk menciptakan sebuah subset masalah yang dapat diselesaikan dengan mudah dan kemudian untuk menyembunyikan sifat super meningkat oleh perkalian modular dan permutasi.Hingga saat ini, Kriptografi merupakan salah satu solusi untuk menjamin penyandian dari suatu data atau informasi. Kriptografi merupakan metode dengan menyandikan isi informasi (*plaintext*) menjadi isi yang sulit atau bahkan tidak dipahami melalui proses *enkripsi*. Untuk memperoleh kembali informasi yang asli dapat dilakukan dengan proses dekripsi, yang tentunya dengan menggunakan kunci yang benar. Untuk melindungi akses data dari pihak-pihak yang tidak berkepentingan tersebut maka sangat diperlukan *enkripsi* dan *dekripsi*. Agar dapat dilakukan dengan baik, dibutuhkan suatu algoritma untuk enkripsi dan dekripsi. Algoritma yang digunakan disini adalah Algoritma *Merkle Hellman Knapsack.*

Penelitian yang dilakukan oleh Akik Hidayat (2016) dengan judul Cryptography Asymmetries Merkle-Hellman Knapsack Digunakan untuk Enkripsi dan Dekripsi Teks. Merkle-Hellman Knapsack merupakan kriptosistem yang

menggunakan algoritma asymmetries. Kelebihan algoritma asymmetries ini adalah proses pendistribusian kunci pada media yang tidak aman seperti internet, tidak memerlukan kerahasiaan. Karena kunci yang didistribusikan adalah kunci publik. Sehingga jika kunci ini sampai hilang atau diketahui oleh orang lain yang tidak berhak, maka pesan sandi yang dikirim akan tetap aman. Sedangkan kunci private (rahasia) tetap disimpan (tidak didistribusikan). Dengan menggunakan Merkle-Hellman Knapsack dapat menggunakan ukuran kunci yang lebih kecil dibandingkan dengan kriptosistem seperti RSA. Kemampuan ini membuat Merkle-Hellman Knapsack mempunyai keamanan yang kuat dengan panjang kunci yang pendek.. Sedangkan tujuan yang ingin dicapai yaitu mengaplikasikan metode kriptosistem Merkle-Hellman Knapsack menggunakan bahasa pemrograman C++.

## II.2. Landasan Teori

Berikut adalah penjelesan dari beberapa teori yang berhubungan dengan judul pada penelitian ini yaitu :

### 1. Perancangan

Perancangan adalah kegiatan yang memiliki tujuan untuk mendesain sistem baru yang dapat menyelesaikan masalahmasalah yang dihadapi perusahaan yang diperoleh dari pemilihan alternatif sistem yang terbaik. Rancang Bangun (desain) adalah tahap dari setelah analisis dari siklus pengembangan sistem yang merupakan pendefinisian dari kebutuhankebutuhan fungsional, serta menggambarkan bagaimana suatu sistem dibentuk yang dapat berupa

penggambaran, perencanaan dan pembuatan sketsa atau pengaturan dari beberapa elemen yang terpisah ke dalam satu kesatuan yang utuh dan berfungsi, termasuk menyangkut mengkonfigurasikan dari komponen-komponen perangkat keras dan perangkat lunak dari suatu system (Ria Apriyani ; 2013 : 2).

**2. Aplikasi**

Aplikasi adalah program yang memiliki aktivitas pemrosesan perintah yang diperlukan untuk melaksanakan permintaan pengguna dengan tujuan tertentu. Aplikasi adalah program atau sekelompok program yang dirancang untuk digunakan oleh pengguna akhir (*end user*). Aplikasi dapat dimanfaatkan untuk keperluan pembelajaran kepada siswa mengingat dalam suatu proses pembelajaran seharusnya terdapat interaksi antar komponen-komponen pembelajaran. Salah satu pendekatan pembelajaran yang memungkinkan antara komponen-komponen pembelajaran tersebut adalah pembelajaran interaktif (Irvan Rizkiansyah ; 2012 : 2).

**3. Keamanan**

Keamanan data menjadi hal yang sangat penting pada saat ini karena untuk setiap pengambilan keputusan, kebijakan harus berdasarkan data. Banyak data yang berisikan informasi penting dan terbatas untuk diketahui pihak yang terkait saja. Pada dunia perbankan banyak kegiatan yang melibatkan data nasabah yang harus diproteksi dan serta sifatnya rahasia. Banyak kegiatan yang akan menimbulkan resiko bilamana informasi yang sensitif dan berharga tersebut

diakses oleh orang-orang yang tidak berhak (*unauthorized person*). Faktor keamanan data menjadi sangat penting dan harus diperhatikan. Salah satu cara untuk meningkatkan keamanan data diperlukan kriptografi dengan metode enkripsi (Pratiwi ; 2016 : 132).

**4. Algoritma Merkle-Hellman Knapsack**

Algoritma Merkle-Hellman knapsack merupakan Algoritma Asimetris. Artinya, algoritma ini memliki kunci publik dan kunci *privat* untuk proses *enkripsi* dan *dekripsinya*. Algoritma ini mempunyai tiga proses mekanisme. Proses yang pertama adalah proses pembangkitan kunci publik dan kunci *privat*, proses *enkripsi* dan proses *dekripsi*. Sebelum melakukan ketiga proses tersebut, beberapa mekanisme dari algoritma ini antara lain:

1. Menentukan deretan *superincreasing* di mana setiap elemen di dalam deretan harus lebih besar daripada jumlah elemen sebelumnya. Deretan *superincreasing* ini digunakan sebagai kunci privat.
2. Memilih bilangan prima *m* dengan ketentuan bilangan tersebut harus lebih besar daripada elemen terakhir deretan *superincreasing*
3. Memilih bilangan *n*, di mana bilangan tersebut relatif prima terhadap bilangan *m*.
4. Membangkitkan kunci publik dengan persamaan 1: $pi{=}si{*}n \ mod \ m$ (1)

Variabel i pada pada persamaan (1) merupakan *indeks* dari deretan *superincreasing*. Jadi, setiap elemen di dalam bilangan *superincreasing* diproses pada persamaan (1) dan menghasilkan kunci publik. Kunci publik tersebut

memiliki panjang elemen yang sama dengan deretan *superincreasing* (Aminudin, 2018:326).

**4. Pengertian Visual Basic**

Microsoft Visual Basic 2010 adalah salah satu komponen Microsoft Visual Studio 2010. *Software* ini diluncurkan Microsoft pada tanggal 12 April 2010 dengan nama kode Dev10 dan menggunakan .Net Framework 4.0. *Integrated Development Environment* (IDE) pada Visual Studio 2010 telah didesain ulang sehingga lebih enak dipandang dan digunakan programmer. Untuk kode editor-nya, Visual Basic 2010 telah menambah fitur *highlights reference*. Ketika satu simbol/kode dalam bahasa pemrogramannya dipilih, maka simbol/kode yang sama, meskipun penggunaannya berbeda akan terlihat berwarna sama. Misal jika kode *math* dipilih, seluruh kode *math* akan terlihat berwarna sama (Fadillah ; 2014 : 10).

**5. Pengertian Database**

*Database* adalah kumpulan *field-field* yang mempunyai kaitan antara satu file dengan field yang lain sehingga membentuk bangunan data untuk menginformasikan kondisi lalu lintas dalam bahasa tertentu (Mhd Bustanur Rahmad ; 2014 : 1333).

Database atau biasa disebut basis data merupakan kumpulan data yang saling berhubungan. Data tersebut biasanya terdapat dalam tabel - tabel yang

saling berhubungan satu sama lain, dengan menggunakan field/kolom pada tiap tabel yang ada (Agus Prayitno ; 2015 : 2)

**6. Pengertian SQL Server**

SQL Server 2008 adalah sebuah terobosan baru dari Microsoft dalam bidang database. SQL Server adalah sebuah DBMS (*Database Management System*) yang dibuat oleh Microsoft untuk ikut berkecimpung dalam persaingan dunia pengolahan data menyusul pendahulunya seperti IBM dan Oracle. SQL Server 2008 dibuat pada saat kemajuan dalam bidang hardware sedemikian pesat. Oleh karena itu sudah dapat dipastikan bahwa SQL Server 2008 membawa beberapa terobosan dalam bidang pengolahan dan penyimpanan data (Komputer ; 2013 : 2).

SQL Server 2008 adalah sebuah terobosan baru dari Microsoft dalam bidang database. SQL Server adalah DBMS (Database Management System) yang dibuat oleh Microsoft untuk ikut berkecimpung dalam persaingan dunia pengolahan data menyusul pendahulunya seperti IBM dan Oracle. SQl Server 2008 dibuat pada saat kemajuan dalam bidang hardware sedemikian pesat. Oleh karena itu sudah dapat dipastikan bahwa SQL Server 2008 membawa beberapa terobosan dalam bidang pengolahan dan penyimpanan data (Wenny Widya ; 2015 : 3)

**7. UML (*Unified Modeling Language*)**

Menurut Windu Gata (2013 : 4) Hasil pemodelan pada OOAD terdokumentasikan dalam bentuk *Unified Modeling Language* (UML). UML

adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membangun perangkat lunak.

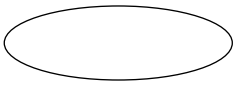
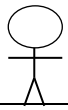
UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem. UML saat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa pemodelan umum dalam industri perangkat lunak dan pengembangan sistem.

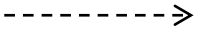
Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan UML adalah sebagai berikut :

- *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, yaitu dapat dilihat pada tabel II.1. berikut :

**Tabel II.1. Simbol Use Case**

| Gambar | Keterangan |
|---|---|
|  | *Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukan pesan antar unit dengan aktor, biasanya 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 |

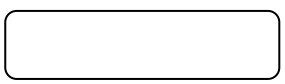
| | |
|---|---|
| | pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. Orang atau sistem bisa muncul dalam beberapa peran. Perlu dicatat bahwa aktor berinteraksi dengan *use case*, tetapi tidak memiliki control terhadap *use case*. |
|  | Asosiasi antara aktor 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. |

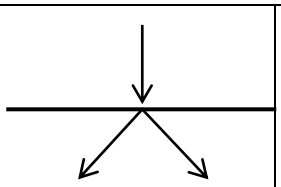
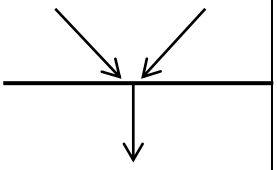
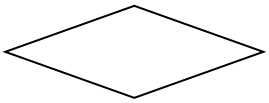
**(Sumber : Windu Gata ; 2013 : 4)**

- 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,* yaitu dapat dilihat pada tabel II.2. berikut:

**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. |

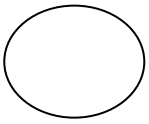
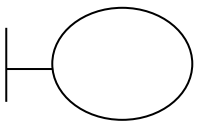
| | |
|---|---|
|  | *Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan pararel menjadi satu. |
|  | *Join* (penggabungan) atau rake, digunakan untuk menunjukkan adanya dekomposisi. |
|  | *Decision Points,* menggambarkan pilihan untuk pengambilan keputusan, *true*, *false.* |
|   | *Swimlane*, pembagian *activity* diagram untuk menunjukkan siapa melakukan apa. |

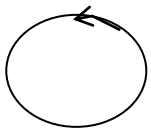
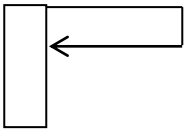
***(Sumber : Windu Gata ; 2013 : 6)***

- 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*, yaitu dapat dilihat pada tabel II.3. berikut:

**Tabel II.3. Simbol 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 *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 : Windu Gata ; 2013 : 7)***

- *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. Diagram dapat dilihat pada tabel II.4. berikut:

**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 : Windu Gata ; 2013 : 8)***