p-ISSN: 2301-5373
e-ISSN: 2654-5101

Aplikasi Website Pengamanan File Dokumen Menggunakan Kriptografi RSA

I Made Ari Widiarsana[a1], I Gusti Ngurah Anom Cahyadi Putra[a2]
I Ketut Gede Suhartana[a3], Luh Gede Astuti[a4], I Putu Gede Hendra Suputra[a5] I Wayan Supriana[a6]

[a]Program Studi Informatika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Udayana
Badung, Bali, Indonesia
[1]awidiarsana.aw@gmail.com
[2]anom.cp@unud.ac.id
[3]ikg.suhartana@unud.ac.id
[4]lg.astuti@unud.ac.id
[5]hendra.suputra@unud.ac.id
[6]wayan.supriana@unud.ac.id

Abstract

Data security is something that needs to be considered in maintaining the confidentiality of information, especially those that only contain information that can be known by the authorized party. There are still many cases of data leakage that occur in Indonesia, especially in documents. Documents can be secured using cryptographic techniques. One of the well-known cryptography is RSA Cryptography. The security of RSA cryptography lies in the difficulty of factoring large numbers into prime factors. In previous research conducted by (Azhar & Yuliany, 2019)[1] *they could only encrypt files with the .pdf extension and also the images contained in the document were not successfully encrypted. The implementation of RSA cryptography will be made using the python programming language based on the website. The system created has a success rate of 100% in encrypting documents for each document, and in decrypting it has a success rate of 85% to 96%.*

Keywords: *Data Security, Cryptography, RSA, Website, Django, Python*

1. Pendahuluan

Pada saat ini penggunaan internet sudah seperti kebutuhan primer di masyarakat dunia. Teknologi internet semakin hari semakin kian berkembang. Keamanan data merupakan hal yang perlu diperhatikan dalam menjaga kerahasiaan informasi, terutama yang hanya memuat informasi yang dapat diketahui oleh pihak yang berwenang. Ada risiko penyadapan saat mengirim data atau informasi tanpa keamanan, dan mudah bagi pihak yang tidak berwenang untuk menemukan informasi yang terkandung di dalamnya. Keamanan dilakukan dengan melindungi keamanan file untuk menghindari masalah dengan pihak jahat yang dapat membuka kerahasiaan informasi. Pencarian celah keamanan adalah proses menyembunyikan atau menyamarkan informasi dengan cara yang tidak jatuh ke pihak lain yang tidak berkepentingan. Pada saat ini di Indonesia masih banyak kasus kebocoran data. Pada awal Mei 2020, sebanyak 91 data pengguna dari Tokopedia bocor dan dijual di situs gelap (dark web). Pada 21 Mei 2020 Daftar Pemilih Tetap (DPT) Pemilu 2014 yang dalam bentuk file pdf bocor di situs dan forum komunitas hacker. Data yang dihimpun mencakup sejumlah informasi sensitif, seperti nama lengkap, nomor kartu keluarga, Nomor Induk Kependudukan (NIK), tempat dan tanggal lahir, alamat rumah, serta beberapa data pribadi lainnya [2]. Pada tahun 2021 kembali lagi terjadi kasus kebocoran data pribadi yang kemungkinan adalah data dari BPJS Kesehatan yang telah di *upload* di internet. Data yang bocor tersebut ditemukan dalam bentuk *file excel* yang sudah di *upload* ke dalam komunitas *hacker*. Diperkirakan data yang bocor sebanyak 20 juta data yang meliputi NIK, nomor HP, alamat, alamat email, Nomor Pokok Wajib Pajak (NPWP), tempat tanggal lahir, jenis kelamin, jumlah tanggungan, serta foto [3]

Ilmu yang mempelajari penyandian atau pengkodean terhadap suatu berkas disebut dengan Kriptografi [4]. Sehingga data yang bersifat pribadi seperti NIK, NPWP, Nomor HP yang tercatat dalam dokumen harus di lindungi agar tidak disalahgunakan oleh pihak-pihak yang tidak berkepentingan [5]. Algortima kriptografi yang paling terkenal yaitu algoritma RSA (Riverst Shamir Adleman). RSA ditemukan pada tahun 1976 oleh tiga peneliti MIT (Massachusetts Institute of Technology), Ron Rivest, Adi Shamir, dan Len Adleman. Pada penelitian yang dilakukan oleh Rakhmat Kurniawan pada tahun 2017 juga menggunakan algortima RSA dalam pengamanan *file* dokumen, akan tetapi pada penelitan tersebut hanya terbatas pada enkripsi *text* dan juga kunci yang digunakan untuk enkripsi dan dekripsi sudah ditetapkan diawal (tidak ada pembangkitan kunci RSA)[6]. Keamanan algoritma RSA adalah sulitnya memfaktorkan bilangan besar menjadi faktor prima. Proses dari pembangkitan kunci RSA adalah :

- Tentukan dua bilangan prima sembarang p dan q
- Hitung nilai n dengan p x q. Disarankan bahwa nilai p dan q tidak sama, karena jika p = q maka nilai n = p2, maka nanti nilai p dapat diperoleh dengan akar kuadrat dari n
- Hitung nilai dari f(n) = (p – 1)(q – 1)
- Pilih kunci publik e yang relatif prima dengan nilai f(n)
- Pembangkitan kunci privat menggunakan persamaan e × d = 1 (mod f(n)).

Yang akan mendapatkan nilai kunci publik adalah (e, n) dan kunci privat adalah (d,n). Proses dari enkripsi RSA adalah $c = m^e \, mod \, n.$ Dan untuk proses dekripsi dari algoritma RSA adalah $m = c^d \, mod \, n$ [7]. Tingkat keamanan dari sebuah pembangkitan kunci RSA adalah semakin panjang kunci yang dibangkitkan (nilai p dan q besar) maka semakin kuat pula kunci yang dibangkitkan.

2. Metode Penelitian

Penelitian ini menggunakan metode kuantitatif, dan menggunakan metode *waterfall* untuk metode pengembangannya. Menggunakan metode ini karena pengembangan dilakukan secara berurutan. Sehingga hasil yang didapatkan akan lebih optimal

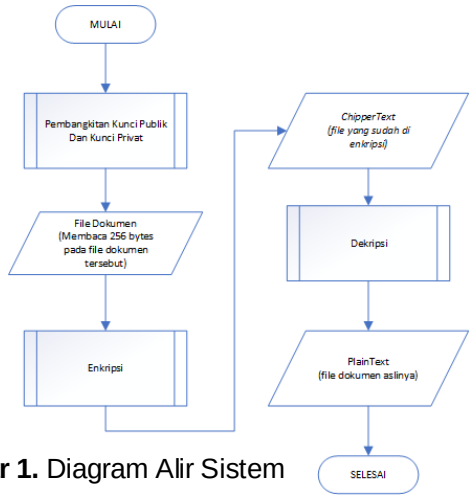
2.1. Gambaran Umum Aplikasi

Implementasi Kriptografi RSA dalam Pengamanan *File* Dokumen dalam paper ini memiliki tiga kategori utama yaitu enkripsi dokumen, dekripsi dokumen dan juga pembangkitan kunci RSA.

a. Pada proses pembangkitan kunci *user* diminta untuk memilih dua buah bilangan prima yang nilainya tidak boleh sama dan juga minimal 3 digit angka.
b. Pada enkripsi dokumen, pengguna akan *mengupload* dokumen berupa *.docx, .pptx, xlsx,* atau *.pdf* dan juga diminta untuk memasukkan kunci publik dari proses pembangkitan kunci RSA. Sistem akan melakukan enkripsi terhadap nilai dari *hexabytes* pada dokumen tersebut.
c. Pada dekripsi dokumen, pengguna akan *mengupload* dokumen berupa *.docx, .pptx, xlsx,* atau *.pdf* dan juga diminta untuk memasukkan kunci privat dari proses pembangkitan kunci RSA. Sistem akan melakukan dekripsi terhadap nilai dari *hexabytes* pada dokumen tersebut menggunakan algoritma RSA.

2.2. Desain Aplikasi

Aplikasi yang dibuat menggunakan bahasa pemrograman python, dengan framework django dan tampilan antarmuka menggunakan html, css, jquery, dan juga bootstrap. Berikut adalah *flowchart* yang digunakan dalam Implementasi Kriptografi RSA dalam Pengamanan File Dokumen di paper ini.



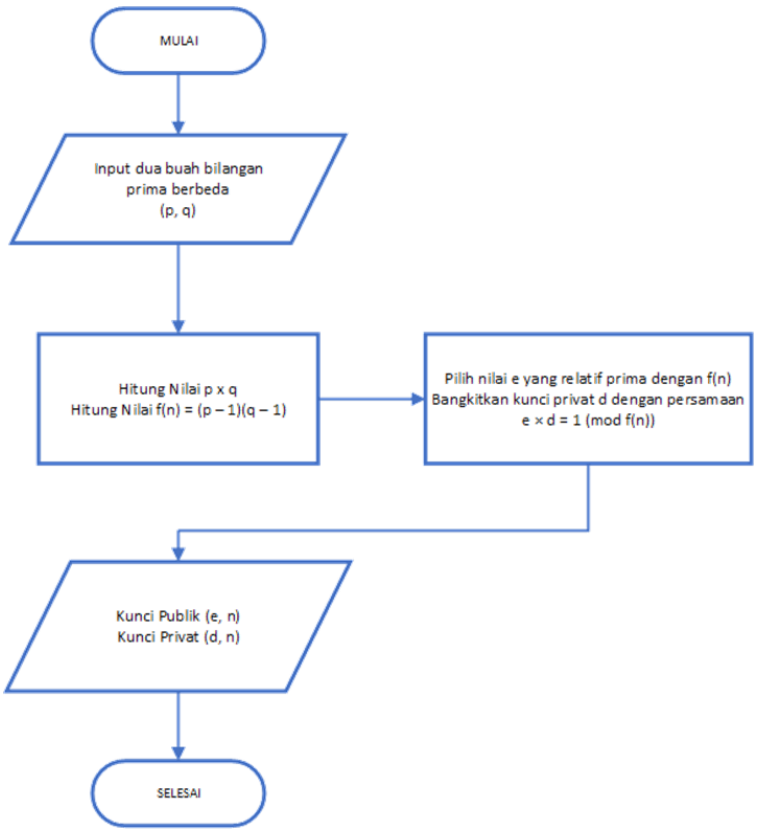
Gambar 1. Diagram Alir Sistem

p-ISSN: 2301-5373
e-ISSN: 2654-5101

Penjelasan diagram alir sistem:

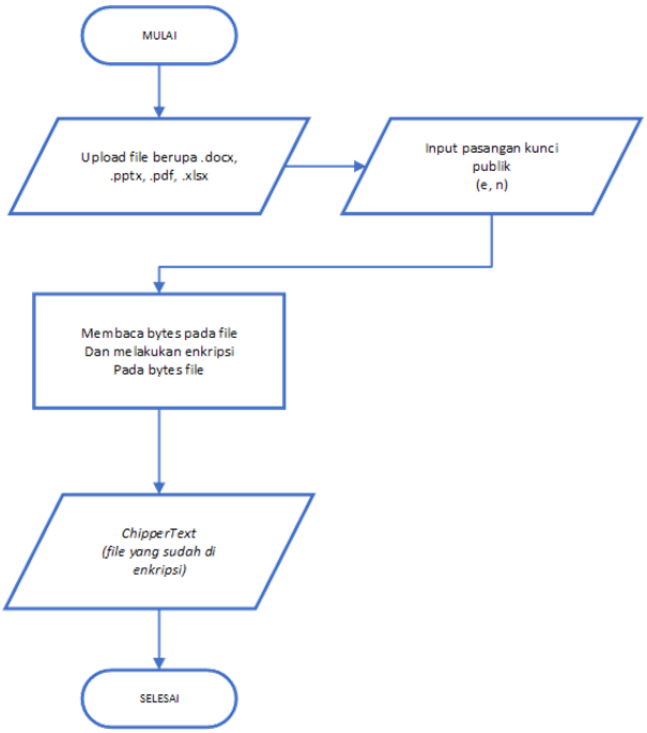
- Mulai
- Membaca file dokumen yang di upload ke sistem, dan membaca 256 bytes di dalam file dokumen tersebut.
- Melakukan proses pembangkitan kunci RSA, disini user diminta untuk memilih dua buah bilangan prima untuk proses pembangkitan kunci RSA
- Melakukan enkripsi 256 bytes dokumen yang sudah di upload menggunakan kunci publik yang sudah di berikan
- Menghasilkan output file chipertext yang dapat di unduh namun tidak dapat dibuka jika belum dilakukan dekripsi terhadap file tersebut
- Proses dekripsi yang dimana nantinya user mengupload file chipertext dan menggunakan kunci privat untuk dekripsinya.
- Akan menghasilkan output berupa plaintext atau file dokumen sama seperti aslinya (sebelum di enkripsi)
- Selesai.

Dalam pembangkitan kunci RSA diperlukan dua buah bilangan prima berbeda p dan q



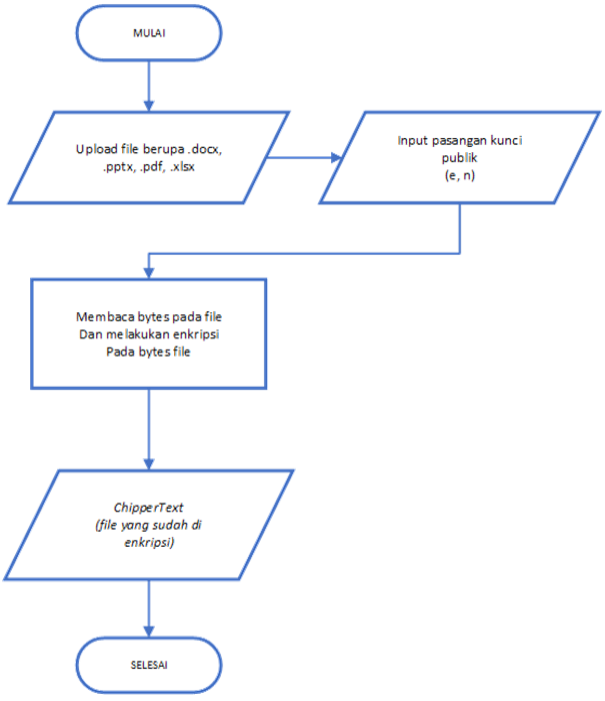
Gambar 2. Diagram Alir Pembangkitan Kunci RSA

Dalam proses enkripsi data plaintext (m) akan di enkripsi menggunakan pasangan kunci publik (e, n).



Gambar 3. Diagram Alir Enkripsi RSA

Dalam proses dekripsi data chippertext akan di dekripsi menggunakan pasangan kunci privat (d, n)



Gambar 3. Diagram Alir Dekripsi RSA

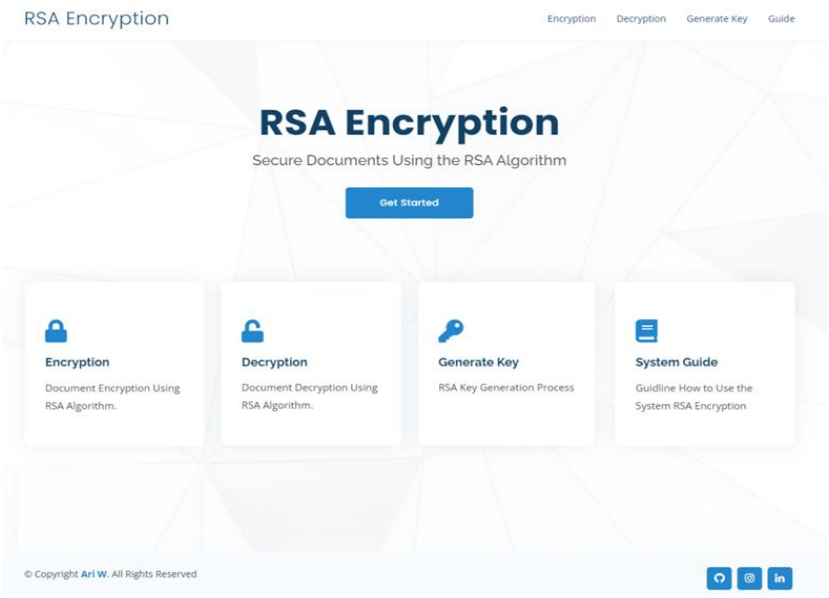
p-ISSN: 2301-5373
e-ISSN: 2654-5101

3. Hasil dan Pembahasan

3.1. Tampilan Awal Aplikasi

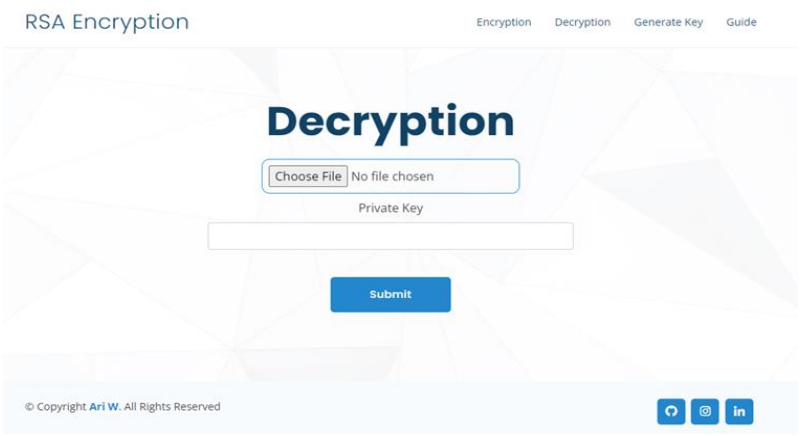
Pada halaman awal menampilkan 4 buah fitur utama yaitu untuk enkripsi, dekripsi, pembangkitan kunci dan juga panduan penggunaan sistem seperti pada gambar 4.

a. Pada halaman enkripsi, di halaman ini pengguna mengunggah sebuah file dokumen berupa .docx, .xlsx, .pptx, atau .pdf. Serta pada halaman ini pengguna memasukkan kunci publik, seperti pada gambar 5.
b. Pada halamn dekripsi pengguna akan mengunggah file dokumen yang telah di enkripsi dan memasukkan nilai dari kunci privat nya, seperti pada gambar 6.
c. Pada halaman pembangkitan kunci RSA, pengguna akan diminta untuk memilih dua buah bilangan prima yang nilainya lebih besar dari 100 (minimal 3 digit angka), dan juga mengisikan alamat email, dikarenakan nantinya hasil dari pembangkitan kunci akan dikirimkan melalui email seperti pada gambar 7.
d. Pada halaman panduan penggunaan, pengguna dapat melihat panduan dari penggunaan sistem. Penjelasan bagaimana cara melakukan enkripsi seperti gambar 8

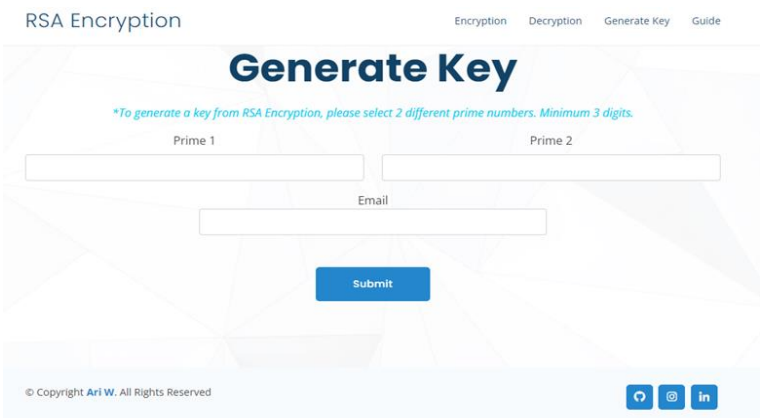


Gambar 4. Tampilan *Home*

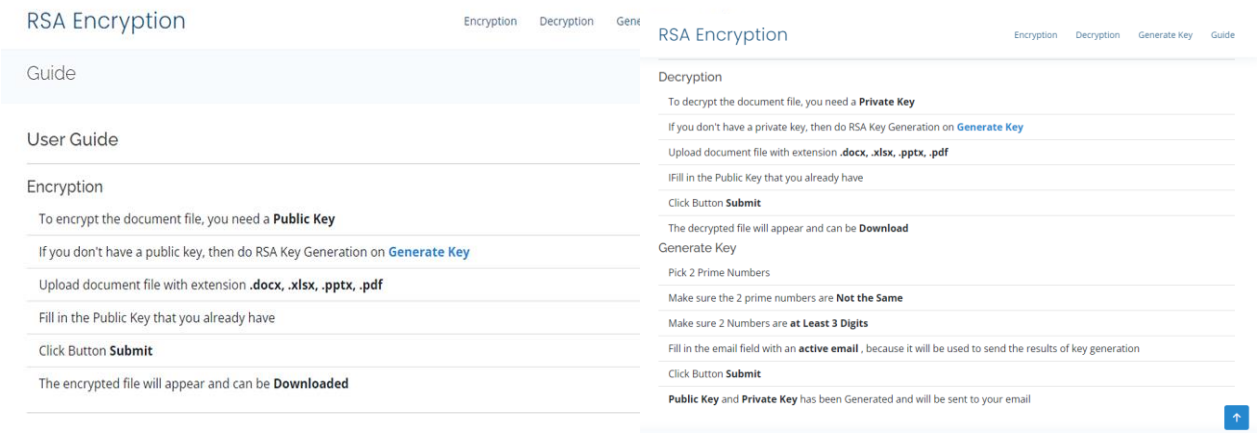
Gambar 5. Tampilan Halaman Enkripsi



Gambar 6. Tampilan Halaman Dekripsi



Gambar 7. Tampilan Halaman Pembangkitan Kunci



Gambar 8. Tampilan Halaman Panduan Penggunaan Sistem

p-ISSN: 2301-5373
e-ISSN: 2654-5101

3.2. Pengujian Sistem

Metode pengujian sistem menggunakan metode blackbox dan juga metode *brute-force* untuk mengetahui tingkat keamanan sistem. Hasil dari pengujian black box:

Tabel 1. Pengujian Enkripsi RSA

No.	Sekenario Pengujian	Uji Kasus	Hasil yang Diharapkan	Hasil Pengujian	Kesimpulan
1	File dokumen dan kunci publik tidak di isikan , klik tombol Save	*File* dokumen: (tidak ada) Kunci publik : (tidak ada)	Sistem akan menampilkan pesan error, bahwa *field* tersebut harus di isi	Sesuai Harapan	Valid
2	File dokumen yang di *upload* valid dan kunci publik tidak di isi kemudian klik tombol Save	*File* dokumen: (valid) Kunci publik : (kosong)	Sistem akan menunjukkan pesan error "*This field is required."*	Sesuai Harapan	Valid
3	File dokumen yang di *upload* kosong dan kunci publik valid kemudian klik tombol Save	*File* dokumen: (kosong) Kunci publik : (valid)	Sistem akan menunjukkan pesan error "*This field is required."*	Sesuai Harapan	Valid
4	Fil dokumen yang di *upload* tidak valid (*upload file* bukan berekstensi *.docx, .pdf, .xlsx, atau .pptx*)	*File* dokumen : (*file* berekstensi .txt) Kunci publik : (valid)	Sistem akan menampilkan pesan error "*Please enter a value with a valid extension."*	Sesuai Harapan	Valid
5	*File* dokumen valid dan kunci publik tidak sesuai format	*File* dokumen ; (valid) Kunci publik : (abcd)	Sistem akan menampilkan pesan error " Kunci Publik tidak sesuai format."	Sesuai Harapan	Valid
6	*File* dokumen valid dan kunci publik valid	*File* dokumen : (valid) Kunci publik : (valid)	Sistem akan menampilkan *pop up* dari *file* yang telah di enkripsi dengan nama acak	Sesuai Harapan	Valid

7	Ketika *pop up* hasil enkripsi tampil dan tombol *download* di klik	Tombol *download* di tekan	*File* hasil yang telah di enkripsi akan langsung terunduh di *device user*	Sesuai Harapan	Valid
8	Ketika *pop up* hasil enkripsi tampil dan tombol *upload drive* di tekan	Tombol *upload drive* di tekan	Sistem akan menampilkan *pop up* untuk meminta pengguna *login* ke akun *google.* Ketika berhaisl *login, file* akan ter-*upload* ke *drive* pengguna	Sesuai Harapan	Valid

Tabel 2. Pengujian *Dekripsi* RSA

No.	Sekenario Pengujian	Test Case	Hasil yang Diharapkan	Hasil Pengujian	Kesimpulan
1	File dokumen dan kunci privat tidak di berikan kemudian klik tombol Save	*File* dokumen: (tidak ada) Kunci privat : (tidak ada)	Sistem akan menampilkan pesan error, bahwa *field* tersebut harus di isi	Sesuai Harapan	Valid
2	File dokumen yang di *upload* valid dan kunci privat tidak di isi kemudian klik tombol Save	*File* dokumen: (valid) Kunci privat : (kosong)	Sistem akan menunjukkan pesan error "*This field is required."*	Sesuai Harapan	Valid
3	File dokumen yang di *upload* kosong dan kunci privat valid kemudian klik tombol Save	*File* dokumen: (kosong) Kunci privat : (valid)	Sistem akan menunjukkan pesan error "*This field is required."*	Sesuai Harapan	Valid
4	Fil dokumen yang di *upload* tidak valid (*upload file* bukan berekstensi *.docx, .pdf, .xlsx, atau .pptx*)	*File* dokumen : (*file* berekstensi .txt) Kunci privat : (valid)	Sistem akan menampilkan pesan error "*Please enter a value with a valid extension."*	Sesuai Harapan	Valid
5	*File* dokumen valid dan kunci privat tidak sesuai format	*File* dokumen ; (valid) Kunci privat : (abcd)	Sistem akan menampilkan pesan error " Kunci Privat tidak sesuai format."	Sesuai Harapan	Valid
6	*File* dokumen valid dan kunci privat valid	*File* dokumen : (valid) Kunci privat : (valid)	Sistem akan menampilkan *pop up* dari *file* yang telah di dekripsi dengan nama dekripsiFile	Sesuai Harapan	Valid

p-ISSN: 2301-5373
e-ISSN: 2654-5101

7	Ketika *pop up* hasil dekripsi tampil dan tombol *download* di klik	Tombol *download* di tekan	*File* hasil yang telah di dekripsi akan langsung terunduh di *device user*	Sesuai Harapan	Valid
8	Ketika *pop up* hasil dekripsi tampil dan tombol *upload drive* di tekan	Tombol *upload drive* di tekan	Sistem akan menampilkan *pop up* untuk meminta pengguna *login* ke akun *google.* Ketika berhaisl *login, file* akan ter-*upload* ke *drive* pengguna	Sesuai Harapan	Valid

Tabel 3. Pengujian Pembangkitan Kunci

No.	Sekenario Pengujian	Test Case	Hasil yang Diharapkan	Hasil Pengujian	Kesimpulan
1	Bilangan 1 dan Bilangan 2 dikosingkan kemudian tombol Save di tekan	Bilangan 1: (tidak ada) Bilangan 2 : (tidak ada)	Sistem akan menampilkan pesan error, bahwa *field* tersebut harus di isi	Sesuai Harapan	Valid
2	Bilangan 1 valid , dan Bilangan 2 tidak di isi	Bilangan 1: (103) Bilangan 2 : (kosong)	Sistem akan menunjukkan pesan error "*This field is required."*	Sesuai Harapan	Valid
3	Bilangan 1 tidak di isi dan Bilangan 2 valid	Bilangan 1 : (kosong) Bilangan 2 : (743)	Sistem akan menunjukkan pesan error "*This field is required."*	Sesuai Harapan	Valid
4	Bilangan 1 atau Bilangan 2 di isi kurang dari 3 digit	Bilangan 1 : (11) Bilangan 2 : (13)	Sistem akan menampilkan pesan error "*Please enter at least 3 characters."*	Sesuai Harapan	Valid
5	Bilangan 1 atau Bilangan 2 di isi dengan huruf	Bilangan 1 : (abc) Bilangan 2 : (xyz)	Sistem akan menampilkan pesan error "*Please enter a valid number."*	Sesuai Harapan	Valid
6	Bilangan 1 atau Bilangan 2 di isi dengan bilangan bukan prima	Bilangan 1: (120) Bilangan 2 : (200)	Sistem akan menampilkan pesan error "Bilangan harus prima."	Sesuai Harapan	Valid
7	Bilangan 1 dan Bilangan 2 di isi dengan bilangan prima (valid)	Bilangan 1 : (139) Bilangan 2 : (743)	Sistem akan memunculkan *field* Kunci publik dan kunci privat yang bisa di *copy paste*.	Sesuai Harapan	Valid

			Sistem juga akan menampilkan *file .txt* yang bisa di *download* oleh pengguna		
8	Sistem menampilkan hasil *generate key RSA*	*File .txt* di tekan	*File .txt* yang di dalamnya berisi nilai kunci publik dan kunci privat akan ter-*download* ke *device user*	Sesuai Harapan	Valid

Pengujian BruteForce

Untuk menguji tingkat keamanan RSA metode yang umum digunakan adalah metode brute-force. Brute-force merupakan melakukan pengecekan list kunci yang ada pada kunci sebenarnya hingga menemukan kunci yang sama [8]. Dari uji coba brute-force didapatkan:

Tabel 4. Brute-force RSA

Prima 1	Prima 2	Kunci Privat	Waktu
101	431	36997|43531	1120.90 detik
101	881	1741|88981	9400.82 detik
431	773	273197|333163	183600.87 detik

Semakin besar nilai dari hasil pembangkitan kunci RSA maka semakin kuat juga nilai dari kunci tersebut, dikarenakan waktu yang dibutuhkan untuk menemukan kunci tersebut juga sangat lama

4. Kesimpulan

Sistem pengamanan dokumen menggunakan algoritma RSA sudah berjalan sesuai fungsionalitas. Dalam pengimplementasiannya sistem yang dibuat akan mengenkripsi nilai dari hexabytes dari file dokumen sehingga ketika dokumen tersebut sudah berhasil di enkripsi dokumen tersebut tidak dapat dibuka. Lama waktu yang dibutuhkan untuk proses enkripsi dan dekripsi ditentukan oleh besaran *file* dokumen dan juga besaran kunci publik dan kunci privatnya, semakin besar maka semakin lama juga waktu yang dibutuhkan. Hasil dari penelitian ini diharapkan mampu membantu orang-orang dalam melindungi *file* dokumen agar aman dipergunakan.

References

[1] J. K. Azhar and S. Yuliany, “Implementasi Algoritma RSA (Rivest, Shamir danAdleman) untuk Enkripsi dan Dekripsi File .pdf,” no. December, 2019.

[2] C. Stephanie, “7 Kasus Kebocoran Data yang Terjadi Sepanjang 2020,” *Kompas.com*, 2021. .

[3] A. Rizal, “Kebocoran Data BPJS Kesehatan Buktikan Lemahnya Perlindungan Data Pribadi,” *Infokomputer.grid.id*, 2021. .

[4] R. Firmansyah, “Implementasi Keamanan Pesan Teks Menggunakan Kriptografi Algoritma Rsa Dengan Metode Waterfall Berbasis Java,” *Joutica*, vol. 4, no. 1, p. 174, 2019, doi: 10.30736/jti.v4i1.265.

p-ISSN: 2301-5373
e-ISSN: 2654-5101

[5] W. Djafar, “Perlindungan Data Pribadi di Indonesia: Lanskap, Urgensi, dan Kebutuhan Pembaruan,” *J. Becoss*, vol. 1, no. 1, pp. 147–154, 2019.

[6] R. Kurniawan, “Rancang Bangun Aplikasi Pengaman Isi File Dokumen Dengan RSA,” *J. Ilmu Komput. dan Inform.*, vol. 01, no. November, pp. 46–52, 2017.

[7] I. R. Munir, “Algoritma RSA dan ElGamal,” *Kriptografi*, p. 13, 2010.

[8] K. Berlin and S. . Henakaran, “An Overview of Cryptanalysis of RSA Public key System,” *Int. J. Eng. Technol.*, vol. 9, no. 5, pp. 3575–3579, 2017, doi: 10.21817/ijet/2017/v9i5/170905312.

This page is intentionally left blank