IMPLEMENTASI KRIPTOGRAFI *FILE* UJIAN SISWA DENGAN METODE RSA BERBASIS *WEBSITE* DI SMAN 84 JAKARTA

I Gusti Ayu Yogie Andhika Putri[1*], Noni Juliasari[2]

[1,2]Fakultas Teknologi Informasi, Teknik Informatika, Universitas Budi Luhur, Jakarta, Indonesia

Email: [1*]igustiayuyogie@gmail.com, [2]noni.juliasari@budiluhur.ac.id
(* : corresponding author)

Abstrak-SMA Negeri 84 Jakarta rutin mengadakan Ujian Akhir Semester (UAS) di setiap semesternya. Ujian Akhir Semester ini dilaksanakan secara serentak di seluruh jurusan, baik jurusan Matematika dan Ilmu Pengetahuan Alam (MIPA), Ilmu Pengetahuan Sosial (IPS), maupun Ilmu Bahasa dan Budaya (IBB), secara tertulis dan *close book*, yang dimaksudkan untuk mengetahui kemampuan seluruh siswa. Kemajuan dan perkembangan teknologi dapat menyebabkan isi dari *file* soal ujian di SMA Negeri 84 Jakarta rentan tersebarluaskan sebelum waktu ujian dimulai. Hal ini mengakibatkan terjadinya ketidakmurnian pada proses penilaian kemampuan siswa. Oleh karena itu, SMA Negeri 84 Jakarta membutuhkan suatu sistem yang dapat menjaga kerahasiaan isi dari *file* soal ujian tersebut. Penelitian ini menerapkan sistem keamanan *file* dengan menggunakan algoritme kriptografi *Rivest Shamir Adleman* (RSA) untuk mengamankan *file* soal ujian siswa berekstensi *.txt*. Dokumen yang dipakai pada proses penelitian ini adalah dokumen soal ujian siswa milik SMA Negeri 84 Jakarta. Hasil peneltian ini berupa suatu sistem aplikasi kriptografi pada *file* berekstensi *.txt* menggunakan algoritme kriptografi RSA. Berdasarkan hasil pengujian yang dilakukan, terdapat perubahan besar ukuran *file* enkripsi, yaitu peningkatan ukuran dari ukuran semula, namun tidak terjadi perubahan besar *file* hasil dekripsi dengan ukuran *file* awal. Kecepatan rata-rata saat melakukan enkripsi *file* adalah 46.8 ms, dan 56.2 ms untuk melakukan dekripsi.

Kata Kunci: Enkripsi, Dekripsi, Kriptografi, RSA, Soal Ujian

CRYPTOGRAPHY IMPLEMENTATION OF STUDENT EXAM FILE USING RSA METHOD AND WEBSITE-BASED AT SMAN 84 JAKARTA

Abstract- *SMA Negeri 84 Jakarta routinely holds the Final Semester Examination in every semester. This Semester Final Examination is carried out simultaneously in all departments, both in Mathematics and Natural Sciences, Social Sciences, and Language and Culture Sciences, in writing and close book, which is intended to determine the ability of all students. Technological advances and developments can cause the contents of the exam question file at SMA Negeri 84 Jakarta to be vulnerable to being disseminated before the exam time begins. This results in impurity in the process of assessing student abilities. Therefore, SMA Negeri 84 Jakarta needs a system that can maintain the contents' confidentiality of the exam file. This research implements a file security system using the Rivest Shamir Adleman (RSA) cryptographic algorithm to secure student exam questions with a .txt extension. The document used in this study is a document about student exams belonging to SMA Negeri 84 Jakarta. The result of this study is an application system for encrypting and decrypting files with .txt extension using RSA cryptography method. Based on the test results, there was a change in the size of the encrypted file to be larger than the original file size, but there was no change at the size of the decrypted file with the initial file size. The average speed when encrypting files is 46.8 ms, and 56.2 ms for decrypting.*

Keywords*: Encryption, Decryption, Cryptography, RSA, Exam File*

1. PENDAHULUAN

Teknologi komputerisasi dan informasi berkembang pesat seiring berjalannya waktu dan memberikan dampak positif serta manfaat, yang sangat memudahkan manusia dalam melakukan pekerjaan sehari-hari. Di samping banyaknya manfaat dan dampak positif, terdapat pula dampak negatif dari kemajuan teknologi yang menjadi ancaman, seperti pencurian dan penyebarluasan data. Kriptografi merupakan salah satu upaya untuk menjaga kerahasiaan dan keamanan data.

SMA Negeri 84 Jakarta merupakan lembaga pendidikan formal milik pemerintah yang terletak di Jalan Peta Barat Nomor 42 Kalideres, Jakarta Barat. Sekolah ini rutin melaksanakan ujian secara *closed book* sebagai bentuk penilaian terhadap para siswa. Ujian ini dilaksanakan untuk mengetahui dan mengukur kemampuan setiap siswa. Oleh karena itu, keamanan dan kerahasiaan isi dari dokumen soal ujian siswa harus terjaga. Sebelum waktu ujian dimulai, isi dokumen soal ujian siswa hanya boleh diketahui oleh pihak yang memiliki otoritas, seperti guru atau staf bidang yang berwenang. Namun, canggihnya teknologi membuat *file* soal ujian rentan tersebarluaskan. Hal

ini mengakibatkan isi dokumen tersebut tidak lagi bersifat rahasia dan menimbulkan ketidakmurnian hasil penilaian siswa. Menanggapi permasalahan yang terjadi, SMAN 84 Jakarta memerlukan suatu aplikasi atau sistem untuk mengamankan isi *file* tersebut, yaitu dengan melakukan enkripsi dan dekripsi pada *file* soal ujian.

Beberapa rumusan masalah yang diambil adalah proses pengamanan *file* yang dilakukan, dan bagaimana sistem dapat memudahkan diakses oleh banyak *user* yang memiliki otoritas. Tujuan dalam melakukan penelitian ini adalah membangun aplikasi untuk menjaga serta meningkatkan keamanan maupun kerahasiaan data dengan enkripsi dan dekripsi *file* soal ujian siswa menggunakan metode RSA, yang memiliki tiga proses dalam mengamankan data, yaitu proses *generate key*, proses enkripsi, dan proses dekripsi. Selain itu, aplikasi juga dibuat dengan *web-based* dengan otorisasi *user* sehingga memudahkan untuk diakses oleh banyak *user*.

Kriptografi merupakan metode menulis pesan menggunakan aksara khusus, seperti kode, sandi, dan memiliki kunci yang hanya diketahui oleh pihak yang dapat memprosesnya. Kriptografi merupakan salah satu ilmu yang membahas masalah keamanan komputer seperti menyembunyikan pesan, kerahasiaan data, keutuhan data dan otentikasi entitas [1]. Kriptografi digunakan untuk mengamankan suatu pesan dengan menggunakan berbagai teknik. Secara umum, kriptografi dapat didefinisikan juga sebagai suatu teknik dalam mengamankan data atau informasi dengan menggunakan kode, karakter, atau sandi umum yang hanya dapat diketahui oleh pihak yang berotoritas.

Algoritma RSA dibuat oleh 3 orang peneliti dari MIT (Massachussets Institute of Technology) pada tahun 1976, yaitu: Ron (R)ivest, Adi (S)hamir, dan Leonard (A)dleman [2]. Algoritme RSA termasuk kriptografi asimetris, yang memerlukan kunci berbeda dalam melakukan proses enkripsi dan proses dekripsi. Algoritme RSA memanfaatkan dua buah bilangan prima untuk kemudian didapatkan sebuah *public key* yang berfungsi untuk mengenkripsi teks, dan sebuah *private key* yang berfungsi untuk mendekripsi teks. RSA mendasarkan proses enkripsi dan dekripsi pada konsep bilangan prima dan aritmatika modulo [3]. Sulitnya untuk melakukan pemfaktoran bilangan besar menjadi bilangan faktor prima merupakan titik keamanan algoritme RSA.

Berdasarkan penelitian sebelumnya dengan judul "Penerapan Kriptografi RSA Dalam Mengamankan *File* Teks Berbasis PHP", algoritma kriptografi RSA didesain sesuai fungsinya sehingga kunci yang digunakan untuk enkripsi berbeda dari kunci yang digunakan untuk dekripsi. Kunci untuk enkripsi pesan disebut publik, sedangkan kunci untuk mendekripsi pesan yang diterima disebut private [4].

Pada penelitian sebelumnya berjudul "Aplikasi Kriptografi Menggunakan Algoritma RSA Dan *Corrected Block TEA* (XXTEA)", algoritma RSA disebut kunci publik karena kunci enkripsi dapat dibuat publik yang berarti semua orang boleh mengetahuinya, namun hanya orang tertentu (si penerima pesan sekaligus pemilik kunci dekripsi yang merupakan pasangan kunci publik) yang dapat melakukan dekripsi terhadap pesan tersebut [5].

Pada penelitian berjudul "Implementasi Kriptografi RSA untuk Peningkatan Keamanan *Database E-commerce*", proses pemfaktoran bilangan besar menjadi bilangan faktor-faktor prima yang sulit merupakan letak keamanan pada algortime kriptografi RSA. Pemfaktoran dilakukan untuk mendapatkan nilai kunci pribadi. Selama pemfaktoran bilangan besar menjadi faktor-faktor prima belum ditemukan algoritma yang mangkus, maka selama itu pula keamanan algoritma RSA tetap terjamin [6].

Berdasarkan penelitian sebelumnya dengan judul "Implementasi Teknik Kriptografi RSA untuk Pengamanan Data Pengiriman SMS", algoritma RSA memiliki keamanan karena pemfaktoran bilangan prima yang dinilai sulit. Pemfaktoran bilangan prima yang cukup besar inilah yang membuat algoritma RSA belum dapat dipecahkan dengan algoritma yang lainnya sehingga sangat dijamin keamanannya [7].

Pada penelitian sebelumnya yang berjudul "Implementasi Algoritma RSA Terhadap Keamanan Data Simpan Pinjam", tingginya keamanan algoritma kriptografi RSA sangat ditentukan oleh besarnya ukuran kunci. Semakin besar ukuran kunci, maka semakin sulit pembobolan terjadi. Kombinasi kunci ini lebih dikenal dengan istilah *brute fore attack* yang bila panjang kuncinya 256 bit, maka menjadi tidak ekonomis dan sia-sia jika *hacker* pun meyadap sandi [8].

Berdasarkan penelitian sebelumnya yang berjudul "Implementasi Algoritma Kriptografi RSA (Rivest Shamir Adleman) untuk Keamanan Data Rekam Medis Pasien", Algoritma kriptografi RSA merupakan salah satu algoritme kriptografi kunci asimetris, di mana memerlukan dua buah kunci yang berbeda untuk melakukan proses enkripsi dan dekripsi. Proses enkripsi memerlukan *public key* sedangkan proses dekripsi memerlukan *private key*. Pembangkitan pasangan kunci dilakukan sebelum melakukan proses enkripsi, yang membutuhkan 2 bilangan bulat prima secara acak, dalam hal ini diharuskan menggunakan sebuah pembangkit yang secara otomatis generasi bilangan tersebut dikarenakan angka-angka tersebut cukup besar dan banyak untuk memperoleh keamanan yang lebih tinggi dan lebih baik [9].

2. METODE PENELITIAN

2.1 Penerapan Metode

Penerapan metode merupakan proses atau tahap demi tahap yang dilalui dari tahap analisis kebutuhan sistem sampai pada tahap pemeliharaan sistem. Penelitian ini menggunakan metode *waterfall* sebagai metode dalam perancangan dan pengembangan aplikasi. Metode *waterfall* memiliki tahap-tahap, antara lain:

a. Analisis kebutuhan, analisis kebutuhan memiliki tujuan untuk memahami kebutuhan *user* terhadap perangkat lunak sebelum melakukan pengembangan sistem. Kebutuhan *user* terhadap sistem menentukan sifat program yang dibangun. Informasi ini didapatkan melalui observasi, dan wawancara di SMA Negeri 84 Jakarta untuk selanjutnya digunakan dalam pengembangan sistem.
b. Desain sistem, perancangan desain sistem merupakan tahapan untuk mengubah hasil analisis kebutuhan *user* terhadap sistem menjadi bentuk *mockup* program. Perancangan desain sistem dilakukan untuk memberikan gambaran mengenai keseluruhan proses pembuatan dan pengembangan sistem, yang merupakan implementasi dari hasil analisis kebutuhan *user* sebelumnya. Perancangan desain ini memberikan gambaran mengenai algoritma yang digunakan, maupun tampilan dan antarmuka sistem yang dibangun.
c. Pembuatan kode program, merupakan tahapan dalam melakukan pengkodingan program untuk menjadi sebuah sistem yang lengkap berdasarkan desain sistem yang sudah dibuat sebelumnya. Desain sistem diimplementasikan menjadi sebuah aplikasi sesuai yang dibutuhkan. Pembuatan kode program tersebut dilakukan menggunakan bahasa pemrograman PHP, dan CSS untuk tampilan aplikasi.
d. Pengujian sistem, dilakukan untuk memeriksa apakah sistem sudah berfungsi dengan baik. Seluruh fungsi yang ada di dalam sistem diuji coba untuk memastikan agar sistem tidak terdapat *error* atau *bug* sehingga dapat digunakan sesuai dengan fungsi yang semestinya. Pengujian juga dilakukan untuk mencegah atau memperbaiki kesalahan yang terjadi pada sistem.
e. Pemeliharaan sistem, tahap pemeliharaan dilakukan untuk melakukan perbaikan sistem jika ditemukan suatu kesalahan saat digunakan oleh *user*. Selain itu, pemeliharaan sistem memungkinkan sistem untuk dikembangkan lebih jauh sesuai dengan kebutuhan yang diinginkan.

Penelitian ini menggunakan metode kriptografi RSA. Metode RSA memiliki tiga tahapan dalam melakukan enkripsi dan dekripsi. Berikut adalah tahapan pada metode RSA :

a. Tahap pembangkitan kunci
   Proses pembangkitan kunci dilakukan menggunakan dua buah bilangan bulat prima, yang bertujuan untuk memperoleh pasangan kunci *public* dan *private*. Berikut proses pembangkitan kunci, yaitu :
   1. Tentukan dua bilangan bulat prima sembarang, diinisialkan dengan p dan q secara acak
   2. Hitung nilai n = p x q.
   3. Cari nilai m = (p – 1) (q – 1)
   4. Tentukan nilai e dengan e > 1, dan GCD(e,m) = 1
   5. Tentukan nilai d dengan (d.e) mod m = 1

Berdasarkan proses di atas, maka diperoleh:

   1. Kunci *public*, yaitu pasangan kunci (e,n).
   2. Kunci *private*, yaitu pasangan kunci (d,n).

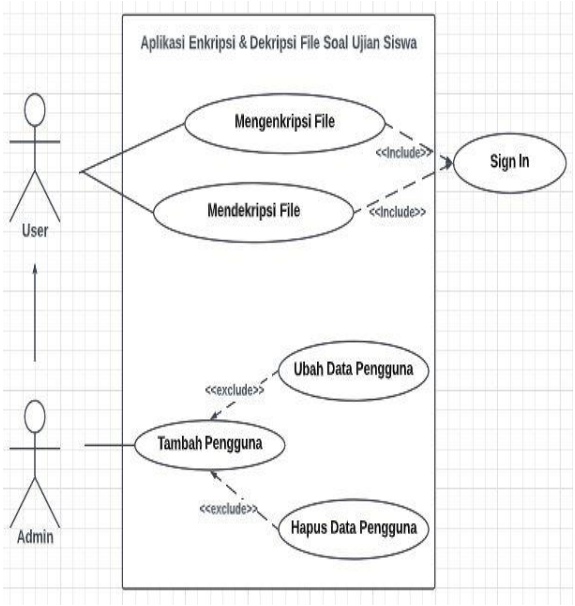
b. Tahap enkripsi
   Enkripsi merupakan tahap untuk mengubah data atau pesan menjadi suatu kode yang hanya dapat dipahami oleh pihak pemegang kunci. Berikut adalah proses enkripsi menggunakan kunci publik :
   1. Menggunakan kunci publik (e,n).
   2. Menggunakan rumus $C = M^e \bmod n$.
c. Tahap dekripsi
   Dekripsi merupakan tahap untuk mengubah kembali pesan yang sudah terenkripsi menjadi pesan semula menggunakan kunci privat yang sudah digunakan sebelumnya. Berikut adalah proses dari tahap dekripsi :
   1. Menggunakan kunci pribadi (d,n).
   2. Tentukan *ciphertext C*.
   3. Menggunakan rumus $M = C^d \bmod n$.

2.2 Perancangan Sistem

2.2.1 *Use Case Diagram*

Use case diagram menggambarkan suatu hubungan yang terdapat antara pengguna, admin atau aktor dengan sistem. Aktor pada sistem aplikasi enkripsi dan dekripsi *file* ini adalah admin dan pengguna aplikasi. *Use case diagram* juga menggambarkan fungsi yang terdapat di dalam sistem serta hubungannya dengan aktor. Selain

itu, *use case diagram* digunakan sebagai gambaran dalam mempresentasikan interaksi antara pengguna dengan sistem. Pada *use case diagram* di bawah ini, digambarkan bahwa admin dapat melakukan seluruh aktivitas atau fungsi di dalam sistem. Namun, *user* tidak dapat melakukan seluruh fungsi dalam sistem, yaitu tidak dapat menambahkan, mengubah, serta menghapus data pengguna. Gambar *use case diagram* sistem terdapat pada Gambar 1 di bawah ini.



Gambar 1. *Use Case Diagram*

2.2.2 Rancangan Basis Data

Sistem aplikasi enkripsi dan dekripsi *file* yang dibuat menggunakan basis data untuk menyimpan data pengguna. Basis data yang dibuat hanya terdiri dari satu *table*, yaitu *table* admin. *Table* admin terdiri dari enam *field*, yaitu *id_admin*, *username*, *password*, nama_lengkap, no_telp, blokir, dan *level*. Yang berperan sebagai *primary key* adalah *id_admin*. Sedangkan, *level* berperan dalam menentukan *level* dari keseluruhan pengguna, apakah pengguna tersebut adalah admin atau *user*. Spesifikasi mengenai rancangan basis data dapat dilihat pada Table 1.

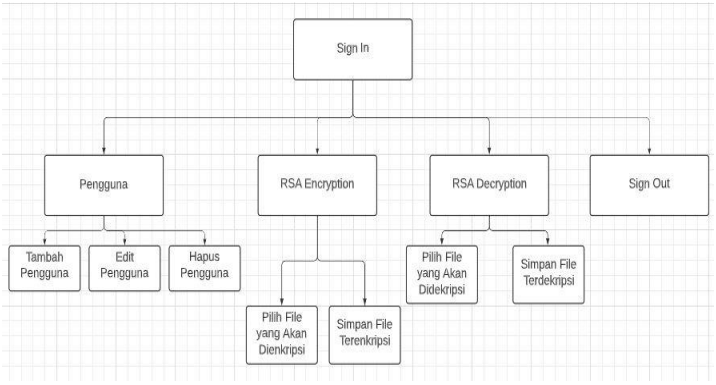
Tabel 1. Table Rancangan Basis Data

Nama *Field*	Tipe Data	Keterangan
id_admin	int(3)	*Primary Key*
username	varchar(100)	
password	varchar(100)	
nama_lengkap	varchar(100)	
no_telp	varchar(100)	
blokir	enum(Y,N)	
level	varchar(25)	

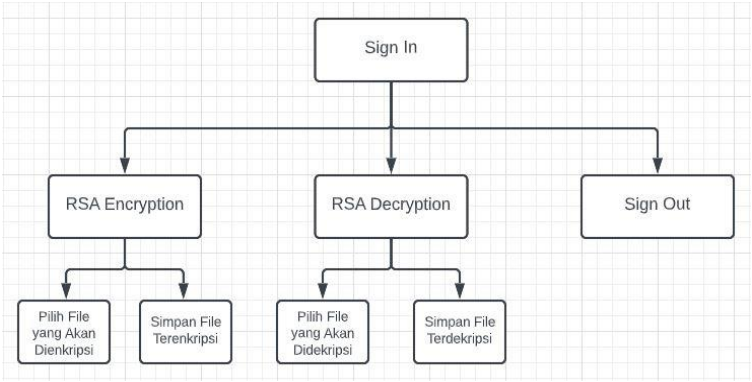
2.2.3 Rancangan Menu

Rancangan menu yang dibuat di dalam sistem aplikasi enkripsi dan dekripsi *file* terdiri dari rancangan menu untuk admin, dan rancangan menu untuk *user*. Admin memiliki beberapa menu yang dapat diakses, yaitu *Sign In*, pengguna, RSA *Encryption*, RSA *Decryption*, dan *Sign Out*. Di dalam menu pengguna, admin dapat

menambahkan, meng*edit*, dan menghapus data pengguna aplikasi ini. Sedangkan pada menu *user*, tidak terdapat menu pengguna seperti yang ada pada admin. Gambar rancangan menu tampilan admin dan *user* disajikan pada Gambar 2 dan Gambar 3.



Gambar 2. Rancangan Menu pada Tampilan Admin



Gambar 3. Rancangan Menu pada Tampilan *User*

3. HASIL DAN PEMBAHASAN

3.1 Analisis Masalah

Selain meningkatkan kecerdasan dan pengetahuan siswa, SMA Negeri 84 Jakarta juga fokus pada pembentukan kedisiplinan, kepribadian dan kemuliaan akhlak. Menurut wawanacara dan observasi, sering terjadi kebocoran dokumen soal ujian sehingga para siswa dapat mengetahui isi dari dokumen tersebut sebelum waktu ujian tiba. Kebocoran data soal ujian dapat menimbulkan indikasi kecurangan yang dilakukan oleh murid atau guru. Selain itu, bocornya data ujian menyebabkan ketidakmurnian nilai yang didapat oleh para siswa sehingga proses penilaian yang dilakukan oleh para guru menjadi tidak sesuai dengan kenyataan lapangan.

3.2 Penyelesaian Masalah

Oleh karena pemasalahan tersebut, peneliti mengambil kesimpulan sebagai evaluasi pada SMA Negeri 84 Jakarta untuk meningkatkan keamanan dokumen soal ujian siswa guna meminimalisir kecurangan dan ketidak murnian dalam proses penilaian yang terjadi dengan membuat suatu sistem aplikasi berbasis *website* dengan menggunakan implementasi kriptografi *Rivest Shamir Adleman* (RSA) untuk mengamankan isi dokumen ujian siswa di SMA Negeri 84 Jakarta. Sistem aplikasi ini mengenkripsi isi dari dari dokumen menjadi kode yang membutuhkan kunci khusus untuk dapat mengembalikannya menjadi seperti semula tanpa adanya perubahan.

3.3 Implementasi Metode RSA

Algoritme RSA mempunyai tiga tahapan, yaitu tahap pembangkitan pasangan kunci, tahap enkripsi, dan dekripsi. Impelementasi metode merupakan implementasi atau contoh penggunaan dari metode yang digunakan. Implementasi ketiga proses tersebut dijabarkan sebagai berikut :

a. Tahap Pembangkitan Pasangan Kunci

Berikut adalah proses dalam mendapatkan pasangan kunci [10]:

1. Menentukan dua bilangan prima acak yang diinisialkan dengan p dan q. Kedua nilai tersebut harus dirahasiakan.

$$p = 3, \quad q = 11 \tag{1}$$

2. Menghitung nilai n dengan rumus (2)

$$n = p\ x\ q \tag{2}$$

$n = 3\ x\ 11$
$n = 33$

3. Menghitung nilai m dengan rumus (3)

$$m = (p-1)(q-1) \tag{3}$$

$m = (3-1)(11-1)$
$m = 20$

4. Tentukan nilai e dengan rumus (4)

$$GCD(e,m) = 1, \quad e > 1 \tag{4}$$

Misal $e = 7$
Buktikan $GCD(7{,}20) = 1$
$20\ mod\ 7 = 6$
$7\ mod\ 6 = 1$
$6\ mod\ 1 = 0$

Angka 7 dapat digunakan sebagai nilai e karena $GCD(7{,}20) = 1$

5. Tentukan nilai d dengan rumus (5)

$$(d,e)\ mod\ m = 1 \tag{5}$$

Misal $d = 3$
Buktikan $(3{,}7) mod\ 20 = 1$
$(3x7) mod\ 20 = 1$

Angka 3 digunakan sebagai nilai d karena syarat terpenuhi.

Berdasarkan proses di atas, maka diperoleh:

1. Kunci publik adalah pasangan kunci (e,n).
   Kunci publik = (7,33)
2. Kunci privat adalah pasangan kunci (d,n).
   Kunci privat = (3, 33)

b. Proses Enkripsi

Proses enkripsi memerlukan kunci publik yang dapat dihitung dengan langkah-langkah berikut:

1. Menggunakan kunci publik (e,n).
2. Tentukan *plaintext* (M)
3. $M = 19$.
4. Enkripsi enggunakan rumus (6).

$$C\ =\ M^e\ mod\ n \tag{6}$$

$C\ =\ 19^7\ mod\ 33$
$C\ =\ 893871739\ mod\ 33$
$C\ =\ 13$

c. Proses Dekripsi

Pada proses dekripsi pesan, algoritma RSA menggunakan kunci privat sebagai berikut:

1. Menggunakan kunci pribadi (d,n).
2. Tentukan *ciphertext C.*
3. Dekripsi menggunakan rumus (7).

$$M\ =\ C^d\ mod\ n \tag{7}$$

$M\ =\ 13^3\ mod\ 33$
$M\ =\ 2197\ mod\ 33$
$M\ =\ 19$

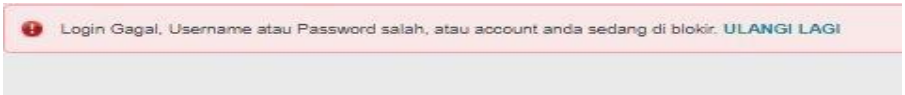
Berdasarkan hasil perhitungan di atas, diperoleh angka 19 yang merupakan *plaintext* awal sehingga dipastikan proses perhitungan sudah benar.

3.4 Tampilan Layar

3.4.1 Tampilan Layar *Sign In*

Di bawah ini adalah tampilan dari layar *sign in*. Untuk melalui proses *sign in*, pengguna harus didaftarkan terlebih dahulu oleh admin karena tidak sembarang orang diperbolehkan menggunakan aplikasi ini. Jika pengguna gagal melakukan proses *sign in*, maka muncul notifikasi gagal *sign in*, dan pengguna dipersilahkan untuk membuat akun terlebih dahulu atau melakukan proses *sign in* kembali. Tampilan layar *sign in* dan tampilan gagal *sign in* dapat dilihat pada Gambar 4 dan Gambar 5.

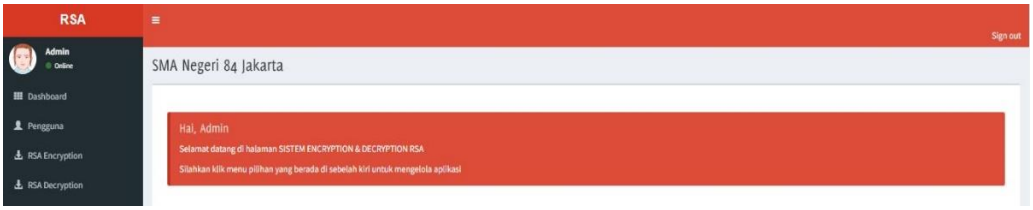
Gambar 4. Tampilan Layar *Sign In*



Gambar 5. Tampilan Layar Gagal *Sign In*

3.4.2 Tampilan Layar Halaman Utama (*Dashboard*)

Berikut adalah tampilan layar *dashboard* dari tampilan admin, dan pengguna. Tampilan ini merupakan tampilan awal aplikasi, di mana hanya terdapat kalimat selamat datang. Pada tampilan admin, terdapat menu pengguna, *RSA encryption*, dan *RSA decryption.* Sedangkan pada tampilan *user*, *user* hanya dapat melihat menu *RSA encryption*, dan *RSA decryption.* Tampilan layar *dashboard* pada tampilan admin dan *user* terdapat pada Gambar 6 dan Gambar 7.

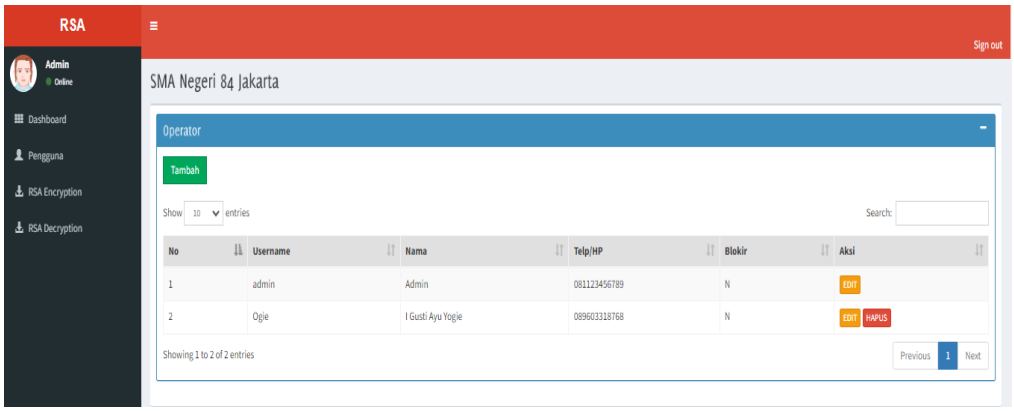


Gambar 6. Tampilan Layar Halaman Utama pada Tampilan Admin

Gambar 7. Tampilan Layar Halaman Utama pada Tampilan *User*

3.4.3 Tampilan Layar Menu Pengguna pada Tampilan Admin

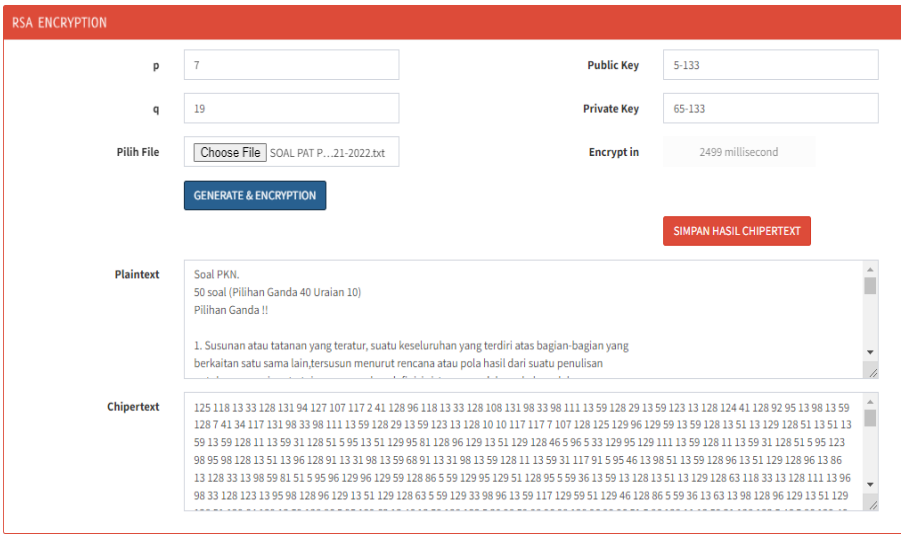
Menu pengguna ini hanya terdapat pada tampilan admin. Layar pengguna terdiri dari *form* tambah pengguna, dan *form* ubah pengguna. Halaman ini memungkinkan admin untuk menambahkan, meng*edit*, dan menghapus data. Berikut adalah tampilan layar halaman pengguna pada Gambar 8.



Gambar 8. Tampilan Layar *Form* Pengguna pada Tampilan Admin

3.4.4 Tampilan Layar Uji Coba Enkripsi *File* pada Menu RSA *Encryption*

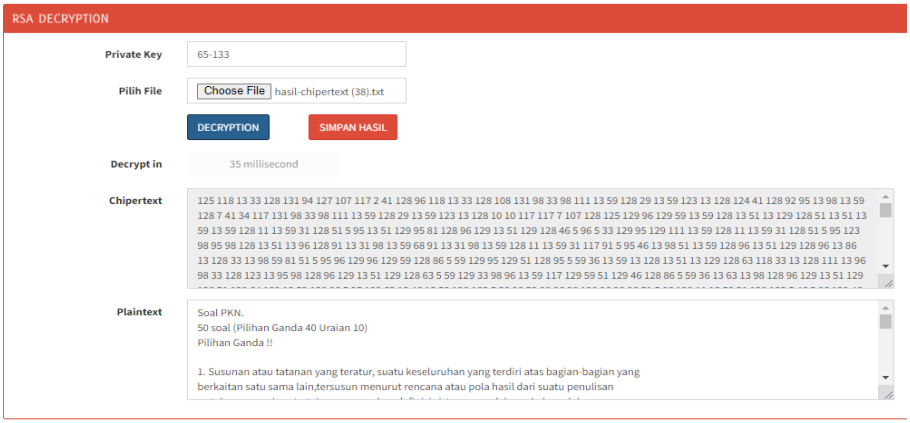
Tampilan layar menu *RSA encryption* pada tampilan admin dan tampilan *user* tidak memiliki perbedaan. Pada halaman ini, pengguna dapat melakukan enkripsi *file* berformat *.txt* dan menyimpan *file* hasil enkripsi tersebut. Untuk melakukan enkripsi *file*, pengguna perlu memasukkan nilai p dan q, kemudian memilih *file* yang ingin dienkripsi. Setelah itu, tekan *button* '*generate* & *encryption'* untuk menemukan nilai dari *public key* dan *private key* yang dipakai, serta menampilkan hasil enkripsi pada aplikasi. *File* hasil enkripsi dapat disimpan dengan menekan *button* 'Simpan Hasil *Ciphertext'* Tampilan layar uji coba enkripsi *file* berformat *.txt* dapat dilihat pada Gambar 9.



Gambar 9. Tampilan Layar Uji Coba Enkripsi *File* pada Menu RSA *Encryption*

3.4.5 Tampilan Layar Uji Coba Dekripsi *File* pada Menu RSA *Decryption*

Tampilan layar menu *RSA decryption* pada tampilan admin dan tampilan *user* tidak memiliki perbedaan. Pada halaman ini, pengguna dapat melakukan dekripsi *file* berformat *.txt* yang sudah dienkripsi sebelumnya. Untuk melakukan dekripsi *file*, pengguna perlu memasukkan nilai *private key* yang sama seperti saat melakukan proses enkripsi, kemudian memilih *file* yang ingin didekripsi. Setelah itu, tekan *button* '*decryption'* untuk mendapatkan dan menampilkan hasil dekripsi pada aplikasi. *File* hasil dekripsi dapat disimpan dengan menekan *button* 'Simpan Hasil'. Tampilan layar uji coba dekripsi *file* berformat *.txt* dapat dilihat pada Gambar 10.



Gambar 10. Tampilan Layar Uji Coba Dekripsi *File* pada Menu RSA *Decryption*

3.5 Pengujian Sistem

Berdasarkan hasil pengujian yang dilakukan dengan menguji 5 *file* soal Penilaian Akhir Tahun SMA Negeri 84 Jakarta, maka peneliti mendapatkan hasil perbandingan seperti yang tersaji pada Tabel 2.

Tabel 2. Table Perbandingan Hasil Pengujian *File* Soal Ujian Siswa

No	File Asli		Setelah Enkripsi		Setelah Dekripsi		Waktu		Perubahan Ukuran
	Nama File	Ukuran File	Nama File	Ukuran File	Nama File	Ukuran File	Enkripsi	Dekripsi	
1	SOAL PAT KLS X BAHASA INGGRIS 2021-2022.txt	19 KB	hasil-ciphertext .txt	60 KB	hasil-plaintext .txt	19 KB	64 ms	85 ms	315.79%
2	SOAL PAT PKN KLS XI TH 2021-2022.txt	10 KB	hasil-ciphertext (1).txt	30 KB	hasil-plaintext (1).txt	10 KB	34 ms	49 ms	300%
3	Indo pg xii.txt	15 KB	hasil-ciphertext (2).txt	46 KB	hasil-plaintext (2).txt	15 KB	58 ms	62 ms	306.67%
4	Sej-10.txt	14 KB	hasil-ciphertext (3).txt	42 KB	hasil-plaintext (3).txt	14 KB	45 ms	48 ms	300%
5	penjas pat 11.txt	10 KB	hasil-ciphertext (4).txt	32 KB	hasil-plaintext (4).txt	10 KB	33 ms	37 ms	320%
Rata-rata							46.8 ms	56.2 ms	308.492%

Berdasarkan hasil pengujian lima dokumen soal ujian di atas, *file* hasil enkripsi mengalami perbesaran ukuran dari ukuran *file* semula, dengan persentase rata-rata perbesaran ukuran adalah 308.492%. *File* hasil dekripsi tidak mengubah ukuran asli *file*. Kecepatan rata-rata saat melakukan enkripsi adalah 46.8 ms, sedangkan kecepatan rata-rata yang diperlukan untuk melakukan dekripsi adalah sebesar 56.2 ms.

3.6 Evaluasi

Berdasarkan hasil pengujian sistem serta spesifikasi program aplikasi, aplikasi ini memiliki beberapa kelebihan dan kelemahan. Berikut merupakan kelebihan dan kelemahan yang terdapat pada program aplikasi enkripsi dan dekripsi *file*:

3.6.1 Kelebihan Sistem

a. Program dapat berjalan secara *cross platform*, seperti Windows, Mac OS, dan Linux.
b. Proses dalam melakukan enkripsi dan dekripsi pada aplikasi berjalan dengan lancar.

c. Ukuran *file* hasil dekripsi tidak mengubah ukuran *file* semula.
d. Keaslian dan kerahasiaan isi *file* terjamin dengan aman.

3.6.2 Kekurangan Sistem

a. Program hanya dapat melakukan enkripsi dan dekripsi *file* dengan ukuran maksimum sebesar 2 MB.
b. Nama *file* yang sudah dienkripsi dan didekripsi tidak dapat langsung di*rename* pada aplikasi.
c. *File* yang sudah dienkripsi mengalami peningkatan ukuran.

4. KESIMPULAN

Berdasarkan pengkajian di atas, penelitian ini memiliki konklusi yang dapat ditarik sebagai berikut : aplikasi kriptografi *file* soal ujian siswa dengan metode RSA yang dibuat mampu melakukan enkripsi dan dekripsi *file* soal yang diujikan, dengan karakter berupa huruf dan angka. Aplikasi ini memiliki fitur pengguna yang hanya dapat digunakan oleh admin untuk menambahkan, mengubah, atau menghapus akun pengguna. Proses enkripsi *file* yang dilakukan pada aplikasi ini memerlukan waktu yang lebih cepat daripada proses dekripsi. Aplikasi ini memerlukan kecepatan rata-rata selama 46.8 ms untuk melakukan enkripsi, sedangkan untuk melakukan dekripsi, aplikasi ini memerlukan kecepatan proses rata-rata 56.2 ms. *File* hasil enkripsi pada aplikasi ini mengalami peningkatan ukuran *file* dengan rata-rata peningkatan sebesar 308.49% dari *file* semula, sedangkan ukuran *file* hasil dekripsi tidak mengalami perubahan dari ukuran *file* semula.

Aplikasi ini masih memiliki kekurangan sehingga memerlukan perbaikan dan pengembangan lebih lanjut. Beberapa masukan yang dapat ditambahkan untuk mengembangkan dan menyempurnakan aplikasi ini, yaitu : aplikasi ini diharapkan untuk dapat disempurnakan lebih jauh dengan memperbanyak ekstensi atau format *file* yang dapat dienkripsi dan didekripsi, seperti *.pdf, .doc, .xls,* dan ekstensi lainnya, serta dapat melakukan enkripsi dan dekripsi *file* berupa gambar dan simbol.

DAFTAR PUSTAKA

[1] R. R. Putra dan V. Tasril, "Implementasi Kriptografi RSA Dalam Pengamanan Data Supplier Bahan Baku Pada Pt. Sinar Aneka Niaga," *Jurnal Teknovasi*, vol. 6, no. 2, pp. 60-66, 2019.
[2] Wahyudi, D. Hartama, I. O. Kirana, Sumarno dan I. Gunawan, "Implementasi Algoritma Kriptografi Rivest Shamir Adlemen untuk Mengamankan Data Ijazah pada SMK Swasta Prama Artha Kab. Simalungun," *Jurnal Ilmu Komputer dan Informatika (JIKI)*, vol. 2, no. 1, pp. 57-66, 2022.
[3] R. Firmansyah dan A. A. Permana, "Implementasi Keamanan Pesan Teks Menggunakan Kriptografi Algoritma RSA Dengan Metode Waterfall Berbasis Java," *JOUTICA*, vol. 4, no. 1, pp. 217-221, 2019.
[4] D. Apdilah dan H. Swanda, "Penerapan Kriptografi RSA Dalam Mengamankan File Teks Berbasis PHP," *Jurnal Teknologi Informasi.*, vol. 2, no. 1 pp. 45–52, 2018.
[5] M. Anif, Siswanto, M. Fikri, dan G. P. Utama, "Aplikasi Kriptografi Menggunakan Algoritma RSA Dan *Corrected Block TEA* (XXTEA)," *Jurnal BIT (Budi Luhur Information Technology)*, vol. 16, no. 2, pp. 13-22, 2019.
[6] S. Rahmadhiyanti, "Implementasi Kriptografi RSA Untuk Peningkatan Keamanan Database E-Commerce," *Jurnal Pelita Informatika*, vol. 8, no. 2, pp. 288-291, 2019.
[7] H. R. Riswanto, K. Safinah, A. N. Muslikah, dan K. F. H. Holle, "Implementasi Teknik Kriptografi RSA untuk Pengamanan Data Pengiriman SMS," *Jurnal Ilmiah Informatika*, vol. 5, no. 1, pp. 61-66, 2020.
[8] B. Anwar, N. B. Nugroho, J. Prayudha, dan Azanuddin, "Implementasi Algoritma RSA Terhadap Keamanan Data Simpan Pinjam," *Sains dan Komputer (SAINTIKOM)*, vol. 18, no. 1, pp. 30-34, 2019.
[9] Sutejo, "Implementasi Algoritma Kriptografi RSA (Rivest Shamir Adleman) untuk Keamanan Data Rekam Medis Pasien,"*Journal of Information Technology and Computer Science (INTECOMS)*, vol. 4, no. 1, pp. 104-114, 2021.
[10] (2019) Algoritma RSA (Contoh Perhitungan Lengkap). [Online]. Available: https://komputerkata.com/algoritma-rsa/