Jurnal Elektronik Ilmu Komputer Udayana
Volume 13, No 2. November 2024

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

Aplikasi Enkripsi Pesan E-Mail Menggunakan Hybrid Cryptosystem AES dan RSA

Gede Krisna Surya Artajaya[a1], Agus Muliantara[a2], I Gusti Ngurah Anom Cahyadi Putra[a3], I Ketut Gede Suhartana[a4]

[a]Program Studi Informatika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Udayana
Jalan Raya Kampus Unud, Jimbaran, Bali, 80361, Indonesia
[1] krisnasurya09@gmail.com
[2] muliantara@unud.ac.id
[3]anom.cp@unud.ac.id
[4] ikg.suhartana@unud.ac.id

Abstract

The security of communication via e-mail is increasingly important in today's digital era, where cyber attacks are increasing. These threats require the development of effective solutions to protect the confidentiality and integrity of information in e-mail. This study presents the development of an e-mail messaging application that employs a hybrid cryptosystem combining Advanced Encryption Standard (AES) and Rivest-Shamir-Adleman (RSA) algorithms. The aim is to improve the security of e-mail communications by utilizing the strengths of both symmetric and asymmetric encryption methods. AES is used for its efficiency in encrypting large amounts of data, while RSA ensures secure key exchange. The application tested with blackbox testing to verify its functionality and brute force testing to assess the encryption's robustness. The results show that the application successfully passed the blackbox testing, demonstrating that it functions as intended. The encryption results of the application are also excellent, as evidenced by brute force testing which shows that the encryption cannot be broken. This hybrid approach, by integrating AES and RSA, provides a practical solution for enhancing e-mail security.

Keywords: *E-mail Security, Hybrid Cryptosystem, Advanced Encryption Standard (AES), Rivest-Shamir-Adleman (RSA), Cyber Attacks*

1. Pendahuluan

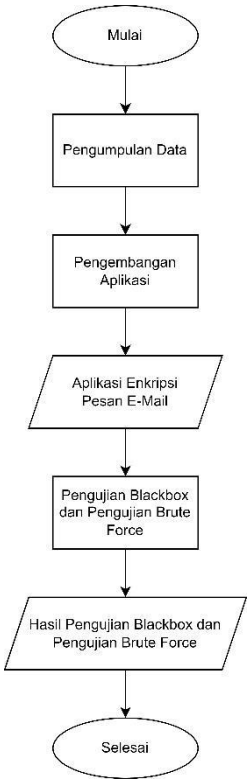
Perkembangan teknologi telah mengubah cara manusia berkomunikasi, dari komunikasi langsung hingga penggunaan media perantara seperti surat. Kemudian, layanan e-mail muncul sebagai solusi modern untuk mengirim pesan dengan cepat dan mudah melalui internet, memungkinkan pertukaran pesan lintas dunia. Namun, e-mail rentan terhadap serangan siber seperti peretasan. Peretasan sendiri merupakan usaha untuk membobol program komputer pihak lain [1] yang memungkinkan pihak tidak sah memperoleh data sensitif atau informasi penting dari pesan e-mail yang dipertukarkan. Oleh karena itu, diperlukan pendekatan atau teknik tertentu untuk memastikan kerahasiaan pesan tersebut, salah satu caranya adalah dengan menerapkan metode kriptografi.

Kriptografi merupakan seni dan ilmu melindungi pesan sehingga pesan tetap aman [2]. Dalam kriptografi suatu pesan akan dienkripsi sehingga tidak memiliki arti lagi, sehingga pesan tidak dapat dibaca oleh pihak luar yang tidak memiliki hak [3]. Terdapat berbagai algoritma dalam kriptografi, di mana algoritma tersebut dapat digabungkan untuk memanfaatkan kelebihan dari masing-masing yang disebut dengan hybrid cryptosystem. Secara umum hybrid cryptography diklasifikasikan menjadi dua model; pertama algoritma simetris digunakan untuk mengenkripsi data dan algoritma asimetris digunakan untuk mengenkripsi kunci rahasia; kedua mengenkripsi data dua kali menggunakan enkripsi simetris atau asimetris secara berurutan [4].

Dengan mengaplikasikan hybrid cryptosystem pada aplikasi, data yang dienkripsi akan lebih aman dari serangan siber. Penelitian sebelumnya [5] telah mengembangkan aplikasi yang dapat mengirim pesan e-mail terenkripsi menggunakan algoritma Blowfish dan algoritma RSA. Dalam penelitian tersebut dilakukan simulasi serangan brute force untuk mendapatkan kunci privat untuk mendekripsi pesan. Hasil dari simulasi tersebut menunjukkan bahwa algoritma yang digunakan aman dan tingkat keamanan terhadap serangan brute force dipengaruhi oleh panjang karakter kunci. Penelitian lain mengenai hybrid cryptosystem [6] membandingkan performa dan efisiensi kombinasi hybrid

cryptosystem dalam hal waktu enkripsi dan enkripsi, rata-rata throughput, serta efisiensi. Hasil perbandingan menunjukkan bahwa kombinasi algoritma AES dan RSA merupakan model yang paling cepat dan efisien dibandingkan dengan kombinasi algoritma Blowfish-RSA, algoritma 3DES-RSA, dan algoritma DES-RSA. Dalam penelitian ini, penulis menggabungkan algoritma AES dan algoritma RSA, di mana algoritma AES digunakan untuk mengenkripsi pesan e-mail dan algoritma RSA digunakan untuk mengenkripsi kunci rahasia AES sehingga pesan dapat terenkripsi saat pengiriman dari pengirim ke penerima.

2. Metode Penelitian



Gambar 1. *Flowchart* Metodologi Penelitian

Dalam penelitian ini penulis mengembangkan program kriptografi untuk mengenkripsi pesan *e-mail* menggunakan algoritma AES dan algoritma RSA. Pada *flowchart* yang dapat dilihat pada Gambar 1, langkah pertama dari penelitian ini adalah mengumpulkan data-data yang berterkaitan dengan penelitian, kemudian melakukan pengembangan aplikasi, di mana setelah aplikasi dibuat maka dilakukan pengujian blackbox untuk menguji fungsionalitas dari aplikasi dan pengujian brute force untuk mengetahui ketahanan dari hasil enkripsi.

2.1 Pengumpulan Data

Data yang digunakan pada penelitian ini adalah data berupa teks pesan yang akan dikirim melalui e-mail. Kemudian untuk metode yang digunakan dalam penelitian ini adalah sebagai berikut:

a. Studi literatur dengan mempelajari textbook, jurnal yang berisikan penelitian terkait enkripsi dan dekripsi menggunakan hybrid cryptosystem, alogiritma AES, algoritma RSA, pembuatan aplikasi pengirim pesan, serta literatur lainnya yang mendukung penelitian ini.
b. Observasi, yakni dengan pengumpulan data dan informasi terkait dengan bagaimana data flow penggunaan *e-mail client*.

2.2 Pengembangan Aplikasi

Pendekatan yang penulis gunakan dalam mengembangkan aplikasi ini adalah pendekatan agile model Extreme Programming (XP), di mana pendekatan agile memungkinkan penulis untuk dengan cepat memperbarui aplikasi bila terjadi perubahan saat pengembangan aplikasi, sehingga aplikasi yang dibuat benar-benar memenuhi kebutuhan pengguna. Berikut tahapan dari Exteme Programming:

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

a. Planning

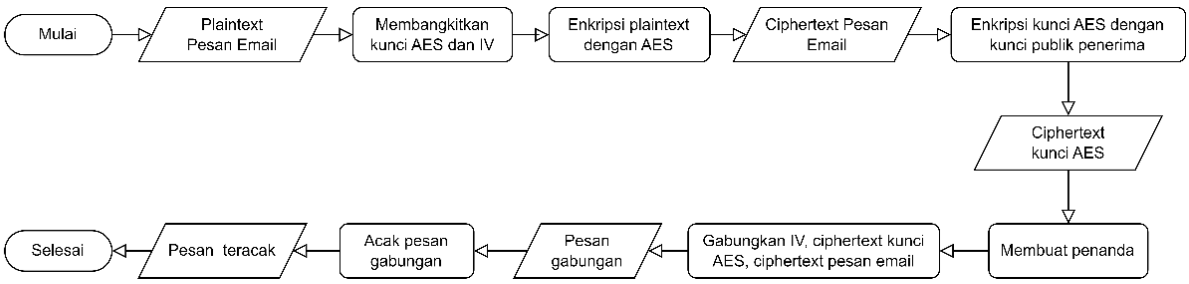
Tahap *planning* merupakan tahap mengidentifikasi masalah yang terjadi kemudian mengumpulkan kebutuhan pengguna dan mendapat gambaran mengenai fitur-fitur aplikasi, analisis kebutuhan sistem, dan keluaran yang diharapkan. Fitur yang akan diterapkan terdiri dari *Login*, Tulis Pesan, E-mail Terkirim, E-Mail Masuk, Draft, Konfigurasi, Bantuan, dan *Logout*. Kemudian analisis kebutuhan sistem terdapat dua kebutuhan yang akan dianalisa yakni kebutuhan fungsional dan kebutuhan non fungsional. Berikut merupakan kebutuhan fungsional dan non fungsional dari aplikasi:

1. Kebutuhan fungsional
- Aplikasi dapat membangkitkan kunci publik dan kunci privat untuk pengguna.
- Aplikasi dapat membangkitkan kunci rahasia untuk mengenkripsi pesan.
- Aplikasi dapat mengenkripsi pesan menjadi *ciphertext* menggunakan kunci rahasia dengan algoritma AES.
- Aplikasi dapat mengenkripsi kunci rahasia menggunakan kunci publik penerima dengan algoritma RSA.
- Aplikasi dapat mengirim pesan yang sudah terenkripsi dari pengirim ke penerima.
- Aplikasi dapat menerima dan mengunduh pesan yang sudah dikirim.
- Aplikasi dapat mendekripsi kunci rahasia menggunakan kunci privat penerima dengan algoritma RSA.
- Aplikmasi dapat mendekripsi *ciphertext* menggunakan kunci rahasia dengan algoritma AES.
- Aplikasi dapat menampilkan pesan yang telah didekripsi.

2. Kebutuhan non fungsional
- Aplikasi harus *user friendly.*
- Aplikasi harus dapat terhubung dengan layanan gmail.

b. Design

Tahap *design* merupakan tahap untuk perancangan model aplikasi berdasarkan kebutuhan yang telah didapat pada tahap planning. Perancangan model ini di antaranya pembuatan *flowchart* dan *mock up* dari desain antarmuka aplikasi. Berikut merupakan *flowchart* enkripsi dan dekripsi dari aplikasi:

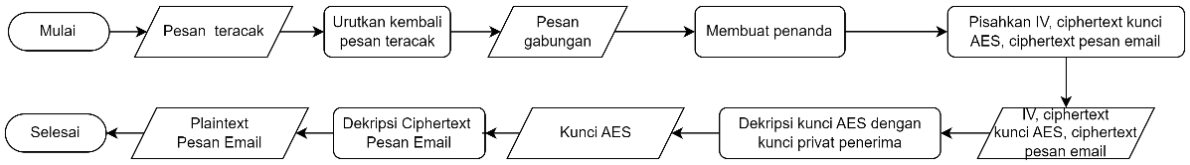


Gambar 2. *Flowchart* Enkripsi

Flowhchart pada Gambar 2 merupakan *flowchart* dari proses enkripsi yang akan diimplementasikan pada aplikasi. Penjelasan *flowchart* enkripsi adalah sebagai berikut:

1. Aplikasi akan membangkitkan kunci AES dan IV. Pembangkitan kunci AES dan IV dilakukan menggunakan modul Random dari library Crypto. IV sendiri merupakan data acak yang diperlukan untuk menginisialisasi proses enkripsi AES.
2. Aplikasi kemudian mengenkripsi *plaintext* menggunakan AES yang menghasilkan *ciphertext*.
3. Kunci AES yang digunakan untuk mengenkripsi pesan *e-mail* akan dienkripsi dengan RSA menggunakan kunci publik penerima dan akan menghasilkan kunci AES terenkripsi.
4. Membuat penanda untuk digabungkan bersama IV, kunci AES terenkripsi, dan *ciphertext*. Penanda dibuat dari operasi perkalian, penjumlahan, serta pengurangan huruf dan angka yang terdapat dalam kunci publik. Penanda dibuat agar aplikasi dapat memisahkan IV, kunci AES terenkripsi, dan *ciphertext* pada proses dekripsi
5. Kemudian IV, kunci AES terenkripsi, *ciphertext*, dan penanda akan digabungkan.
6. Setelah digabungkan, urutan pesan akan diacak. Untuk metode pengacakannya yakni dengan menghitung nilai ASCII tiap karakter dari hasil penggabungan pada langkah lima

untuk digunakan sebagai seed dalam modul Random agar urutan acak konsisten dan urutan dapat dikembalikan.



Gambar 3. *Flowchart* Dekripsi

Flowhchart pada Gambar 3 merupakan *flowchart* dari proses dekripsi yang akan diimplementasikan pada aplikasi. Penjelasan *flowchart* dekripsi adalah sebagai berikut:

1. Aplikasi akan mengurutkan kembali urutan pesan yang diacak, sehingga mendapatkan pesan gabungan yang uruta1nnya benar. Metode yang digunakan sama seperti mengacak urutan dari pesan pada saat enkripsi.
2. Membuat penanda untuk memisahkan IV, kunci AES terenkripsi, dan *ciphertext*.
3. Pisahkan IV, kunci AES terenkripsi, *ciphertext* untuk keperluan dekripsi.
4. Dekripsi kunci AES terenkripsi dengan kunci privat penerima.
5. Dekripsi pesan terenkripsi menggunakan AES.

c. Coding

Tahap *coding* merupakan tahap pembuatan kode program berdasarkan perancangan aplikasi yang telah dibuat. Hasil dari coding ini merupakan protoipe dari aplikasi. Pembuatan kode program aplikasi ini menggunakan bahasa pemrograman Python. Dalam pembuatan aplikasi digunakan protokol SMTP untuk mengirim e-mail, IMAP untuk mengunduh e-mail, SSL untuk mengamankan komunikasi antara klien dan server, dan Gmail API untuk kebutuhan autentikasi.

d. Testing

Tahap *testing* merupakan tahap untuk menguji aplikasi yang telah dibuat. Pada pengujian aplikasi ini dilakukan pengujian *blackbox* dan pengujian keamanan dengan melakukan penyerangan *brute force*.

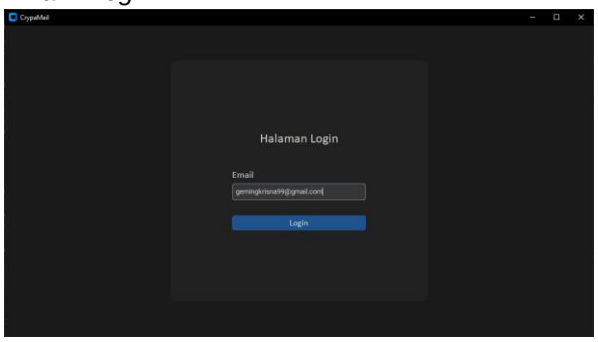
e. *Software Increment*

Pada tahap ini dilakukan pengembangan lebih lanjut dengan menambahkan fitur baru sehingga kemampuan fungsional aplikasi bertambah.

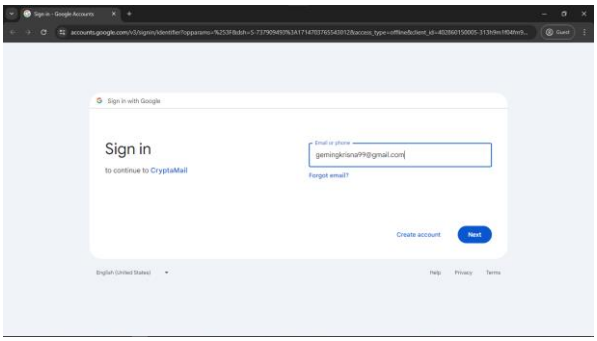
3. Hasil dan Diskusi

3.1. Penggunaan Aplikasi

a. Login

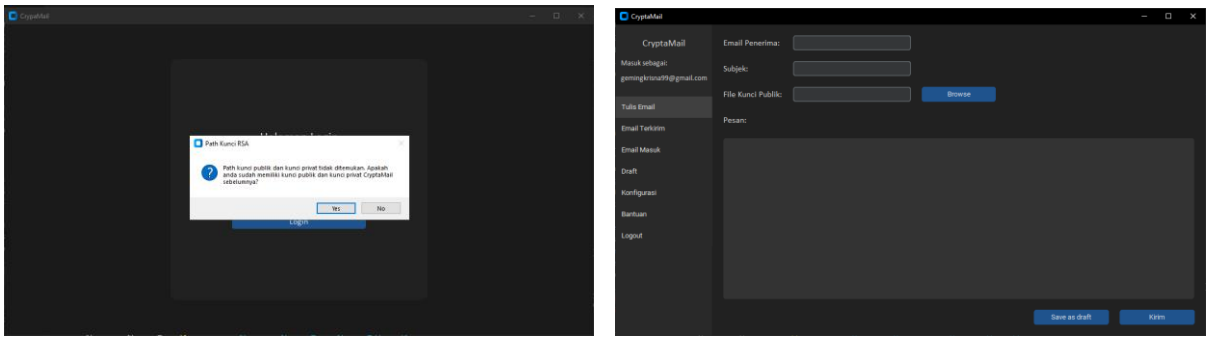


(a) halaman *login* aplikasi



(b) halaman *sign in* gmail

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



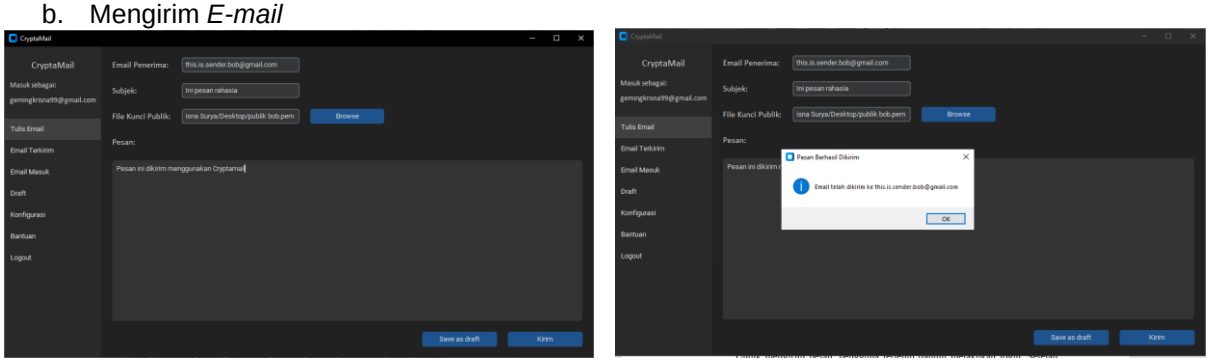
(c) *pop up path* Kunci RSA (d) halaman tulis *e-mail*

Gambar 4. *Login* Aplikasi

Login merupakan proses pertama untuk menggunakan aplikasi. Proses login dapat dilihat pada Gambar 4. Berikut merupakan langkah-langkah untuk melakukan login:

1. Mengisi alamat *e-mail* pada kolom "*E-mail*", kemudian tekan tombol "Login" seperti yang terlihat pada Gambar 4 (a)
2. *Browser* akan muncul dengan halaman *sign in* dari google untuk melakukan autentikasi seperti yang terlihat pada Gambar 4 (b). Ikuti langkah yang disediakan oleh google.
3. Setelah proses autentikasi selesai, apabila pengguna baru pertama kali menggunakan aplikasi atau *login* pada perangkat lain, akan muncul pop up pembuatan kunci RSA seperti yang terlihat pada Gambar 4 (c). Tekan "Yes" apabila pengguna sudah memiliki kunci RSA dan tekan "No" apabila pengguna belum memiliki kunci RSA.
4. Setelah proses *Login* selesai aplikasi akan menampilkan halaman Tulis *E-mail* seperti yang terlihat pada Gambar 4 (d).

b. Mengirim *E-mail*



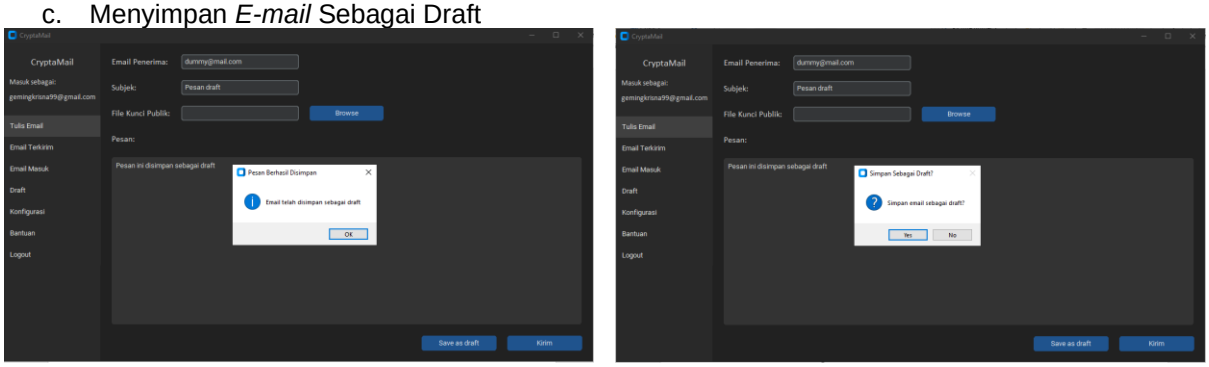
(a) halaman tulis *e-mail* yang terisi semua (b) *pop up e-mail* terkirim

Gambar 5. Mengirim *E-mail*

Proses mengirim e-mail dapat dilihat pada Gambar 5. Berikut merupakan langkah untuk mengirim *e-mail*:

1. Mengisi alamat *e-mail* penerima, subjek, dan pesan pada kolom yang disediakan serta memasukkan kunci publik penerima seperti pada Gambar 5 (a).
2. Setelah semua terisi, tekan Kirim dan tunggu hingga *pop up* muncul seperti pada Gambar 5 (b).

c. Menyimpan *E-mail* Sebagai Draft



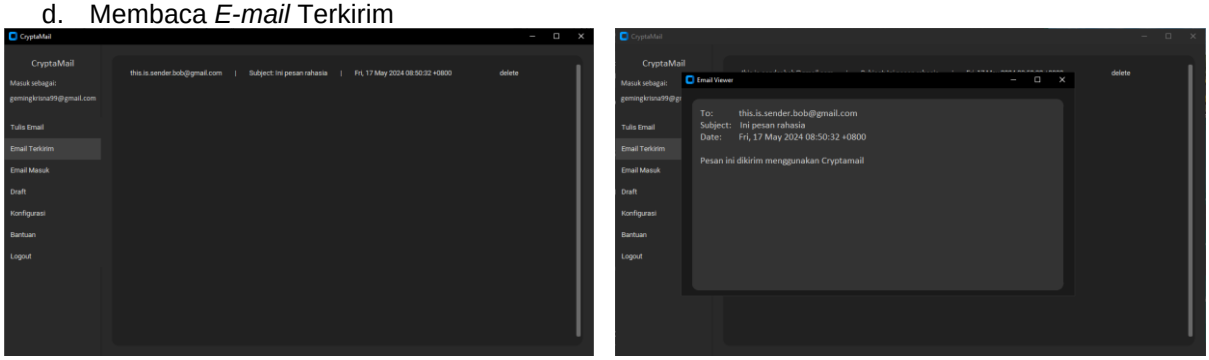
(a) *pop up e-mail* disimpan sebagai draft (b) *pop up* simapn *e-mail* sebagai draft

Gambar 6. Menyimpan *E-mail* Sebagai Draft

Untuk melakukan penyimpanan e-mail sebagai draft dapat dilihat pada Gambar 6. Langkah untuk menyimpan *e-mail* sebagai draft adalah sebagai berikut:

1. Mengisi data *e-mail* pada halaman Tulis *E-mail*.
2. Menekan tombol Save as draft dan tunggu sampai *pop up* muncul seperti pada Gambar 6 (a).
3. *E-mail* juga dapat disimpan ketika pengguna memilih halaman lain dan terdapat data *e-mail* pada halaman Tulis *E-mail* seperti pada Gambar 6 (b).

d. Membaca *E-mail* Terkirim



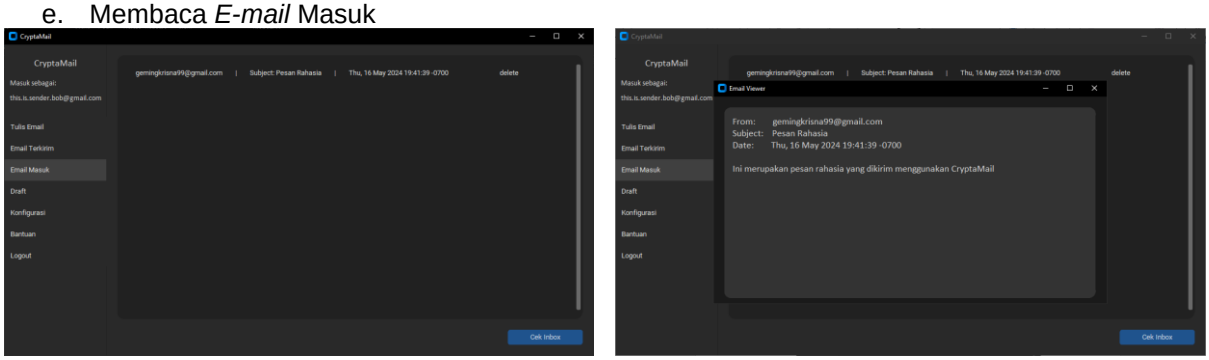
(a) halaman *E-mail* Terkirim (b) membaca *e-mail* terkirim

Gambar 7. Membaca *E-mail* Terkirim

Untuk membaca e-mail terkirim dapat dilihat pada Gambar 7. Berikut langkah untuk membaca *e-mail* terkirim:

1. Menekan *E-mail* Terkirim pada panel navigasi sehingga aplikasi menampilkan halaman e-mail terkirim seperti yang terlihat pada Gambar 7 (a).
2. Menekan *e-mail* yang ingin dibaca sehingga aplikasi menampilkan e-mail yang terkirim seperti yang terlihat pada Gambar 7 (b).

e. Membaca *E-mail* Masuk



(a) halaman *E-mail* Masuk (b) membaca *e-mail* masuk

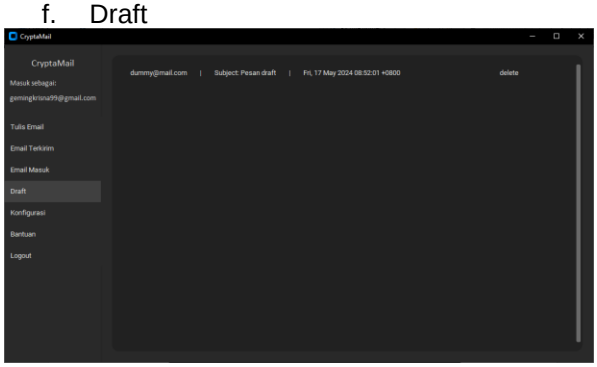
Gambar 8. Membaca *E-mail* Masuk

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

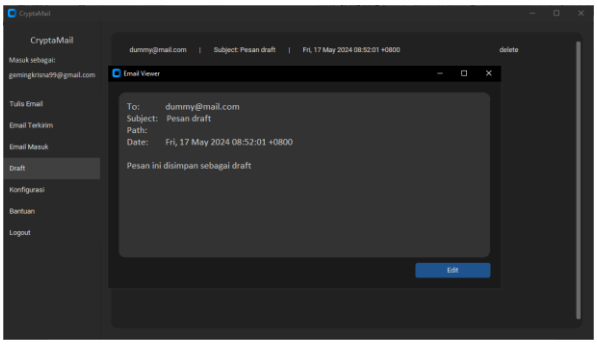
Untuk membaca e-mail masuk dapat dilihat pada Gambar 8. Berikut langkah untuk membaca *e-mail* masuk:

1. Menekan *E-mail* Masuk pada panel navigasi sehingga aplikasi menampilkan halaman e-mail masuk seperti yang terlihat pada Gambar 8 (a).
2. Menekan *e-mail* yang ingin dibaca sehingga aplikasi menampilkan e-mail yang masuk seperti yang terlihat pada Gambar 8 (b).

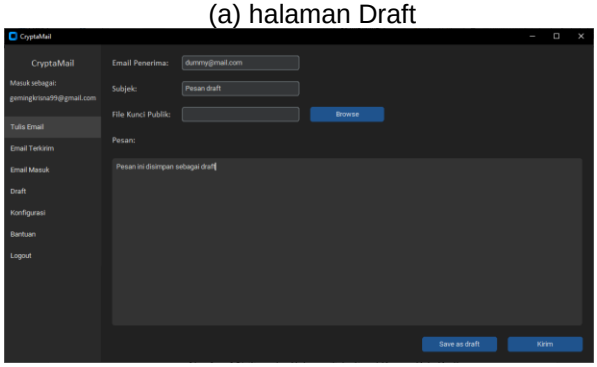
f. Draft



(a) halaman Draft



(b) membaca draft



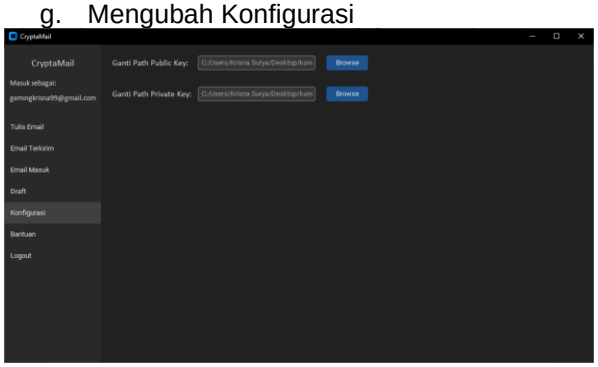
(c) halaman Tulis *E-mail* dengan data draft

Gambar 9. Membaca dan Mengedit Draft

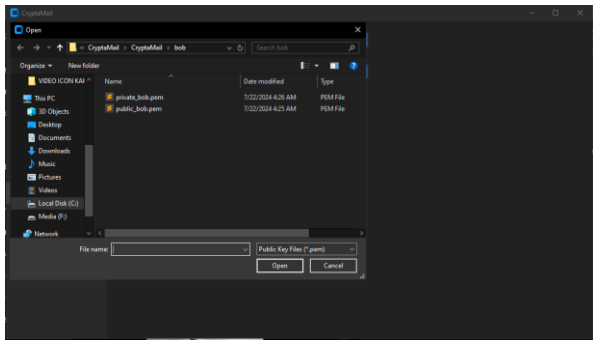
Untuk membaca dan mengedit draft dapat dilihat pada Gambar 9. Berikut langkah untuk membaca dan mengedit draft:

1. Menekan Draft pada panel navigasi sehingga halaman Draft muncul seperti pada Gambar 9 (a).
2. Menekan draft yang ingin dibaca sehingga aplikasi menampilkan draft yang dipilih seperti yang terlihat pada Gambar 9 (b).
3. Apabila ingin mengedit draft, tekan tombol Edit.
4. Aplikasi akan berpindah ke halaman Tulis *E-mail* untuk mengedit draft seperti yang terlihat pada Gambar 9 (c).

g. Mengubah Konfigurasi



(a) halaman Konfigurasi



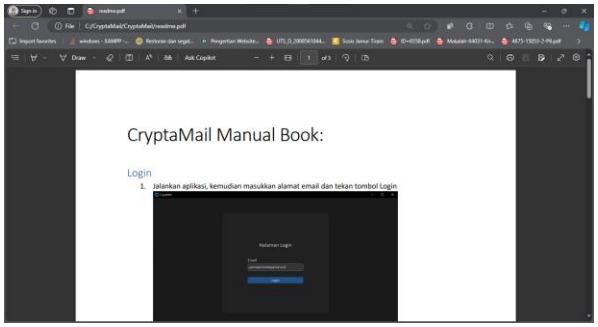
(b) jendela baru untuk memasukkan kunci publik atau kunci privat

Gambar 10. Konfigurasi Aplikasi

Untuk konfigurasi aplikasi dapat dilihat pada Gambar 9. Berikut langkah untuk mengubah konfigurasi:

1. Menekan Konfigurasi pada panel navigasi sehingga aplikasi menampilkan halaman konfigurasi seperti yang terlihat pada Gambar10 (a).
2. Menekan tombol Browse untuk memilih kunci publik atau kunci privat sehingga aplikasi menampilkan jendela baru untuk memilih kunci publik atau kunci privat seperti yang terlihat pada Gambar 10 (b).

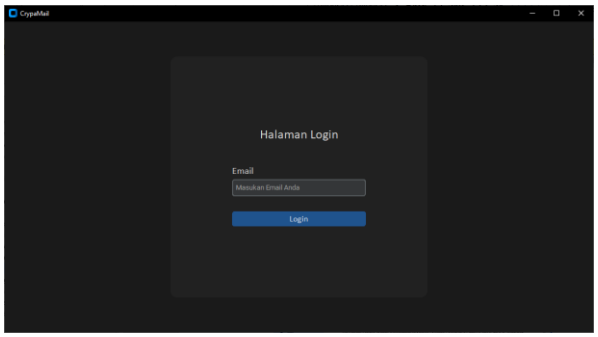
h. Menampilkan Bantuan



Gambar 11. File Bantuan Yang Dibuka Menggunakan Browser

Untuk menampilkan bantuan pengguna menekan Bantuan pada panel navigasi, file bantuan akan dibuka dengan *browser* atau aplikasi penampil pdf seperti yang terlihat pada Gambar 11.

i. *Logout*



Gambar 12. File Bantuan Yang Dibuka Menggunakan Browser

Logout dapat dilakukan dengan menekan Logout pada panel navigasi kemudian aplikasi akan kembali ke halaman Login seperti yang terlihat pada Gambar 12

3.2. Pengujian *Blackbox*

Pengujian blackbox pada aplikasi dilakukan dengan tujuan untuk memastikan aplikasi dapat berjalan sesuai dengan fungsionalitas yang telah ditetapkan sebelumnya. Hasil pengujian *blackbox* dapat dilihat pada tabel 1 sebagai berikut:

Tabel 1. Hasil Pengujian Blackbox

No	Rancangan Proses	Hasil yang Diharapkan	Hasil	Keterangan
1.	Mengisi setiap data yang diperlukan untuk melakukan login dengan benar.	Aplikasi melakukan proses autentikasi	Berhasil	Jika semua kolom pada halaman login diisi dengan benar.
2.	Mengisi setiap data yang diperlukan untuk mengirim pesan *e-mail*.	Aplikasi melakukan proses enkripsi subjek dan isi pesan yang dimasukkan, mengirimkan pesan *e-mail* dan membuat	Berhasil	Jika semua kolom pada halaman tulis *e-mail* diisi dengan benar.

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

		arsip dari *e-mail* yang dikirim.		
3.	Menekan tombol save as draft dengan minimal satu kolom terisi.	Aplikasi menyimpan data yang telah dimasukkan sebagai draft dan menampilkan pesan pop up pesan *e-mail* telah disimpan sebagai draft setelah data berhasil disimpan.	Berhasil	Minimal terdapat satu kolom yang terisi.
4.	Menampilkan daftar *e-mail* terkirim.	Aplikasi menampilkan dafttar *e-mail* terkirim	Berhasil	Jika terdapat minimal satu *e-mail* terkirim.
5.	Menampilkan daftar *e-mail* masuk.	Aplikasi menampilkan dafttar *e-mail* masuk	Berhasil	Jika terdapat minimal satu *e-mail* masuk.
6.	Menekan tombol cek inbox.	Aplikasi akan melakukan pengecekan apakah terdapat *e-mail* masuk atau tidak	Berhasil	Menekan tombol cek inbox.
7.	Memuat ulang daftar *e-mail* masuk.	Aplikasi memuat ulang daftar *e-mail* masuk sehingga *e-mail* masuk terbaru dapat dilihat oleh pengguna.	Berhasil	Minimal terdapat satu *e-mail* masuk ketika tombol cek inbox ditekan.
8.	Menampilkan daftar draft *e-mail*.	Aplikasi menampilkan dafttar draft *e-mail*	Berhasil	Jika terdapat minimal satu draft *e-mail*.
9.	Menekan tombol edit.	Aplikasi akan mengisi data dari draft *e-mail* terpilih ke kolom yang berada di halaman tulis *e-mail*. Kemudian aplikasi akan pindah halaman ke halaman tulis *e-mail*.	Berhasil	
10.	Menekan salah satu *e-mail*.	Aplikasi mendekripsi *e-mail* yang dipilih dan menampilkan window baru untuk melihat isi pesan.	Berhasil	
11.	Menekan tombol delete.	Aplikasi menampilkan pop up pilihan apakah pengguna yakin menghapus *e-mail* atau tidak	Berhasil	
12.	Menekan tombol yes pada pop up pilihan menghapus *e-mail*.	Aplikasi akan menghapus *e-mail* terkait.	Berhasil	
13.	Menekan tombol no pada pop up pilihan menghapus *e-mail*.	Aplikasi tidak akan menghapus *e-mail* terkait.	Berhasil	
14.	Mengubah path kunci publik	Aplikasi akan menggunakan path dari kunci publik yang dipilih untuk mengganti path kunci publik yang lama.	Berhasil	File yang dipilih harus file pem dari kunci publik.
15.	Mengubah path kunci privat	Aplikasi akan menggunakan path dari kunci privat yang dipilih	Berhasil	File yang dipilih harus file pem dari kunci privat.

		untuk mengganti path kunci privat yang lama.	
16.	Menekan tombol bantuan pada panel navigasi.	Menampilkan file bantuan pada browser atau aplikasi pembuka file pdf yang digunakan.	Berhasil
17.	Menekan tombol logout pada panel navigasi.	Menampilkan halaman login dan mengosongkan variabel yang berisikan data pengguna sebelumnya	Berhasil

Pada tabel 1 hasil pengujian *blackbox* dapat dilihat bahwa aplikasi telah berhasil melewati pengujian Blackbox dengan sangat baik, di mana dari semua rancangan proses yang diujikan aplikasi berhasil memberikan hasil yang diharapkan.

3.3. Pengujian *Brute Force*

Pengujian ini bertujuan untuk mengetahui seberapa tahan hasil enkripsi menggunakan aplikasi terhadap serangan *Brute Force*. Dalam pengujian ini diasumsikan penyerang memiliki *ciphertext* dan kunci publik, di mana kunci publik digunakan untuk membuat kunci privat secara *brute force*. Berikut langkah pembuatan kunci privat yang dilakukan pada penelitian ini:

a. Mengambil nilai eksponen publik dari kunci publik target.
b. Membuat bilangan prima p dan q.
c. Membuat modulus n dengan mengalikan p dan q.
d. Menghitung fungsi totien Euler dengan (p - 1) * (q - 1)
e. Memeriksa apakah nilai eksponen publik dan fungsi totien adalah saling prima, jika tidak maka proses akan kembali ke langkah membuat bilangan prima p dan q.
f. Jika nilai eksponen publik dan fungsi totien adalah saling prima, hitung d dengan cara menghitung invers modular dari nilai eksponen publik modulo fungsi totien Euler.
g. Setelah mendapatkan nilai d, langkah selanjutnya adalah membuat kunci privat menggunakan data yang telah dikumpul yakni modulus n, nilai eksponen publik, d, p, dan q.
h. Ulangi pembuatan kunci privat sebanyak sepuluh kali.

Setelah selesai membuat kunci privat yang akan digunakan dalam membrute force *ciphertext*, langkah selanjutnya adalah mencoba mendekripsi paksa *ciphertext* sebanyak sepuluh. Hasil dari pengujian brute force pada *ciphertext* dapat dilihat pada Tabel 2 sebagai berikut:

Tabel 2. Hasil Pengujian Brute Force

No	Kunci Yang Digunakan	Hasil
1.	private_key_0.pem	*Ciphertext* with incorrect length
2.	private_key_1.pem	*Ciphertext* with incorrect length
3.	private_key_2.pem	*Ciphertext* with incorrect length
4.	private_key_3.pem	*Ciphertext* with incorrect length
5.	private_key_4.pem	*Ciphertext* with incorrect length
6.	private_key_5.pem	*Ciphertext* with incorrect length
7.	private_key_6.pem	*Ciphertext* with incorrect length
8.	private_key_7.pem	*Ciphertext* with incorrect length
9.	private_key_8.pem	*Ciphertext* with incorrect length
10.	private_key_9.pem	*Ciphertext* with incorrect length

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

Dapat dilihat dari tabel 2 hasil pengujian brute force didapatkan bahwa terjadi error saat proses pendekripsian yang dikarenakan panjang *ciphertext* yang salah. Hal ini disebabkan karena RSA digunakan dalam pengenkripsian kunci AES di mana panjang kunci AES berbeda dengan panjang *ciphertext*. *Ciphertext* yang dikirim sendiri merupakan gabungan dari IV, kunci AES yang dienkripsi menggunakan RSA, dan pesan yang dienkripsi menggunakan AES yang kemudian urutannya diacak. Sehingga perlu langkah awal sebelum mendekripsi *ciphertext* yang dikirim, yakni dengan mengurutkan kembali urutannya kemudian memisahkan IV, kunci AES yang dienkripsi menggunakan RSA, dan pesan yang dienkripsi menggunakan AES.

4. Kesimpulan

Berdasarkan hasil pengujian dalam penelitian ini, terdapat beberapa kesimpulan yang dapat ditarik, antara lain:

a. Aplikasi yang dikembangkan menggunakan hybrid crytosystem AES dan RSA dapat berfungsi dengan sangat baik dan memenuhi kebutuhan fungsional, hal ini dibuktikan dari hasil pengujian blackbox. Di mana dalam pengujian blackbox aplikasi berhasil melewati semua pengujian dengan hasil yang sesuai harapan.
b. Ketahanan enkripsi terhadap serangan brute force sangat baik, di mana dari setiap percobaan brute force yang telah dilakukan menghasilkan error saat proses pendekripsian yang dikarenakan panjang *ciphertext* yang salah.

Adapun saran untuk pengembangan aplikasi lebih lanjut adalah penambahan fitur enkripsi dan dekripsi untuk file multimedia seperti gambar, audio, video, serta dokumen seperti PDF atau file Word. Sehingga penggunaan aplikasi dapat lebih luas lagi..

References

[1] S. A. M. Babys, “ANCAMAN PERANG SIBER DI ERA DIGITAL DAN SOLUSI KEAMANAN NASIONAL INDONESIA,” *JURNAL ORATIO DIRECTA*, vol. 3, pp. 425–442, 2021.

[2] B. Schneier, “Applied Cryptography, Protocols, Algorithms, and Source Code in C,” 1995.

[3] S. I. Lestariningati, “Rekayasa Internet.”

[4] S. H. Murad and K. H. Rahouma, “Hybrid Cryptography for Cloud Security: Methodologies and Designs,” in *Lecture Notes in Networks and Systems*, Springer Science and Business Media Deutschland GmbH, 2022, pp. 129–140. doi: 10.1007/978-981-16-2275-5_7.

[5] M. Iqbal Zulfikar, G. Abdillah, and A. Komarudin, “Kriptografi untuk Keamanan Pengiriman *E-mail* Menggunakan Blowfish dan Rivest Shamir Adleman (RSA),” 2019.

[6] S. H. Murad and K. H. Rahouma, “Implementation and Performance Analysis of Hybrid Cryptographic Schemes applied in Cloud Computing Environment,” in *Procedia Computer Science*, Elsevier B.V., 2021, pp. 165–172. doi: 10.1016/j.procs.2021.10.070.

This page is intentionally left blank.