IMPLEMENTASI ALGORITMA VIGENERE CIPHER DAN RIVEST CODE 4 UNTUK ENKRIPSI DAN DEKRIPSI PRIVATE MAIL PADA PT. EMBEE PLUMBON TEKSTIL

Disusun Oleh:

N a m a : Radhian Fariz Muhammad

NIM : 14523175

PROGRAM STUDI INFORMATIKA – PROGRAM SARJANA

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ISLAM INDONESIA

2020

HALAMAN PENGESAHAN DOSEN PEMBIMBING

IMPLEMENTASI ALGORITMA VIGENERE CIPHER DAN RIVEST CODE 4 UNTUK ENKRIPSI DAN DEKRIPSI PRIVATE MAIL PADA PT. EMBEE PLUMBON TEKSTIL

TUGAS AKHIR

Disusun Oleh:

N a m a : Radhian Fariz Muhammad

NIM : 14523175

Yogyakarta, 15 Agustus 2020

Pembimbing,

(Dr. Yudi Prayudi, S.Si, M.Kom.)

HALAMAN PENGESAHAN DOSEN PEMBIMBING

IMPLEMENTASI ALGORITMA VIGENERE CIPHER DAN RIVEST CODE 4 UNTUK ENKRIPSI DAN DEKRIPSI PRIVATE MAIL PADA PT. EMBEE PLUMBON TEKSTIL

TUGAS AKHIR

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer dari Program Studi Informatika di Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, 15 Agustus 2020

Tim Penguji

Dr. Yudi Prayudi, S.Si., M.Kom. _____

Anggota 1

Fietyata Yudha, S.Kom., M.Kom. _____

Anggota 2

Septia Rani, S.T., M.Cs. _____

Mengetahui,

Ketua Program Studi Informatika – Program Sarjana

Fakultas Teknologi Industri

Universitas Islam Indonesia

(Dr. Raden Teduh Dirgahayu, Dr., S.T., M.Sc.)

iv

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama : Radhian Fariz Muhammad

NIM : 14523175

Tugas akhir dengan judul:

IMPLEMENTASI ALGORITMA VIGENERE CIPHER DAN RIVEST CODE 4 UNTUK ENKRIPSI DAN DEKRIPSI PRIVATE MAIL PADA PT. EMBEE PLUMBON TEKSTIL

Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila dikemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, tugas akhir yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung resiko dan konsekuensi apapun.

Demikian surat pernyataan ini dibuat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 15 Agustus 2020

(Radhian Fariz Muhammad)

HALAMAN PERSEMBAHAN

Assalamu'alaikum Warahmatullahi Wabarakatuh

Alhamdulillahirobbil'alamin

Untuk Keluarga

Kupersembahkan Tugas Akhir ini untuk kedua orang tuaku Ibu dan Bapak yang selalu sabar dan percaya kepada anaknya. Terima kasih atas doa, dukungan, dan nasehat yang selalu diberikan. Tidak lupa saya ucapkan maaf yang setulus-tulusnya karena saya begitu lama untuk mengerjakan tugas akhir ini.

Kepada kakakku, terima kasih atas segala dukungannya sehingga tugas akhir ini akhirnya bisa terselesaikan.

Untuk sahabat dan teman-teman ku

Sahabat yang selalu ada saat aku membuatuhkan mereka. Terima kasih kepada Rani dan Akbar yang selalu mendukung dan memotivasi agar saya cepat-cepat menyelesaikan tugas akhir ini. Terima kasih kepada geng barate maminggo dan keluarga kost Santoso, atas segala bentuk dukungan yang pernah kalian berikan.

Wassalamu'alaikum warahmatullahi wabarakatuh

HALAMAN MOTO

"Sesungguhnya Allah bersama dengan orang-orang yang sabar"

(Q.S. Al-Baqaarah : 153)

"*Maka sesungguhnya bersama kesulitan itu ada kemudahan. Sesungguhnya bersama kesulitan itu ada kemudahan*"

(Q.S. Al-Insyirah : 5-6)

"Jika kalian bersyukur maka akan Aku tambahkan nikmat-Ku untuk kalian"

(Q.S. Ibrahim : 7)

"Ingatlah kepada-Ku niscaya Aku ingat kepada kalian"

(Q.S. Al-Baqarah : 152)

"*Pendidikan memiliki akar yang pahit tapi buahnya manis*"

(Aristoteles)

Hari ini harus lebih baik dari kemarin dan hari esok harus lebih baik dari hari ini

KATA PENGANTAR

Assalamualaikum Wareahmatullahi Wabarakatuh

Segala puji syukur kami panjatkan kepada Tuhan Yang Maha Esa. Atas rahmat dan karunia-Nya, kami dapat menyelesaikan tugas penulisan makalah mata kuliah bahasa Indonesia tepat waktu. Tidak lupa shalawat serta salam tercurah kepada Rasulullah SAW yang syafa'atnya kita nantikan kelak.

Penulisan laporan Tugas Akhir yang berjudul "Implementasi Algoritma Vigenere Cipher dan Rivest Code 4 Untuk Enkripsi dan Dekripsi Private Mail Pada PT. Embee Plumbon Tekstil" ini dibuat sebagai salah satu syarat untuk memperoleh gelar Sarjana di Jurusan Informatika, Fakultas Teknologi Industri, Universitas Islam Indonesia, Yogyakarta. Selama proses penulisan laporan Tugas Akhir ini saya sebagai penulis mengalami banyak kesulitan namun dapat diatasi berkat bantuan dari berbagai pihak. Oleh karena itu, pada kesempatan ini penulis mengucapkan terima kasih kepada:

Allah SWT atas segala karunia-Nya sehingga penulis diberi kemudahan selama menyusun Tugas Akhir.

Bapak Dr. Yudi Prayudi, S. Si, M. Kom., selaku Dosen Pembimbing Tugas Akhir yang sudah memberi bimbingan serta nasehat kepada penulis sehingga dapat membantu dalam penyusunan tugas akhir ini.

Seluruh Staf yang bekerja di Universitas Islam Indonesia khususnya dosen-dosen yang telah memberikan ilmu selama kuliah, penulis ucapkan terima kasih sebesar-besarnya.

Penulis menyadari bahwa dalam penulisan Tugas Akhir ini masih memiliki banyak kekurangan yang harus diperbaiki. Oleh karena itu kritik serta saran yang membangun sangat penulis harapkan. Semoga laporan ini dapat bermanfaat bagi para pembaca.

Wassalamu'alaikum warahmatullahi wabarakatuh

Yogyakarta, 15 Agustus 2020

(Radhian Fariz Muhammad)

SARI

Email sebagai media komunikasi melalui internet saat ini sering digunakan untuk kebutuhan bertukar pesan. Tetapi penggunaannya memiliki permasalahan dalam aspek keamanan. Terkadang data yang dikirimkan merupakan data yang bersifat rahasia yang isinya tidak boleh diketahui oleh orang lain. Salah satu upaya yang dapat dilakukan untuk meningkatkan keamanan pada informasi adalah penerapan teknik kriptografi. Kriptografi adalah ilmu yang melakukan proses enkripsi dan dekripsi pada data dan informasi sehingga membantu mengurangi resiko bocornya data dan informasi. Penulis sadar bahwa penerapan satu teknik kriptografi hanya mengurangi sedikit resiko dari kebocoran data dan informasi oleh karna itu penulis menggunakan dua teknik kriptografi.

Pada penelitian ini penulis membangun aplikasi untuk mengamankan data yang dikirimkan melalui email dengan menggunakan metode vigenere cipher dan Rivest Code 4. Kemudian aplikasi diimplementasikan dan dilakukan pengujian terhadap responden yaitu Tim IT Support pada PT. Embee Plumbon Tekstil.

Pengujian yang telah dilaksanakan menunjukan hasil yang baik. Hasil yang diperoleh memperlihatkan bahwa penelitian ini telah berhasil mengimplementasikan algoritma Vigenere Cipher dan Rivest Code 4. Selain itu aplikasi yang dibangun sudah mampu memenuhi tujuan kriptografi yaitu c*onfidentiality, integrity, authentication* serta *non-repidiation.*

Kata kunci: Kriptografi, RC4, Vigenere Cipher, Email, Web

GLOSARIUM

ASCII	standar internasional dalam kode huruf dan simbol seperti Hex dan Unicode.
Ciphertext	pesan tersandi hasil enkripsi, tidak dapat dibaca dengan jelas.
Dekripsi	proses mengubah *ciphertext* menjadi *plaintext*
Email Server	perangkat lunak yang berguna untuk mengatur proses pengiriman serta penerimaan *email* yang ada di internet.
Enkripsi	proses mengubah *plaintext* menjadi *ciphertext*
Kriptanalis	ilmu dan seni untuk memecahkan cipherteks menjadi plainteks, tanpa memerlukan kunci yang digunakan.
Kriptografi	seni dan ilmu untuk menjaga keamanan pesan
Plaintext	pesan yang dirahasikan.
SMTP	protokol untuk berkomunikasi dengan server guna mengirimkan email dari lokal email ke server, sebelum akhirnya dikirimkan ke server email penerima.
UML	metodologi untuk mengembangkan sistem OOP dan sekelompok perangkat tool untuk mendukung pengembangan sistem tersebut.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1. 1 Latar Belakang

Dengan perkembangan teknologi informasi dan komunikasi yang berkembang sangat pesat namun juga dibarengi dengan tuntutan keamanan dan kerahasiaan informasi yang disampaikan, banyak perangkat keras dan perangkat lunak dengan fitur dan teknologi baru yang diciptakan untuk memenuhi tuntutan dalam bidang tersebut. Media komunikasi yang banyak digunakan tentu harus merupakan media yang mudah oleh semua orang. Salah satu contoh media komunikasi yang saat ini sering digunakan adalah email (Try, 2009).

Email disebut salah satu jenis service yang tidak bisa terlepas dari aktivitas yang terjadi di internet. Email banyak dimanfaatkan di berbagai bidang salah satunya untuk bisnis dan social network. Hal ini karena electronic mail atau email dapat melakukan pengiriman pesan secara cepat serta dapat mengirimkan file penting ataupun hal lainnya yang tidak lepas dari proses transfer informasi. Aplikasi Microsoft Outlook dan Mozila Thunderbird sendiri merupakan apliaksi yang dibuat khusus untuk pengiriman serta penerimaan email dari pihak ketiga. Pengiriman serta penerimaan email sehingga bisa dibaca secara offline merupakan kemudahan yang ditawarkan oleh dua aplikasi tersebut.

Kecepatan pengiriman dan penerimaan email merupakan salah satu hal terpenting dalam proses pengiriman dan penerimaan email. Dua aplikasi yang sudah disebutkan diatas yaitu Microsoft Outlook dan Thunderbird sendiri sudah menerapkan penerimaan email. Dalam penerapan kedua aplikasi tersebut memiliki kekurangan yaitu sangat selektif dalam hal pemilihan attachment yang ada dalam email sehingga sangat sulit untuk membaca semua attachment.

Email server adalah perangkat lunak yang berguna untuk mengatur proses pengiriman serta penerimaan email yang ada di internet. Biasanya email server sudah tersedia dalam layanan hosting yang ditawarkan provider internet, dalam pembuatan skripsi ini penulis menggunakan sebuah software open source bernama HmailServer untuk membuat sebuah mail server sendiri yang bisa digunakan secara offline dengan mengandalkan jaringan LAN, penggunaan layanan ini memberikan hak akses untuk proses POP3 diluar layanan aplikasi sehingga bisa berkomunikasi dengan email client pihak ketiga dalam hal ini adalah aplikasi yang penulis buat.

PT. Embee Plumbon Tekstil adalah sebuah perusahaan industri yang berkembang di industri tekstil di pasar global. Dalam melakukan proses komunikasi, semua elemen yang ada di perusahaan PT. Embee Plumbon Tekstil menggunakan layanan email untuk saling bertukar informasi. Beberapa data yang dikomunikasian menggunakan email bersifat rahasia dan penting, seperti data finance, data production, data accounting merupakan salah satu data rahasia perusahaan. Informasi yang demikian tentunya akan berdampak buruk apabila jatuh ke tangan yang tidak bertanggung jawab, yang dapat dapat berakibat terhadap kinerja perusahaan. Sudah seharusnya informasi yang disampaikan melalui perantara email terjamin keamanannya pada saat didistribusikan dan disimpan di media penyimpanan baik publik maupun privat. Terlebih lagi isi dari email tersebut bersifat penting dan rahasia, sehingga tidak mudah dicuri maupun dimanipulasi oleh pihak yang tidak bertanggung jawab.

Dari permasalahan yang ada agar dapat meningkatkan keamanan data / informasi yang dikirim email di perlukannya suatu metode pengamanan yang dapat dipercaya. Salah satunya adalah menerapkan metode kriptografi untuk menjaga keaslian dan kerasahasiaan data / informasi yang dikirim melalui email, sehingga penyadap akan kesulitan untuk mengetahui isi informasi yang sebenarnya.

Kriptografi merupakan ilmu serta seni untuk melindungi keamanan pesan kala pesan dikirim dari suatu tempat ke tempat lain (Pradipta, 2016). Pesan yang dirahasiakan dinamakan plaintext, sebaliknya pesan hasil penyandian disebut ciphertext. Enkripsi atau disebut juga proses penyandian plaintext jadi ciphertext serta proses membalikkan ciphertext jadi plaintext asalnya disebut dekripsi. Ada 2 metode dalam kriptografi yang digunakan untuk penyandian teks ialah kriptografi klasik (kriptografi simetri) serta kriptografi modern (kriptografi asimetri). Contoh dari kriptografi simetri yakni vigenere cipher serta rivest code.

Vigenere cipher adalah salah satu dari kriptografi klasik simetri dan merupakan bentuk sederhana dari cipher substitusi polialfabetik. Dalam proses enkripsinya vigenere cipher menggunakan metode substitusi abjad-majemuk, dimana setiap huruf yang ada akan dienkripsi menggunakan kunci yang berbeda, tidak seperti cipher substitusi lainnya (Muhammad, 2017).

Rivest code 4 adalah salah satu jenis stream cipher, yaitu memproses unit atau input data pada satu saat. Unit atau data pada umumnya sebuah byte atau bahkan kadang kadang bit (byte dalam hal RC4). Panjang kunci yang digunakan 1 sampai 256 byte algoritma kriptografi ini sederhana dan mudah diimplementasikan dengan kecepatan proses enkripsi dan dekripsi yang terbilang cukup cepat (Halim & Budiman, 2015).

Vigenere cipher dan RC4 merupakan algoritma kriptografi yang memiliki kinerja bagus yaitu dengan mendeskripsikan setiap huruf dengan kunci yang berbeda serta panjang kunci yang digunakan mencapai 256 byte. Dengan menggabungkan dua metode akan mempersempit kemungkinan kebocoran data enkripsi sehingga data menjadi lebih aman. Oleh karena itu dua metode tersebut dipilih untuk enkripsi dan dekripsi privat mail pada PT. Embee Plumbon Tekstil.

1. 2 Rumusan Masalah

Rumusan masalah pada penelitian ini adalah bagaimana merancang sistem yang dapat mengamankan isi data yang dikirim melalui email dengan menggunakan metode vigenere cipher dan rivest code 4?

1. 3 Batasan Masalah

Agar pembahasan dalam Tugas Akhir ini terfokuskan, penelitian ini memiliki beberapa batasan masalah sebagai berikut:

a. Penelitian di lakukan di perusahaan PT. Embee Plumbon Tekstil.
b. Membahas enkripsi dan dekripsi email menggunakan algoritma *vigenere cipher* dan *rivest code 4* saat mengirim dan menerima email, bukan membahas keamanan jalur transfernya.
c. Tipe data email yang di enkripsi dan dekripsi ialah *plaintext* dan *attachment file*.
d. Tipe *file* yang dikirim hanya berupa format docx, xls, pdf, dan ppt.

1. 4 Manfaat Penelitian

Dari hasil penelitian ini di harapkan dapat memberikan kontribusi yang bermanfaat antara lain:

a. Menambah pehamanan mengenai kriptografi terutama tentang metode vigenere cipher dan rivest code 4.
b. Aplikasi yang dirancang dapat menyandikan isi pesan yang hendak di kirimkan melalui email tanpa takut dibaca oleh orang yang tidak memiliki hak dengan menggunakan aplikasi ini.

1. 5 Tujuan Penelitian

Tujuan dari penelitian ini adalah sebagai berikut:

Merancang dan membangun purwarupa email client yang mampu melakukan enkripsi dan dekripsi dengan menerapkan ilmu kriptografi *vigenere cipher* dan *rivest code* 4 sehingga data yang dikirim melalui email tidak dapat dibaca oleh pihak yang tidak berkepentingan.

1. 6 Metodelogi Penelitian

Dalam penelitian ini digunakan metode sebagai berikut:

a. Pengumpulan Data dan Studi *Literature*

Pada tahap ini dilakukan pencarian dan pemahaman *Literature* serta pengumpulan informasi tentang email dan algoritma kriptografi *vigenere cipher* serta algoritma kriptografi *rivest code* 4 yang akan diimplementasikan untuk keamanan data yang dikirim melalui email. *Literature* yang digunakan dapat berupa artikel, jurnal, buku, dan berbagai sumber informasi lainnya.

b. Perumusan Masalah dan Penyelesainnya

Tahap ini meliputi perumusan masalah, batasan-batasan masalah dan penyelesainnya.

c. Analisis dan Perancangan Sistem

Pada tahap ini dilakukan analisis terhadap sistem dengan menentukan batasan sistem dan alur kerja dari sistem berdasarkan data dan informasi yang diperoleh dari studi literatur. Tahap perancangan sistem diawali dengan merancang sistem sesuai dengan analisis, tujuan, dan batasan penelitian. Perancangan sistem dilakukan dengan membuat flowchart dan interface sistem.

d. Implementasi Sistem

Pada tahap ini akan dilakukan pembangunan aplikasi untuk mengirim dan menerima data email, yang mana dalam pembangunan aplkasi tersebut diterapkan teori/ algoritma yang telah dipelajari yaitu algoritma *vigenere cipher* dan *rivest code* 4.

e. Pengujian Aplikasi

Pada tahap ini dilakukan uji coba terhadap sistem yang sudah dibangun guna mengetahui apakah sistem tersebut sudah sesuai dengan hasil rancangan yang sudah dilakukan, sehingga mencapai hasil akhir yang sesuai.

f. Penyusunan Laporan

Penyusunan laporan akhir merupakan proses dokumentasi dari proses pelaksanaan atau penyusunan penelitian dan diharapkan dapat bermanfaat bagi penelitian lebih lanjut.

1. 7 Sistematika Pembahasan

Untuk memahami lebih jelas laporan tugas akhir ini, penulis melakukan pengelompokan materi menjadi beberapa bab serta disusun secara sistematis dalam lima bab, dan tiap-tiap bab tersebut dibagi menjadi sub-sub bab. Adapun penulisannya adalah sebagai berikut:

BAB I PENDAHULUAN

Bab ini berisi penjelasan latar belakang permasalahan, rumusan masalah, batasan masalah, manfaat penelitian, tujuan penelitian, metodologi penelitian, serta sistematika penulisan penelitian.

BAB II LANDASAN TEORI

Bab ini berisikan teori-teori yang digunakan untuk mendukung penulisan penelitian, selain itu dalam bab 2 juga dijelaskan mengenai metode yang digunakan dalam penelitian.

BAB III METODOLOGI PENELITIAN

Bab ini berisikan tahapan proses pengerjaan tugas akhir yang meliputi studi literatur, analisis masalah, strategi pemecahan masalah, analisis kebutuhan serta perancangan sistem.

BAB IV HASIL DAN PEMBAHASAN

Bab ini berisi penjelasan implementasi dari tampilan sistem serta pengujian sistem yang telah diimplementasikan.

BAB V KESIMPULAN DAN SARAN

Bab ini berisi kesimpulan dan saran dari hasil penelitian yang berkaitan dengan hasil analisa yang telah diuraikan pada bab-bab sebelumnya.

BAB II
LANDASAN TEORI

2. 1 Keamanan Data

Persoalan keamanan serta kerahasiaan data ialah salah satu aspek penting dari suatu informasi. Secara umum keamanan komputer meliputi beberapa aspek, yaitu :

a. *Privacy / Confidentiality*

Privacy lebih kearah data-data yang sifatnya rahasia, sedangkan *confidentiality* berhubungan dengan data yang diberikan ke pihak lain untuk keperluan tertentu dan hanya diperbolehkan untuk keperluan tertentu.

b. *Integrity*

Integrity menekankan bahwa informasi tidak boleh diubah tanpa izin pemilik informasi. informasi yang diterima harus sesuai serta sama persis seperti ketika informasi dikirimkan. Bila ada perbedaan antara informasi atau data yang dikirim dengan yang diterima maka aspek integrity tidak tercapai.

c. *Authenticity*

Aspek ini berhubungan dengan metode atau cara untuk menyatakan bahwa informasi benar-benar asli, orang yang mengakses atau memberikan informasi adalah benar -benar orang yang dimaksud.

d. *Availability*

Aspek ini behubungan dengan ketersediaan data dan informasi. Data dan informasi yg tidak selaras dalam suatu sistem komputer tersedia dan bisa dimanfaatkan oleh orang yang berhak.

e. *Access Control*

Acces Control berhubungan dengan cara pengaturan akses pada informasi. Hal ini umumnya berhubungan dengan klasifikasi data, mekanisme authentication dan juga privacy. *Access control* acapkali dilakukan dengan memakai kombinasi user id/password atau dengan menggunakan mekanisme lain.

2.1. Email

Electronic mail atau email artinya sebuah metode yang digunakan untuk mengubah, mengirim, menyimpan, serta menerima pesan melalui sistem komunikasi elektronik. kata email mencakup sistem yang berdasar pada Simple Mail Transfer Protocol (SMTP) dan sistem

intranet yang memungkinkan pengguna dalam satu organisasi mengirimkan pesan pada satu sama lain. internet protocol sering kali dipakai oleh kelompok organisasi sebagai layanan email internal. Format dari sebuah pesan email dari internet didefinisikan di RFC 2822 dan seri dari RFC yang secara keseluruhan diklaim sebagai Multipurpose Internet Mail Extensions (MIME). Email terdiri dari dua bagian besar, yaitu:

a. *Header*

   Disusun menjadi beberapa *field*, biasanya nama *field* dimulai pada karakter pertama di suatu baris, diikuti oleh tanda ':', diikuti oleh nilai non-null, bukan spasi atau bukan tab pada karakter pertamanya. Nama *field* serta nilainya masuk dalam karakter ASCII sebesar 7 bit. Bagian *Header* serta *body* dipisahkan oleh satu baris kosong. Pesan pada umumnya paling sedikit mempunyai 4 *field* berikut:

   1. *From* : Alamat email dan terkadang diikuti nama pengirim pesan
   2. *To :* Alamat email dan terkadang diikuti nama penerima pesan.
   3. *Subject* : Rangkuman isi pesan
   4. *Date* : Waktu dan tanggal setempat saat pesan dikirim

b. *Body*

   Pesan yang diterima berupa teks tanpa struktur, terkadang mengandung tanda pengenal di bagian akhir. pada awalnya dirancang memakai 7-bit ASCII, akan tetapi kini sebagian besar memakai 8-bit, tetapi belum bersifat universal.

2.2. Kriptografi

Kriptografi artinya ilmu perihal teknik enkripsi dimana data diacak menggunakan suatu kunci enkripsi menjadi sesuatu yang sulit dibaca oleh seseorang yg tidak memiliki kunci dekripsi (Kromodimoeljo, 2009).

Tetapi di pengertian terbaru kriptografi merupakan ilmu yang bersandarkan pada teknik matematika untuk menjaga keamanan informasi seperti kerahasiaan, keutuhan data serta otentikasi entitas. Jadi pengertian kriptografi terkini ialah tidak hanya berurusan dengan penyembunyian pesan, tetapi lebih pada sekumpulan teknik yang menyediakan keamanan informasi.

2. 3. 1. Tujuan Kriptografi

Tujuan kriptografi untuk memberikan layanan keamanan (Paar, 2010) sebagai berikut:

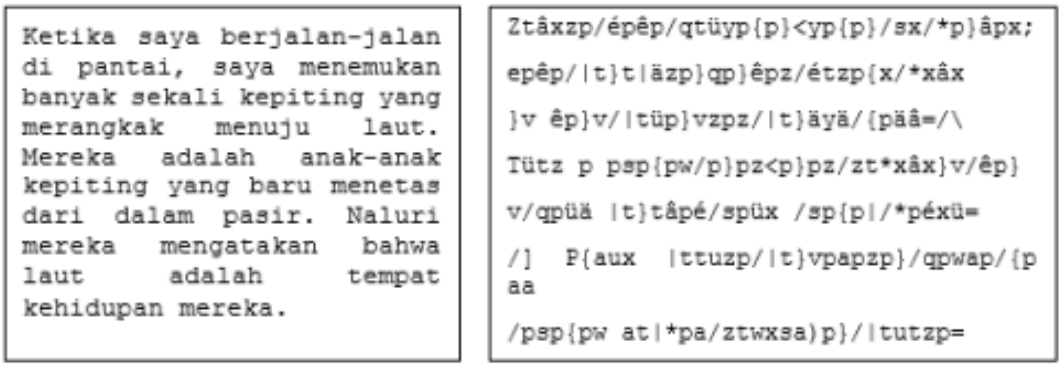
a. Kerahasian (Confidentiality) Informasi dirahasiakan dari semua pihak yang tidak berwenang.
b. Keutuhan Data (*Integrity*) Pesan tidak berubah dalam proses pengiriman hingga pesan diterima oleh penerima.
c. Autentikasi (*Aunthentication*) Kepastian terhadap identitas setiap entitas yang terlibat dan keaslian sumber data.
d. Nirpenyangkalan (*Non-repidiation*) Setiap entitas yang berkomunikasi tidak dapat menolak atau menyangkal atas data yang telah dikirim atau diterima.

2. 3. 2. Terminologi Kriptografi

Di dalam kriptografi, sering ditemukan berbagai istilah atau terminologi sebagai berikut:

a. Plaintext dan Ciphertext.

Informasi atau data yang dapat dibaca dan dimengerti maknanya disebut dengan pesan. Nama lain dari pesan sendiri adalah plainteks (plaintext) atau teks-jelas (cleartext). Pesan perlu disandikan ke dalam bentuk lain yang tidak dapat dipahami oleh orang lain. Cipherteks (ciphertext) atau kriptogram (cryptogram) merupakan bentuk dari pesan yang diberi sandi. Cipherteks juga harus dapat ditransformasikan kembali menjadi plainteks semula agar pesan yang diterima bisa dibaca.



(a) Plainteks (b) Cipherteks

Gambar 2. 1 Plainteks dan Cipherteks

Sumber : (Munir, 2006)

Pengirim dan Penerima

Pengirim (*sender*) adalah entitas yang mengirim pesan kepada entitas lainnya. Penerima (*receiver*) adalah entitas yang menerima pesan. Entitas di sini dapat berupa orang, mesin (komputer), kartu kredit, dan sebagainya.

Enkripsi dan Dekripsi

Enkripsi (*encryption*) atau enciphering adalah Proses menyandikan plainteks menjadi cipherteks. Sedangkan, proses mengembalikan cipherteks menjadi plainteks semula dinamakan dekripsi *(decryption)* atau deciphering.

Kriptanalis

Kriptografi berkembang sedemikian rupa sehingga melahirkan bidang yang berlawanan yaitu kriptanalisis (Munir, 2006). Kriptanalis (cryptanalysis) adalah ilmu dan seni untuk memecahkan cipherteks menjadi plainteks, tanpa memerlukan kunci yang digunakan. Pelakunya disebut dengan cryptanalyst. Kriptanalis berusaha memecahkan cipherteks tersebut untuk menemukan plainteks atau kunci. Kriptologi (cryptology) adalah studi mengenai kriptografi dan kriptanalis.

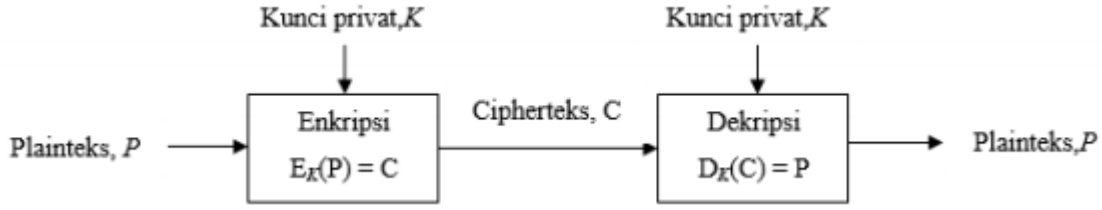
2. 3. 3. Jenis-Jenis Algoritma Kriptografi

Secara umum ada dua jenis kriptografi berdasarkan kuncinya, yaitu :

a. Algoritma Simetris

Algoritma Simetris adalah algoritma yang menggunakan kunci yang sama pada enkripsi dan dekripsinya. Aplikasi kriptografi simetri yang utama adalah melindungi kerahasiaan data yang dikirim melalui saluran tidak aman dan melindungi kerahasiaan data yang disimpan pada media yang tidak aman.

Kelemahan dari sistem ini adalah baik pengirim maupun penerima pesan harus memiliki kunci yang sama, sehingga pengirim pesan harus mencari cara yang aman untuk memberitahukan kunci kepada penerima pesan (Munir, 2006). Skema kriptografi simetris disebut juga sebagai skema atau algoritma kunci simetris (*symmetric-key*), kunci rahasia (*secret-key*), dan *single-key* (Paar, 2010).



Gambar 2. 2 Skema Kriptografi Simetris

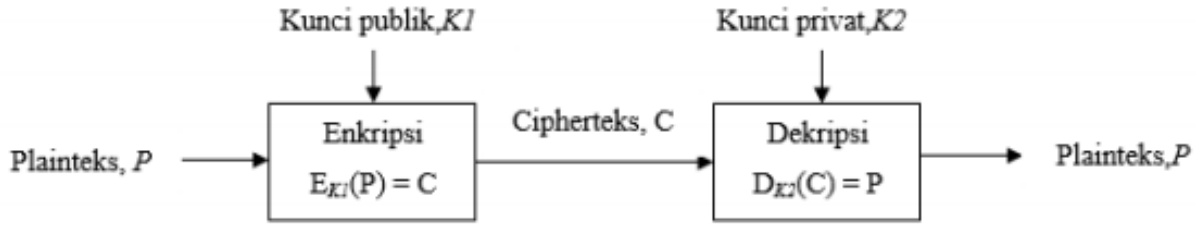
Sumber : (Munir, 2006)

Algoritma yang termasuk dalam algoritma simetri adalah DES (Data *Encryption* Standard), Blowfish, Twofish, Triple-DES, IDEA, Serpent, RC4, RC4A, Hill Cipher, Affine Cipher, OTP, AES (Advanced *Encryption* Standard), dsb.

b. Algoritma Asimetris

Algoritma Asimetris adalah algoritma kriptografi yang mempergunakan kunci yang berbeda pada enkripsi dan dekripsinya. Nama lainnya adalah kriptografi kunci publik (*public-key cryptography*), sebab kunci untuk enkripsi tidak rahasia dan dapat diketahui siapapun (diumumkan ke publik), sementara kunci untuk dekripsi hanya diketahui oleh penerima pesan (karena itu rahasia).

Pada kriptografi jenis ini, setiap orang yang berkomunikasi mempunyai sepasang kunci, yaitu kunci privat dan kunci publik. Pengirim mengenkripsi pesan dengan menggunakan kunci publik si penerima pesan (*receiver*). Hanya penerima pesan yang dapat mendekripsi pesan karena hanya ia yang mengetahui kunci privatnya sendiri (Munir, 2006).



Gambar 2. 3 Skema Kriptografi Asimetris

Sumber : (Munir, 2006)

2.3. Algoritma Vigenere Cipher

Pada tahun 1986 salah satu algoritma kriptografi klasik diperkenalkan pada masyarakat, Vigenere Cipher merupakan sebutan algoritma tersebut. Orang yang mempublikasikannya adalah diplomat dan juga kriptologis asal Prancis yaitu Blaise de Vigenere. Namun pada dasarnya algoritma tersebut sudah pernah dituliskan oleh Giovan Batista Belaso pada tahun 1553 disebuah buku dengan judul La Cifra del Sig.

Cara kerja algoritma Vigenere cipher sangat mirip dengan algoritma Caesar Cipher yang juga mengenkripsi plainteks pada pesan. Algoritma Casar Cipher sendiri menerapkan metode substitusi abjad-tunggal yaitu berarti semua hurup pada suatu pesan dienkripsikan menggunakan kuncu yang sama. Akan tetapi Vigenere Cipher menggunakan metode substitusi abjad majemuk yang berarti mengenkripsikan setiap huruf yang ada menggunakan kunci yang berbeda.

Sebagai contoh Caesar cipher jika terdapat plainteks:

MAKALAH KRIPTOGRAFI

Maka jika dienkripsi dengan dengan nilai kunci 2 akan didapat cipherteks:

OCMCNCJ MTKRVQITCHK

Dari cipherteks yang didapat dapat kita lihat bahwa huruf M dienkripsi menjadi O, huruf A dienkripsi menjadi huruf C, dan seterusnya dimana huruf pada pesan digeser sejauh nilai kunci. Algoritma Caesar cipher sangat sederhana sehingga sanga berisiko untuk dipecahkan karena hanya dibutuhkan pengetahuan satu huruf dari plainteks untuk mengetahui kunci yang digunakan.

Vigenère cipher yang menerapkan metode substitusi abjad-majemuk tidak memiliki permasalahan tersebut karena setiap huruf pada pesan yang dienkripsi dengan Vigenère cipher ini akan digeser dengan nilai yang berbeda tergantung dengan kunci yang diberikan. Kunci yang digunakan pada Vigenère cipher berbeda denganyang digunakan pada Caesar cipher. Jika pada Caesar cipher kuncinya hanya satu nilai saja, maka pada Vigenère cipher kunci yang digunakan berbentuk deretan huruf.

Kunci yang berbetuk deretan kata tersebut akan memungkinkan setiap huruf plainteks untuk dienkripsi dengan kunci yang berbeda. Jika panjang kunci yang digunakan lebih pendek dari panjang plainteks maka kunci akan diulang sampai panjang kunci samdengan panjang

plainteks. Algoritma ini akanmeminimalkan kemungkinan dipecahkannya cipherteks jika satu huruf plainteks diketahui. Model matematika dari enkripsi pada algoritma Vigenère cipher ini adalah seperti berikut:

Ci= Ek(Mi) = (Mi + Ki) mod 26

Dan model matematika untuk deskripsinya adalah:

Mi= Dk(Ci) = (Ci - Ki) mod 26

Dengan C memodelkan cipherteks, M memodelkan Plainteks, dan K memodelkan kunci. Contoh dari penerapan algoritma Vigenère cipher adalah jika kita memiliki sebuah plainteks yang ingin dienkripsi:

MAKALAH KRIPTOGRAFI

Dan kita menggunakan kunci:

TUGAS

Maka plainteks akan dienkripsi dengan cara:

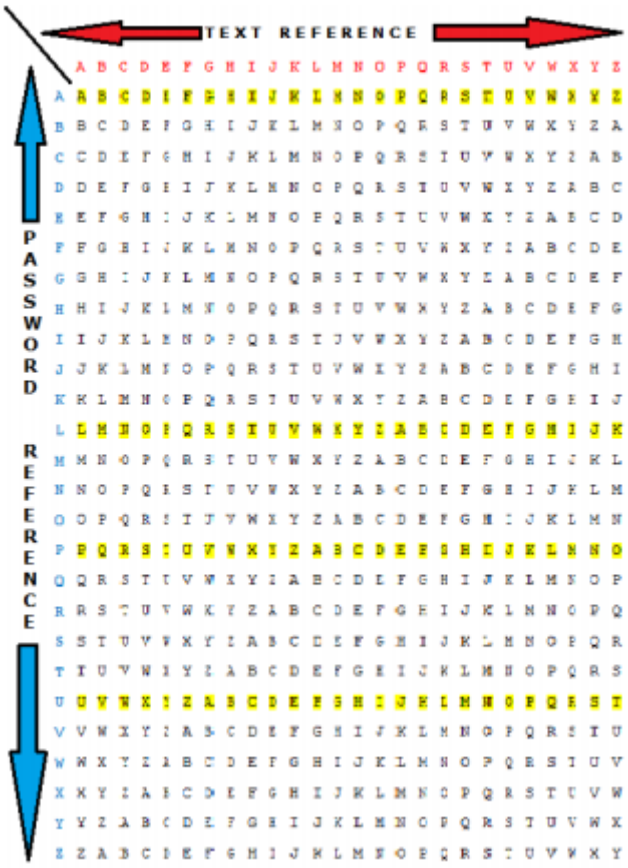
Plaintext : MAKALAH KRIPTOGRAFI

Kunci : TUGASTU GASTUGASTUG

Ciphertext : FUQADTB QRAINUGJTZO

Huruf pada kunci akan dikonversi menjadi sebuah nilai, misalnya A = 0, B = 1, sampai dengan Z = 25. Setelah itu prosesnya sama seperti pada Caesar cipher dimana setiap huruf pada plainteks akan digeser sejauh nilai kunci yang posisinya bersesuaian. Pergeseran huruf-huruf ini bisa dipetakan dalam bentuk tabel 26x26 yang memetakan antara huruf pada plainteks dengan huruf pada kunci.

Tabel 2. 1 Tabel Pemetaan Vigenere Cipher



Selain menggunakan Algoritma Vigenere Cipher bujur sangkar Vigenere untuk melakukan algoritma ini dapat dilakukan dengan menjumlahkan *plaintext* dengan kunci kemudian di modulo 26. Dengan asumsi a = 0, b=1, c=2, ….. z = 25.

2.4. Algoritma RC4

2.5.1. Pengertian Algoritma RC4

Menurut Kromodimoedjo (Halim & Budiman, 2015) Agoritma RC4 (Rivest Code 4) merupakan salah satu algoritma kunci simetris yang dibuat oleh RSA Data Security Inc (RSADSI) yang berbentuk stream chipper. Algoritma ini ditemukanpada tahun 1978 oleh Ronald Rivest dan menjadi *simbol* keamanan RSA (Rivest Shamir Adleman). RC4 merupakan salah satu jenis stream cipher sehingga RC4 memproses unit atau input data, pesan atau informasi pada satu saat. Dengan cara ini enkripsi atau dekripsi dapat dilaksanakan pada panjang yang variabel.

Algoritma ini tidak harus menunggu sejumlah input data tertentu sebelum diproses, atau menambahkan *byte* tambahan untuk mengenkripsi. RC4 menggunakan panjang variabel kunci dari 1 sampai 256 *byte* untuk menginisialisasi *state table*. *State table* digunakan untuk pengurutan menghasilkan *byte* pseudo-random yang kemudian menjadi stream pseudo-random. Setelah di XOR dengan *plaintext* sehingga didapatkan *ciphertext*. Tiap elemen pada *state table* di *swap* sedikitnya sekali. Kunci RC4 sering dibatasi sampai 40 bit, tetapi dimungkinkan untuk menggunakan kunci 128-bit. RC4 memiliki kemampuan penggunaan kunci antara 1 sampai 2048 bit. Panjang kunci merupakan factor utama dalam sekuritas data. RC4 dapat memili kunci sampai 128-bit.

Algoritma RC4 dapat digunakan untuk mengenkripsi dengan mengombinasikannya dengan plainteks dengan menggunakan bit-wise Xor (Exclusive-or). Proses dekripsinya dilakukan dengan cara yang sama (karena Xor merupakn fungsi simetrik). Untuk menghasilkan *key*stream cipher menggunakan state internal yang meliputi dua bagian:

a. Sebuah permutasi dari 256 kemungkinan *byte*.
b. Indeks pointer 8-bit.

2. 5. 2. Tahap *Key* Scheduling Algorithm (KSA)

Menurut Kromodimoedjo (dalam Halim & Budiman, 2015) *Key* Scheduling dipergunakan untuk menentukan inisial dari permutasi di array S panjang kunci diartikan sebagai jumlah *byte* di kunci dan mempunyai rentang panjang kunci dari 1 sampai 256. Khususnya antara 5-16 tergantung dari panjang kunci 40-128 bit. Berikut adalah langkah langkah algoritma KSA:

Nilai S diisi dengan nilai sesuai indeksnya: S[0] = 0, S[1] = 1, ... , S[255] = 255.

Lakukan inisialisasi suatu nilai (j = 0 dan i = 0) kemudian lakukan perhitungan nilai j yang baru. Nilai j yang baru dirumuskan sebagai berikut :

j = (j + S[i] + *key*[i mod *keylength*]) mod 256

Dimana:

S[i] = Nilai S pada indeks ke-i.

key = Kunci enkripsi.

keylength = Besar kunci enkripsi dalam *bytes*.

Kemudian tukarkan nilai S[i] dengan S[j].

Lakukan langkah kedua dan ketiga untuk setiap nilai i = 0, i = 1, ... , i = 255.

Tabel 2. 2 Tabel Larik 256 bit Hasil *Key* Scheduling

78	103	71	171	47	127	223	143	216	66	153	185	15	69	244	167
152	8	29	147	97	227	195	174	108	155	100	224	7	182	188	146
133	176	10	255	180	58	88	84	92	122	228	193	54	205	93	5
207	197	181	177	51	62	63	115	86	45	237	48	72	126	186	194
111	156	144	231	253	136	64	238	160	137	19	217	89	193	91	61
28	23	50	105	94	221	172	132	229	211	166	98	44	230	178	130
134	198	90	0	245	241	34	30	27	18	208	79	95	117	249	85
140	119	179	219	254	52	39	168	138	169	112	32	14	204	201	25
129	110	246	40	26	120	35	16	21	199	159	13	202	184	191	175
162	109	251	247	225	77	113	65	96	81	139	164	70	236	183	215
104	125	239	20	43	76	206	187	114	75	190	1	60	83	234	149
243	73	189	82	57	46	86	141	101	220	161	200	4	56	131	53
87	33	242	6	250	3	222	74	9	173	165	24	12	106	26	233
213	123	17	22	42	142	99	37	148	163	196	218	49	203	210	41
68	107	154	212	170	235	252	128	118	158	11	214	147	59	248	135
124	55	240	102	151	121	2	67	145	31	232	209	116	226	150	38

2. 5. 3. Tahap Pseudo-random Generation (PRGA)

Menurut Kromodimoedjo (dalam Halim & Budiman, 2015) Tahap selanjutnya dari algoritma RC4 dinamakan Pseudo-random Generation Algorithm (PRGA). Tahap ini menghasilkan nilai pseudo-random yang kemudian dilakukan operasi XOR dengan *plaintext* untuk proses enkripsi atau dengan *ciphertext* pada proses dekripsi. Untuk langkah-langkah PRGA adalah sebagai berikut:

Inisialisasikan nilai i = 0 dan j = 0.

Hitung nilai i dan j yang baru dengan cara :

i = (i + 1) mod 256

j = (j + S[i]) mod 256

Tukarkan nilai S[i] dengan S[j].

Hitung nilai t = (S[i] + S[j]) mod 256.

Ambil nilai S[t] kemudian lakukan operasi XOR pada *plaintext* ataupun *ciphertext* dengan indeks k. Lakukan langkah kedua hingga kelima untuk setiap nilai k = 0, k = 1, ... , k = panjang pesan – 1.

Misalkan pada tahap *key* schedulling menghasilkan larik S seperti yang ditunjukkan oleh tabel 2.1 dan pesan yang akan dienkripsi adalah “jaya” dengan kode ASCII 106, 97, 121 dan 97. *Ciphertext* akan dihitung dengan menginisialisasikan nilai i dan j dengan nol. Selanjutnya, dari nilai nol sampai dengan panjang dari pesan dikurangi satu yaitu bernilai tiga, dilakukan perhitungan nilai i dan j yang baru. Hasil perhitungan pertama adalah:

i = (0 + 1) mod 256 = 1

j = (0 + S[1]) mod 256 = (0 + 103) mod 256 = 103

Nilai S[1] = 103 dan S[103] = 30 ditukarkan sehingga dihasilkan S[1] = 30 dan S[103] = 103. Selanjutnya akan dilakukan perhitungan nilai t. Hasil perhitungan nilai t adalah:

t = (S[1] + S[103]) mod 256 = (30 + 103) mod 256 = 133

Sehingga didapat nilai S[t] = S[133] dimana S[133] = 120. Kemudian nilai S[133] akan di-XORkan dengan pesan[k] = pesan[0] dimana pesan[0] yaitu “j” dengan kode ASCII sama dengan 106. Hasil perhitungannya adalah:

120 = 0111 1000

106 = <u>0110 1010</u> ⊕

0001 0010 = 18, yaitu kode ASCII untuk karakter “DC2”

Proses perhitungan tersebut diulang sampai karakter terakhir *plaintext*. Algoritma yang sama juga digunakan untuk proses dekripsi.

2.5. ASCII (American Standart Code for Information Interchange)

American Standart Code for Information Interchange atau ASCII merupakan suatu standar *internasional* kode atau *simbol* seperti Hex dan Unicode tetapi ASCII lebih bersifat universal, contohnya 124 adalah untuk karakter "|". Kode ASCII selalu digunakan oleh komputer dan alat komunikasi lain untuk menunjukkan teks. Kode ASCII sebenarnya memiliki komposisi bilangan biner sebanyak 7 bit. Namun, ASCII disimpan sebagai sandi 8 bit dengan menambakan satu angka 0 sebagai bit signifikan paling tinggi.

Tabel 2. 3 Tabel ASCII

Dec	Hexa	Char	Dec	Hexa	Char	Dec	Hexa	Char	Dec	Hexa	Char
0	0	Nul	65	41	A	130	82	,	195	C3	Ã
1	1	SOH	66	42	B	131	83	*ƒ*	196	C4	Ä
2	2	STX	67	43	C	132	84	„	197	C5	Å
3	3	ETX	68	44	D	133	85	…	198	C6	Æ
4	4	EOT	69	45	E	134	86	†	199	C7	Ç
5	5	ENQ	70	46	F	135	87	‡	200	C8	È
6	6	ACK	71	47	G	136	88	ˆ	201	C9	É
7	7	BEL	72	48	H	137	89	‰	202	CA	Ê
8	8	BS	73	49	I	138	8A	Š	203	CB	Ë
9	9	TAB	74	4A	J	139	8B	‹	204	CC	Ì
10	A	LF	75	4B	K	140	8C	Œ	205	CD	Í
11	B	VT	76	4C	L	141	8D	•	206	CE	Î
12	C	FF	77	4D	M	142	8E	Ž	207	CF	Ï
13	D	CR	78	4E	N	143	8F	•	208	D0	Đ
14	E	SO	79	4F	O	144	90	•	209	D1	Ñ
15	F	SI	80	50	P	145	91	‘	210	D2	Ò
16	10	DLE	81	51	Q	146	92	’	211	D3	Ó
17	11	DC1	82	52	R	147	93	“	212	D4	Ô
18	12	DC2	83	53	S	148	94	”	213	D5	Õ
19	13	DC3	84	54	T	149	95	•	214	D6	Ö
20	14	DC4	85	55	U	150	96	–	215	D7	×
21	15	NAK	86	56	V	151	97	—	216	D8	Ø
22	16	SYN	87	57	W	152	98	˜	217	D9	Ù
23	17	EBT	88	58	X	153	99	™	218	DA	Ú
24	18	CAN	89	59	Y	154	9A	š	219	DB	Û
25	19	EM	90	5A	Z	155	9B	›	220	DC	Ü
26	1A	SUB	91	5B	[	156	9C	œ	221	DD	Ý
27	1B	ESC	92	5C	\	157	9D	•	222	DE	Þ
28	1C	FS	93	5D	]	158	9E	ž	223	DF	ß
29	1D	GS	94	5E	^	159	9F	Ÿ	224	E0	à
30	1E	RS	95	5F	_	160	A0		225	E1	á
31	1F	US	96	60	`	161	A1	¡	226	E2	â
32	20	SPACE	97	61	a	162	A2	¢	227	E3	ã
33	21	!	98	62	b	163	A3	£	228	E4	ä
34	22	"	99	63	c	164	A4	¤	229	E5	å

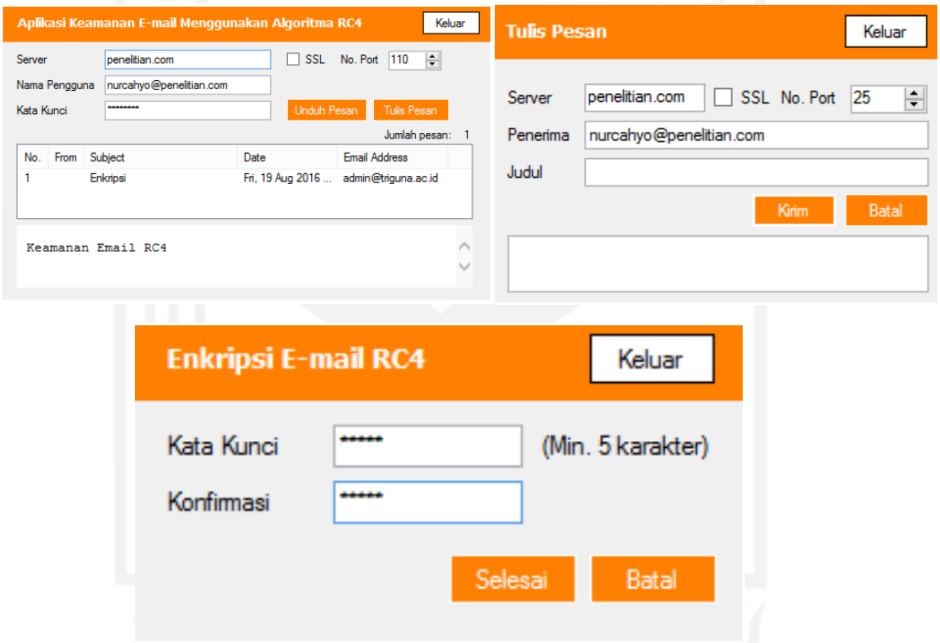
35	23	#	100	64	d	165	A5	¥	230	E6	æ
36	24	$	101	65	e	166	A6	¦	231	E7	ç
37	25	%	102	66	f	167	A7	§	232	E8	è
38	26	&	103	67	g	168	A8	¨	233	E9	É
39	27	'	104	68	h	169	A9	©	234	EA	Ê
40	28	(	105	69	i	170	AA	ª	235	EB	Ë
Dec	Hexa	Char	Dec	Hexa	Char	Dec	Hexa	Char	Dec	Hexa	Char
41	29	)	106	6A	j	171	AB	«	236	EC	Ì
42	2A	*	107	6B	k	172	AC	¬	237	ED	Í
43	2B	+	108	6C	l	173	AD		238	EE	Î
44	2C	,	109	6D	m	174	AE	®	239	EF	Ï
45	2D	-	110	6E	n	175	AF	¯	240	F0	Ð
46	2E	.	111	6F	o	176	B0	°	241	F1	Ñ
47	2F	/	112	70	p	177	B1	±	242	F2	Ò
48	30	0	113	71	q	178	B2	²	243	F3	Ó
49	31	1	114	72	r	179	B3	³	244	F4	Ô
50	32	2	115	73	s	180	B4	´	245	F5	Õ
51	33	3	116	74	t	181	B5	µ	246	F6	Ö
52	34	4	117	75	u	182	B6	¶	247	F7	÷
53	35	5	118	76	v	183	B7	·	248	F8	Ø
54	36	6	119	77	w	184	B8	¸	249	F9	Ù
55	37	7	120	78	x	185	B9	¹	250	FA	Ú
56	38	8	121	79	y	186	BA	º	251	FB	Û
57	39	9	122	7A	z	187	BB	»	252	FC	Ü
58	3A	:	123	7B	{	188	BC	¼	253	FD	Ý
59	3B	;	124	7C	|	189	BD	½	254	FE	Þ
60	3C	<	125	7D	}	190	BE	¾	255	FF	Ÿ
61	3D	=	126	7E	~	191	BF	¿			
62	3E	>	127	7F	•	192	C0	À			
63	3F	?	128	80	€	193	C1	Á			
64	40	@	129	81	•	194	C2	Â			

2.6. Penelitian Relevan

Berikut adalah penelitian yang terkait dengan algoritma vigenere cipher dan RC4:

a. Aplikasi Keamanan E-mail Menggunakan Algoritma RC4

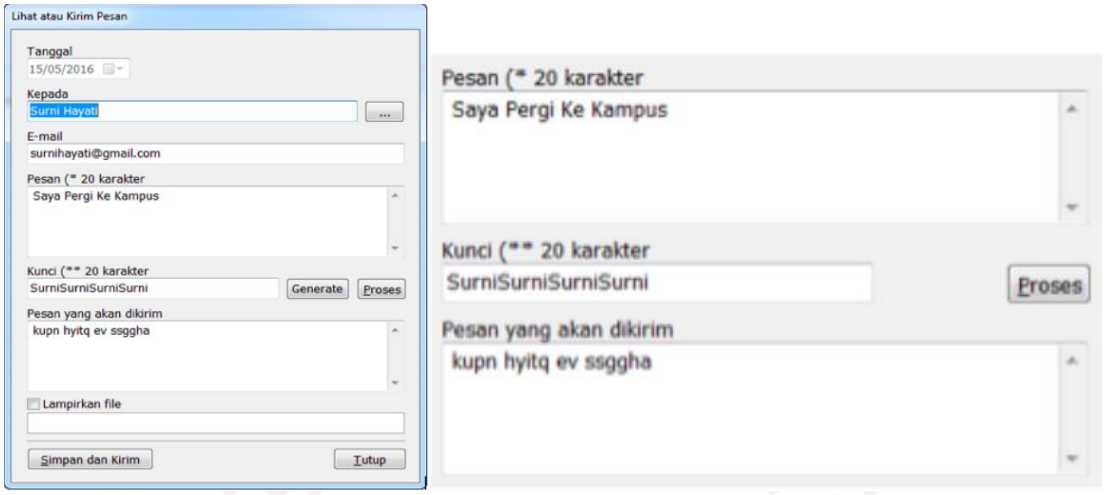
Pada tahun 2016 penelitian oleh Nurcahyo Budi Nugroho, Zulfian Azmi, dan Saiful Nur Arif dari Program Studi Sistem Komputer, STMIK Triguna Dharma pada jurnal penelitian berjudul Aplikasi Keamanan Email Menggunakan Algoritma Rc4. Penelitian ini membahas proses enkripsi pesan email yang dikirim menggunakan algoritma RC4. Sebuah aplikasi keamanan email di buat menggunakan visual studio code dengan mercury main server untuk implementasi algoritma RC4. Hasil pengujian menunjukkan bahwa algoritma RC4 berhasil di implementasikan untuk mengenkripsi dan mendekripsikan semua karakter pada *body* email, aplikasi yang dibuat sudah mampu memenuhi kebutuhan aplikasi email client yang menerapkan algoritma kriptografi RC4.



Gambar 2. 4 Aplikasi Keamanan E-mail Menggunakan Algoritma RC4

b. Aplikasi Pengamanan Email Menggunakan Vigenere Cipher

Pada tahun 2017 penelitian oleh Dahlan Abdullah dan Surnihayati dari Program Studi Teknik Informatika, Fakultas Teknik, Universitas Malikussaleh, Aceh, pada jurnal penelitian berjudul Pengamanan Email Menggunakan Vigenere Cipher. Penelitian ini membahas proses enkripsi dan dekripsi dan dekripsi email menggunakan metode kriptografi vigenere cipher, menyimpulkan bahwa Penggunaan algoritma Vigenere Cipher merupakan salah satu cara untuk mengatasi kelemahan metode kriptografi klasik abjad majemuk. Kelebihan dari metode vigenere cipher adalah tidak begitu rentan terhadap metode pemecahan cipher yang disebut analisis frekuensi.



Gambar 2. 5 Aplikasi Pengamanan Email Menggunakan Vigenere Cipher

c. Algoritma RC4 Sebagai Metode Enkripsi

Pada tahun 2009 penelitian oleh Karina Novita dari Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika ITB yang berjudul Algoritma Rc4 Sebagai Metode Enkripsi. Pada penelitian ini, peneliti menjelaskan mekanisme dasar kerja algoritma RC4 secara rinci dan jelas, Algoritma RC4 ini digunakan untuk meningkatkan keamanan data, melindungi data atau pesan agar tidak terbaca oleh pihak lain. Algoritma RC4 merupakan metode enkripsi tercepat dibandingan DES, Triple-DES, Blowfish-256, AES-128, AES-258. Namun, algoritma RC4 juga memiliki kelemahan antara lain, tingginya peluang untuk menghasilkan *array* S yang sama ataupun berulang dan *Bit-Flipping Attack.*

BAB III
METODOLOGI PENELITIAN

Pada bab ini dilakukan pembahasan proses pengerjaan tugas akhir yang meliputi studi literatur, analisis masalah, strategi pemecahan masalah, analisis kebutuhan serta perancangan sistem. Perancangan sistem dilakukan dengan membuat flowchart dan *interface* sistem dengan menggunakan pemodelan UML (Unified Modeling Language).

3.1 Studi Literatur

Studi literatur dilakukan dengan cara mengumpulkan sumber referensi yang berkaitan dengan sistem email dan algoritma kriptografi vigenere cipher dan RC4. Dapat berupa informasi dari artikel, jurnal, buku, dan berbagai sumber informasi lainnya. Setelah itu hasil yang dipeeroleh dari studi literatur ini adalah referensi yang terdapat pada BAB II LANDASAN TEORI laporan ini.

3.2 Analisis

3. 2. 1. Analisis Masalah

Sejak terbentuknya internet, email sudah digunakan orang dan merupakan salah satu fasilitas yang ada pada saat itu. Selain itu orang menggunakan email untuk menyimpan berbagai dat. Hal ini dikarenakan orang-orang takut lupa mengenai informasi penting tersebut dan pada akhirnya email dipilih sebagai tempat penyimpanannya. Namun hal tersebut berakibat pada berkembangnya kecurangan yang dilakukan orang untuk mencari tahu mengenai informasi tersebut.

Salah satu cara yang dilakukan yaitu dengan melakukan hack ke email sang korban. Permasalahan tersebut dapat diatasi dengan proses enkripsi. Salah satu algoritma enkripsi yang cukup dikenal adalah dengan algoritma enkripsi vigenere cipher dan RC4. Untuk itulah dibutuhkan sebuah aplikasi yang dapat mengirimkan email dengan kemampuan untuk melakukan enkripsi dan dekripsi pesan teks pada email.

3. 2. 2. Analisis Kebutuhan

Tahapan analisis kebutuhan dibagi menjadi dua bagian, yaitu kebutuhan fungsional dan kebutuhan non-fungsional. Kebutuhan fungsional mendeskripsikan aktifitas yang dapat

dilakukan oleh suatu sistem. Sedangkan kebutuhan non-fungsional mendeskripsikan fitur, karakteristik, dan batasan lainnya pada sistem.

a. Kebutuhan Fungsional

Kebutuhan fungsional dari sistem ini adalah untuk dapat melakukan pengamanan pesan email menggunakan Algoritma vigenere cipher dan RC4, kebutuhan fungsional yang harus dipenuhi antara lain sebagai berikut:

1. Menerima input *plaintext*

   Sistem dapat menerima input *plaintext* yang akan dikirim, mencari dan membaca *plaintext* yang telah tersedia di dalam lokasi penyimpanan perangkat dengan format *file* yang telah ditentukan, yaitu doc, docx, dan txt.

2. Menyimpan *plaintext*

   Sistem menyimpan *plaintext* yang telah didekripsi ke dalam lokasi penyimpanan perangkat yang dipilih.

3. Menerima dan menyimpan input kunci

   Pada proses enkripsi dan dekripsi, sistem menerima dan menyimpan input kunci yang sudah diinputkan secara manual ke dalam program, sistem mencari dan membaca kunci yang telah tersimpan di dalam lokasi penyimpanan perangkat.

4. Mengirim dan mengenkripsi pesan

Sistem melakukan enkripsi pesan dengan menggunakan kunci algoritma vigenere cipher lalu dilanjutkan dengan kunci algoritma RC4, kemudian menghasilkan satu pesan rahasia atau *ciphertext* yang akan dikirim melalui email.

1. Menyimpan hasil enkripsi pesan

   Sistem menyimpan pesan hasil dari enkripsi yang dilakukan.

2. Menerima dan mendekirpsi pesan

   Sistem dapat menerima pesan email yang terenkripsi, lalu mendekripsi pesan yang akan dibaca dengan menggunakan kunci algoriotma RC4 dan kunci algoritma vigenere cipher yang sama. *Ciphertext* yang pada pesan email akan didekirpsi menjadi *plaintext*.

b. Kebutuhan Non-Fungsional

Kebutuhan non-fungsional yang ada pada sistem meliputi karakteristik sebagai berikut:

1. Performa

Perangkat lunak yang akan dibangun dapat menunjukkan hasil dari fungsi kriptografi yang dilakukan oleh sistem.

2. Hemat biaya

   Perangkat lunak yang akan dibangun menggunakan teknologi open source dan bebas digunakan.

3. Dokumentasi

   Perangkat lunak yang dibangun akan memberikan panduan atau pengarahan penggunaan.

4. Mudah dipelajari dan digunakan

   Perangkat lunak yang akan dibangun memiliki tampilan yang *user* friendly dan mudah digunakan karena memiliki halaman petunjuk.

5. Kontrol

   Perangkat lunak yang dibangun akan menampilkan pesan kesalahan atau error *message* jika setiap input yang dimasukkan kosong atau tidak sesuai

3.3 Strategi Pemecahan Masalah

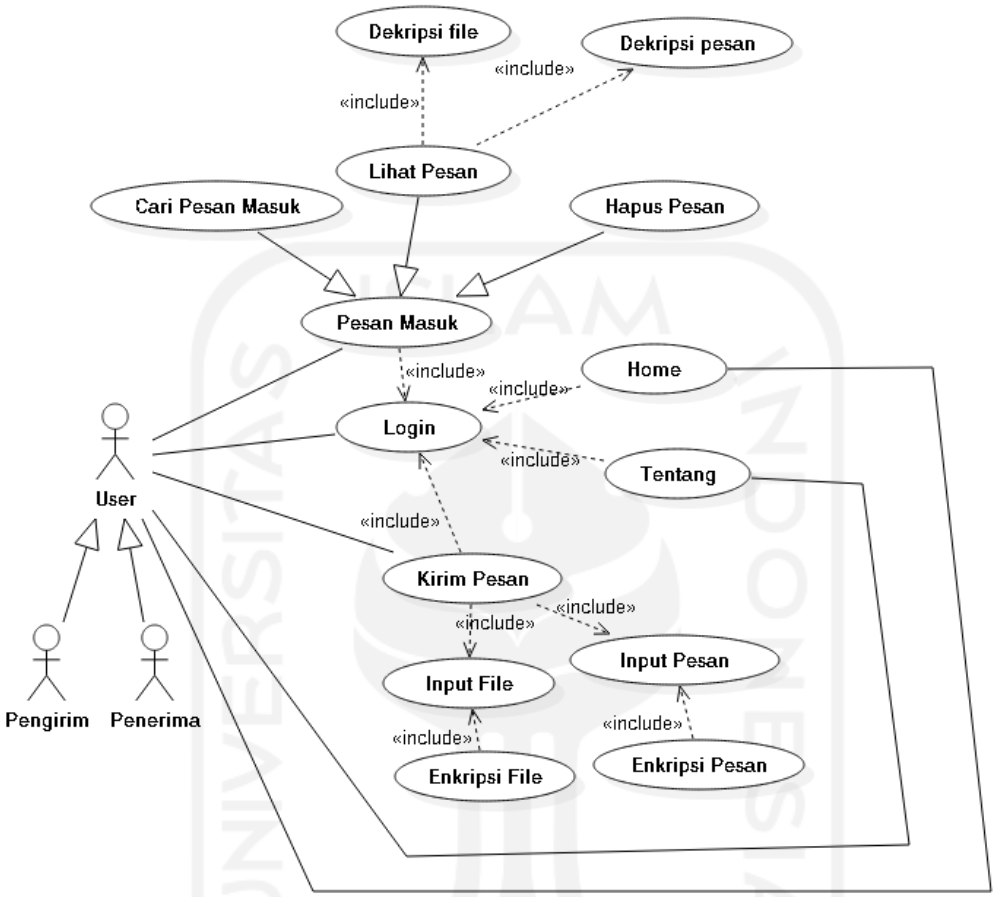
Strategi dalam melakukan pemecahan masalah yang dianalisa oleh penulis mengenai aplikasi keamanan data email menggunakan algoritma vigenere dan algoritma rc4 adalah sebagai berikut:

a. Membuat aplikasi keamanan data pesan dengan menggunakan algoritma vigenere dan algoritma rc4.
b. Meningkatkan keamanan dan privasi data pesan saat pengguna melakukan komunikasi.
c. Mengetahui tingkat keamanan enkripsi data menggunakan algoritma vigenere dan algoritma rc4.
d. Mengetahui seberapa besar efektivitas enkripsi pada algoritma vigenere dan algoritma rc4.

3.4. Perancangan Sistem

Perancangan sistem merupakan gambaran dari keseluruhan rincian bagaimana sistem akan berjalan. Tujuan dari hal tersebut adalah untuk menghasilkan sebuah sistem yang sesuai dengan kebutuhan *user*. Dalam penelitian yang dilakukan ini, perancangan sistem terdiri dari perancangan use case diagram, perancangan activity diagram, flowchart, dan perancangan *interface* atau antarmuka sistem.

3.4.1 Perancangan Use Case Diagram



Gambar 3. 1 Use Case Diagram Sistem

Use Case Diagram menggambarkan kebutuhan sistem secara fungsional dengan mengidentifikasi aktor-aktor yang terlibat dan berinteraksi dengan fungsi dasar pada sistem. *Use Case Diagram* akan menjelaskan fungsi apa saja yang dikerjakan oleh sistem. Aktor yang yang terlibat pada penelitian ini adalah pengirim dan penerima namun didefinisikan dengan satu aktor karena aktor tersebut dapat melakukan pengiriman dan penerimaan pesan sekaligus.

Definisi Aktor

Deskripsi pendefinisian aktor pada penggunaan aplikasi keamanan email sebagai berikut:

Tabel 3. 1 Definisi Aktor Use Case

No.	Aktor	Deskripsi
1	*User*	*User* adalah orang yang menggunakan aplikasi keamanan email untuk mengirim dan menerima pesan email.

Definisi *Use Case*

Berikut adalah deskripsi pendefinisian *use case* pada penggunaan aplikasi keamanan email sebagai berikut:

Tabel 3. 2 Definisi Use Case

No.	Use Case	Deskripsi
1	*Login*	Proses awal yang dilakukan untuk masuk ke halaman *Home*
2	*Home*	Merupakan menu halaman awal saat *user* berhasil masuk ke dalam aplikasi
4	Kirim Pesan	Merupakan salah satu menu yang dapat di akses untuk mengirim pesan dan *file* email yang terenkripsi.
5	Pesan Masuk	Merupakan salah satu menu yang dapat di akses untuk melihat *list* pesan masuk dan membaca pesan tersebut.
6	Tentang	Merupakan salah satu menu yang digunakan untuk melihat informasi tentang aplikasi

Skenario *Use Case*

Berikut adalah deskripsi pendefinisian skenario *use case* pada penggunaan aplikasi keamanan email sebagai berikut:

a. Skenario use case Login

Tabel 3. 3 Skenario Use Case *Login*

Nama Use Case	*Login*
Deskripsi	Digunakan untuk masuk ke halaman *Home*
Aktor	*User*
Kondisi Awal	Email dan *Password* di isi
Skenario	
Aktor	Sistem
1. Menginputkan email dan *password*	
	2. Validasi email dan *password* jika benar maka *login* berhasil dan sistem menampilkan halaman *Home*
Kondisi Akhir	*User* berhasil *login* dan masuk ke halaman *Home*

b. Skenario *Use Case* Kirim Pesan

Tabel 3. 4 Skenario Use Case Kirim Pesan

Nama Use Case	Kirim Pesan
Deskripsi	Digunakan untuk mengirim pesan email yang terenkripsi.
Aktor	*User*
Kondisi Awal	Menampilkan form halaman menu Kirim Pesan

Skenario	
Aktor	Sistem
1. memilih menu kirim pesan, dan menginput *to*, *Subject*, isi pesan, dan *attachment*.	
	2. Menampilkan form halaman enkripsi email.
3. klik tombol Enkrip dan Kirim Email	
	4. Memulai enkripsi isi pesan dengan menggunakan Vigenere Ciher dan Rivest Code 4.
	5. Melanjutkan enkripsi *file* dengan Rivest Code 4.
Kondisi Akhir	Sistem mengirim pesan email terenkripsi dan menyimpan data ke dalam database

c. Skenario *Use Case* Pesan Masuk

Tabel 3. 5 Skenario Use Case Pesan Masuk

Nama Use Case	Kotak Masuk
Deskripsi	Digunakan melihat dan membaca pesan email yang masuk.
Aktor	*User*
Kondisi Awal	Menampilkan form halaman menu dekripsi email

Skenario	
Aktor	Sistem
1. memilih menu Kotak Masuk	
	2. Menampilkan halaman Kotak Masuk dan daftar pesan yang diterima.
3. klik Lihat Pesan	
	4. Memulai dekripsi isi pesan menggunakan Rivest Code 4 dan Vigenere Cipher.
	5. Melanjutkan dekripsi *file* menggunakan Rivest Code 4.
Kondisi Akhir	Sistem menampilkan detail isi pesan dan *attachment* yang sudah terdekripsi dari pesan yang dipilih.

d. Skenario *Use Case* Tentang

Tabel 3. 6 Skenario *Use Case* Tentang

Nama Use Case	Tentang
Deskripsi	Digunakan untuk menampilkan informasi tentang aplikasi
Aktor	Pengirim / Penerima
Kondisi Awal	Masuk ke tampilan tentang
Skenario	
Aktor	Sistem

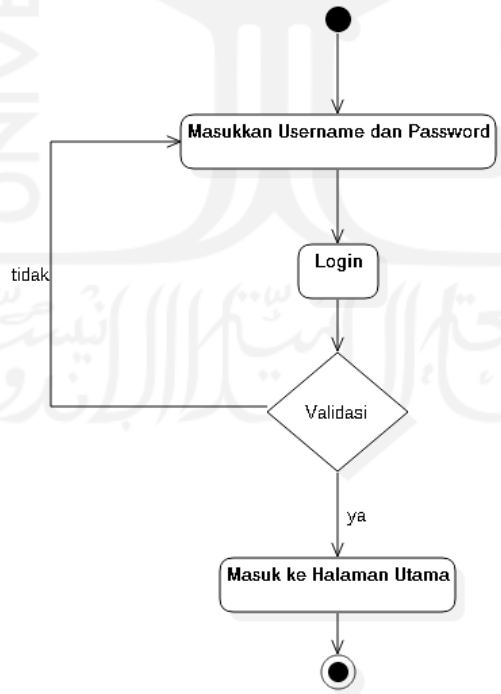
1. Memilih menu Tentang	2. Menampilkan halaman tentang
Kondisi Akhir	Menampilkan informasi tentang aplikasi.

3.4.2 Perancangan Activity Diagram

Activity Diagram merupakan bentuk khusus dari *state machine* yang bertujuan memodelkan komputasi-komputasi dan aliran-aliran kerja yang terjadi dalam sistem/perangkat lunak yang sedang dikembangkan. *Activity Diagram* menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi dan bagaimana keseluruhan aktivitas berakhir. Adapun activity diagram tersebut adalah sebagai berikut:

Activity Diagram *Login*

Activity diagram *Login* merupakan gambaran dari aktivitas aktor melakukan *login* untuk dapat masuk ke dalam halaman utama sistem.

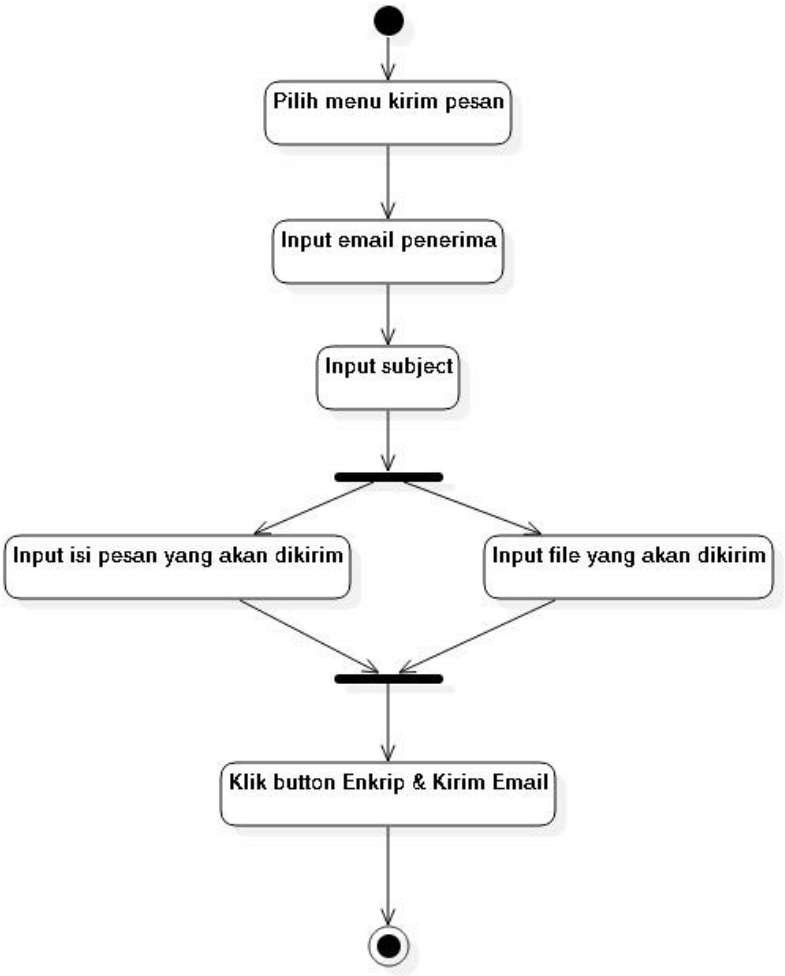


Gambar 3. 2 Activity Diagram *Login* Aplikasi

Pada gambar activity diagram *login* menunjukkan bahwa aktivitas pertama, *user* harus masukkan *username* dan *password* dengan benar. Jika *username* dan *password* yang dimasukkan salah, maka *user* akan kembali mengisi hingga *username* dan *password* benar dan tepat penulisannya.

***Activity* Diagram Kirim Pesan**

Activity diagram Kirim pesan adalah gambaran dari saat *user* akan mengirimkan pesan email yang terenkripsi.



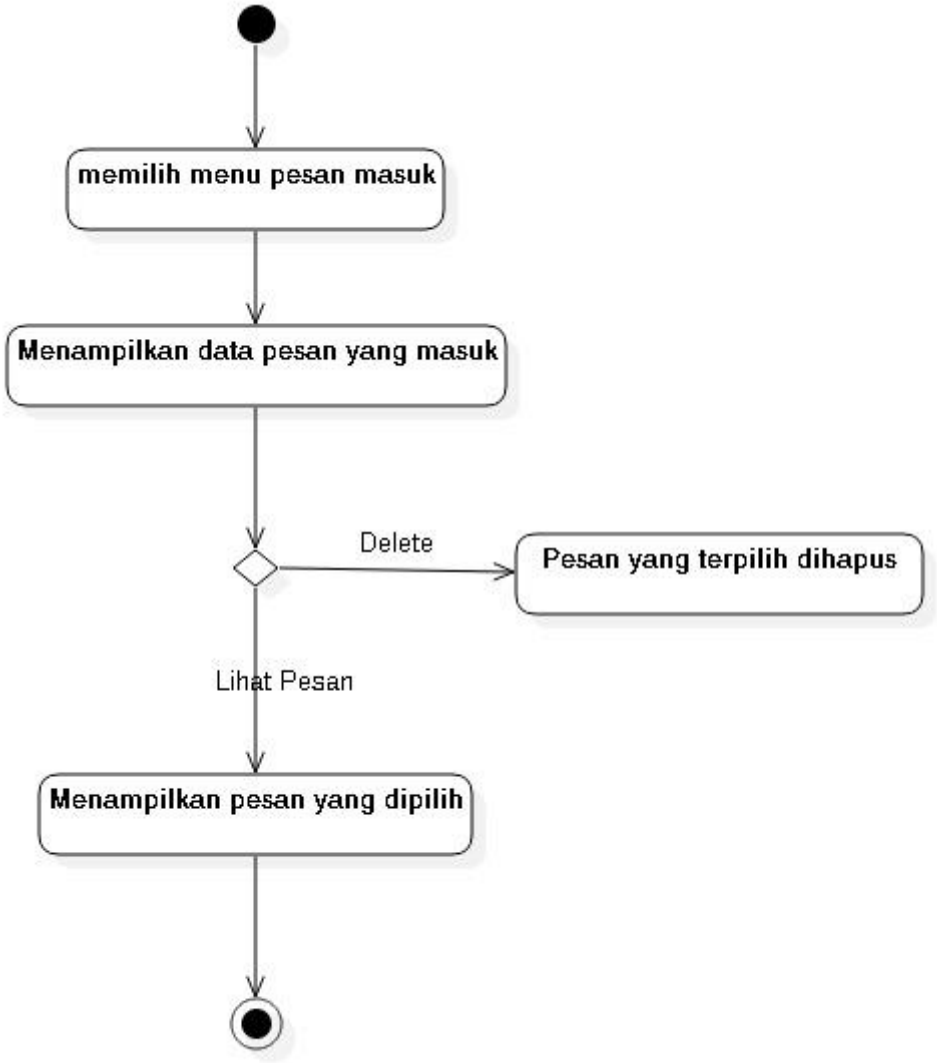
Gambar 3. 3 Activity Diagram Kirim Pesan

Pada gambar activity diagram kirim pesan, *user* perlu memilih menu kirim pesan jika ingin mengirimkan pesan email terenkripsi. Sebelum mengirim pesan terenkripsi, *user* perlu menginput email penerima, menginputkan judul/*Subject* pesan, lalu menginputkan pesan

maupun *file* yang akan dikirim. Setelah nya *user* meng-klik *button* enkrip & kirim email, untuk memulai proses pengenkripsian sekaligus pengiriman pesan/*file* yang sudah diinputkan. Hasil dari proses ini akan akan mengubah pesan/*file* yang dikirim menjadi *ciphertext*.

***Activity* Diagram Pesan Masuk**

Activity diagram Kirim pesan adalah gambaran dari saat *user* akan melihat dan membaca pesan email yang yang sudah didekripsi.



Gambar 3. 4 Activity Diagram Pesan Masuk

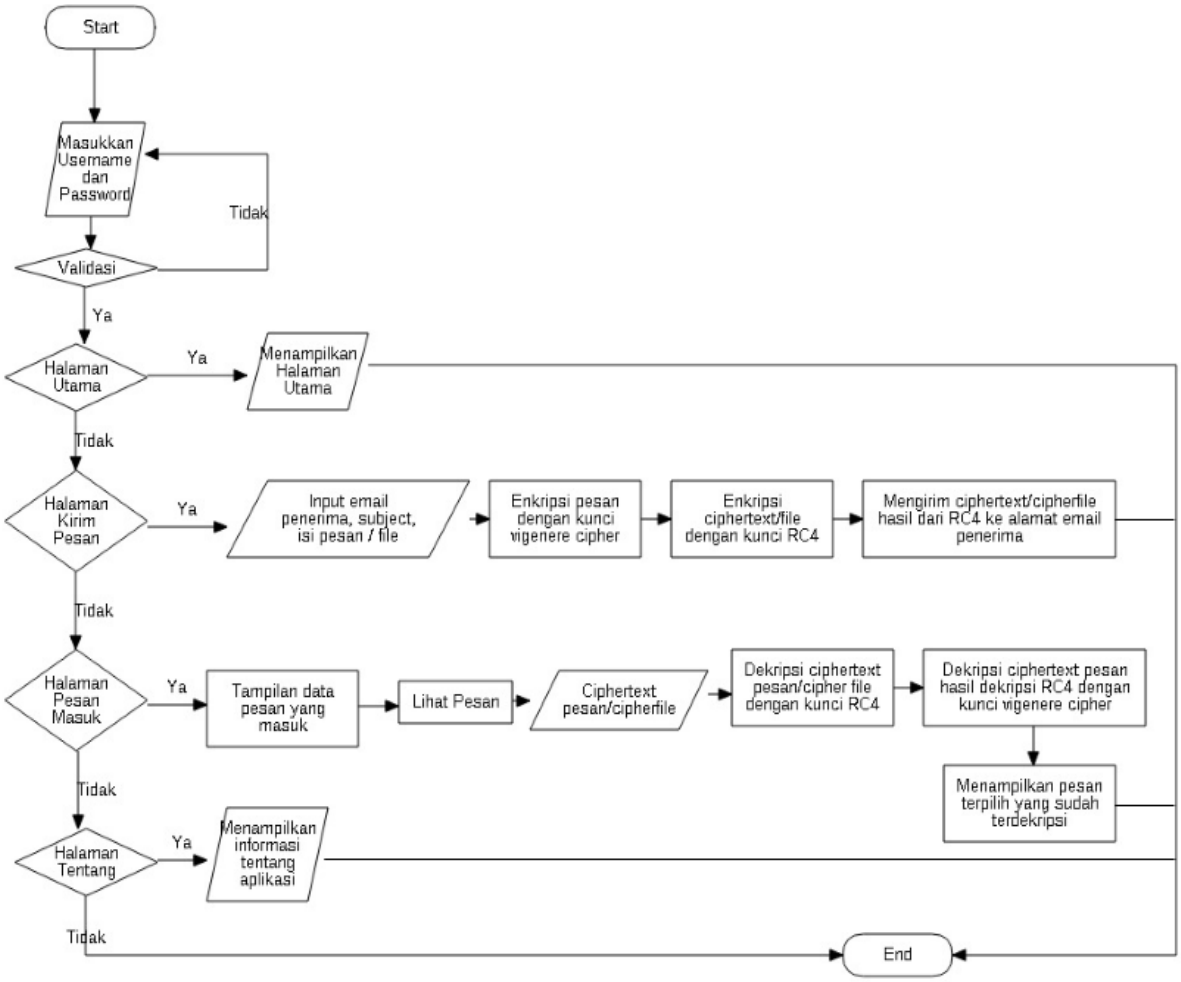
Pada gambar activity diagram pesan masuk, *user* perlu memilih menu pesan masuk jika ingin melihat dan membaca pesan email yang masuk. Klik *button* Lihat Pesan, untuk membaca pesan masuk yang terpilih, saat *button* tersebut di klik, maka akan memulai proses

mendekripsikan pesan/*file* yang akan dibaca. Setelah proses dekripsi selesai, maka akan menampilkan isi pesan masuk yang dipilih.

3.4.3 Flowchart

Flowchart adalah penggambaran secara grafik dari langkah-langkah dan urutan-urutan prosedur dari suatu program.

Flowchart Sistem



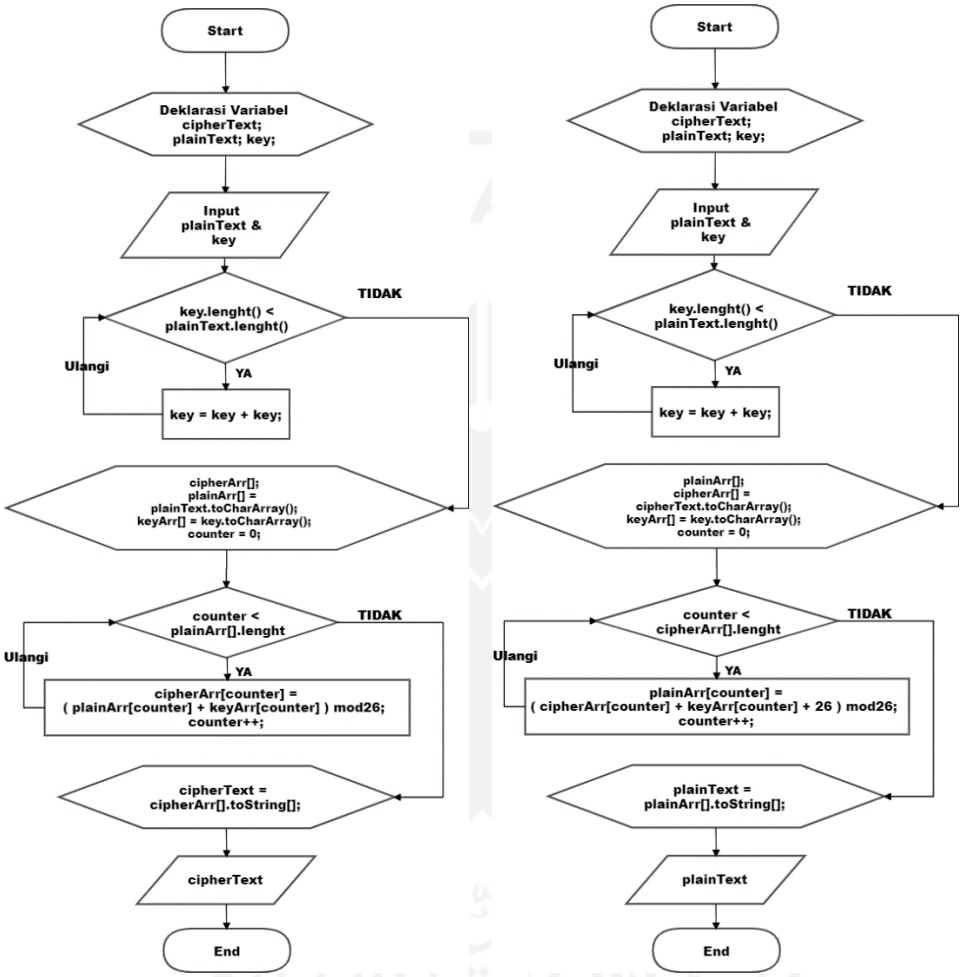
Gambar 3. 5 Flowchart Sistem

Flowchart Sistem berfungsi sebagai tempat menggambarkan alur kerja pada sistem secara sistematis. Flowchart sistem yang ada pada gambar 3.5 diatas memiliki empat halaman utama yang dapat dipilih oleh pengguna, yaitu halaman awal atau utama yang akan pertama kali diliat setelah *user* berhasil *login* ke dalam sistem. Halaman kirim pesan berisi halaman

yang digunakan untuk melakukan pengiriman pesan/*file* email. Halaman pesan masuk berisi halaman yang menampilkan beberapa data pesan email yang masuk. Halaman tentang berisi halaman tentang informasi aplikasi.

a. Flowchart Algoritma

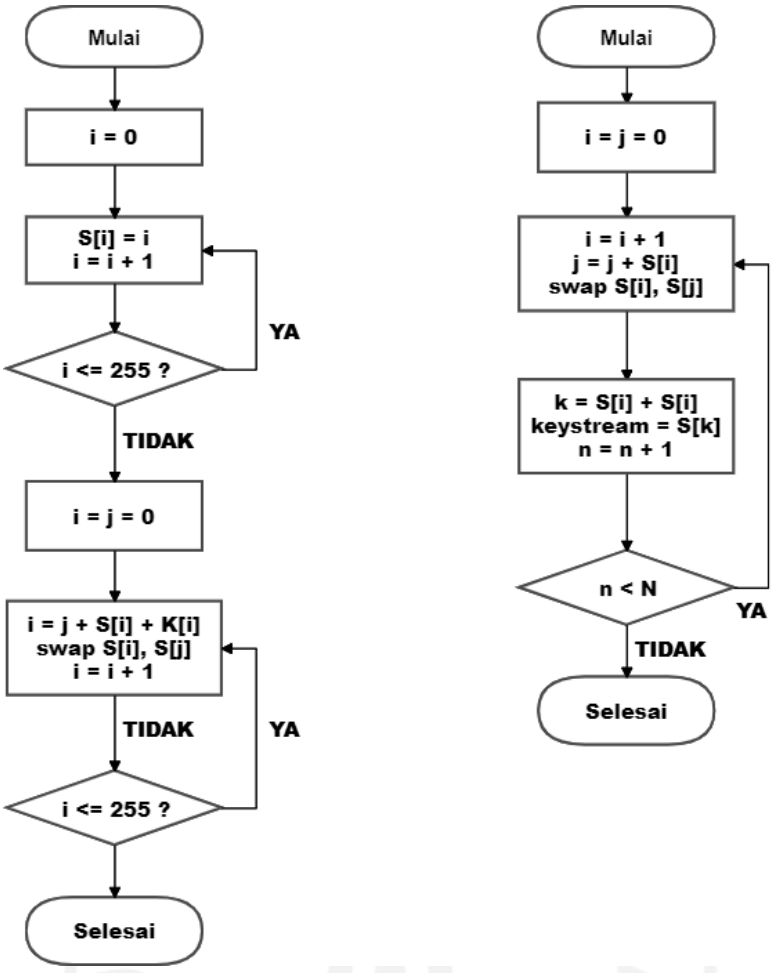
1. Algoritma Vigenere Cipher



(a) Enkripsi (b) Dekripsi

Gambar 3. 6 Flowchart Algoritma Vigenere Cipher

2. Flowchart Algoritma RC4



(a) KSA (b) PRGA

Gambar 3. 7 Flowchart Algoritma RC4

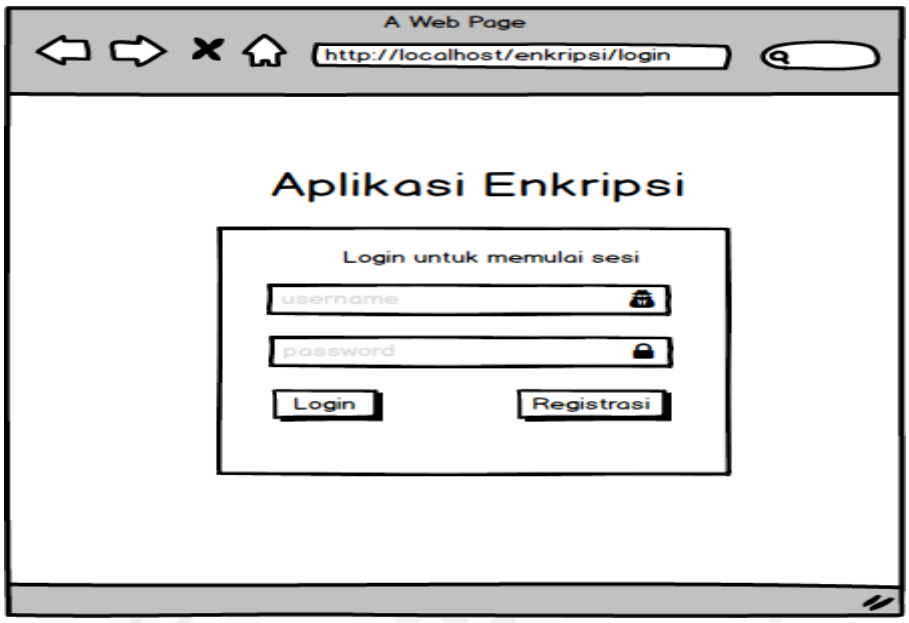
Gambar 3.7 diatas merupakan bagaimana algoritma RC4 bekerja. Algoritma RC4 memiliki dua proses utama dalam enkripsi dan dekripsi pesan, yaitu proses *Key* Schedulling Algorithm (KSA) yang ditunjukkan oleh flowchart pada sisi sebelah kiri dan proses Pseudo-Random Generation Algorithm (PRGA) yang ditunjukkan pada sisi sebelah kanan flowchart. Dalam algoritma spritz, proses enkripsi dan dekripsi pesan menggunakan flowchart yang sama.

3.4.4 Perancangan Antarmuka

Perancangan tampilan berfungsi untuk memastikan suatu sistem aplikasi mudah untuk digunakan dan dapat dimengerti oleh *user*. Rancangan dibutuhkan sebelum implementasi diterapkan dalam bentuk program.

a. Halaman *Login*

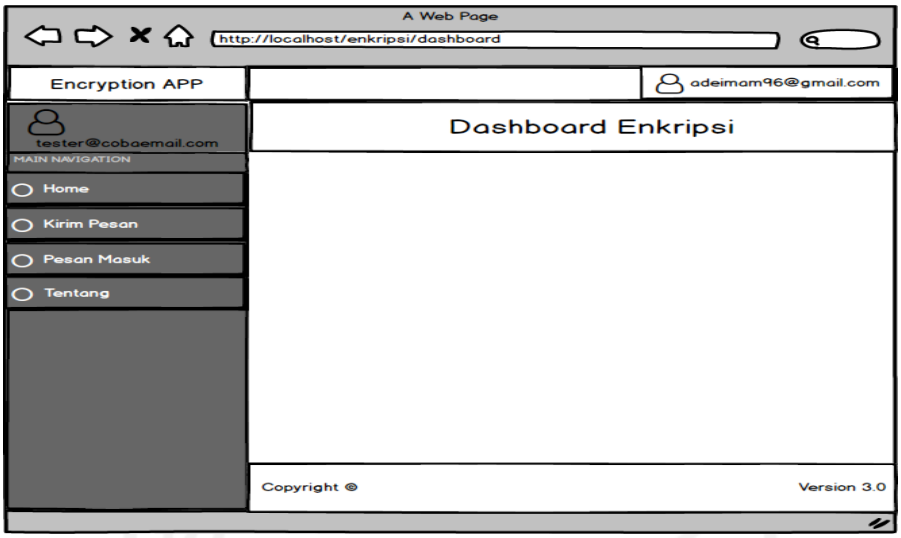
Halaman *login* merupakan halaman pertama muncul ketika *user* membuka aplikasi *encryption* app, seperti dibawah ini:



Gambar 3. 8 Rancangan Halaman *Login*

b. Halaman Utama

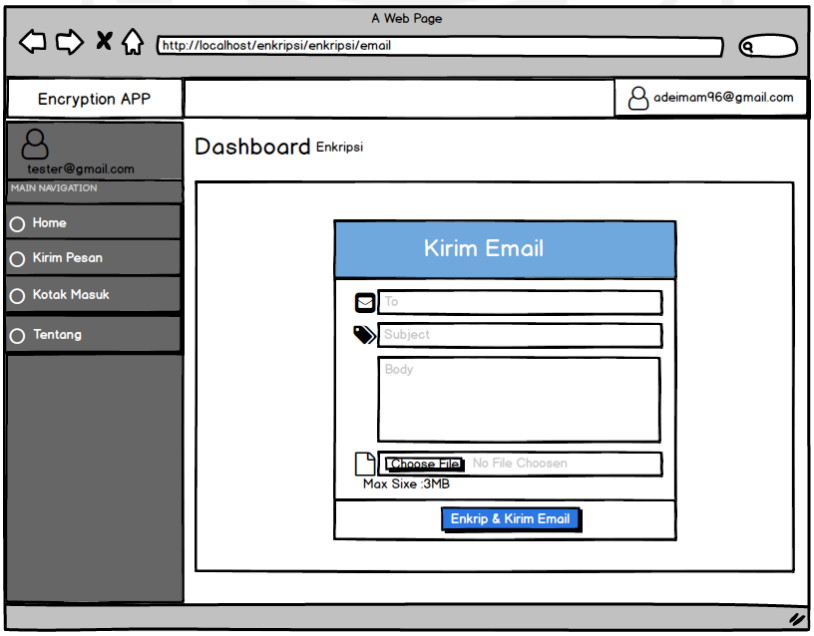
Halaman Dashboard merupakan halaman pertama kali muncul ketika *user* berhasil *login* masuk ke dalam aplikasi *encryption* app, seperti dibawah ini:



Gambar 3. 9 Rancangan Halaman Utama

c. Halaman Kirim Pesan

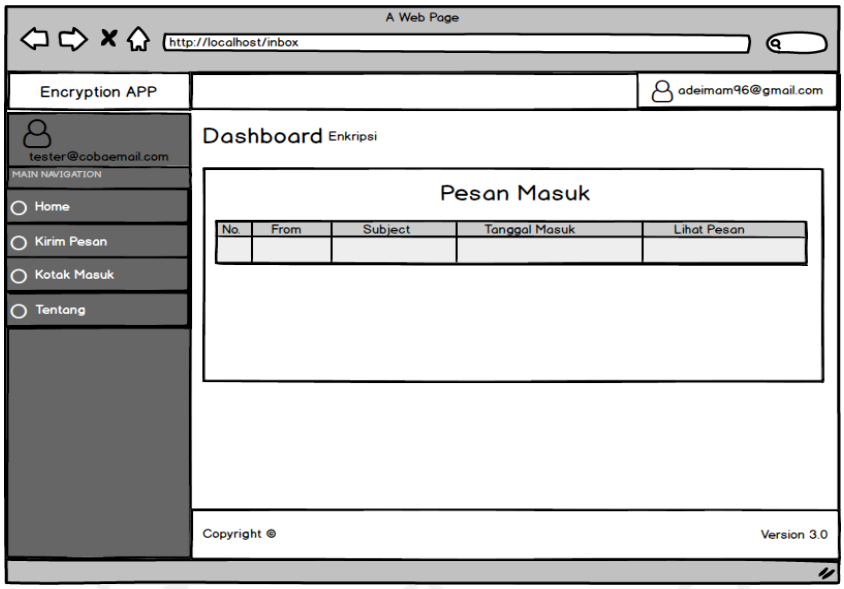
Pada halaman kirim pesan *user* dapat mengisi *field* yang ada saat ingin mengirim pesan email, seperti dibawah ini:



Gambar 3. 10 Rancangan Halaman Kirim Pesan

d. Halaman Kotak Masuk

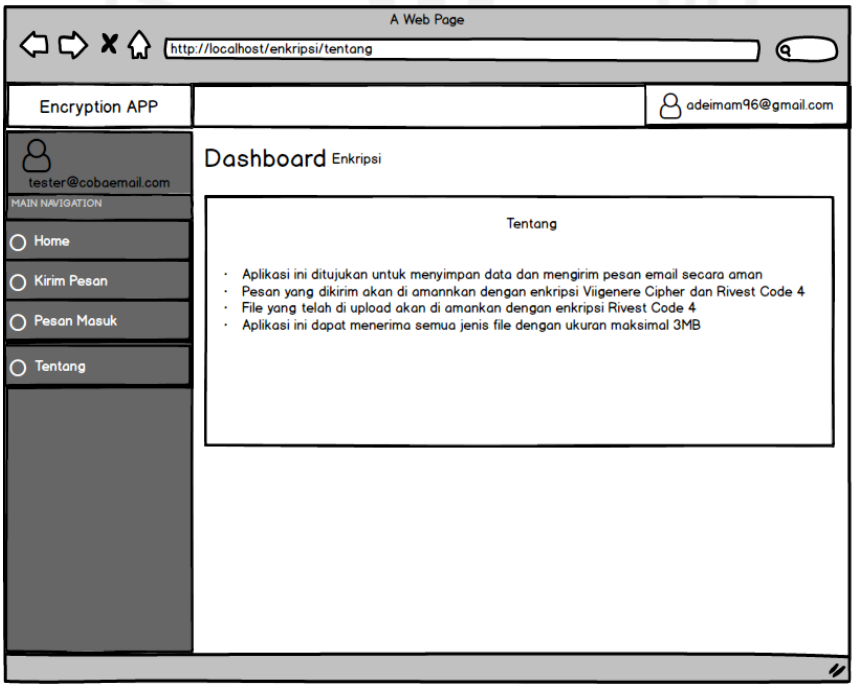
Pada halaman kotak masuk *user* dapat melihat daftar pesan yang diterima dan membaca pesan yang dipilih, seperti dibawah ini:



Gambar 3. 11 Rancangan Halaman Kotak Masuk

e. Halaman Tentang

Pada halaman Tentang berisi tentang informasi aplikasi *encryption* app dan metode kriptografi yang digunakan, berikut rancangan halaman tentang:



Gambar 3. 12 Rancangan Halaman Tentang

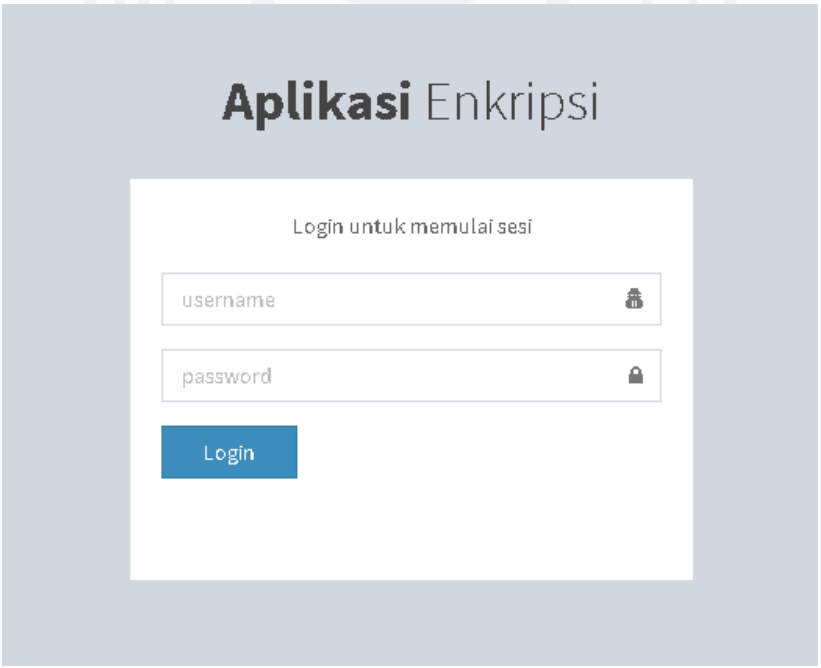
BAB IV
IMPLEMENTASI DAN PENGUJIAN

4.1 Implementasi

Dalam tugas akhir ini, sistem dibangun dengan menggunakan text editor Sublime Text berbahasa pemrograman PHP dan berbasis web. Sistem terdiri lima halaman utama, yaitu halaman *login*, halaman *home*, halaman kirim pesan, halaman pesan masuk, dan halaman tentang.

4.1.1 Halaman *Login*

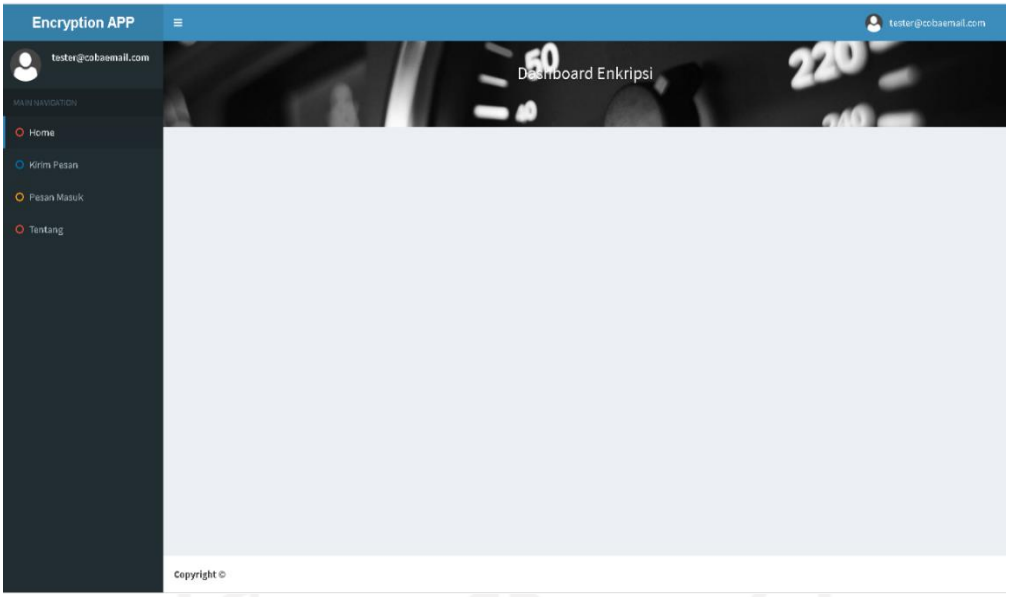
Halaman *Login* merupakan tampilan awal yang muncul saat aplikasi dijalankan. Pada Halaman ini terdapat *username* dan *password* yang perlu diisi untuk dapat masuk ke dalam halaman utama aplikasi.



Gambar 4. 1 Halaman *Login*

4.1.2 Halaman Utama

Halaman Utama merupakan tampilan yang pertama kali muncul saat *user* berhasil masuk kedalam aplikasi, seperti dibawah ini:



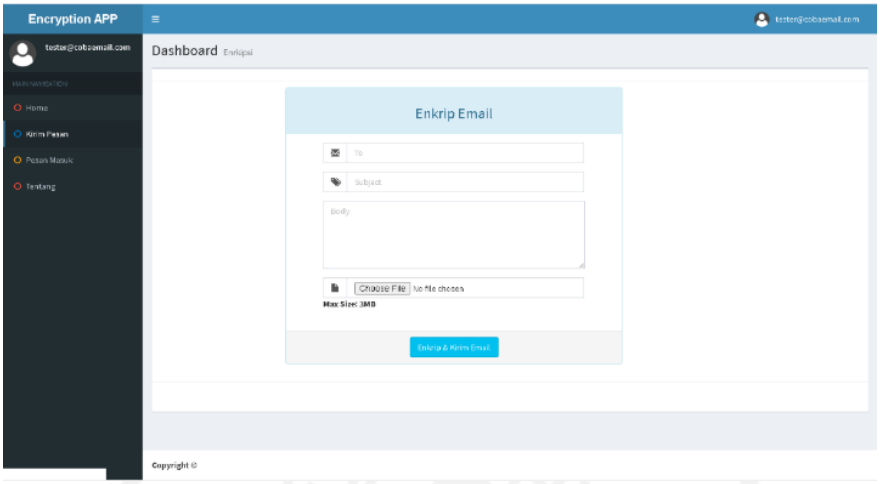
Gambar 4. 2 Halaman Utama

4.1.3 Halaman Kirim Pesan

Halaman Kirim Pesan ialah halaman yang digunakan *user* untuk mengirimkan pesan email terenkripsi. Sebelum mengirimkan pesan terenkripsi, *user* perlu mengisi *field-field* yang pada halaman kirim pesan.

Field To ditujukan kepada email si penerima, *field Subject* berisi judul/*Subject* pesan yang dikirim, *field Body* berisi pesan yang akan *user* kirimkan, dan *user* dapat mengirimkan *file* dengan cara mengklik *button Choose file*. *File* yang dikirim hanya *file* dokumen dengan format doc, docx, xls, xlsx, pdf, ppt, dan notepad.

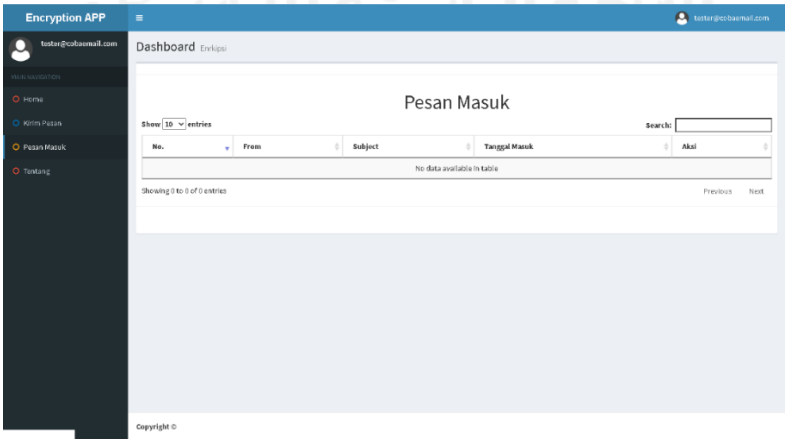
Jika semua *field* sudah terisi klik *button* Enkrip & Kirim Email, maka pesan dan *file* akan di enkripsi dengan algoritma Vigenere Cipher lalu dilanjutkan dengna algoritma RC4, dan dikirimkan kepada email penerima yang sebelumnya sudah diisi pada *field* To. Berikut tampilan halaman kirim pesan:



Gambar 4. 3 Halaman Kirim Pesan

4.1.4 Halaman Pesan Masuk

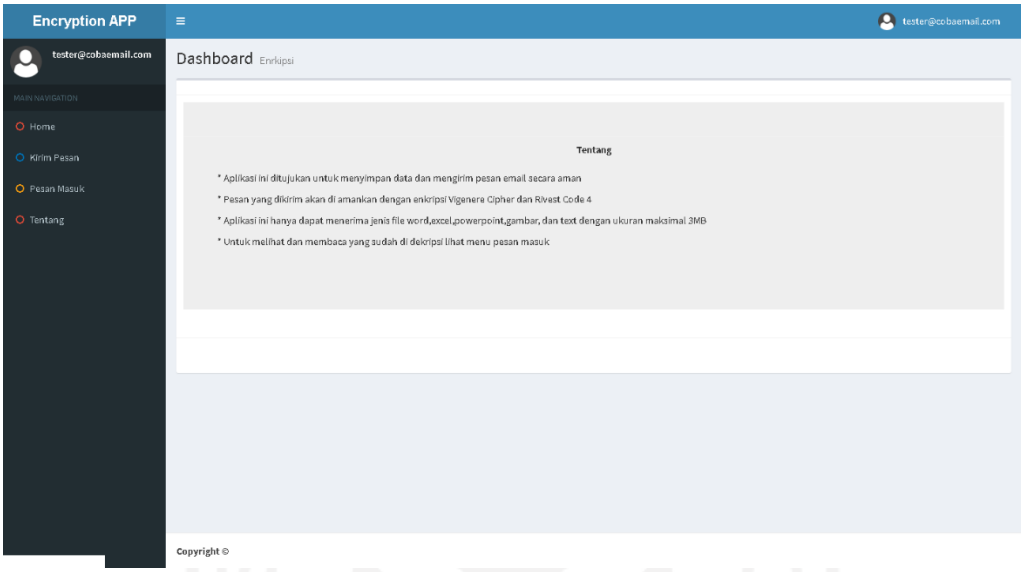
Halaman Pesan Masuk ialah halaman yang digunakan *user* untuk melihat pesan dan mendekripsikan pesan terenkripsi sebelumnya yang juga dikirimkan menggunakan aplikasi ini. Pada saat *user* memilih menu Pesan Masuk, maka akan muncul beberapa *list* daftar pesan masuk jika memang ada pesan email yang diterima. Untuk melihat pesan masuk tersebut dan mendekripsikannya *user* hanya perlu mengklik *button* lihat pesan, pada pesan yang akan dibaca, masa secara otomatis akan melakukan proses dekripsi yang dimulai dengan algoritma RC4 dan dilanjutkan Vigenere Cipher, setelah proses dekripsi selesai maka pesan email terdekripsi yang akan dibaca muncul. Berikut tampilan halaman pesan masuk:



Gambar 4. 4 Halaman Pesan Masuk

4.1.5 Halaman Tentang

Halaman Tentang berisi informasi tentang aplikasi enkripsi email yang dibuat. Berikut tampilan halaman tentang:



Gambar 4. 5 Halaman Tentang

4.1.6 Implementasi Enkripsi Vigenere Cipher

Berikut ini merupakan *libraries* Vigenere.php yaitu file php yang didalamnya berisi *code* untuk mengenkripsi dan dekripsi teks dimana proses enkripsi dan dekripsi menggunakan algorimta kriptografi vigenere. Berikut merupakan potongan *code* yang dipakai untuk proses enkripsi dan dekripsi:

```
class vigenere {
function encrypt($key, $text)
{
$pswd = strtolower($key);
$code = "";
$ki = 0;
$kl = strlen($key);
$length = strlen($text);
for ($i = 0; $i < $length; $i++)
{
if (ctype_alpha($text[$i]))
{
if (ctype_upper($text[$i]))
{
$text[$i] = chr(((ord($key[$ki]) - ord("a") + ord($text[$i]) - ord("A")) % 26) +
ord("A"));
}
```

```
else
{
$text[$i] = chr(((ord($key[$ki]) - ord("a") + ord($text[$i]) - ord("a")) % 26) +
ord("a"));
}
$ki++; //menambah nilai variabel ki +1
if ($ki >= $kl)
{
$ki = 0;
}
}}
return $text;
}
```

Gambar 4. 6 Implementasi Enskripsi Vigenere Cipher

Class vigenere

Di dalam *class vigenere* terdapat *function encrypt* dan *decrypt*, dimana *class vigenere* ini dapat di panggil di *class* atau fungsi lainnya yang melibatkan proses enkripsi dan dekripsi. Cara pemanggilannya perlu memuat vigenere.php di class atau fungsi yang diperlukan, lalu membuat sebuah *object* baru dari class vigenere yang bertujuan untuk mengakses fungsi *encrypt* dan *decrypt* pada class vigenere.

Function encrypt

Function encrypt adalah proses enkripsi vigenere cipher, disaat *function encrypt* dipanggil dan kedua parameter diisi dengan benar maka jalankan proses enkripsi. Membuat sebuah perulangan for untuk meng iterasi tiap kata pada teks yang akan di enkripsi dan akan berhenti berdasarkan panjang *length text*.

```
function decrypt($key, $text)
{
$pswd = strtolower($key);
$code = "";
$ki = 0;
$kl = strlen($key);
$length = strlen($text);
for ($i = 0; $i < $length; $i++)
{if (ctype_alpha($text[$i]))
{
if (ctype_upper($text[$i]))
{
$x = (ord($text[$i]) - ord("A")) - (ord($key[$ki]) - ord("a"));
if ($x < 0)
}
$x = $x + ord("A");
$text[$i] = chr($x);
}
else
{
$x = (ord($text[$i]) - ord("a")) - (ord($key[$ki]) - ord("a"));
if ($x < 0)
{
$x += 26;
```

```
$x = $x + ord("a");
$text[$i] = chr($x);
}
$ki++;
if ($ki >= $kl)
{
$ki = 0;
}
}
}
return $text;
}
}
?>
```

Gambar 4. 7 Implementasi Enskripsi Vigenere Cipher – Fungsi Enskripsi

Function decrypt

Penjelasan fungsi *decrypt* sama dengan fungsi *encrypt*, dimana fungsi *decrypt* fungsi yang di akan digunakan untuk melakukan proses pendekripsian text yang terenkrisp. Fungsi *decrypt* dan fungsi *encrypt* memiliki perhitungan yang berbeda.

4.1.7 Implementasi Enkripsi RC4

Berikut ini merupakan *libraries* RC4.php yaitu *file* php yang digunakan untuk melakukan proses enkripsi dan dekripsi menggunakan algoritma kriptografi RC4. Alur proses untuk melakukan enkripsi dan dekripsi sama dengan vigenere perbedaan nya hanya jika saat melakukan enkripsi hasil dari enkripsi akan di ubah menjadi tipe bilangan hexa, sedangkan saat melakukan dekripsi, teks yang didapat diubah dahulu dari tipe hexa menjadi biner. Berikut merupakan potongan *code* yang digunakan:

```
class rc4 {
private $key;
private $maxLength;
public function __construct()
{
$this->key = '';
$this->maxLength = 256;
}
public function setKey($key)
{
$length = strlen($key);
if ($length < 1 || $length > 256) {
throw new OutOfRangeException('key must be between 1 and 256 bytes', 40);
}
$this->key = $key;
}
public function encrypt($plainText, $convertToHex = true)
{
return    $convertToHex    ?    bin2hex($this->process($plainText))    :    $this-
>process($plainText);
}
```

```
public function decrypt($encryptedText, $convertToBin = true)
{
return $this->process($convertToBin ? hex2bin($encryptedText) : $encryptedText);
}
private function process($string)
  {
    $stringPermutation = $this->performKeyScheduling(range(0, $this->maxLength -
1));
    return $this->generationAlgorithm($stringPermutation, $string);
```

Dalam enkripsi dan dekripsi algoritma kriptografi RC4 melibatkan 256 kararakter ASCII, dimana teks yang akan dienkripsi/dekripsi tidak hanya alphabet namun angka dan juga spasi maupun simbol dapat di enkripsi/dekripsi, hasil nya pun dari yang asli nya alphabet bisa menjadi simbol/angka.

Algoritma kriptografi RC4 terdapat 2x proses pada saat melakukan enkrip dan dekripsi, yang pertama adalah melakukan proses *Key Scheduling Algorithm* (KSA) dimana pada proses ini akan dilakukan pengacakan kunci sebanyak 256 kali pengacakan kunci, kenapa 256 kali? Karena total karakter ASCII adalah 256. Setelah proses Key Scheduling Algoritma (KSA) dilanjutkan dengan proses *Pseudo-random Generation Algorithm* (PRGA). Dimana proses PRGA akan menghasilkan angka *pseudo-random* yang kemudian dilakukan operasi XOR dengan *plaintext* untuk proses enkripsi atau dengan *ciphertext* pada proses dekripsi.

4.2 Pengujian Sistem

Pengujian sistem merupakan tahap lanjutan dari implementasi sistem. Pengujian sistem dilakukan guna membuktikan sistem yang dibangun telah berjalan sesuai dengan analisis dan perancangan sistem yang telah dibuat sebelumnya.

Spesifikasi perangkat keras yang digunakan untuk pengujian sistem adalah:

a. Prosesor Intel® Core™ i3 CPU 2.2GHz
b. Memori (RAM) 4.00 GB

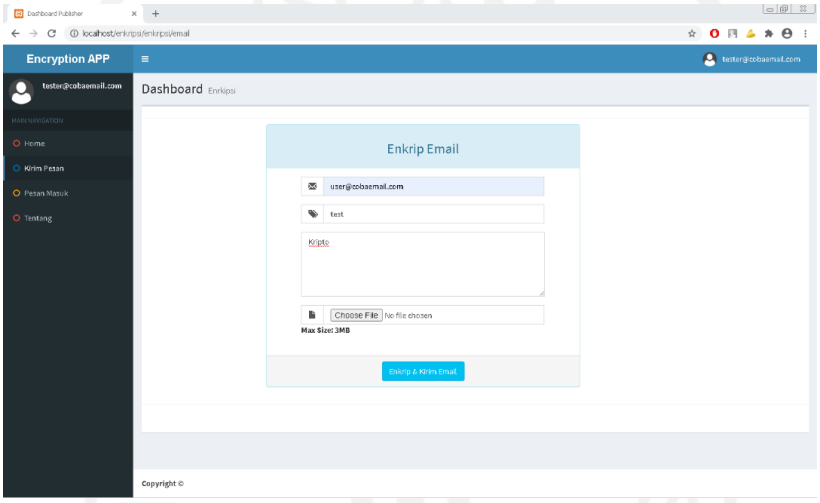
Spesifikasi perangkat lunak yang digunakan untuk pengujian sistem adalah:

a. Sistem Operasi Windows 10 Pro
b. XAMPP
c. GmailServer
d. Google Chrome
e. Mozilla Thunderbird

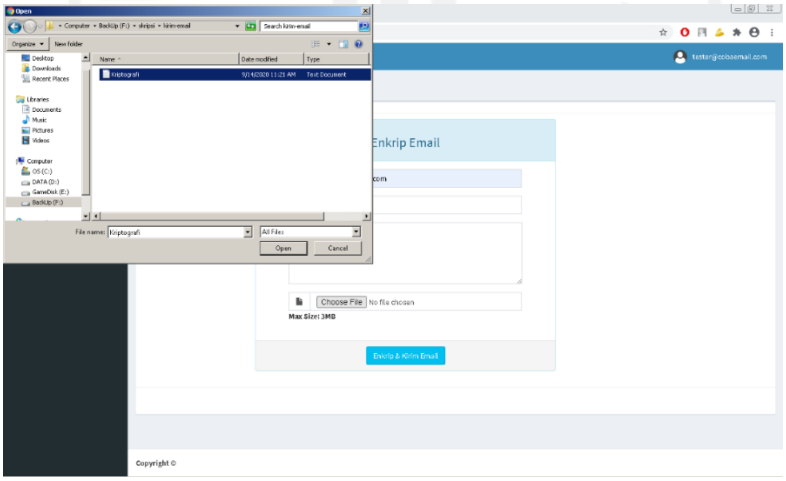
4.2.1 Pengujian Proses Pengiriman Pesan Email

Untuk melakukan proses pengiriman pesan email tahap awal yang dilakukannya adalah *user* memilih halaman kirim pesan. Setelah itu halaman kirim pesan akan muncul, selanjutkan dilakukan langkah-langkah berikut ini untuk melakukan proses pengiriman pesan email:

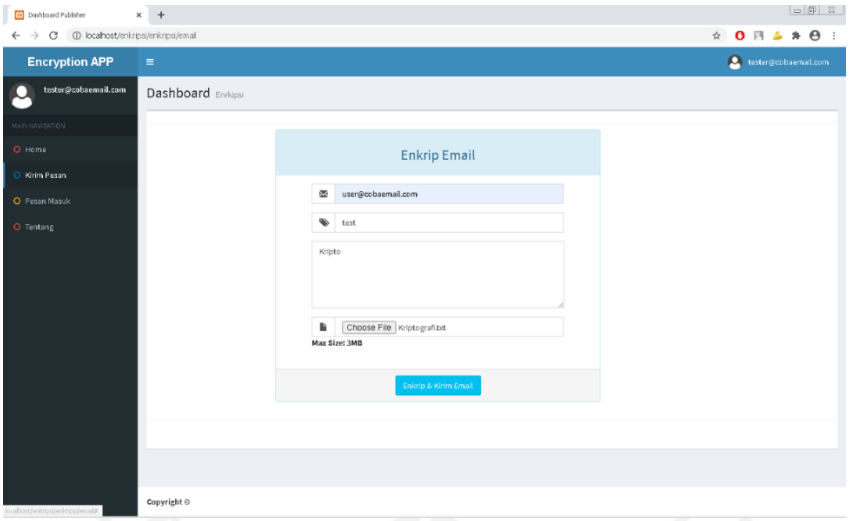
a. Mengisi *field-field* yang ada, masukkan *file* jika ingin mengirim *file* dengan cara menekan tombol *choose file*, lalu pilih *file* yang akan kirim. *File* yang dikirim hanya format txt, .doc, .docx, .xls, xlsx, .pdf, .ppt. dengan ukuran *file* dibawah 3MB. Pada Gambar 4. 6, Gambar 4. 7 serta Gambar 4. 8 ditampilkan proses pengiriman pesan, sebagai berikut:



Gambar 4. 8 Tampilkan Kirim Pesan Tanpa *File*

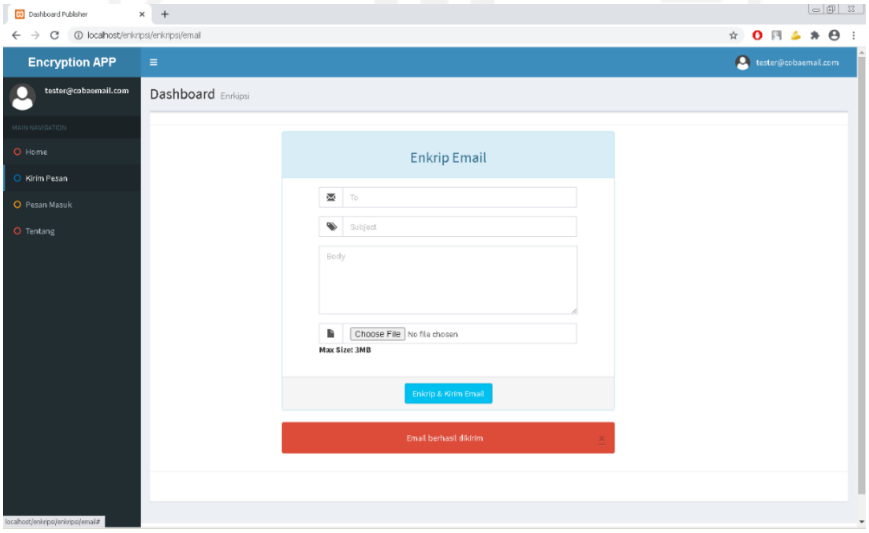


Gambar 4. 9 *Choose File* Pilih *File* Dikirim



Gambar 4. 10 Tampilan Kirim Pesan Dengan *File*

b. Selanjutkan tekan tombol Enkrip & Kirim Email untuk memulai proses enkripsi isi pesan beserta *file* yang akan dikirim, jika melampirkan *file* pada saat akan melakukan proses pengiriman email. Proses enkripsi isi pesan menggunakan algoritma vigenere cipher dan diikuti RC4, sedangkan enkripsi *file* hanya menggunakan algoritma RC4. Jika proses berhasil akan ditampilkan pesan bahwa email berhasil dikirim. Proses yang dimaksud bisa dilihat pada Gambar 4. 11 sebagai berikut:

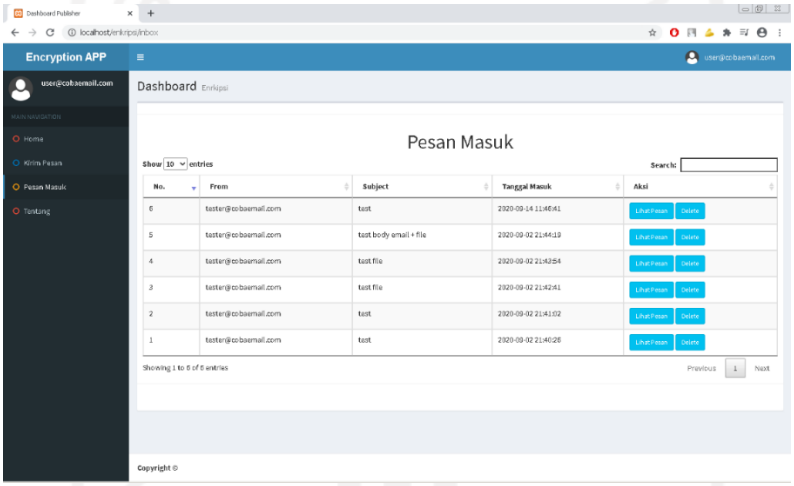


Gambar 4. 11 Tampilan Email Berhasil Dikirim

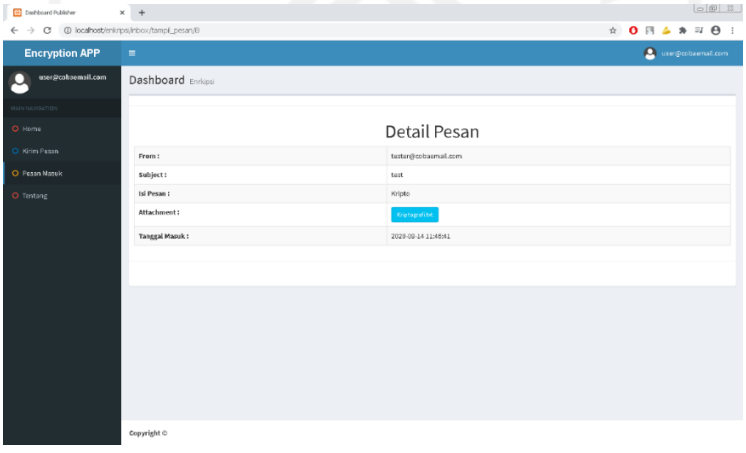
4.2.2 Pengujian Proses Membaca Pesan Email

Untuk melakukan proses membaca pesan email tahap awal yang dilakukan adalah *user* memlilih halaman pesan masuk. Setelah itu halaman pesan masuk akan muncul, dan menampilkan *list* pesan masuk, jika memang ada pesan yang diterima.

Tekan *button* lihat pesan, untuk memulai proses dekripsi dan membaca pesan email yang dipilih. Sebelum pesan dapat dibaca, pesan akan didekripsi terlebih dahulu, dekripsi isi pesan menggunakan algoritma RC4 dan diikuti dengan algoritma vigenere cipher, sedangkan dekripsi *file* hanya menggunakan algoritma RC4. Proses pembacaan pesan email tersebut bisa dilihat pada gambar berikut:

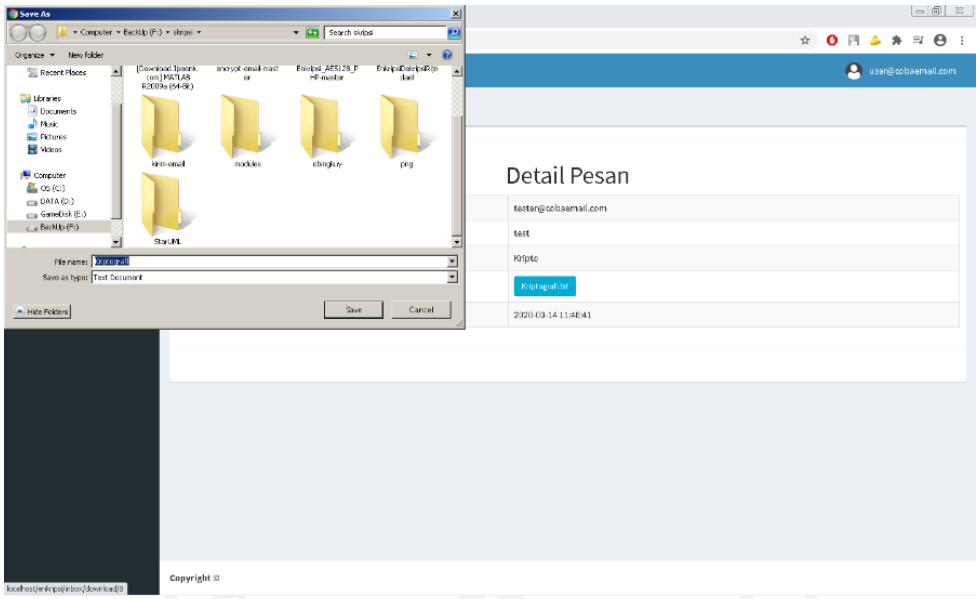


Gambar 4. 12 Tampilan *List* Pesan Masuk



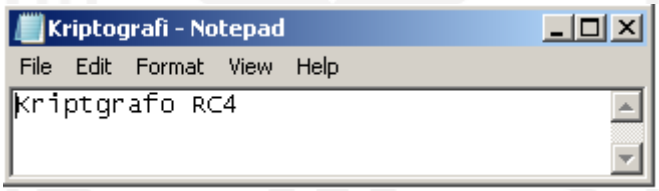
Gambar 4. 13 Tampilan Lihat Pesan

Tekan *button* nama *file*, untuk mengunduh dan menyimpan *file* yang dilampikan, sebagai berikut:



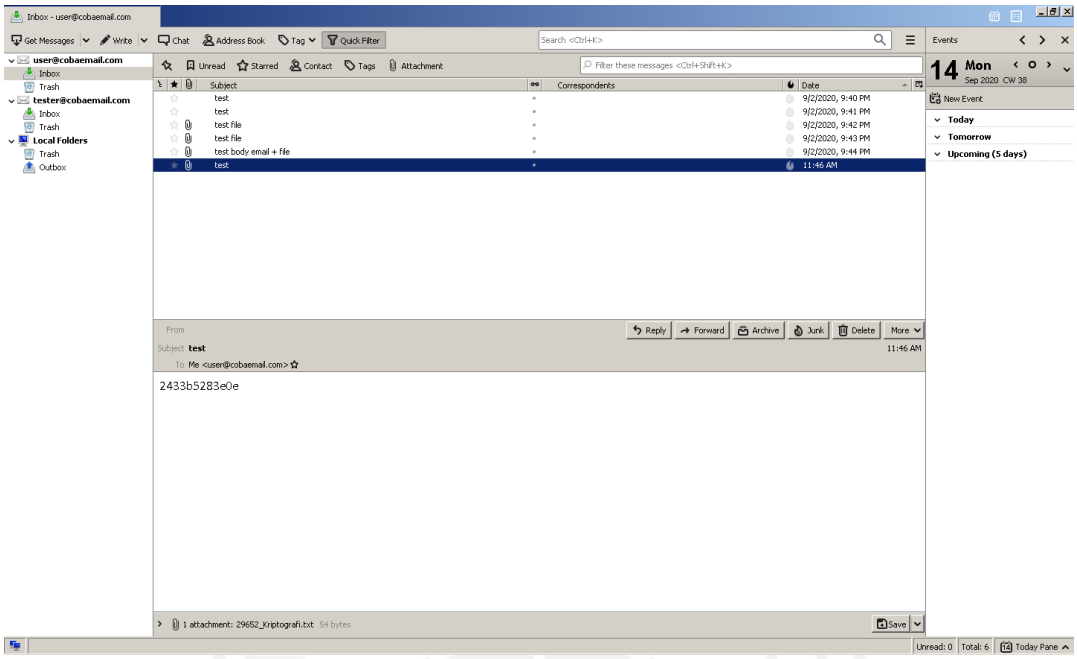
Gambar 4. 14 Tampilan Menyimpan *File*

Buka *file* yang disimpan sebelumnya, isi dari *file* tersebut akan kembali seperti sedia kala sebelum terenkripsi, sebagai berikut:



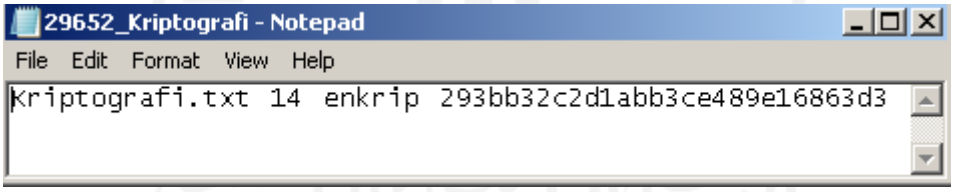
Gambar 4. 15 Tampilan Isi *File* Terdekripsi

Untuk melihat apakah isi pesan dan *file* dikirim terenkripsi apa tidak, *user* dapat melihat membuka Mozilla thunderbird, *login* akun email kedalam Mozilla thunderbird, dalam kasus ini yaitu email *user*@cobaemail.com akan *login* menggunakan Mozilla thunderbird. Setelah berhasil *login* lihat inbox pada Mozilla thunderbird. Isi inbox Mozilla thunderbird sama dengan isi dari aplikasi email yang dibuat. Isi pesan email yang dikirim melalui aplikasi email yang dibuat tidak akan bisa dibaca jika dibuka selain menggunakan aplikasi email yang dibuat, tampilannya sebagai berikut:



Gambar 4. 16 Pesan Email Dibuka Dengan Mozilla Thunderbird

Tekan *attachment* pada pesan email dipilih. Untuk menyimpan cipher*file*. *File* yang terenkripsi akan tersimpan dengan nama berbeda. Yang dienkripsi dari *file* ialah hanya isi nya, bukan *file* itu sendiri. Berikut murupakan tampilan isi *file* yang terenkripsi:



Gambar 4. 17 Tampilan Isi *File* Yang Terenkripsi

4.2.3 Pengujian Enkripsi Pesan dengan Perhitungan Manual

Pengujian hasil enkripsi dengan perhitungan manual ini dilakukan dengan *plaintext* dienkripsi dengan algoritma Vigenere Cipher dan hasilnya adalah *ciphertext* hasil enkripsi vigenere cipher. Selanjutnya *ciphertext* tersebut akan di enkripsi kembali dengan algoritma RC4, yang nanti nya akan menghasilkan *ciphertext* baru. Cipertext baru ini lah yang akan dikirim saat proses pengiriman pesan email. *Plaintext* untuk kasus ini ialah “Kripto” dengan

kunci “vigenere” untuk algoritma vigenere cipher, dan kunci “rc4” untuk algoritma RC4. Untuk lebih jelasnya proses enkripsi tersebut dijelaskan sebagai berikut:

a. Enkripsi Pesan dengan Algoritma Vigenere Cipher

Pada algoritma ini, *plaintext* dan kunci yang digunakan terlebih dahulu diubah kedalam bentuk ASCII. Kode ASCII dapat dilihat pada gambar 4.16 berikut:

CHAR	DEC	HEX	CHAR	DEC	HEX	CHAR	DEC	HEX	CHAR	DEC	HEX
[NUL]	0	00		32	20	@	64	40	`	96	60
[SOH]	1	01	!	33	21	A	65	41	a	97	61
[STX]	2	02	"	34	22	B	66	42	b	98	62
[ETX]	3	03	#	35	23	C	67	43	c	99	63
[EOT]	4	04	$	36	24	D	68	44	d	100	64
[ENQ]	5	05	%	37	25	E	69	45	e	101	65
[ACK]	6	06	&	38	26	F	70	46	f	102	66
[BEL]	7	07	'	39	27	G	71	47	g	103	67
[BS]	8	08	(	40	28	H	72	48	h	104	68
[HT]	9	09	)	41	29	I	73	49	i	105	69
[LF]	10	0A	*	42	2A	J	74	4A	j	106	6A
[VT]	11	0B	+	43	2B	K	75	4B	k	107	6B
[FF]	12	0C	,	44	2C	L	76	4C	l	108	6C
[CR]	13	0D	-	45	2D	M	77	4D	m	109	6D
[SO]	14	0E	.	46	2E	N	78	4E	n	110	6E
[SI]	15	0F	/	47	2F	O	79	4F	o	111	6F
[DLE]	16	10	0	48	30	P	80	50	p	112	70
[DC1]	17	11	1	49	31	Q	81	51	q	113	71
[DC2]	18	12	2	50	32	R	82	52	r	114	72
[DC3]	19	13	3	51	33	S	83	53	s	115	73
[DC4]	20	14	4	52	34	T	84	54	t	116	74
[NAK]	21	15	5	53	35	U	85	55	u	117	75
[SYN]	22	16	6	54	36	V	86	56	v	118	76
[ETB]	23	17	7	55	37	W	87	57	w	119	77
[CAN]	24	18	8	56	38	X	88	58	x	120	78
[EM]	25	19	9	57	39	Y	89	59	y	121	79
[SUB]	26	1A	:	58	3A	Z	90	5A	z	122	7A
[ESC]	27	1B	;	59	3B	[	91	5B	{	123	7B
[FS]	28	1C	<	60	3C	\	92	5C	|	124	7C
[GS]	29	1D	=	61	3D	]	93	5D	}	125	7D
[RS]	30	1E	>	62	3E	^	94	5E	~	126	7E
[US]	31	1F	?	63	3F	_	95	5F	[DEL]	127	7F

CHAR	DEC	HEX	CHAR	DEC	HEX	CHAR	DEC	HEX	CHAR	DEC	HEX
€	128	80		160	A0	À	192	C0	à	224	E0
(n/a)	129	81	¡	161	A1	Á	193	C1	á	225	E1
‚	130	82	¢	162	A2	Â	194	C2	â	226	E2
ƒ	131	83	£	163	A3	Ã	195	C3	ã	227	E3
„	132	84	¤	164	A4	Ä	196	C4	ä	228	E4
…	133	85	¥	165	A5	Å	197	C5	å	229	E5
†	134	86	¦	166	A6	Æ	198	C6	æ	230	E6
‡	135	87	§	167	A7	Ç	199	C7	ç	231	E7
ˆ	136	88	¨	168	A8	È	200	C8	è	232	E8
‰	137	89	©	169	A9	É	201	C9	é	233	E9
Š	138	8A	ª	170	AA	Ê	202	CA	ê	234	EA
‹	139	8B	«	171	AB	Ë	203	CB	ë	235	EB
Œ	140	8C	¬	172	AC	Ì	204	CC	ì	236	EC
(n/a)	141	8D		173	AD	Í	205	CD	í	237	ED
Ž	142	8E	®	174	AE	Î	206	CE	î	238	EE
(n/a)	143	8F	¯	175	AF	Ï	207	CF	ï	239	EF
(n/a)	144	90	°	176	B0	Ð	208	D0	ð	240	F0
‘	145	91	±	177	B1	Ñ	209	D1	ñ	241	F1
’	146	92	²	178	B2	Ò	210	D2	ò	242	F2
“	147	93	³	179	B3	Ó	211	D3	ó	243	F3
”	148	94	´	180	B4	Ô	212	D4	ô	244	F4
•	149	95	µ	181	B5	Õ	213	D5	õ	245	F5
–	150	96	¶	182	B6	Ö	214	D6	ö	246	F6
—	151	97	·	183	B7	×	215	D7	÷	247	F7
˜	152	98	¸	184	B8	Ø	216	D8	ø	248	F8
™	153	99	¹	185	B9	Ù	217	D9	ù	249	F9
š	154	9A	º	186	BA	Ú	218	DA	ú	250	FA
›	155	9B	»	187	BB	Û	219	DB	û	251	FB
œ	156	9C	¼	188	BC	Ü	220	DC	ü	252	FC
(n/a)	157	9D	½	189	BD	Ý	221	DD	ý	253	FD
ž	158	9E	¾	190	BE	Þ	222	DE	þ	254	FE
Ÿ	159	9F	¿	191	BF	ß	223	DF	ÿ	255	FF

Gambar 4. 18 Kode ASCII

Maka diketaui nilai ASCII dari *plaintext* “Kripto” adalah sebagai berikut:

K = 75, r = 114, i = 105, p = 112, t = 116, o = 111.

Dan kunci “vigenere” bernilai sebagai berikut:

v = 118, i = 105, g = 103, e = 101, n = 110, e = 101, r = 114, e = 101.

Dalam algoritma vigenere cipher, proses enkripsi dibagi menjadi 2, yaitu berdasarkan huruf capital, dan huruf kecil. Rumus enkripsi vigenere cipher jika *plaintext* adalah huruf capital sebagai berikut:

$C_i = ((K_{iASCII} - 97) + P_{iASCII} - 65 \bmod 26) + 65)$

Sedangkan rumus enkripsi vigenere cipher jika *plaintext* adalah huruf kecil sebagai berikut:

$C_i = (((K_{iASCII} - 65 + P_{iASCII} - 65) \bmod 26) + 97)$

Keterangan:

C_i = nilai decimal karakter *ciphertext*

K_{iASCII} = nilai decimal ASCII karakter kunci ke-i

P_{iASCII} = nilai decimal ASCII karakter *plaintext* ke-i

Memulai proses enkripsi *plaintext* "Kripto" dengan menggunakan algoritma vigenre cipher dengan kunci "vigenere".

Karakter K

Karena karakter K adalah huruf capital maka rumus enkripsi yang digunakan ialah rumus enkripsi untuk huruf capital. Karakter ini akan di enkripsi dengan *key* v

$C_i = (((K_{iASCII} - 97 + P_{iASCII} - 65) \bmod 26) + 65)$

$C_i = (((v - 97 + K - 65) \bmod 26) + 65)$

$C_i = (((118 - 97 + 75 - 65) \bmod 26) + 65)$

$C_i = ((31 \bmod 26) + 65)$

$C_i = (5 + 65)$

$C_i = 70 = F.$

Karakter r

Karena karakter r adalah huruf kecil maka rumus enkripsi yang digunakan ialah rumus enkripsi untuk huruf kecil. Karakter ini akan di enkripsi dengan *key* i

$C_i = (((K_{iASCII} - 97 + P_{iASCII} - 65) \bmod 26) + 97)$

$C_i = (((i - 97 + r - 97) \bmod 26) + 97)$

$C_i = (((105 - 97 + 114 - 97) \bmod 26) + 97)$

$C_i = ((25 \bmod 26) + 97)$

$C_i = (25 + 97)$

$C_i = 122 = z.$

Karakter i

Karena karakter i adalah huruf kecil maka rumus enkripsi yang digunakan ialah rumus enkripsi untuk huruf kecil. Karakter ini akan di enkripsi dengan *key* g

$C_i = (((K_{iASCII} - 97 + P_{iASCII} - 65) \bmod 26) + 97)$

C_i = (((g – 97 + i – 97) mod 26) + 97)

C_i = (((103– 97 + 105– 97) mod 26) + 97)

C_i = ((14 mod 26) + 97)

C_i = (14 + 97)

C_i = 111 = o.

Karakter p

Karena karakter p adalah huruf kecil maka rumus enkripsi yang digunakan ialah rumus enkripsi untuk huruf kecil. Karakter ini akan di enkripsi dengan *key* e

C_i = ((K_{iASCII} – 97 + P_{iASCII} – 65) mod 26) + 97)

C_i = (((e – 97 + p – 97) mod 26) + 97)

C_i = (((101– 97 + 112– 97) mod 26) + 97)

C_i = ((19 mod 26) + 97)

C_i = (19 + 97)

C_i = 116 = t.

Karakter t

Karena karakter t adalah huruf capital maka rumus enkripsi yang digunakan ialah rumus enkripsi untuk huruf kecil. Karakter ini akan di enkripsi dengan *key* n

C_i = ((K_{iASCII} – 97 + P_{iASCII} – 65) mod 26) + 97)

C_i = (((n – 97 + t – 97) mod 26) + 97)

C_i = (((110– 97 + 116– 97) mod 26) + 97)

C_i = ((32 mod 26) + 97)

C_i = (6 + 97)

C_i = 103 = g.

Karakter o

Karena karakter o adalah huruf capital maka rumus enkripsi yang digunakan ialah rumus enkripsi untuk huruf kecil. Karakter ini akan di enkripsi dengan *key* e

C_i = ((K_{iASCII} – 97 + P_{iASCII} – 65) mod 26) + 97)

C_i = (((e – 97 + o – 97) mod 26) + 97)

C_i = (((101– 97 + 111– 97) mod 26) + 97)

C_i = ((18 mod 26) + 97)

$C_i = (18 + 97)$

$C_i = 115 = s.$

Setelah dilakukan proses perhitungan manual, didapat sebuah *ciphertext* yaitu “Fzotgs”.

b. Enkripsi Pesan dengan Algoritma Rivest Code 4

Tahap *Key* Scheduling Algorithm (KSA)

Key Scheduling Algorithm diawali dengan menginisiasikan *state* awal yang berisi baris sebanyak 256 elemen. Baris 256 pada state awal dapat dilihat pada tabel dibawah ini:

Tabel 4. 1 Larik S awal

1	2	3	4	5	6	7	8	9	10	11	…	255
1	2	3	4	5	6	7	8	9	10	11	…	255

Selanjutnya kunci “rc4” akan diambil nilai ASCII-nya yaitu sebagai berikut :

r = 114

c = 99

4 = 52

Setalah itu dapat dilakukan perhitungan nilai j yang pertama dengan nilai awal i = 0 dan j = 0 sebagai berikut :

j = (j + S[i] + *key*[i *mod keylength*]) mod 256

j = (0 + 0 + 114) mod 256 = 114

Tukarkan nilai S[0] dengan S[114]. Kemudian dilakukan perhitungan kembali untuk nilai j yang kedua dengan nilai i = 1 dan j = 114 sebagai berikut :

j = (j + S[i] + *key*[i *mod keylength*]) mod 256

j = (114 + 1 + 99) *mod* 256 = 214

Tukarkan nilai S[1] dengan S[214]. Langkah ini diulangi hingga i mencapai nilai 255 dan hasil dari tahap *key scheduling* dapat dilihat pada tabel 4.2 dimana nilai i berada pada baris yang berwarna biru.

Tabel 4. 2 Hasil Perhitungan *Key* Scheduling Algorithm

0	1	2	3	4	5	6	7	8	9	10	11	12	13
114	92	12	64	47	172	32	122	154	186	236	102	218	15
14	15	16	17	18	19	20	21	22	23	24	25	26	27
163	74	127	132	183	191	129	31	55	6	220	88	166	131
28	29	30	31	32	33	34	35	36	37	38	39	40	41
178	7	243	142	105	224	44	19	23	71	40	136	59	152
42	43	44	45	46	47	48	49	50	51	52	53	54	55
149	79	66	41	126	35	87	98	81	60	107	113	97	20
56	57	58	59	60	61	62	63	64	65	66	67	68	69
106	193	109	11	90	116	125	240	29	253	101	210	43	91
70	71	72	73	74	75	76	77	78	79	80	81	82	83
117	2	223	84	50	137	77	185	121	226	175	158	228	65
84	85	86	87	88	89	90	91	92	93	94	95	96	97
16	192	234	28	171	78	135	214	213	145	34	169	0	205
98	99	100	101	102	103	104	105	106	107	108	109	110	111
159	161	56	151	227	139	217	237	209	207	76	202	24	189
112	113	114	115	116	117	118	119	120	121	122	123	124	125
85	249	180	254	246	221	115	245	168	195	22	244	219	230
126	127	128	129	130	131	132	133	134	135	136	137	138	139
72	157	206	201	104	89	197	231	200	53	176	26	9	112
140	141	142	143	144	145	146	147	148	149	150	151	152	153
54	177	232	199	196	75	25	241	33	18	1	208	58	165
154	155	156	157	158	159	160	161	162	163	164	165	166	167
138	212	86	190	83	93	167	252	63	30	68	82	17	250
168	169	170	171	172	173	174	175	176	177	178	179	180	181
204	248	100	61	148	62	118	10	155	67	247	130	80	21
182	183	184	185	186	187	188	189	190	191	192	193	194	195
153	5	203	242	238	111	37	233	110	188	134	229	13	211
196	197	198	199	200	201	202	203	204	205	206	207	208	209
120	173	147	162	144	251	57	164	235	124	38	14	48	108
210	211	212	213	214	215	216	217	218	219	220	221	222	223
46	27	222	69	94	45	70	181	216	141	156	239	187	146
224	225	226	227	228	229	230	231	232	233	234	235	236	237

215	123	194	119	255	73	184	182	170	150	49	128	225	42
238	239	240	241	242	243	244	245	246	247	248	249	250	251
143	4	96	140	179	198	3	52	36	160	174	8	51	99
252	253	254	255										
39	103	133	95										

Tahap Pseudo-Random Generation Algorithm(PRGA)

Langkah berikutnya setelah melakukan tahap KSA adalah tahap Pseudo-Random Generation Algorithm (PRGA). Pada tahapan ini, *ciphertext* dari hasil enkripsi vigenere cipher sebelumnya yaitu “Fzotgs” akan dienkripsi dengan memroses setiap karakter. Untuk proses enkripsi sebagai berikut:

Karakter F

Tahap awal lakukan inisialisasi nilai i, j, dan z = 0 dan setiap baris akan di modulo dengan 256. Kemudian dilakukan perhitungan dengan cara sebagai berikut:

i = i + 1 ➔ i = (0 + 1) mod 256 = 1

j = j + S[i] ➔ j = (0 + 92) mod 256 = 92

Selanjutnya nilai S[i] dan S[j] ditukarkan kemudian dilakukan perhitungan nilai z dengan cara sebagai berikut:

S[i], S[j] ➔ S[i] = S[1] = 92, S[j] = S[92] = 213

Swap S[i] with S[j] ➔ S[i] = S[1] = 213, S[j] = S[92] = 92

z = S[i] + S[j] ➔ z = (213 + 92) mod 256 = 49

Selanjutnya lakukanlah operasi XOR untuk karakter pertama pada *ciphertext* hasil vigenere, yaitu karakter “F” dengan kode ASCII 70 dan nilai S[z] = S[49] yang didapat adalah 98 sebagai berikut.

0100 0110
0110 0010 ⊕
0010 0100 = 36_{10}

Pads tabel ASCII nilai 36_{10} merupakan karakter “$”, *convert* nilat 36_{10} menjadi nilai hexadecimal, bilangan hexadecimal dari nilai 36_{10} adalah “24”. Bilangan hexadecimal inilah yang akan menjadi *ciphertext*.

Karakter z

Pada tahap ini nilai i = 1, j = 92 dan z = 0 dan setiap baris akan di modulo dengan 256. Kemudian dilakukan perhitungan nilai i dan j baru dengan cara sebagai berikut:

i = i + 1 ➔ i = (1 + 1) mod 256 = 2

j = j + S[i] ➔ j = (92 + 12) mod 256 = 104

Selanjutnya nilai S[i] dan S[j] ditukarkan kemudian dilakukan perhitungan nilai z dengan cara sebagai berikut:

S[i], S[j] ➔ S[i] = S[2] = 12, S[j] = S[104] = 217

Swap S[i] with S[j] ➔ S[i] = S[2] = 217, S[j] = S[104] = 12

- z = S[i] + S[j] ➔ z = (217 + 12) mod 256 = 229

Selanjutnya lakukanlah operasi XOR untuk karakter pertama pada *ciphertext* hasil vigenere, yaitu karakter "z" dengan kode ASCII 102 dan nilai S[z] = S[229] yang didapat adalah 73 sebagai berikut.

0111 1010
0100 1001 ⊕
0011 0011 = 51_{10}

Pada tabel ASCII nilai 51_{10} merupakan karakter "3", *convert* nilat 51_{10} menjadi nilai hexadecimal, bilangan hexadecimal dari nilai 36_{10} adalah "33". Bilangan hexadecimal inilah yang akan menjadi *ciphertext*.

Karakter o

Pada tahap ini nilai i = 2, j = 104 dan z = 0 dan setiap baris akan di modulo dengan 256. Kemudian dilakukan perhitungan nilai i dan j baru dengan cara sebagai berikut:

i = i + 1 ➔ i = (2 + 1) mod 256 = 3

j = j + S[i] ➔ j = (104 + 64) mod 256 = 168

Selanjutnya nilai S[i] dan S[j] ditukarkan kemudian dilakukan perhitungan nilai z dengan cara sebagai berikut:

S[i], S[j] ➔ S[i] = S[3] = 64, S[j] = S[168] = 204

Swap S[i] with S[j] ➔ S[i] = S[3] = 204, S[j] = S[168] = 64

z = S[i] + S[j] ➔ z = (204 + 64) mod 256 = 12

Selanjutnya lakukanlah operasi XOR untuk karakter pertama pada *ciphertext* hasil vigenere, yaitu karakter "o" dengan kode ASCII 111 dan nilai S[z] = S[12] yang didapat adalah 218 sebagai berikut.

0110 1111
1101 1010 ⊕
1011 0101 = 181_{10}

Pada tabel ASCII nilai 181_{10} merupakan karakter "µ", *convert* nilat 181_{10} menjadi nilai hexadecimal, bilangan hexadecimal dari nilai 36_{10} adalah "b5". Bilangan hexadecimal inilah yang akan menjadi *ciphertext*.

Karakter t

Pada tahap ini nilai i = 3, j = 168 dan z = 0 dan setiap baris akan di modulo dengan 256. Kemudian dilakukan perhitungan nilai i dan j baru dengan cara sebagai berikut:

i = i + 1	➔	i = (3 + 1) mod 256 = 4
j = j + S[i]	➔	j = (168 + 47) mod 256 = 215

Selanjutnya nilai S[i] dan S[j] ditukarkan kemudian dilakukan perhitungan nilai z dengan cara sebagai berikut:

S[i], S[j]	➔	S[i] = S[4] = 47, S[j] = S[215] = 45
Swap S[i] with S[j]	➔	S[i] = S[4] = 45, S[j] = S[92] = 47
z = S[i] + S[j]	➔	z = (45 + 47) mod 256 = 92

Selanjutnya lakukanlah operasi XOR untuk karakter pertama pada *ciphertext* hasil vigenere, yaitu karakter "t" dengan kode ASCII 116 dan nilai S[z] = S[92] yang didapat adalah 92 sebagai berikut.

0111 0100
<u>0101 1100</u> ⊕
0010 1000 = 40_{10}

Pada tabel ASCII nilai 40_{10} merupakan karakter "(", *convert* nilat 40_{10} menjadi nilai hexadecimal, bilangan hexadecimal dari nilai 40_{10} adalah "28". Bilangan hexadecimal inilah yang akan menjadi *ciphertext*.

Karakter g

Pada tahap ini nilai i = 4, j = 215 dan z = 0 dan setiap baris akan di modulo dengan 256. Kemudian dilakukan perhitungan nilai i dan j baru dengan cara sebagai berikut:

i = i + 1	➔	i = (4 + 1) mod 256 = 5
j = j + S[i]	➔	j = (215 + 172) mod 256 = 131

Selanjutnya nilai S[i] dan S[j] ditukarkan kemudian dilakukan perhitungan nilai z dengan cara sebagai berikut:

S[i], S[j]	➔	S[i] = S[5] = 172, S[j] = S[131] = 89
Swap S[i] with S[j]	➔	S[i] = S[5] = 89, S[j] = S[131] = 172
z = S[i] + S[j]	➔	z = (89 + 172) mod 256 = 5

Selanjutnya lakukanlah operasi XOR untuk karakter pertama pada *ciphertext* hasil vigenere, yaitu karakter "g" dengan kode ASCII 103 dan nilai S[z] = S[5] yang didapat adalah 89 sebagai berikut.

0110 0111
0101 1001 ⊕
0011 1110 = 62_{10}

Didalam tabel ASCII nilai 62_{10} merupakan karakter ">", *convert* nilat 62_{10} menjadi nilai hexadecimal, bilangan hexadecimal dari nilai 62_{10} adalah "3e". Bilangan hexadecimal inilah yang akan menjadi *ciphertext*.

Karakter s

Pada tahap ini nilai i = 5, j = 131 dan z = 0 dan setiap baris akan di modulo dengan 256. Kemudian dilakukan perhitungan nilai i dan j baru dengan cara sebagai berikut:

i = i + 1	➔	i = (5 + 1) mod 256 = 6
j = j + S[i]	➔	j = (131 + 32) mod 256 = 163

Selanjutnya nilai S[i] dan S[j] ditukarkan kemudian dilakukan perhitungan nilai z dengan cara sebagai berikut:

S[i], S[j]	➔	S[i] = S[6] = 32, S[j] = S[163] = 30
Swap S[i] with S[j]	➔	S[i] = S[6] = 30, S[j] = S[163] = 32
z = S[i] + S[j]	➔	z = (30 + 32) mod 256 = 62

Selanjutnya lakukanlah operasi XOR untuk karakter pertama pada *ciphertext* hasil vigenere, yaitu karakter "s" dengan kode ASCII 115 dan nilai S[z] = S[62] yang didapat adalah 125 sebagai berikut.

0111 0011
0111 1101 ⊕
0000 1110 = 14_{10}

dalam tabel ASCII nilai 14_{10} merupakan karakter "SO", *convert* nilat 14_{10} menjadi nilai hexadecimal, bilangan hexadecimal dari nilai 14_{10} adalah "0e". Bilangan hexadecimal inilah yang akan menjadi *ciphertext*.

Setelah dilakukanya perhitungan manual enkripsi pesan email, dengan menggunakan algoritma vigenere cipher dan RC4 diperoleh *ciphertext* yang berbentuk bilangan hexadecimal yaitu "2433b5283e0e". *Ciphertext* inilah yang akan terkirim saat *user* mengirimkan pesan "Kripto". Hasil *ciphertext* dari perhitungan manual ini sama dengan yang ada pada gambar 4.18.

4.2.4 Pengujian Dekripsi Pesan dengan Perhitungan Manual

Pengujian hasil Dekripsi dengan perhitungan manual ini dilakukan dengan *ciphertext* yang akan didekripsi dengan algoritma Rivest Code 4 terlebih dahulu, kemudian hasil dari dekripsi RC4 akan di dekripsi kembali dengan algoritma vigenere cipher. Yang nanti nya akan menghasilkan *plaintext* baru. *plaintext* baru ini lah yang akan muncul saat proses membaca pesan email. *Ciphertext* untuk kasus ini ialah "2433b5283e0e" dengan kunci "rc4" untuk algoritma rivest code 4, dan kunci "vigenere" untuk algoritma vigenere cipher. Untuk lebih jelasnya proses dekripsi tersebut dijelaskan sebagai berikut:

a. Dekripsi Pesan dengan Algoritma Rivest Code 4

Pada proses dekripsi, algoritma Spritz memiliki tahapan yang sama pada saat proses enkripsi. Perbedaannya adalah terdapat pada tahap Pseudo-Random Generation Algorithm (PRGA) dimana pada tahap ini terdapat perbedaan pada nilai saat operasi XOR dilakukan. sedangkan pada proses dekripsi, ciphertext akan di XOR-kan dengan perhitungan yang sama pada tahap Pseudo-Random Generation Algorithm (PRGA) sebelumya dan menggunakan tabel yang sama pada tahap Key Scheduling Algorithm (KSA) yang ditunjukkan pada tabel 4.2 di atas.

Ciphertext yang akan didekripsi adalah *ciphertext* yang telah diperoleh pada enkripsi sebelumnya yaitu "2433b5283e0e".

Hexadecimal "24"

Tahap awal lakukan inisialisasi nilai i, j, dan z = 0 dan setiap baris akan di modulo dengan 256. Kemudian dilakukan perhitungan dengan cara sebagai berikut:

i = i + 1	➔	i = (0 + 1) mod 256 = 1
j = j + S[i]	➔	j = (0 + 92) mod 256 = 92

Selanjutnya nilai S[i] dan S[j] ditukarkan kemudian dilakukan perhitungan nilai z dengan cara sebagai berikut:

S[i], S[j]	➔	S[i] = S[1] = 92, S[j] = S[92] = 213
Swap S[i] with S[j]	➔	S[i] = S[1] = 213, S[j] = S[92] = 92
z = S[i] + S[j]	➔	z = (213 + 92) mod 256 = 49

Selanjutnya lakukanlah operasi XOR untuk karakter pertama pada *ciphertext* sebelumnya, yaitu Hexadecimal "24" dengan kode ASCII 36 dan nilai S[z] = S[49] yang didapat adalah 98 sebagai berikut.

0010 0100
0110 0010 ⊕
0100 0110 = 70_{10} dalam tabel ASCII nilai 70_{10} merupakan karakter “F".

Hexadecimal “33”

Pada tahap ini nilai i = 1, j = 92 dan z = 0 dan setiap baris akan di modulo dengan 256. Kemudian dilakukan perhitungan nilai i dan j baru dengan cara sebagai berikut:

i = i + 1 ➔ i = (1 + 1) mod 256 = 2

j = j + S[i] ➔ j = (92 + 12) mod 256 = 104

Selanjutnya nilai S[i] dan S[j] ditukarkan kemudian dilakukan perhitungan nilai z dengan cara sebagai berikut:

S[i], S[j] ➔ S[i] = S[2] = 12, S[j] = S[104] = 217

Swap S[i] with S[j] ➔ S[i] = S[2] = 217, S[j] = S[104] = 12

- z = S[i] + S[j] ➔ z = (217 + 12) mod 256 = 229

Selanjutnya lakukanlah operasi XOR untuk karakter pertama pada *ciphertext* hasil vigenere, yaitu Hexadecimal “z” dengan kode ASCII 51 dan nilai S[z] = S[229] yang didapat adalah 73 sebagai berikut.

0011 0011
0100 1001 ⊕
0111 1010 = 122_{10} dalam tabel ASCII nilai 122_{10} merupakan karakter “z”.

Hexadecimal “b5”

Pada tahap ini nilai i = 2, j = 104 dan z = 0 dan setiap baris akan di modulo dengan 256. Kemudian dilakukan perhitungan nilai i dan j baru dengan cara sebagai berikut:

i = i + 1 ➔ i = (2 + 1) mod 256 = 3

j = j + S[i] ➔ j = (104 + 64) mod 256 = 168

Selanjutnya nilai S[i] dan S[j] ditukarkan kemudian dilakukan perhitungan nilai z dengan cara sebagai berikut:

S[i], S[j] ➔ S[i] = S[3] = 64, S[j] = S[168] = 204

Swap S[i] with S[j] ➔ S[i] = S[3] = 204, S[j] = S[168] = 64

z = S[i] + S[j] ➔ z = (204 + 64) mod 256 = 12

Selanjutnya lakukanlah operasi XOR untuk karakter pertama pada *ciphertext* hasil vigenere, yaitu Hexadecimal “b5” dengan kode ASCII 181 dan nilai S[z] = S[12] yang didapat adalah 218 sebagai berikut.

1011 0101
1101 1010 ⊕
0110 1111 = 111_{10} dalam tabel ASCII nilai 111_{10} merupakan karakter “o”.

Hexadecimal “28”

Pada tahap ini nilai i = 3, j = 168 dan z = 0 dan setiap baris akan di modulo dengan 256. Kemudian dilakukan perhitungan nilai i dan j baru dengan cara sebagai berikut:

i = i + 1	➔	i = (3 + 1) mod 256 = 4
j = j + S[i]	➔	j = (168 + 47) mod 256 = 215

Selanjutnya nilai S[i] dan S[j] ditukarkan kemudian dilakukan perhitungan nilai z dengan cara sebagai berikut:

S[i], S[j]	➔	S[i] = S[4] = 47, S[j] = S[215] = 45
Swap S[i] with S[j]	➔	S[i] = S[4] = 45, S[j] = S[92] = 47
z = S[i] + S[j]	➔	z = (45 + 47) mod 256 = 92

Selanjutnya lakukanlah operasi XOR untuk karakter pertama pada *ciphertext* hasil vigenere, yaitu Hexadecimal “28” dengan kode ASCII 40 dan nilai S[z] = S[92] yang didapat adalah 92 sebagai berikut.

0010 1000
0101 1100 ⊕
0111 0100 = 116_{10} dalam tabel ASCII nilai 116_{10} merupakan karakter “t”.

Hexadecimal “3e”

Pada tahap ini nilai i = 4, j = 215 dan z = 0 dan setiap baris akan di modulo dengan 256. Kemudian dilakukan perhitungan nilai i dan j baru dengan cara sebagai berikut:

i = i + 1	➔	i = (4 + 1) mod 256 = 5
j = j + S[i]	➔	j = (215 + 172) mod 256 = 131

Selanjutnya nilai S[i] dan S[j] ditukarkan kemudian dilakukan perhitungan nilai z dengan cara sebagai berikut:

S[i], S[j]	➔	S[i] = S[5] = 172, S[j] = S[131] = 89
Swap S[i] with S[j]	➔	S[i] = S[5] = 89, S[j] = S[131] = 172
z = S[i] + S[j]	➔	z = (89 + 172) mod 256 = 5

Selanjutnya lakukanlah operasi XOR untuk karakter pertama pada *ciphertext* hasil vigenere, yaitu karakter “3e” dengan kode ASCII 62 dan nilai S[z] = S[5] yang didapat adalah 89 sebagai berikut.

0011 1110

0101 1001 ⊕
0110 0111= 103_{10} dalam tabel ASCII nilai 103_{10} merupakan karakter “g”.

Hexadecimal “0e”

Pada tahap ini nilai i = 5, j = 131 dan z = 0 dan setiap baris akan di modulo dengan 256. Kemudian dilakukan perhitungan nilai i dan j baru dengan cara sebagai berikut:

i = i + 1 ➔ i = (5 + 1) mod 256 = 6

j = j + S[i] ➔ j = (131 + 32) mod 256 = 163

Selanjutnya nilai S[i] dan S[j] ditukarkan kemudian dilakukan perhitungan nilai z dengan cara sebagai berikut:

S[i], S[j] ➔ S[i] = S[6] = 32, S[j] = S[163] = 30

Swap S[i] with S[j] ➔ S[i] = S[6] = 30, S[j] = S[163] = 32

z = S[i] + S[j] ➔ z = (30 + 32) mod 256 = 62

Selanjutnya lakukanlah operasi XOR untuk karakter pertama pada *ciphertext* hasil vigenere, yaitu karakter “0e” dengan kode ASCII 14 dan nilai S[z] = S[62] yang didapat adalah 125 sebagai berikut.

0000 1110
0111 1101 ⊕
0111 0011= 115_{10} dalam tabel ASCII nilai 115_{10} merupakan karakter “s”.

Setelah dilakukan proses perhitungan manual dekripsi pesan dengan algoritma RC4, didapat dekriptext pertama yaitu “Fzotgs”.

b. Dekripsi Pesan dengan Algoritma Vigenere Cipher

Pada algoritma ini, dekriptext dan kunci yang digunakan terlebih dahulu diubah kedalam bentuk ASCII. Kode ASCII dapat dilihat pada gambar 4.18. Maka diketaui nilai ASCII dari dekriptext dari hasil dekripsi rc4 sebelumnya yaitu “Fzotgs” adalah sebagai berikut:

F = 70, z = 122, o = 111, p = 112, t = 116, o = 115.

Dan kunci “vigenere” bernilai sebagai berikut:

v = 118, i = 105, g = 103, e = 101, n = 110, e = 101, r = 114, e = 101.

Dalam algoritma vigenere cipher, proses enkripsi dibagi menjadi 2, yaitu berdasarkan huruf capital, dan huruf kecil. Rumus enkripsi vigenere cipher jika *ciphertext* adalah huruf capital sebagai berikut:

$P_i = ((((C_{iASCII} - 65) - (K_{iASCII} - 97)) \bmod 26) + 65)$

Sedangkan rumus enkripsi vigenere cipher jika *plaintext* adalah huruf kecil sebagai berikut:

$P_i = ((((C_{iASCII} - 97) + (K_{iASCII} - 97)) \bmod 26) + 97)$

Keterangan:

P_i = nilai decimal karakter *plaintext* ke-i

K_{iASCII} = nilai decimal ASCII karakter kunci ke-i

C_{iASCII} = nilai decimal ASCII karakter *ciphertext* ke-i

Memulai proses dekripsi dekriptext sebelumnya yaitu “Fzotgs” dengan menggunakan algoritma vigenre cipher dengan kunci “vigenere”.

Karakter “F”

Karena karakter “F” adalah huruf capital maka rumus enkripsi yang digunakan ialah rumus enkripsi untuk huruf capital. Karakter ini akan di enkripsi dengan *key* v

$P_i = ((((C_{iASCII} - 65) - (K_{iASCII} - 97)) \bmod 26) + 65)$

$P_i = ((((F - 65) - (v - 97)) \bmod 26) + 65)$

$P_i = ((((70 - 65) - (101 - 97)) \bmod 26) + 65)$

$P_i = ((-16 \bmod 26) + 65)$

$P_i = (10 + 65)$

$P_i = 75 = K.$

Karakter “z”

Karena karakter “z” adalah huruf kecil maka rumus enkripsi yang digunakan ialah rumus enkripsi untuk huruf kecil. Karakter ini akan di enkripsi dengan *key* i

$P_i = ((((C_{iASCII} - 97) + (K_{iASCII} - 97)) \bmod 26) + 97)$

$P_i = ((((z - 97) + (i - 97)) \bmod 26) + 97)$

$P_i = (((122 - 97 + 105 - 97) \bmod 26) + 97)$

$P_i = ((17 \bmod 26) + 97)$

$P_i = (17 + 97)$

$P_i = 114 = r.$

Karakter “o”

Karena karakter “o” adalah huruf kecil maka rumus enkripsi yang digunakan ialah rumus enkripsi untuk huruf kecil. Karakter ini akan di enkripsi dengan *key* g

$P_i = ((((C_{iASCII} - 97) + (K_{iASCII} - 97)) \bmod 26) + 97)$

$P_i = ((((o - 97) + (g - 97)) \bmod 26) + 97)$

$P_i = (((111 - 97 + 103 - 97) \bmod 26) + 97)$

$P_i = ((8 \bmod 26) + 97)$

$P_i = (8 + 97)$

$P_i = 105 = i.$

Karakter "t"

Karena karakter "t" adalah huruf kecil maka rumus enkripsi yang digunakan ialah rumus enkripsi untuk huruf kecil. Karakter ini akan di enkripsi dengan *key* e

$P_i = ((((C_{iASCII} - 97) + (K_{iASCII} - 97)) \bmod 26) + 97)$

$P_i = ((((t - 97) + (e - 97)) \bmod 26) + 97)$

$P_i = (((116 - 97 + 101 - 97) \bmod 26) + 97)$

$P_i = ((15 \bmod 26) + 97)$

$P_i = (15 + 97)$

$P_i = 112 = p.$

Karakter "g"

Karena karakter "g" adalah huruf capital maka rumus enkripsi yang digunakan ialah rumus enkripsi untuk huruf capital. Karakter ini akan di enkripsi dengan *key* n

$P_i = ((((C_{iASCII} - 97) + (K_{iASCII} - 97)) \bmod 26) + 97)$

$P_i = ((((g - 97) + (n - 97)) \bmod 26) + 97)$

$P_i = (((103 - 97 + 110 - 97) \bmod 26) + 97)$

$P_i = ((-7 \bmod 26) + 97)$

$P_i = (19 + 97)$

$P_i = 116 = t.$

Karakter "s"

Karena karakter "s" adalah huruf kecil maka rumus enkripsi yang digunakan ialah rumus enkripsi untuk huruf kecil. Karakter ini akan di enkripsi dengan *key* e

$P_i = ((((C_{iASCII} - 97) + (K_{iASCII} - 97)) \bmod 26) + 97)$

$P_i = ((((s - 97) + (e - 97)) \bmod 26) + 97)$

$P_i = (((115 - 97 + 101 - 97) \bmod 26) + 97)$

$P_i = ((14 \bmod 26) + 97)$

$P_i = (14 + 97)$

$P_i = 111 = o.$

Setelah dilakukan proses perhitungan manual dekripsi dengan menggunakan algoritma vigenere cipher, didapat sebuah *plaintext* baru yaitu “Kripto”. Setelah dilakukanya perhitungan manual dekripsi pesan email, dengan menggunakan algoritma RC4 dan vigenere cipher diperoleh *plaintext* yaitu “Kripto”. *Ciphertext* inilah yang akan terbaca oleh *user* saat *user* menekan tombol lihat pesan. Hasil dekripsi *plaintext* dari perhitungan manual ini memiliki output yang sama dengan yang ada pada gambar 4.13.

4. 3. Hasil Pengujian Sistem

Setelah dilakukan proses implementasi dan pengujian sistem maka diperoleh hasil pengujian sistem. Guna memperoleh hasil pengujian sistem dilakukan wawancara kepada Tim IT Support PT. Embee sebagai responden. Wawancara dilakukan setelah responden menggunakan aplikasi. Pertanyaan dan jawaban dari proses wawancara dapat dilihat pada Tabel 4. 3.

Tabel 4. 3 Hasil Wawancara Pengujian Sistem

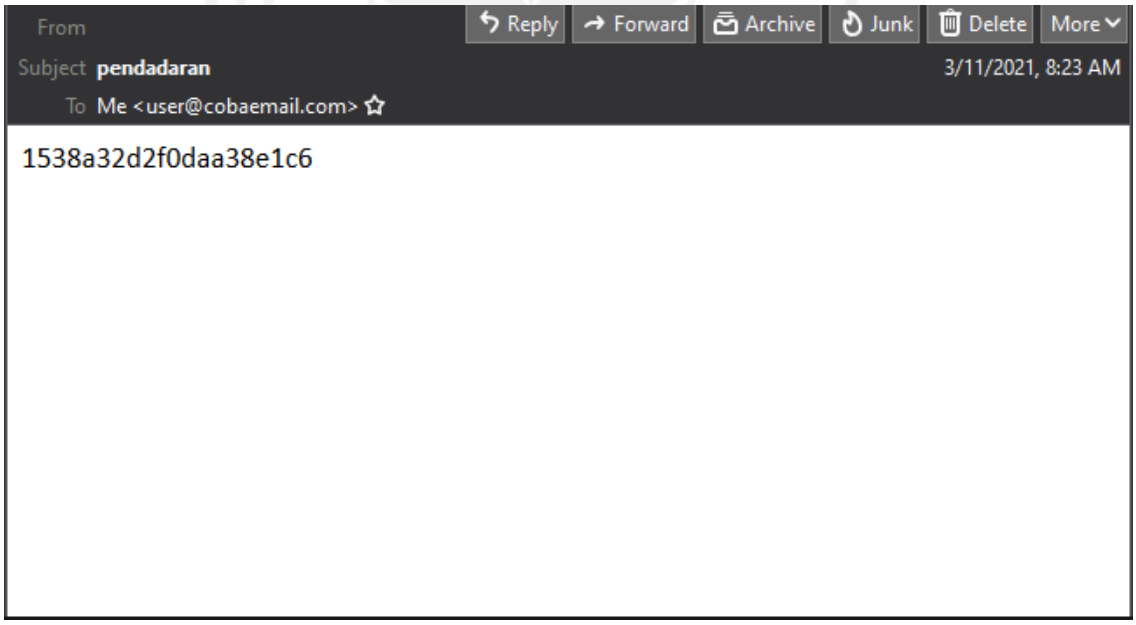
No	Pertanyaan	Jawaban
1.	Apakah aplikasi ini sudah berhasil merahasiakan data dari pihak yang tidak berkenan?	Iya sudah, karena melalui aplikasi ini data menjadi terenkripsi sehingga terjamin kerahasiaannya.
2.	Apakah melalui aplikasi ini data pada pesan yang dikirimkan mengalami perubahan saat diterima oleh email tujuan?	Tidak, data yang dikirimkan tidak mengalami perubahan ketika dikirimkan, jumlah data sesuai dengan data yang pertama kali dikirimkan.
3.	Apakah melalui aplikasi ini data yang dikirimkan terjamin keasliannya?	Iya, karena email yang telah di dekripsi hanya dapat dibaca oleh penerima email yang dituju.
4.	Apakah ada pihak yang menyangkal bahwa ia telah melakukan pengiriman pesan atau telah menerima pesan?	Tidak ada, karena di setiap email sudah tertera alamat penerima dan pengirimnya
5.	Apakah anda akan menggunakan aplikasi ini untuk pertukaran data dengan rekan kerja?	Iya, tentu saja saya akan menggunakan aplikasi ini sebagai alat pertukaran data. Karena dengan menggunakan aplikasi ini data yang dikirim sudah melalui proses enkripsi sehingga data tersebut hanya dapat dilihat oleh email penerima.

6.	Apakah tampilan dari aplikasi ini cukup jelas?	Sudah cukup jelas, namun ukuran tulisannya terlalu kecil.
7.	Apakah aplikasi ini mudah digunakan?	Sangat mudah digunakan.

4.4 Keterkaitan Hasil Pengujian dengan Kriteria-kriteria Algorita Kriptografi

a. Apakah aplikasi ini sudah berhasil merahasiakan data dari pihak yang tidak berkenan?

Pada Tabel 4. 3 poin No. 1 penulis memperoleh jawaban yaitu melalui aplikasi ini data menjadi terenkripsi sehingga terjamin kerahasiaannya. Hal tersebut berarti aplikasi ini memenuhi kriteria algoritma kriptografi yaitu confidentiality. Gambar dibawah ini merupakan tampilan aplikasi yang menunjukan kerahasiaan isi email terjamin, karena isi email tersebut hanya dapat dibaca melalui inbox yang ada di alamat penerima. Ketika email tersebut tidak dibuka melalui inbox penerima maka yang terbaca adalah kata-kata yang terenkripsi. Berikut tampilan inbox yang telah terenkripsi:

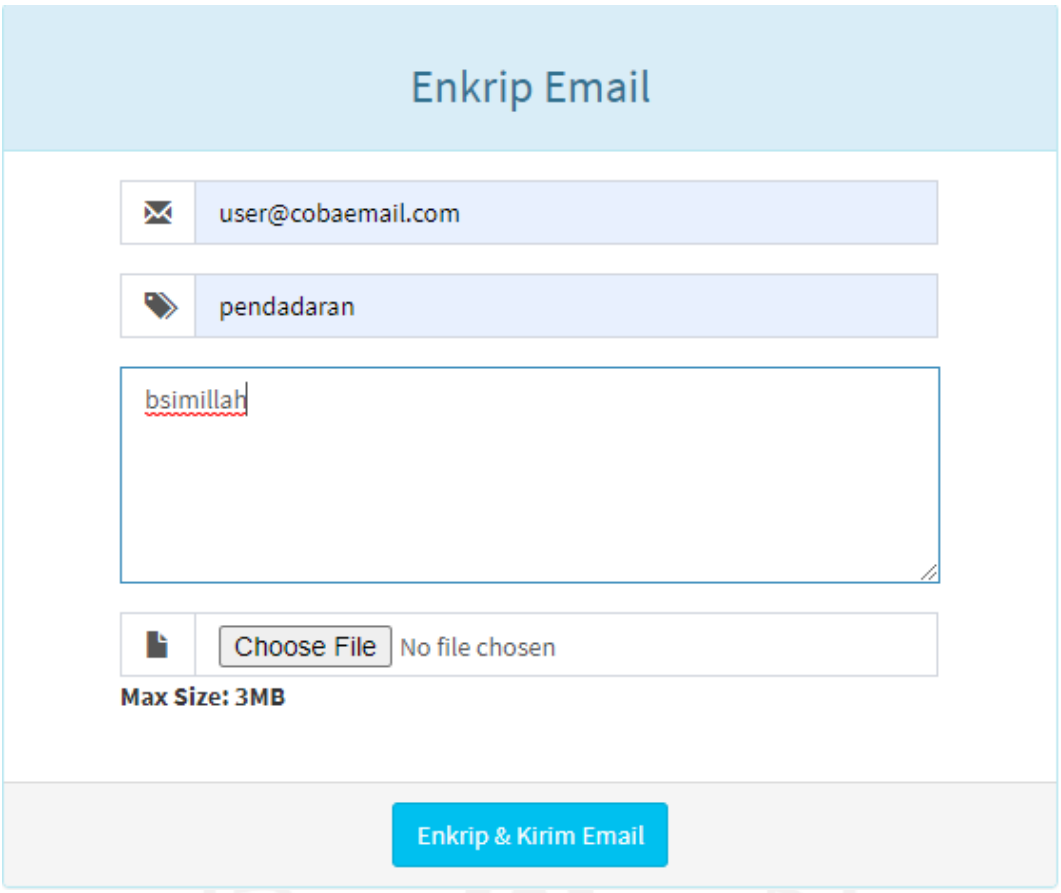


Gambar 4. 19 Hasil Pengujian – Confidentiality

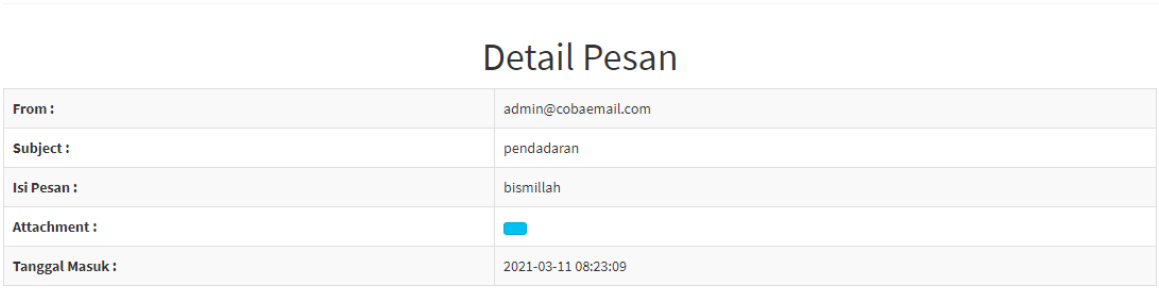
b. Apakah melalui aplikasi ini data pada pesan yang dikirimkan mengalami perubahan saat diterima oleh email tujuan?

Pada Tabel 4. 3 poin No. 2 penulis memperoleh jawaban yaitu Tidak, data yang dikirimkan tidak mengalami perubahan ketika dikirimkan, jumlah data sesuai dengan data yang

pertama kali dikirimkan. Hal tersebut berarti aplikasi ini memenuhi kriteria algoritma kriptografi yaitu integrity. Gambar dibawah ini merupakan bukti bahwa aplikasi sudah memenuhi kriteria algoritma kriptografi integrity:



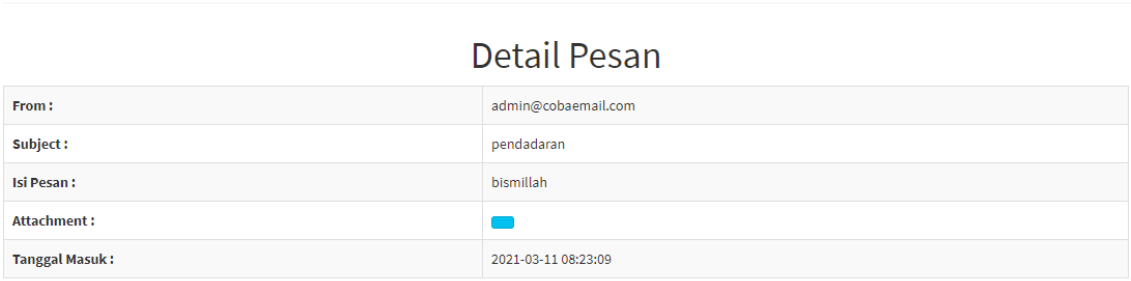
Gambar 4. 20 Pesan yang dikirimkan



Gambar 4. 21 Pesan yang diterima

c. Apakah melalui aplikasi ini data yang dikirimkan terjamin keasliannya?

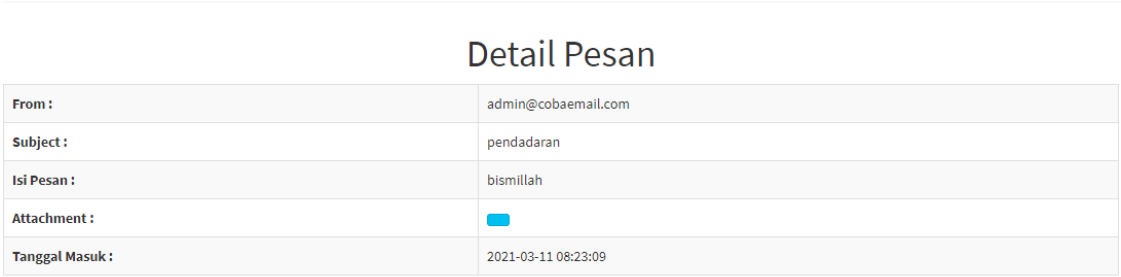
Pada Tabel 4. 3 poin No. 3 penulis memperoleh jawaban yaitu email yang telah di dekripsi hanya dapat dibaca oleh penerima email yang dituju. Keaslian penerima terjamin karna inbox hanya bisa dibuka oleh orang yang sudah terdaftar emailnya. Hal tersebut berarti aplikasi ini memenuhi kriteria algoritma kriptografi yaitu Authentificaton. Gambar dibawah ini merupakan bukti bahwa aplikasi sudah memenuhi kriteria algoritma kriptografi Authentificaton:



Gambar 4. 22 Hasil Pengujian – Authentificaton

d. Apakah ada pihak yang menyangkal bahwa ia telah melakukan pengiriman pesan atau telah menerima pesan?

Pada Tabel 4. 3 poin No. 4 penulis memperoleh jawaban yaitu Tidak ada, karena di setiap email sudah tertera alamat penerima dan pengirimnya. Tidak dapat menyangkal karna disitu tertulis alamat pengirim dan tujuannya adalah ke alamat email penerima. Hal tersebut berarti aplikasi ini memenuhi kriteria algoritma kriptografi yaitu non repudiation. Gambar dibawah ini merupakan bukti bahwa aplikasi sudah memenuhi kriteria algoritma kriptografi non repudiation:



Gambar 4. 23 Hasil Pengujian – Non repudiation

Pada saat responden diminta untuk menggunakan aplikasi ini, responden tidak mengalami kesulitan dan justru responden mengatakan bahwa aplikasi sangat mudah digunakan. Selain itu aplikasi berhasil melakukan enkripsi dan dekripsi tanpa mengurangi isi data. Responden juga memberikan komentar bahwa dengan aplikasi ini juga dapat memberikan jaminan agar tidak ada penyangkalan mengenai pengiriman data ataupun penerimaan data. Berdasarkan hasil wawancara di atas, maka dapat disimpulkan bahwa aplikasi yang dibangun sudah mampu memenuhi tujuan kriptografi. Selain itu aplikasi juga mudah digunakan untuk melakukan pertukan data.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil studi literatur, analisis, perancangan, implementasi, dan pengujian sistem ini, maka kesimpulan yang didapat adalah sebagai berikut:

a. Pada penelitian ini telah berhasil mengimplementasikan algoritma Vigenere Cipher dan Rivest Code 4 pada Aplikasi pengamanan pesan email berbasis web.
b. Aplikasi yang dibangun sudah mampu memenuhi tujuan kriptografi yaitu c*onfidentiality, integrity, authentication* serta *non-repidiation.*

5.2 Saran

Pada aplikasi yang dibangun masih terdapat beberapa temuan yang kedepannya dapat dikembangkan. Berikut merupakan beberapa hal tersebut sebagai berikut:

a. Aplikasi dapat dikembangkan dengan menggunakan fungsi cryptographically secure pseudorandom generator (CSPRNG).
b. Aplikasi dapat dikembangkan dengan menggunakan algoritma lain sehingga kecepatan dalam mengenkripsi dan mendekripsi file menjadi lebih cepat dan lebih aman.

DAFTAR PUSTAKA

Abdullah, D., & Surnihayati. (2017). Pengamanan Email Menggunakan Vigenere Cipher. *Journal of Information System, Applied, Management, Accounting and Research*.

Halim, A., & Budiman. (2015). Implementasi Affine Chiper Dan Rc4 Pada Enkripsi *File* Tunggal. *Seminar Nasional Teknologi dan Informatika*.

Kromodimoeljo, S. (2009). *Teori dan Aplikasi Kripto*g*rafi.* Jakarta: SPK IT Consulting.

Mollin, R. A. (2007). *An Introduction* to *Cryptography Second Edition.* New York: Taylor & Francis Group, LLC.

Muhammad, D. I. (2017, July). Implementasi Kriptografi Vigenere Cipher dengan PHP. *Jurnal Teknologi Informasi, 1*, 12-25.

Munir, R. (2006). *Kripto*g*rafi.* Bandung: Informatika.

Nurcahyo B., N., Zulfian A., & Saiful N., A. (2016). Aplikasi Keamanaan Email Menggunakan Algoritma RC4. *Jurnal SAINTIKOM*.

Paar, C. e. (2010). *Understanding Crypto*g*raphy.* New York: Springer Heidelberg.

Pradipta, A. (2016). Implementasi Metode Caesar Cipher Alphabet Majemuk dalam Kriptografi untuk Pengamanan Informasi. *Indonesian Journal on Networking and Security*.

Suryani, K. N. (2009). Algoritma Rc4 Sebagai Metode Enkripsi. *Makalah If2091 Struktur Diskrit*.

Try, S. A. (2009). Rancang Bangun Email Client Pada Perangkat Mobile. *Jurnal Penelitian*.

LAMPIRAN