# PENGAMANAN TABLE DATABASE MENGGUNAKAN KRIPTOGRAFI ALGORITMA RSA

**Lutfi Pratama**[1)], **Subandi,**[2)]
[1]Program Studi Teknik Informatika, Fakultas Teknologi Informasi, Universitas Budi Luhur
[1,2]Jl. Raya Ciledug, Petukangan Utara, Kebayoran Lama, Jakarta Selatan 12260
E-mail : 1311503377@student.budiluhur.ac.id[1)], subandionline@gmail.com[2)]

***Abstrak***

*Database* (*Basis Data) adalah kumpulan data yang tersimpan di dalam komputer sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi dari basis data tersebut. Menyimpan informasi atau data yang terintegrasi dengan baik di dalam komputer adalah kegunaan dari database . Keamanan dalam penyimpanan dan proses pertukaran data merupakan hal yang sangat penting pada perkembangan teknologi saat ini yang semakin pesat. Namun kita kurang menyadari keamanan pada saat penyimpanan dan proses pertukaran data tersebut. Salah satu dampak yang mungkin terjadi adalahpengambilan data dari pihak yang tidak berhak . Dari permasalahan tersebut memunculkan gagasan untuk mengembangkan suatu aplikasi berbasis desktop, yang di dalamnya dapat melakukan pengamanan table database. Metodologi yang diterapkan pada penelitian kali ini menggunakan Algoritma RSA. Aplikasi yang dikembangkan menggunakan bahasa pemrograman Java. Menggunakan database MySQL-Front 6.0. Tools dan editor menggunakan XAMPP 7.2.0-0 dan NetBeans IDE 8.2. Dengan adanya Aplikasi Pengamanan Table Database ini diharapkan akan mempermudah administrator dalam mengamankan table database yang bersifat rahasia, sehingga data yang ada di dalamnya aman dari pencurian data dan hanya orang yang berhak untuk membukanya sajalah yang dapat mengakses dan merubahnya. Sebelum pengguna mengamankan table database, terlebih dahulu pengguna harus membangkitkan kunci public dan private yang dibutuhkan untuk melakukan proses enkripsi.*

**Kata kunci:** Database, Keamanan, Kriptografi, Algoritma RSA

## 1. PENDAHULUAN

Berkembangnya Ilmu Pengetahuan dan Teknologi di zaman sekarang sangatlah cepat. Memudahkan kita untuk melakukan pertukaran data dengan orang lain secara cepat. Namun terkadang kita kurang menyadari keamanan dalam proses pertukaran data tersebut. Dari sekian banyak dampak negatif yang ditimbulkan salah satunya yaitu pencurian data, dimana hal tersebut adalah salah satu masalah yang ditakuti. Dengan adanya pencurian data maka masalah keamanan dalam proses pertukaran informasi serta penyimpanan data dianggap penting.

SMAN 57 Jakarta adalah salah satu Sekolah yang berada di Jalan Raya Kedoya Kebon Jeruk, yang sudah menggunakan database untuk penyimpanan data siswanya. Namun di SMAN 57 Jakarta belum mempunyai aplikasi untuk mengamankan database. Untuk mengamankan data-data siswa di database dapat menggunakan kriptografi. Oleh sebab itu, bantuan untuk keamanan database diperlukan bagi pengguna agar database aman dari pencurian data.

Algoritma yang digunakan oleh penulis adalah RSA. Alasan dipilihnya Algoritma RSA untuk mengamankan database adalah karena Algoritma RSA memiliki tingkat keamanan tinggi setelah datanya terenkripsi.

## 2. METODE PENELITIAN

### 2.1. Kriptografi

Kriptografi merupakan sebuah teknik yang bersifat rahasia dengan cara mengubah suatu data yang telah disimpan menjadi sebuah teks, simbol atau pun tulisan-tulisan yang sulit diartikan, sehingga data yang tersimpan tadi tidak mudah disalahgunakan oleh pihak yang tidak berkepentingan.

Kriptografi klasik adalah teknik enkripsi yang digunakan dalam enkripsi simetris dimana kunci dekripsi sama dengan kunci enkripsi. Untuk public key cryptography, diperlukan teknik enkripsi asimetris dimana kunci dekripsi tidak sama dengan kunci enkripsi. Enkripsi, dekripsi dan pembuatan kunci untuk teknik enkripsi asimetris memerlukan komputasi yang lebih intensif dibandingkan enkripsi simetris, karena enkripsi asimetris menggunakan bilangan-bilangan yang sangat besar [1].

### 2.2 Algoritma RSA

RSA merupakan algoritma kriptografi asimetris, dimana kunci yang digunakan untuk mengenkripsi berbeda dengan yang digunakan untuk mendekripsi. Kunci yang digunakan untuk mengenkripsi disebut dengan kunci public, dan yang digunakan untuk mendekripsi disebut dengan kunci private. RSA adalah salah satu algoritma kriptografi yang menggunakan konsep kriptografi kunci public. RSA membutuhkan tiga langkah dalam prosesnya, yaitu pembangkitan kunci, enkripsi, dan dekripsi. Proses enkripsi dan dekripsi merupakan proses yang

hampir sama. Jika bilangan acak yang dibangkitkan kuat, maka akan lebih sulit untuk melakukan cracking terhadap pesan. Parameter kuat tidaknya suatu kunci terdapat pada besarnya bilangan acak yang digunakan.

### 2.3 Algoritma Pembangkitan Pasangan Kunci

Untuk pembangkitan pasangan kunci RSA, maka digunakan algoritma sebagai berikut:

1) Memilih dua bilangan prima sembarang yang diberi nama p dan q. Nilai p dan q harus dirahasiakan.
2) Menghitung nilai n = p x q besaran n tidak perlu dirahasiakan.
3) Menghitung m = (p-1)(q-1)
4) Memilih nilai e ( kunci publik) yang relatif prima terhadap m.
5) Relatif prima terhadap m artinya faktor pembagi keduanya adalah 1, secara matematis disebut gcd (e,m) = 1. Untuk mencarinya dapat digunakan algoritma Euclid.
6) Menghitung d (kunci pribadi), untuk mencari nilai d secara matematis (e x d) mod m = 1. Dapat juga menggunakan algoritma Extended Euclid.

Maka hasil dari algoritma tersebut diperoleh:

1) Public key adalah pasangan (e,n).
2) Private key adalah pasangan (d,n).

### 2.4 Enkripsi Pesan

1) Menggunakan public key (e,n).
2) Plaintext M dinyatakan menjadi blok-blok m1, m2, m3,…
3) Setiap blok mi di enkripsi menjadi ci, dengan rumus ci = mie mod n.

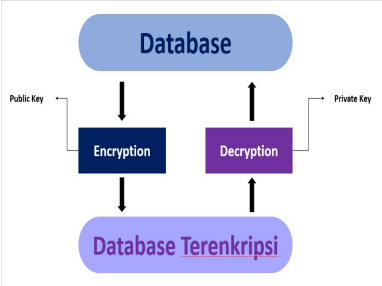
### 2.5 Dekripsi Pesan

1) Menggunakan private key (d,n).
2) Pilih ciphertext C.
3) Setiap blok ci di dekripsi menjadi blok mi, dengan rumus mi = cid mod n.

## 3 HASIL DAN PEMBAHASAN

### 3.1. Perancangan Program

Program yang akan dibuat mempunyai beberapa form, yaitu terdiri dari form main menu, generate key, encryption, decryption, help, help generate key, help encryption, help decryption dan about us. Untuk dapat melakukan enkripsi database pengguna dapat menggunakan form encryption dengan memilih database yang ingin di enkripsi. Namun, sebelum melakukan enkripsi memerlukan public key agar database bisa terenkripsi. Sedangkan untuk mengembalikan database yang sudah di enkripsi kembali seperti semula, pengguna juga dapat memilih form decryption, namun pengguna membutuhkan private key untuk mengembalikan database tersebut. Jika pengguna membutuhkan bantuan dalam menjalankan program ini, pengguna dapat melihatnya di menu help. Untuk mendapatkan public key dan private key pengguna dapat menggunakan form generate key. Berikut ini adalah arsitektur kerja aplikasinya.

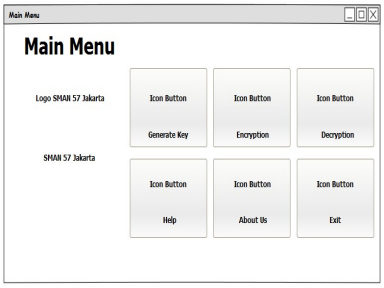



*Gambar 1. Arsitektur Kerja Aplikasi*

Rancangan menu sangat penting dalam tahap pembuatan program. Oleh karena itu rancangan menu yang dibuat harus mudah dipahami agar pengguna dapat langsung mengerti saat menggunakannya. Dalam program ini, akan di gambarkan rancangan menu masing-masing form, yaitu rancangan layar form main menu, generate key, encryption, decryption, help, help generate key, help encryption, help decryption dan about us.

### 3.2. Rancangan Layar form Main Menu

Pada rancangan layar form Main Menu, menampilkan beberapa tombol yang ada di Main Menu, yaitu terdiri dari tombol generate key, tombol encryption, tombol decryption, tombol help, tombol about us dan tombol exit. Rancangan layar form Main Menu dapat dilihat pada gambar 5. dibawah ini:

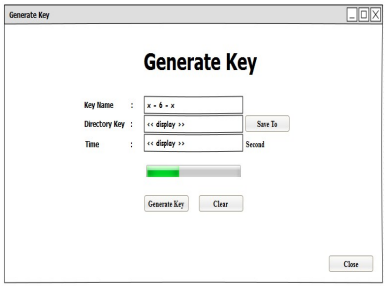



*Gambar 2. Rancangan Layar form Main Menu*

### 3.3. Rancangan Layar form Generate Key

Form yang berfungsi untuk membuat dua buah kunci, yaitu terdiri dari public key dan private key. Dua kunci ini nantinya akan digunakan pada saat melakukan enkripsi dan dekripsi database. Pengguna harus membuat nama kunci dan dilanjutkan dengan menyimpan kunci tersebut untuk disimpan di direktori yang diinginkan oleh pengguna. Rancangan

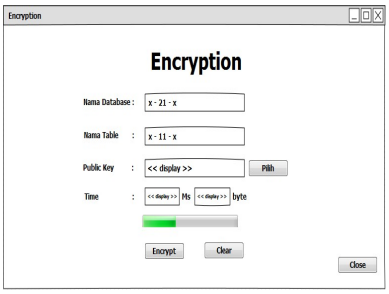
layar form generate key dapat dilihat pada gambar 6. dibawah ini:




*Gambar 3. Rancangan Layar form Generate Key*

### 3.4. Rancangan Layar form Encryption

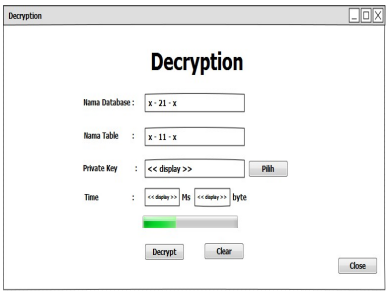
Pada rancangan layar form encryption, form ini berfungsi untuk melakukan enkripsi pada sebuah database. Pertama pengguna harus memasukan nama database terlebih dahulu, setelah itu pengguna memasukan nama table yang akan di enkripsi, selanjutnya pengguna memilih public key yang sebelumnya sudah dibuat di form generate key. Rancangan layar form encryption dapat dilihat pada gambar 7. dibawah ini:




*Gambar 4. Rancangan Layar form Encryption*

### 3.5. Rancangan Layar form Decryption

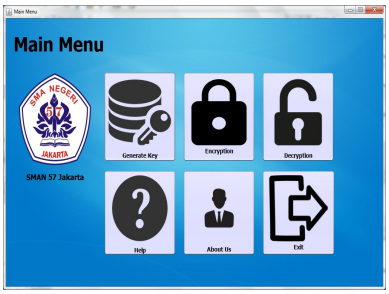
Pada rancangan layar form decryption, form ini berfungsi untuk melakukan dekripsi pada sebuah database. Sama seperti proses di form encryption, pertama pengguna harus memasukan nama database terlebih dahulu, setelah itu pengguna memasukan nama table yang mau di dekripsi, selanjutnya pengguna memilih private key yang sebelumnya sudah dibuat di form generate key. Rancangan layar form decryption dapat dilihat pada gambar 8. dibawah ini:




*Gambar 5. Rancangan Layar form Decryption*

### 3.6. Tampilan Layar pada form Main Menu

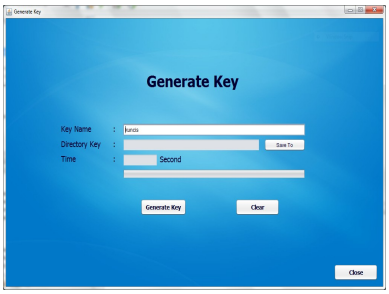
Tampilan layar dari form main menu ini muncul setelah pengguna berhasil login ke dalam aplikasi, di dalam form ini berisi beberapa menu yang terdiri dari Form Generate key, Form Encryption, Form Decryption, Form About us, Form Help, dan Exit. Berikut adalah tampilan layar dari form main menu pada gambar 9.:



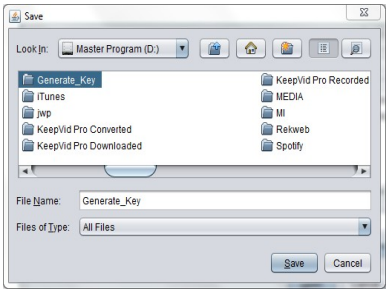

*Gambar 6. Tampilan Layar form Main Menu*

### 3.2 Tampilan Layar pada form Generate Key

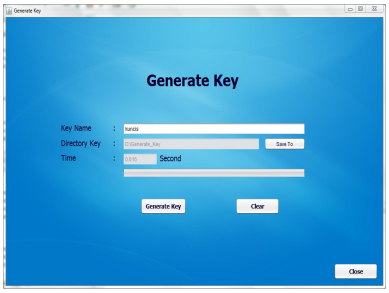
Di form ini pengguna dapat membuat kunci yang nantinya kunci ini bisa digunakan untuk melakukan enkripsi dan dekripsi database. Ada 2 kunci yang dihasilkan dari form ini, yaitu public key dan private key. Setelah pengguna mengisi key name, selanjutnya pengguna memilih lokasi penyimpanan kunci. Setelah pengguna sudah menentukan dimana public key dan private key disimpan, pengguna tinggal meng-klik tombol generate key. Setelah generate key berhasil dijalankan, maka di form generate key akan menampilkan informasi waktu berapa detik saat generate key selesai dijalankan. Berikut adalah tampilannya:




*Gambar 7.: Tampilan Layar form Generate Key*




*Gambar 8.: Memilih lokasi penyimpanan kunci*

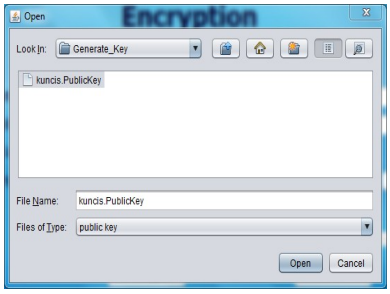



*Gambar 9: Tampilan informasi waktu generate key yang berhasil diproses*

### 3.3 Tampilan Layar pada form Encryption

Ini adalah tampilan layar dari form encryption, di form ini pengguna bisa mengenkripsi database yang ingin di enkripsi, pengguna mengisi semua form, dimulai dengan mengisi nama database, kemudian nama table yang ada di dalam database tersebut dan pengguna memilih public key di deriktori folder penyimpanan, dimana public key sebelumnya disimpan. Setelah public key dipilih langkah selanjutnya pengguna tinggal meng-klik tombol Encrypt untuk memulai proses enkripsi database, jika proses enkripsi telah selesai. Akan menampilkan berapa waktu yang dibutuhkan setelah proses enkripsi berhasil dijalankan. Berikut adalah tampilannya:


*Gambar 10. Tampilan Layar form Encryption*

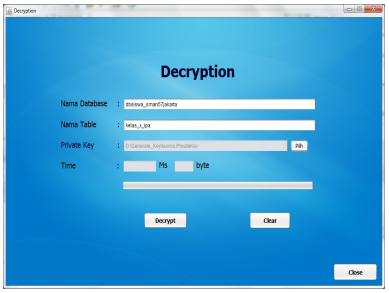



*Gambar 11. Memilih public key di direktori folder penyimpanan*

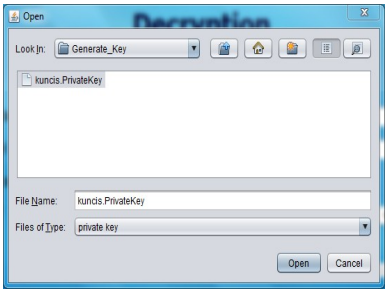

*Gambar 12. Tampilan form encryption setelah berhasil melakukan encrypt*

### 3.4 Tampilan Layar pada form Decryption

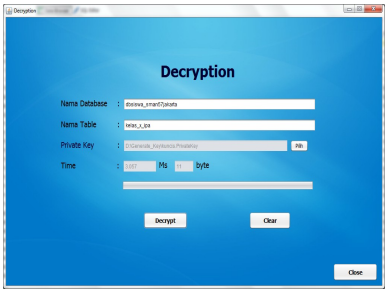
Ini adalah tampilan layar dari form decryption, di form ini pengguna bisa dekripsi database yang ingin di dekripsi, pengguna mengisi semua form, dimulai dengan mengisi nama database, kemudian nama table yang ada di dalam database tersebut dan pengguna memilih private key di deriktori folder penyimpanan, dimana private key sebelumnya disimpan. Setelah private key dipilih langkah selanjutnya pengguna tinggal meng-klik tombol Decrypt untuk memulai proses dekripsi database, jika proses dekripsi telah selesai. Akan menampilkan berapa waktu yang dibutuhkan setelah proses dekripsi berhasil dijalankan. Berikut adalah tampilannya:




*Gambar 13. Tampilan Layar form Decryption*



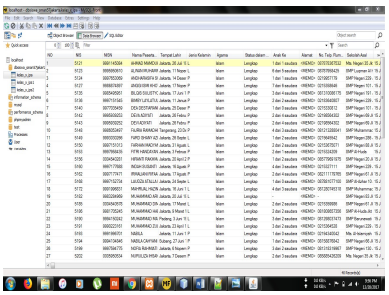

*Gambar 14. Memilih private key di direktori folder penyimpanan*




*Gambar 15. Tampilan form decryption setelah berhasil melakukan decrypt*

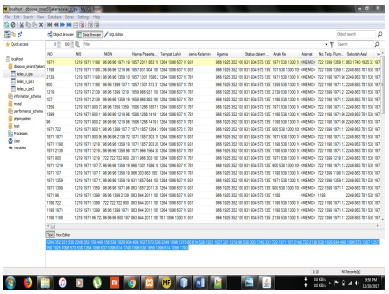
### 3.5 Proses Encryption dan Decryption pada Database

Untuk melakukan encryption dan decryption pada database, pengguna bisa melakukannya lewat form encryption dan form decryption yang sudah dijelaskan sebelumnya. Di bawah ini adalah tampilan table database yang ingin di enkripsi, yang ada pada gambar 19.:



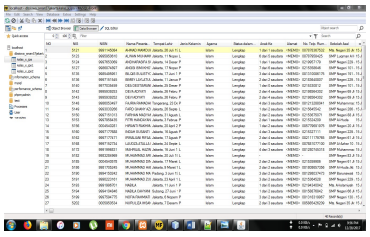
*Gambar 16. Table Database sebelum di Encrypt*

Setelah database berhasil dilakukan enkripsi, maka database yang sudah di enkripsi tidak akan bisa dibaca lagi seperti terlihat pada gambar dibawah ini:



*Gambar 17. Table Database setelah di Encrypt*

Jika database yang sebelumnya sudah dilakukan enkripsi dan ingin mengembalikannya seperti semula. Maka pengguna dapat melakukannya dengan dekripsi database tersebut, setelah database berhasil di dekripsi, maka database yang sudah di dekripsi akan bisa dibaca lagi dan kembali seperti semula. Seperti terlihat pada gambar dibawah ini:



*Gambar 18. Table Database setelah di dekripsi*

### 3.6 Tabel Pengujian

Berikut ini adalah perbandingan antara proses encrypt dan decrypt. Pengujiannya yaitu meliputi ukuran awal tabel pada database yang ingin di enkripsi, records per-tabel, key name, waktu proses encrypt dan decrypt, ukuran sebelum dan setelah database di encrypt maupun decrypt, dan status

yang di tampilkan dalam proses encrypt maupun decrypt.

Tabel 1. Hasil Pengujian Proses Encrypt

| Table | Column | Records | Key Name | Waktu (Ms) | Sebelum Encrypt | Sesudah Encrypt | Status |
|---|---|---|---|---|---|---|---|
| kelas_x_ipa | 20 | 40 | kuncis.public | 3.642 | 16.384 bytes | 98.304 bytes | Berhasil |
| kelas_x_ips1 | 20 | 40 | kuncis.public | 3.546 | 16.384 bytes | 98.304 bytes | Berhasil |
| kelas_x_ips2 | 20 | 38 | kuncis.public | 2.742 | 16.384 bytes | 98.304 bytes | Berhasil |

Tabel 2. Hasil Pengujian Proses Decrypt

| Table | Column | Records | Key Name | Waktu (Ms) | Hasil Encrypt | Setelah Decrypt | Status |
|---|---|---|---|---|---|---|---|
| kelas_x_ipa | 20 | 40 | kuncis.private | 3.349 | 98.304 bytes | 16.384 bytes | Berhasil |
| kelas_x_ips1 | 20 | 40 | kuncis.private | 3.858 | 98.304 bytes | 16.384 bytes | Berhasil |
| kelas_x_ips2 | 20 | 38 | kuncis.private | 4.907 | 98.304 bytes | 16.384 bytes | Berhasil |

### 3.7 Kelebihan dan Kekurangan Program

**a. Kelebihan Program**

1) Dapat digunakan oleh siapapun untuk mengamankan isi database.
2) Aplikasi ini mudah di operasikan.
3) Memiliki tampilan yang sederhana.
4) Database yang sudah di enkripsi tidak bisa dibaca isinya sebelum dilakukan dekripsi.
5) Isi database yang telah dilakukan dekripsi tidak mengalami perubahan setelah di dekrip atau bisa dibilang kembali seperti semula database-nya.

**b. Kekurangan Program**

1) Semakin besar ukuran database yang ingin di enkrip dan dekrip, maka proses enkrip maupun dekrip-nya semakin lama.
2) Waktu yang dibutuhkan untuk proses enkrip dan dekrip masih terbilang kurang cepat.
3) Pengguna yang menggunakan aplikasi ini harus mengerti tentang database.
4) Saat melakukan enkripsi dan dekripsi, waktu yang diperlukan dalam setiap proses berbeda-beda.

## 4 KESIMPULAN

Berdasarkan perancangan, pengembangan, serangkaian uji coba dan analisa program dari aplikasi pengamanan database ini. Maka dapat diambil suatu kesimpulan, diantaranya:

a. Dengan adanya aplikasi pengamanan database, proses pertukaran informasi serta penyimpanan data menjadi lebih aman dan nyaman.
b. Satu kunci public dan private dapat digunakan berulang kali dengan database yang berbeda.
c. Proses decrypt dengan kunci yang sesuai, akan mengembalikan table database kembali seperti semula tanpa mengalami perubahan sedikitpun.
d. Waktu yang digunakan untuk melakukan proses enkrip dan dekrip berbanding lurus dengan ukuran table database yang diproses (semakin kecil ukuran table yang akan diproses, maka akan semakin cepat proses enkripsi dan dekripsinya. Semakin besar ukuran table yang akan diproses, maka akan semakin lama proses enkripsi dan dekripsinya).

## 5 DAFTAR PUSTAKA

[1] Kromodimoeljo, S, 2010, Teori dan Aplikasi Kriptografi, Jakarta: SPK IT Consulting.