BAB II

TINJAUAN PUSTAKA

II.1. Penelitian Terkait

Berdasarkan penelitian yang dilakukan oleh Irmawati (2016) mengenai Pengembangan Aplikasi Kriptografi *File* Dokumen, *Audio* dan Gambar Dengan Algoritma DES, Irmawati menyimpulkan bahwa kekuatan kunci enkripsi *Data Encryption Standard* lebih baik dibandingkan dengan MD5 dan RC4, hal ini dibuktikan dengan hasil pengujian dengan manipulasi ekstensi *file* dan penggunaan *software* pembuka enkripsi yaitu *file repair*. *Data Encryption Standard* merupakan algoritma kriptografi simetris karena hanya mampu menghasilkan satu kunci *private* dengan ukuran 56 bit. Kecepatan enkripsi yang mampu dihasilkan DES yaitu 5549,9 *KiloByte* / detik atau 5,41 *MegaByte* / detik.

Berdasarkan penelitian yang dilakukan oleh Raja Nasrul Fuad dan Haikal Nando Winata (2017) mengenai Aplikasi Keamanan *File Audio* WAV *(Waveform)* Dengan Terapan Algoritma RSA, peneliti menyimpulkan bahwa perangkat lunak *(software)* dapat melakukan penyandian data *audio* dengan menerapkan algoritma RSA dan stuktur data *audio* wav. Ukuran berkas *audio* wav menjadi bertambah besar setelah dilakukan enkripsi menggunakan algoritma RSA berdasarkan besar kunci yang digunakan.

II.2. Landasan Teori

Berdasarkan penelitian terkait diatas dapat disimpulkan bahwa algoritma RSA masih rentan terhadap serangan *software* pembuka kunci enkripsi yaitu *file repair* dan ukuran hasil dari enkripsi yang dihasilkan bertambah besar berdasarkan besaran kunci yang digunakan. Landasan teori yang penulis gunakan dalam penelitian ini berdasarkan penelitian yang terkait adalah sebagai berikut:

II.2.1. Kriptografi

Kata kriptografi *(cryptography)* berasal dari Bahasa Yunani, yaitu *kriptos* yang artinya *secret* (rahasia), dan *graphien,* yang artinya *writing* (tulisan). Jadi, kriptografi berarti *secret writing* (tulisan rahasia). Ada beberapa definisi kriptografi yang didapat dari berbagai sumber, antara lain:

1. Kriptografi *(Cryptography)* didefinisikan sebagai ilmu sekaligus seni untuk menjaga kerahasiaan pesan (data atau informasi) yang mempunyai pengertian, dengan cara menyamarkannya (mengacak) menjadi bentuk yang tidak dapat dimengerti menggunakan suatu algoritma tertentu (Mulya, 2008).
2. Kriptografi merupakan ilmu yang mempelajari tentang bagaimana cara membuat suatu pesan hanya bisa dibaca oleh pihak yang berwenang untuk membacanya (Setiawan, 2010).
3. Kriptografi merupakan ilmu sekaligus seni untuk menjaga kerahasian data atau informasi agar tidak dapat dilihat, dibaca, dimengerti oleh pihak

ketiga yang tidak memiliki wewenang terhadap data atau informasi tersebut (Frengky Fernando, 2014).

Berdasarkan beberapa definisi kriptografi diatas, dapat disimpulkan bahwa Kriptografi adalah bidang ilmu yang menggunakan seni penyandian secara rahasia untuk menjaga dan menyamarkan bentuk asli suatu objek, data, maupun informasi dari pihak lain dengan menggunakan suatu algoritma tertentu sehingga tidak dapat dimengerti oleh pihak ketiga.

II.2.2. Algoritma

"Algoritma adalah susunan yang logis dan sistematis untuk memecahkan suatu masalah untuk mencapai tujuan tertentu." (Gun Gun Maulana, 2017). Kata logis merupakan kata kunci dalam algoritma, dalam dunia komputer algoritma memiliki peranan yang sangat penting dalam pembuatan suatu *software* (program komputer). Langkah-langkah dalam algoritma harus ditentukan dengan logis dan harus dapat ditentukan bernilai salah atau benar.

II.2.3. *Audio*

Audio adalah suara atau bunyi yang dihasilkan ketika molekul di udara berubah oleh suatu gerakan yang ditimbulkan sebuah objek yang menghasilkan sebuah getaran (Affan Bachri, 2016). Jumlah getaran yang menghasilkan suara atau bunyi dalam satuan tertentu disebut sebagai frekwensi. Satuan untuk frekwensi adalah *cycle per second* atau cps. Satuan ini biasa dikenal dengan sebutan Hertz (Hz). Terkadang getaran yang dihasilkan sangat cepat, dan

perbandingan gerakan yang dihasilkan adalah seribu, maka satuan yang digunakan adalah kilohertz (kHz).

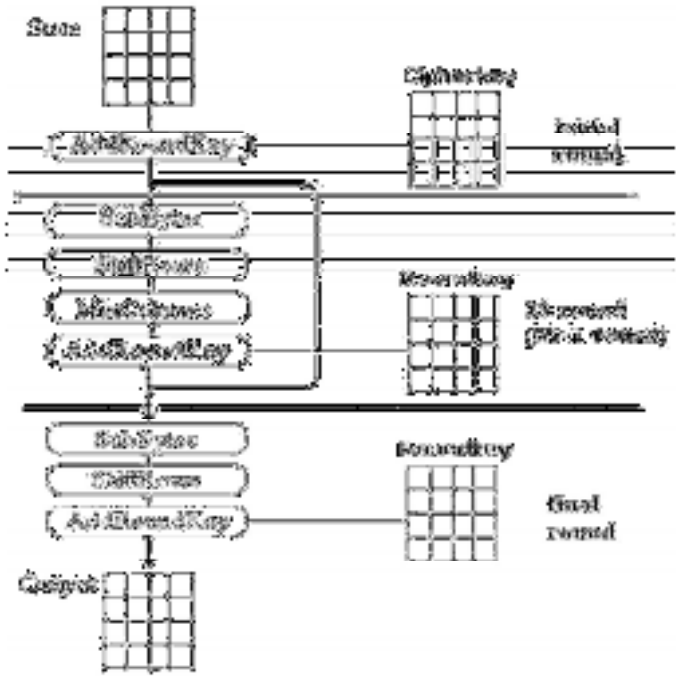
II.2.4. *Advanced Encryption Standard (AES)*

Advanced Encryption Standard (AES) merupakan algoritma kriptografi yang dapat digunakan untuk mengamankan data. Algoritma AES adalah blok *ciphertext* simetrik yang dapat mengenkripsi *(encipher)* dan dekripsi *(decipher)* informasi. Enkripsi merubah data yang tidak dapat lagi dibaca disebut *ciphertext,* sebaliknya dekripsi adalah merubah *ciphertext* data menjadi bentuk semula yang kita kenal sebagai *plaintext*. Algoritma AES menggunakan kunci kriptografi 128, 192, dan 256 bits untuk mengenkripsi dan mendekripsi data (Ami Aisiah Ibrahim, 2017). *Advanced Encryption Standard (AES)* merupakan algoritma standard keamanan data yang digunakan saat ini yang ditetapkan oleh NIST *(National Institute of Standard Technology)* pada bulan November 2001 untuk menggantikan algoritma DES *(Data Encryption Standard)* karena dianggap sudah tidak aman lagi. Metode yang digunakan dalam penelitian ini menggunakan algoritma *Advanced Encryption Standard (AES)* yang akan diterapkan kedalam nilai bit dari *file audio* untuk mengacak dan merubah nilai bit asli dari *file audio*. Garis besar Algoritma Rijndael yang beroperasi pada blok 128-bit dengan kunci 128-bit adalah sebagai berikut:

1. *AddRoundKey*: Melakukan *XOR* anatara *state* awal (*plaintext)* dengan *cipher key.* Tahap ini disebut juga *initial round.*

2. Putaran sebanyak *Nr*-1 kali. Proses yang dilakukan pada setiap putaran adalah:
    a. *SubBytes*: Subtitusi *byte* dengan menggunakan tabel subtitusi *(S-box)*.
    b. *ShiftRows*: Pergeseran baris-baris *array state* secara *wrapping.*
    c. *MixColumns*: Mengacak data di masing-masing kolom *array state*.
    d. *AddRoundKey*: Melakukan *XOR* antara *state* sekarang *round key.*
3. *Final Round*: Proses untuk putaran terakhir:
    a. *SubBytes*
    b. *ShiftRows*
    c. AddRoundKey

Berikut gambar diagram proses enkripsi menggunakan metode AES:



Gambar II.1. Diagram Proses Enkripsi.

II.2.5. *Android*

Android adalah sistem operasi dan *platform* pemrograman yang dikembangkan oleh Google untuk ponsel cerdas dan perangkat seluler lainnya (seperti *tablet*). *Android* bisa berjalan di beberapa macam perangkat dari banyak produsen yang berbeda. *Android* menyertakan *kit development* perangkat lunak untuk penulisan kode asli dan perakitan modul perangkat lunak untuk membuat aplikasi pengguna *android*. *Android* juga menyediakan pasar untuk mendistribusikan aplikasi. Secara keseluruhan, *android* menyatakan ekosistem untuk aplikasi seluler (*Android Developer Fundamentals Course*, 2016).

II.2.6. *Android Studio*

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

II.2.7. *Unified Modeling Language (UML)*

Menurut Windu Gata (2013) 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.

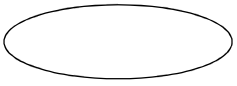
UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem (Gellysa Urva dan Helmi Fauzi Siregar, 2015). UML mulai diperkenalkan oleh *Object Management Group,* sebuah organisasi yang telah mengembangkan model, teknologi, dan standar OOP *(Object Oriented Programming)* sejak tahun 1980. Saat ini UML sudah mulai banyak digunakan oleh para praktisi OOP, UML merupakan dasar bagi perangkat *(tool)* desain berorientasi objek dari IBM.

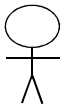
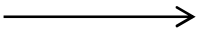
Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan UML adalah sebagai berikut (Gellysa Urva dan Helmi Fauzi Siregar, 2015):

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	Arti	Keterangan
	*Use Case*	*Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukar pesan antar unit dengan aktor, biasanya dinyatakan

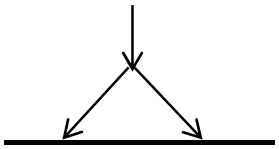
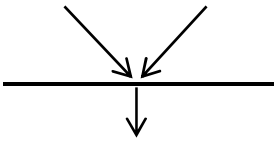
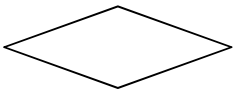
		dengan menggunakan kata kerja di awal nama *use case*.
	*Actor*	Aktor adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan aktor, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. Orang atau sistem bisa muncul dalam beberapa peran. Perlu dicatat bahwa aktor berinteraksi dengan *use case*, tetapi tidak memiliki control terhadap *use case*.
	Association	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.
	*Generalization*	Asosiasi antara aktor dan *use case* yang menggunakan panah terbuka untuk mengidinkasikan bila aktor berinteraksi secara pasif dengan sistem.
	Include	*Include,* merupakan di dalam *use case* lain (*required*) atau pemanggilan *use case* oleh *use case* lain, contohnya adalah pemanggilan sebuah fungsi program.
	*Extend*	*Extend,* merupakan perluasan dari *use case* lain jika kondisi atau syarat terpenuhi.

(Sumber : Gellysa Urva dan Helmi Fauzi Siregar, 2015).

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

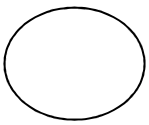
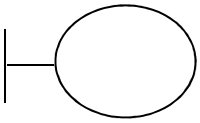
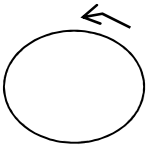
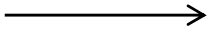
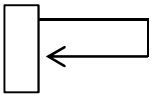
(Sumber : Gellysa Urva dan Helmi Fauzi Siregar, 2015).

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 berikut ini:

Tabel II.3. Simbol *Sequence Diagram.*

Gambar	Arti	Keterangan
	*Entity Class*	*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*	*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*	*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*	*Message*, simbol mengirim pesan antar *class*.
	*Recursive*	*Recursive,* menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri.
	*Activation*	*Activation, activation* mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivitas sebuah operasi.
	Lifeline	*Lifeline,* garis titik-titik yang terhubung dengan objek, sepanjang *lifeline* terdapat *activation*.

(Sumber : Gellysa Urva dan Helmi Fauzi Siregar, 2015).

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 / Methode), 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 : Gellysa Urva dan Helmi Fauzi Siregar, 2015).