PENERAPAN METODE KRIPTOGRAFI RSA-CRT DAN METODE STEGANOGRAFI LSB PADA SISTEM PENGAMANAN PESAN DENGAN MEDIA VIDEO

SKRIPSI

DIANA VITA
NIM. 0701163137

PROGRAM STUDI ILMU KOMPUTER
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI SUMATERA UTARA
MEDAN
2021

PENERAPAN METODE KRIPTOGRAFI RSA-CRT DAN METODE STEGANOGRAFI LSB PADA SISTEM PENGAMANAN PESAN DENGAN MEDIA VIDEO

SKRIPSI

Diajukan Untuk Memenuhi Syarat Mencapai Gelar Sarjana Komputer

DIANA VITA

NIM. 0701163137

PROGRAM STUDI ILMU KOMPUTER

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS ISLAM NEGERI SUMATERA UTARA

MEDAN

2021

PERSETUJUAN SKRIPSI

Hal : Surat Persetujuan Skripsi
Lamp : -

Kepada Yth,
Dekan Fakultas Sains dan Tekmologi
UIN Sumatera Utara Medan

Assalamu'alaikum Wr. Wb.

Setelah membaca, meneliti, memberikan petunjuk dan mengoreksi serta mengatakan perbaikan, maka kami selaku pembimbing berpendapat bahwa skripsi saudara,

Nama	:	Diana Vita
Nomor Induk Mahasiswa	:	0701163137
Program Studi	:	Ilmu Komputer
Judu;	:	Penerapan Metode Kriptografi RSA-CRT dan Metode Steganografi LSB-LCG Pada Sistem Pengamanan Pesan Dengan Media Video

Dapat disetujui untuk segera di*munaqasyah*kan. Atas perhatiannya kami ucapkan terimakasih.

Medan, 30 Agustus 2021

Komisi Pembimbing,

Dosen Pembimbing I,	Dosen Pembimbing II,
(Dr. Mhd. Furqan, S.Si., M.Comp.Sc.)	(Sriani, S.Kom., M.Kom)
NIP. 198008062006041003	NIB. 1100000108

SURAT PERNYATAAN KEASLIAN SKRIPSI

Saya yang bertanda tangan dibawah ini:

Nama	:	Diana Vita
Nomor Induk Mahasiswa	:	0701163137
Program Studi	:	Ilmu Komputer
Judul	:	Penerapan Metode Kriptografi RSA-CRT dan Metode Steganografi LSB Pada Sistem Pengamanan Pesan Dengan Media Video.

Dengan ini menyatakan bahwa skripsi ini adalah hasil karya saya sendiri, kecuali beberapa kutipan dan ringkasan yang masing-masing disebutkan sumbernya. Apabila dikemudian hari ditemukan plagiat dalam skripsi ini maka saya bersedia menerima sanksi pencabutan gelar akademik yang saya peroleh dan sanksi lainnya sesuai dengan peraturan yang berlaku.

Medan, 30 Agustus 2021

Diana Vita
NIM. 070116313

KEMENTERIAN AGAMA REPUBLIK INDONESIA
UNIVERSITAS ISLAM NEGERI SUMATERA UTARA MEDAN
FAKULTAS SAINS DAN TEKNOLOGI
Jl. IAIN No. 1 Medan 20235
Telp. (061) 6615683-6622925, Fax. (061) 6615683
Url: http://saintek.uinsu.ac.id, E-mail: saintek@uinsu.ac.id

PENGESAHAN SKRIPSI

Nomor: B.013/ST/ST.V/PP.01.1/02/2022

Judul	:	Penerapan Metode Kriptografi RSA-CRT Dan Metode Steganografi LSB Pada Sistem Pengamanan Pesan Dengan Media Video
Nama	:	Diana Vita
Nomor Induk Mahasiswa	:	0701163137
Program Studi	:	Ilmu Komputer
Fakultas	:	Sains Dan Teknologi

Telah dipertahankan di hadapan Dewan Penguji Skripsi Program Studi Ilmu Komputer Fakultas Sains dan Teknologi UIN Sumatera Utara Medan dan dinyatakan **LULUS**.

Pada hari/tanggal	:	Kamis, 09 September 2021
Media	:	Zoom Meeting

Tim Ujian Munaqasyah,
Ketua,

Ilka Zufria, M.Kom
NIP. 198506042015031006

Dewan Penguji,

Penguji I,	Penguji II,
Ilka Zufria, M.Kom NIP. 198506042015031006	Rakhmat Kurniawan R, S.T, M.Kom NIP. 198503162015031003
Penguji III,	Penguji IV,
Dr. Mhd Furqan, S.Si, M.Comp. Sc. NIP. 198008062006041003	Sriani, M.Kom NIP. 55201002

Mengesahkan,
Dekan Fakultas Sains dan Teknologi
UIN Sumatera Utara Medan,

Dr. Mhd Syahnan, M.A.
NIP. 196609051991031002

ABSTRAK

Penelitian ini dilakuakn dengan mengkombinasikan kriptografi dan steganografi untuk pengamanan pesan rahasia. Kriptografi digunakan untuk mengenkripsi data atau pesan rahasia, kemudian dilakukan proses Steganografi dengan menyembunyikan pesan yang telah dilakukan proses enkripsi kedalam media video sehingga data atau pesan rahasia lebih terjamin kerahasiaannya. RSA merupakan metode yang dipilih dalam penelitian ini, RSA adalah salah satu algoritma kunci *public* yang kekuatannya terletak pada proses eksponensial dan memfaktoran bilangan menjadi dua bilangan prima dan RSA-CRT ditetapkan untuk meningkatkan efisiansi waktu operasi dikombinasikan dengan metode Steganografi, dimana pesan disisipkan kedalam bit ke 7 pada data piksel yang menyusun *file* gambar secara berurutan. Berdasarkan hasil pengujian proses pengamanan pesan rahasia kedalam media video diperoleh tingkat keberhasilan aplikasi mencapai 100% dan keberhasilkan pengujian berdasarkan aspek *imperceptibility, recovery, dan fidelity*. Sedangkan berdasarkan pengujian ketahanan stego-video diperoleh hasl yang belum bisa memenuhi aspek *robustness*.

Kata Kunci: Kriptografi, Steganografi, RSA-CRT, LSB

ABSTRACT

This research was conducted by combining cryptography and steganography for securing secret messages. Cryptography is used to encrypt secret data or messages, then the Steganography process is carried out by hiding messages that have been encrypted into video media so that confidential data or messages are more guaranteed. RSA is th method chosen in this study, RSA is one of the public key algorithms whose strength lies in the exponential process and factoring numbers into two prime numbers and RSA-CRT is applied to increase operating time efficiency combined with the LSB Steganography method, where messages are inserted into the 7^{th} bit in the pixel data that composes the image file in sequence. Based on the result of testing the process of securing secret messages into video media, the application success rate 100% and the success of the test based on imperceptibility, recovery, and fidelity aspects. Meanwhile, based on the stego-video resistance test, the results were not able to meet the robustness aspect.

Keywords: Cryptography, Steganography, RSA-CRT, LSB

KATA PENGANTAR

Assalamu'alaikum Warahmatullahi Wabarakatuh.

Bismillahirrahmaanirrahiim, puji dan syukur penulis ucapkan kepada Allah Subhana Wata'ala yang telah memberikan rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan penelitian "Penerapan Metode Steganografi LSB Dan Metode Kriptografi RSA-CRT Pada Sistem Pengamanan Pesan Dengan Media Video" sebagai syarat untuk memperoleh gelar Sarjana Ilmu Komputer. Dalam penelitian ini penulis dapat belajar dan menerapkan teori-teori yang pernah didapat selama perkuliahan di Program Studi Ilmu Komputer khususnya mengenai pengolahan citra.

Terima kasih penulis ucapkan kepada semua pihak yang telah membantu penulis ssejak awal hingga akhir penyusunan penelitian ini. Secara khusus penulis ingin mengucapkan terimakasih kepada orang tua penulis yaitu Ayahanda Supriadi dan Ibunda Fitriati Ritonga yang telah memberi moril, material, semangat dan do'a kepada penulis.

Penulis menyadari bahwa tersusunnya skripsi ini atas do'a, perhatian, bantuan, bimbingan, motivasi serta dukungan dari berbagai pihak, sehingga dengan keikhlasan dan kerendahan hati pada kesempatan ini penulis mengucapkan terima kasih yang sebesar-besarnya kepada:

1. Bapak Prof. Dr. Syahrin Harahap, M.A, selaku Rektor Universitas Islam Negeri Sumatera Utara Medan.
2. Bapak Dr. Mhd. Syahnan, M.A, selaku Dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Sumatera Utara Medan.
3. Bapak Ilka Zufria, M.Kom selaku Ketua Program Studi Ilmu Komputer.
4. Bapak Dr.Mhd Furqon, S,Si., M.Comp.Sc selaku dosen pembimbing skripsi I yang telah berkontribusi membantu penulis dalam memberikan ide, saran, kritik, dan bimbingannya kepada penulis selama penulis mengerjakan skripsi ini.

5. Ibu Sriani, S.Kom., M.Kom selaku dosen pembimbing skripsi II yang telah berkontribusi membantu penulis dalam memberikan ide, saran, kritik, dan bimbingannya kepada penulis selama penulis mengerjakan skripsi ini.
6. Bapak Abdul Halim Hasugian, M.Kom selaku dosen Penasehat Akademik penulis.
7. Seluruh dosen pengajar dan staf pegawai Fakultas Sains dan Teknologi Universitas Islam Negeri Sumatera Utara Medan.
8. Seluruh teman-teman Ilmu Komputer khususnya Angkatan 2015, 2016, dan semua pihak yang tidak dapat disebutkan satu persatu yang selama ini telah menjadi teman untuk saling berbagi dan memberi dukungan dari awal hingga selesai penelitian skripsi ini.

Penulis menyadari bahwa masih banyak kekurangan dalam penulisan penelitian ini. Oleh karena itu, penulis mengharapkan kritik dan saran yang bermanfaat terhadap penelitian ini. Akhir kata penulis ucapan terima kasih.

Medan, 30 Agustus 2021

Penulis

Diana Vita

DAFTAR ISI

Halaman

LAMPIRAN-LAMPIRAN

DAFTAR GAMBAR

Gambar **Judul Gambar** **Halaman**

DAFTAR TABEL

DAFTAR LAMPIRAN

Lampiran **Judul Lampiran**

BAB I
PENDAHULUAN

1.1 Latar Belakang

Pada saat ini ada banyak sekali cara untuk mengamankan informasi karena dianggap menjadi perhatian penting. Cara untuk mengamankannya terus diupayakan demi mencegah penyalahgunaan informasi dari pihak ketiga yang tidak memiliki hak untuk mengetahuinya karena sudah jelas hal tersebut sangat merugikan.

Islam adalah agama yang peduli dengan masalah keamanan informasi. Salah satunya adalah kisah Nabi Zulkarnaen 'Alaihissalam dari QS. Surat Al-Kahfi 90-98. Saat itu, Nabi Zulkarnaen ‘Alaihissalam diminta untuk membangun tembok tinggi dan tebal yang tidak bisa dilewati Ya’juj dan Ma’juj untuk melindungi umatnya dari kejahatan. Nabi Zulkarnaen 'Alaihissalam kemudian membangun tembok yang terbuat dari tembaga dan besi panas untuk orang-orang yang membutuhkan keamanan ini. Dalam dunia teknologi, konsep dinding digunakan untuk mencegah pihak yang tidak diinginkan mengakses informasi atau data seseorang (Soediro, 2018).

Salah satu teknik yang digunakan untuk keamanan informasi adalah kriptografi. Kriptografi dapat mengubah pesan rahasia menjadi pesan acak yang tidak berarti (ciphertext), sehingga informasi rahasia hanya dapat dibaca oleh pihak yang berwenang. Kelemahan dari teknik ini, bagaimanapun, adalah bahwa pesan acak yang ditampilkan menimbulkan kecurigaan, memungkinkan penjahat untuk mengutak-atik informasi.

Teknik lain yang dapat digunakan untuk keamanan informasi adalah steganografi. Steganografi merupakan suatu teknik untuk menyembunyikan (embedding) informasi yang memanfaatkan kekurangan dari sistem indera manusia seperti mata dan telinga (A & Painem, 2020). Dengan menggunakan steganografi, informasi rahasia dimasukkan ke dalam media lain seperti teks, gambar digital, suara dan video sehingga orang lain tidak akan mengetahui keberadaan informasi rahasia di media tersebut.

Dengan ditemukannya steganografi, bukan berarti pengiriman pesan benar-benar aman, dan selain memiliki kelebihan dibandingkan kriptografi, steganografi juga memiliki kelemahan. Oleh karena itu, dalam menangani permasalahan tersebut, penelitian ini akan menggunakan kombinasi kriptografi dan steganografi untuk keamanan informasi sehingga menghasilkan sistem keamanan yang aman dan tangguh terhadap serangan. Biasanya teknik yang digunakan yaitu dengan mengenkripsi pesan terlebih dahulu pada proses kriptografi, kemudian menyisipkannya ke media *cover* yang disebut steganografi.

Banyaknya metode dalam melindungi keamanan data pada kriptografi, maka dipilihlah teknik kriptografi RSA. Algoritma ini melakukan pemfaktoran bilangan yang sangat besar, oleh karena itu RSA dianggap aman untuk membangkitkan dua kunci, dipilih dua bilangan prima acak yang besar. Sistem kriptografi RSA dapat dimodifikasi menggunakan teorema *Chinese Remainder Theorem* (CRT) yang disebut RSA-CRT. Pada dasarnya RSA-CRT sama dengan RSA biasa, namun memanfaatkan teorema CRT untuk memperpendek ukuran bit, dan terbukti bahwa sistem kriptografi RSA-CRT memiliki waktu komputasi yang lebih singkat dibandingkan sistem kriptografi RSA biasa, sekitar 4 kali lebih cepat (Landriandani, 2020). Namun di sisi lain, penelitian tentang penambahan CRT pada algoritma RSA menunjukkan bahwa RSA-CRT membutuhkan *resource memory* yang lebih banyak daripada RSA ketika panjang kunci lebih besar dari 2048. Hal ini dikarenakan penggunaan lebih banyak variable pada operasi teorema CRT, oleh karena itu pemanfaatan memori oleh RSA-CRT lebih besar daripada RSA (Landriandani, 2020).

Selain itu, saat ini ada beberapa metode steganografi yang umum digunakan. Salah satunya adalah LSB (*Least Significant Bit*). Cara kerja metode ini adalah dengan menyisipkan pesan pada bit ke-7 dari data *pixel* yang menyusun *file* gambar secara berurutan.

Studi tentang kombinasi kriptografi RSA-CRT dengan steganografi LSB telah dilakukan beberapa kali pada penelitian sebelumnya, antara lain seperti pada penerapan kriptografi RSA-CRT dan steganografi LSB menggunakan *file* audio sebagai data yang disembunyikan. Hasil dalam penelitian tersebut membuat proses

dekripsi menjadi lebih cepat ditinjau dari waktu komputasi algoritma (Abdullah, Abdulameeer, & Hussein, 2015). Selain itu penelitian dilakukan dengan metode yang sama menggunakan citra digital sebagai media nya. *Stegoimage* dari kedua metode tersebut adalah menghasilkan nilai PSNR yang tergolong cukup tinggi dengan nilai lebih dari 57,737 desibels (Nasution, 2017). Oleh karena itu, perlu untuk melakukan suatu pengujian dalam menerapkan kombinasi metode kriptografi dan steganografi pada *file* video.

Berdasarkan latar belakang tersebut maka hal yang akan dilakukan dalam penelitian ini adalah membahas tentang penerapan metode steganografi LSB dikombinasi dengan metode kriptografi RSA-CRT pada sistem pengamanan pesan dengan media video.

Penelitian ini fokus pada kemampuan sistem dalam pengamanan suatu pesan. Sistem akan melakukan proses pengamanan informasi dengan kombinasi kriptografi dengan steganografi.

1.2 Rumusan Masalah

Berdasarkan uraian pada latar belakang yang telah di jabarkan maka dapat dirumuskan permasalahan untuk diselesaikan pada penelitian ini antara lain:

1. Bagaimana menganalisis metode kriptografi RSA-CRT dikombinasi dengan metode steganografi LSB dalam menyisipkan pesan kedalam *file* video?
2. Bagaimana menghasilkan sebuah aplikasi sistem yang dapat menyisipkan dan mengekstraksi pesan dengan metode metode kriptografi RSA-CRT dengan metode steganografi LSB?

1.3 Batasan Masalah

Adapun permasalahan yang harus dibatasi untuk menghindari permasalahan yang meluas maka batasan masalah pada penelitian ini antara lain:

1. Format *file* video yang digunakan dengan ekstensi (*.mp4) dengan ukuran *frame* minimal 30 fps (*frames/second*).

2. Aplikasi yang dibuat mencakup aplikasi pengamanan pesan yang berfungsi untuk menyisipkan dan mengekstraksi informasi dari *file* video.
3. Informasi yang disisipkan hanya dapat menyimpan pesan rahasia berupa teks.

1.4 Tujuan Penelitian

Adapun tujuan penelitian ini antara lain:

1. Mengembangkan aplikasi perangkat lunak dengan implementasi metode kriptografi RSA-CRT dikombinasi dengan metode steganografi LSB dalam menyisipkan pesan kedalam *file* video.
2. Untuk menghasilkan sebuah aplikasi sistem yang dapat menyisipkan dan mengekstraksi pesan dengan menerapkan kombinasi metode kriptografi RSA-CRT dengan metode steganografi LSB.

1.5 Manfaat Penelitian

Adapun manfaat yang diharapkan pada penelitian ini antara lain:

1. Menerapkan dan mengembangkan ilmu pengetahuan tentang metode kriptografi dikombinasi dengan steganografi.
2. Menghasillkan aplikasi perangkat lunak yang dapat mengimplementasikan penggabungan metode kriptografi RSA-CRT dengan metode steganografi LSB.
3. Membantu mengamankan informasi yang bersifat rahasia dan mempersulit orang yang tidak bertanggung jawab dalam mengambil data tersebut.

BAB II
TINJAUAN PUSTAKA

2.1 Pengertian Citra

Citra merupakan fungsi menerus (*continue*) atas intensitas cahaya pada bidang dua dimensi. Sumber cahaya menerangi objek, objek memantulkan kembali seluruh atau sebagian berkas cahaya kemudian ditangkap oleh alat optis atau elektro-optis (Sriani, Triase, & Khairuna., 2017)

Pada dasarnya citra dapat diartikan sebagai representasi visual dari suatu objek. Selain itu, citra juga dapat digunakan sebagai gambar yang mewakili suatu objek tertentu, sehingga dapat menyampaikan pengertian tentang objek tersebut. Inilah sebabnya mengapa dapat digunakan untuk menyampaikan rasa tentang objek tersebut. Jika ingin mendefinisikannya lebih bebas lagi, citra dapat didefinisikan sebagai elemen visual yang dapat dipahami oleh indera penglihatan, apapun bentuk dan fungsinya. Citra digital atau *image*, demikian sebutannya dalam dunia komputasi, adalah representasi visual dari suatu objek tertentu setelah mengalami berbagai transformasi dari berbagai bentuk rangkaian numerik (Andrian, 2019).

2.2 Jenis Citra

Citra merupakan suatu keluaran dari suatu sistem perekaman data yang bersifat optik, analog ataupun digital. Perekaman data citra dapat dibagi menjadi dua yaitu:

2.2.1 Citra Analog

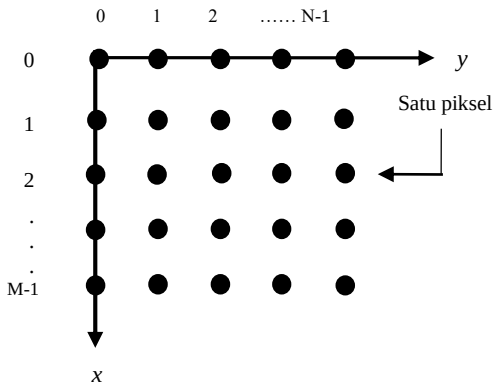
Citra analog adalah gambar kontinu, seperti gambar pada monitor televisi, foto sinar X, foto yang dicetak di atas kertas foto, lukisan, pemandangan alam, CT *scan*, gambar yang direkam pada pita kaset, dan sebagainya. Citra analog tidak dapat dipresentasikan di komputer, sehingga untuk dapat diproses di komputer, terlebih dahulu harus dilakukan proses konversi analog ke digital. Citra analog dihasilkan oleh peralatan analog seperti kamera analog, kamera video, CT *scan*,

sensor sinar-X untuk foto dada, sensor gelombang pendek dalam sistem radar, sensor *ultrasound* dalam sistem *ultrasound*, dll (Suryansah, Habibi, & Awangga, 2020).

2.2.2 Citra Digital

Citra digital adalah representasi dari citra yang diambil oleh mesin dengan metode berdasarkan sampling dan kuantisasi. Sampling menggambarkan ukuran kotak yang disusun dalam baris dan kolom. Dengan kata lain, pengambilan sampel gambar mewakili ukuran piksel (titik) pada gambar, dan kuantisasi mewakili nilai skala abu-abu (graysacle) sesuai dengan jumlah digit biner yang digunakan oleh mesin, yaitu kuantisasi gambar menjelaskan jumlah warna dalam gambar (Suryansah, Habibi, & Awangga, 2020).

Secara umum sistem koordinat yang digunakan untuk merepresentasikan citra digital ditunjukkan Gambar 2.1. Citra digital diwakili oleh matriks M (baris) dan N (kolom), dimana perpotongan antara baris dan kolom disebut piksel. Sebuah piksel memiliki dua parameter, yaitu koordinat (x,y) adalah f(x,y), yang merupakan intensitas atau warna piksel pada titik tersebut (Andono, Sutojo, & Muliono, 2017).



Gambar 2.1 Sistem koordinat yang dipergunakan untuk mewakili citra (Andono, Sutojo, & Muliono, 2017)

Sebuah citra digital dapat ditulis dalam bentuk matriks berikut:

$$f(x,y)=\begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,M-1) \\ f(1,0) & \dots & \dots & f(1,M-1) \\ \dots & \dots & \dots & \dots \\ f(N-1,0) & f(N-1,1) & \dots & f(N-1,M-1) \end{bmatrix}$$

Berdasarkan uraian tersebut, citra digital dapat dituliskan secara matematis sebagai fungsi dari intensitas *f*(x,y), dimana nilai x (baris) dan y (kolom) adalah koordinat posisi dan *f*(x,y) adalah setiap titik (x,y) nilai fungsi yang mewakili intensitas atau tingkat keabuan citra (Andono, Sutojo, & Muliono, 2017).

2.3 Jenis-Jenis Citra Digital

Secara umum terdapat 3 jenis citra yang sering digunakan para peneliti, diantaranya adalah:

2.3.1 Citra Biner

Citra biner atau citra hitam putih (*black and white image*) adalah citra yang setiap pikselnya hanya memiliki 2 kemungkinan nilai, yaitu 0 atau 1. Nilai 0 akan menampilkan hitam, dan nilai 1 akan menampilkan putih. Oleh karena itu, citra jenis ini sering digunakan dalam proses *masking* atau proses segmentasi citra (Hidayatullah, 2017).

Untuk mendapatkan citra biner, kita perlu melakukan ambang batas (tresholding) citra skala abu-abu sesuai dengan ambang batas yang ditentukan. Mengonversi nilai piksel menjadi 1 jika nilai piksel pada gambar skala abu-abu melebihi atau sama dengan ambang batas. Namun, jika nilai piksel kurang dari ambang batas, ubah nilai piksel menjadi 0 (Hidayatullah, 2017).

Contoh citra biner seperti Gambar 2.2.

Gambar 2.2 Citra Biner (pemrogramanmatlab.com)

2.3.2 Citra *Grayscale*

Citra *grayscale* adalah citra yang hanya memiliki 1 kanal, sehingga hanya nilai intensitas yang ditampilkan, yang disebut juga dengan skala keabuan. Karena jenis citra ini hanya memiliki 1 kanal, maka citra *grayscale* memiliki tempat penyimpanan yang lebih hemat. Jenis citra ini disebut juga sebagai citra 8-bit karena setiap nilai pikselnya memerlukan penyimpanan 8-bit. Foto dan gambar hitam putih yang ditampilkan oleh TV hitam putih sebenarnya menggunakan citra *grayscale*, bukan hitam dan putih (Hidayatullah, 2017).

Contoh citra biner seperti Gambar 2.3

Gambar 2.3 Citra *Grayscale* (pemrogramanmatlab.com)

2.3.3 Citra Warna

Citra berwarna atau *true color image* sering disebut sebagai citra berwarna 24-bit karena setiap nilai pikselnya memerlukan penyimpanan sebesar 24-bit. Setiap piksel disetiap kanal memiliki 256 kemungkinan nilai, dengan nilai antara 0-255. Ini membuktikan bahwa 8-bit data diperlukan untuk setiap piksel pada kanal. Karena citra berwarna memiliki 3 kanal, satu piksel pada citra berwarna membutuhkan 3 × 8-bit = 24-bit. Oleh karena itu, citra berwarna disebut sebagai 24-bit *color image*. Dengan kombinasi warna yang ada, citra berwarna memiliki 256×256×256 = 2^{24} = 16.777.216 kemungkinan variasi warna. (Hidayatullah, 2017).

Contoh citra biner seperti Gambar 2.4.

Gambar 2.4 Citra Warna (pemrogramanmatlab.com)

2.4 Pengolahan Citra Digital

Pengolahan citra digital adalah disiplin ilmu yang berkaitan dengan perbaikan kualitas citra (peningkatan kontras, transformasi warna, resolusi citra), transformasi gambar (rotasi, translasi, skala, transformasi geometrik), melakukan seleksi citra ciri (*feature image*) yang optimal untuk tujuan analisis, melakukan proses penarikan informasi atau deskripsi objek atau pengenalan objek yang terkandung pada citra, melakukan kompresi atau reduksi data untuk tujuan penyimpanan data, transmisi data, dan waktu proses data. *Input* dari pengolahan citra adalah citra, sedangkan *output*-nya adalah citra hasil pengolahan (Ali, 2014).

2.5 Video

Video adalah media penyembunyian informasi yang potensial dan banyak digunakan. Data yang dapat disisipkan atau disembunyikan pada media video memiliki kapasitas yang lebih besar dibandingkan media lain seperti teks, gambar dan audio. Selain itu, video tersedia dalam berbagai format sehingga dapat digunakan untuk berbagai keperluan (Malvi & Painem, 2020).

Video adalah teknik untuk menangkap, merekaman, memproses, menyimpana, mentransimisikan, dan merekonstruksi urutan gambar diam dengan menyajikan adegan bergerak secara elektronik. Kata video berasal dari kata latin "*vidi*" atau "*visum*", yang berarti "melihat" atau memiliki daya penglihatan.

Ada dua jenis video, video analog dan video digital. Video analog adalah gambar atau audio yang direkam pada pita dalam bentuk sinyal magnetik. Sedangkan video digital adalah video yang proses perekamannya menggunakan sensor atau komputer dan file yang dihasilkan berupa file atau data. Dalam penelitian ini digunakan video digital.

Video digital pada dasarnya terdiri dari serangkaian *frame*. Serangkaian *frame* tersebut ditampilkan di layar pada kecepatan tertentu, tergantung pada kecepatan *frame rate* yang diberikan (diukur dalam *frame per second*). Jika *frame rate* cukup tinggi, mata manusia tidak dapat menangkap gambar atau *frame*, tetapi menangkapnya sebagai rangkaian yang kontinu/berkelanjutan (video) (Anti, Kridalaksana, & Khairina, 2017).

Setiap *frame* adalah citra digital. Suatu citra digital diwakili oleh sebuah matriks, dan setiap matriks mewakili nilai intensitas, jika I adalah matriks dua dimensi, I (x,y) adalah nilai intensitas yang sesuai dengan posisi baris x dan kolom y dalam matriks tersebut. Titik-titik yang ditempatkan pada gambar sampel disebut *picture elements*, atau lebih sering disebut sebagai piksel. *Pixel* atau piksel (*picture element* / elemen gambar) adalah titik-titik kecil. Gambar apapun yang muncl di layar komputer sebenarnya terdiri dari titik-titik kecil (Anti, Kridalaksana, & Khairina, 2017).

Jika meletakkan beberapa piksel dalam satu baris, maka yang muncul adalah sebuah garis. Jadi semua garis, sehalus apapun yang terlihat pada layar komputer, sebenarnya adalah deretan piksel. Piksel dapat dianggap sebagai sebuah titik, tetapi pada kenyataannya, piksel-piksel lebih seperti persegi panjang kecil yang tingginya tidak proporsional dengan lebarnya (Anti, Kridalaksana, & Khairina, 2017).

a. *Frame Rate*

Keajaiban terjadi ketika mata manusia melihat serangkaian gambar diam yang bersambung. Jika gambar tersebut diputar dengan cepat maka terlihat sebuah pergerakan halus, yang merupakan prinsip dasar film, video dan animasi. Jumlah gambar yang terlihat per detik disebut *frame rate*. *Frame rate* yang diperlukan minimal 10 fps (*frame rate per second*) agar menghasilkan pergerakan gambar yang halus. Film-film yang kita tonton di 11 gedung bioskop adalah film yang ditayangkan dengan *frame rate* 24 fps, sedangkan video yang kita lihat di TV sekitar 30 fps (tepatnya 29.97 fps) untuk negara yang memakai format standar NTSC (*National Television Standards Comitte*) yaitu Amerika Serikat, Jepang, Kanada, Meksiko dan Korea. Untuk negara Indonesia, Inggris, Australia, Eropa dan China format video standar yang digunakan adalah PAL (*Phase Alternate Line*) dengan *frame rate* sebesar 25 fps. Sedangkan negara Perancis. Timur Tengah dan Afrika menggunakan format video standar SECAM (*Sequential Couleur Avec Memoire*) dengan *frame rate* sebesar 25 fps (Anti, Kridalaksana, & Khairina, 2017).

b. Resolusi dan *Frame Size*

Lebar dan tinggi *frame* video disebut *frame size* dan diukur dalam satuan *pixel*, misalnya video dengan *framei size* 640x480 *pixel*. Dalam dunia video digital, *frame size* disebut juga dengan resolusi. Semakin tinggi resolusi gambar, semakin banyak informasi yang dimuat, yang berarti semakin besar pula kebutuhan memori untuk membaca informasi. Misalnya untuk format PAL DI/DV adalah 720x576 *pixel*, format NTSC DV 720x480 *pixel* dan format PAL VCD/VHS (MPEG1) adalah 352x288 *pixel* sedangkan format NTSC VCD adalah 320x240 *pixel* (Anti, Kridalaksana, & Khairina, 2017).

c. Kedalaman *Pixel*

Kedalaman bit menentukan jumlah bit yang dipakai untuk mewakili setiap piksel pada sebuah *frame* & dinyatakan menggunakan *bit/pixel*. Semakin banyak bit yang dipakai untuk mewakili suatu piksel, maka semakin meningkat kualitasnya. Kedalam *pixel* paling rendah terdapat pada *binary-value image* yang hanya menggunakan bit untuk tiap *pixel*, maka hanya ada dua kemungkinan untuk tiap *pixel*, yaitu 0 (hitam) atau 1 (putih). Nilai 1 *byte* (8 bit) per *pixel*, diperoleh 28 atau 256 tingkat intensitas. Kemudian video kedalaman 16 bit sering disebut video *high color*, dimana setiap pikselnya diwakili oleh 2 *byte* atau 16 bit dengan 65.356 warna. Selama pembentukan bit, nilai merah dan biru terjadi di kanan dan kiri 5 bit. Komponen hijau memiliki 5 bit ditambah 1 bit ekstra, pemilihan komponen hijau dengan garis 6 bit karena penglihatan manusia lebih sensitif terhadap warna hijau. Oleh karena itu, semakin rendah jumlah bit yang digunakan per *pixel*, maka semakin rendah kualitas gambarnya (Anti, Kridalaksana, & Khairina, 2017).

2.6 Kriptografi

2.6.1 Pengertian Kriptografi

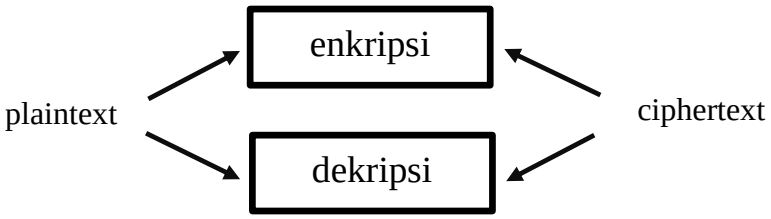
Kriptografi (atau kriptologi; berasal dari kata Yunani *kryptos*, "tersembunyi, rahasia", dan *graphein*, "menulis", atau *logi*, "ilmu") adalah ilmu dan seni pengamanan pesan.

Kriptografi adalah ilmu tentang teknik enkripsi dimana data diacak menggunakan kunci enkripsi menjadi sesuatu yang sulit dibaca oleh seseorang tanpa kunci dekripsi (Kromodimoeljo, 2010). Dalam kriptografi terdapat dua proses yaitu enkripsi dan dekripsi. Pesan yang akan dienkripsi disebut *plaintext* (teks biasa). Disebut demikian karena informasi ini mudah dibaca dan dipahami oleh siapa saja. Algoritma yang digunakan untuk mengenkripsi dan mendekripsi *plaintext* melibatkan penggunaan beberapa bentuk kunci. Pesan *plaintext* yang terenkripsi (atau dikodekan) disebut *chipertext* (teks sandi) (Munir, 2019).

Saat melakukan enkripsi dan dekripsi, diperlukan aturan khusus agar penerima dapat memahami pesan yang dimodifikasi oleh pengirim. Aturan atau algoritma yang dimaksud adalah algoritma enkripsi. Algortirma enkripsi yang baik harus dapat membingungkan (sulit untuk membuat ulang pesan yang sebenarnya tanpa algoritma dekripsi) dan membekukan (menghilangkan karakteristik dari pesan yang asli). Selanjutnya, algoritma enkripsi yang baik dan dapat diandalkan adalah yang kekuatan keamanannya terletak pada kuncinya, bukan pada rahasia dari algoritma itu sendiri.

Algoritma enkripsi dapat dibedakan berdasarkan beberapa hal berikut:

- Menurut era: klasik dan modern
- Berdasarkan kerahasiaan kunci: kunci rahasia dan kunci *public*
- Berdasarkan kesamaan kunci enkripsi dan dekripsi: kunci simetris dan kunci asimetris.

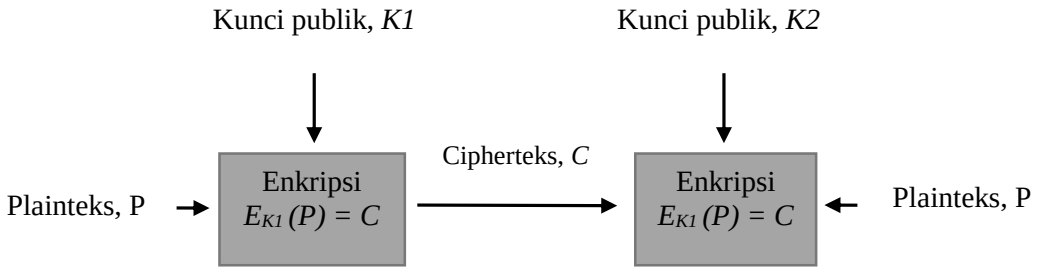


Gambar 2.5 Diagram enkripsi-dekripsi secara umum

2.6.2 Kriptografi Kunci Asimetris

Ada dua model algoritma enkripsi yang menggunakan kunci, yaitu kunci simetris dan kunci asimetris. Enkripsi kunci simetris adalah kriptografi yang kunci enkripsi dank unci dekripsinya menggunakan kunci yang sama, oleh karena itu disebut kriptografi simetris. Sedangkan kriptografi kunci asimetris adalah kriptografi yang diimplementasikan sedemikian rupa sehingga kunci yang digunakan untuk enkripsi berbeda dengan kunci yang digunakan untuk dekripsi. Kriptografi asimetris juga dikenal sebagai kriptografi kunci *public*. Disebut demikian karena kunci yang digunakan untuk enkripsi dapat disebarluaskan ke public sedangkan kunci yang digunakan untuk dekripsi hanya disimpan oleh orang

yang tepat. Pengirim melakukan enkripsi menggunakan kunci *public*, sedangkan penerima pesan dapat mendekripsi jika dan hanya jika mereka mengetahui kunci privat. Beberapa contoh algoritma kriptografi kunci asimetris antara lain RSA, LUC, DSA, Electif Curve Elgamal (Landriandani, 2020).



Gambar 2.6 Skema Kunci Asimetris (Munir, 2019)

2.6.3 Algortima RSA

RSA adalah algoritma kriptografi kunci publik atau kunci asimetrik (kunci enkripsi dan kunci dekripsi yang berbeda) yang tidak memerlukan saluran yang aman untuk distribusi kunci. RSA telah ditemukan oleh tiga peneliti: Ronald Linn Rivest, Adi Shamir, dan Len Adleman 1977. Keamanan algoritma RSA terletak pada sulitnya memfaktorkan bilangan yang besar menjadi faktor-faktor prima (Arief & Saputra, 2016). Kunci privat dan kunci publik yang digunakan berupa bilangan bulat. Kunci privat dibangkitkan dari beberapa bilangan prima dan kunci publiknya (Munir, 2019).

Untuk mendapatkan kunci privat, harus terlebih dahulu memfaktorkan bilangan non prima ke dalam faktor primanya. Tentu saja ini sangat sulit, sehingga keamanan algoritma RSA tergolong sangat baik. Semakin besar bilangan prima semakin sulit memfaktorkannya (Landriandani, 2020).

Algoritma RSA dibagi menjadi 3 langkah (Arief & Saputra, 2016):

1. Pembangkit Kunci
    a. Pilih 2 bilangan prima besar untuk nilai *p* dan *q*
    b. Hitung nilai modulus n = $p \times q$ (1)

c. Hitung menggunakan fungsi Euler $\varphi(n) = (p-1)x(q-1)$ (2)

d. Piilih nilai *integer e* acak sebagai kunci publik, dengan syarat memenuhi *Greater Common Divisor* (GCD) $(e, \varphi(n)) = 1, 1 < e < \varphi(n)$ (3)

e. Hitung kunci privat d sehingga *d x e* = $1 = (mod\ \varphi(n))$ (4)

2. Enkripsi

$C = M^e\ mod\ n$ (5)

3. Dekripsi

$M = C^d\ mod\ n$ (6)

2.6.4 CRT (*Chinese Remainder Theorem*)

CRT (*Chinese Remainder Theorem*) merupakan suatu algoritma untuk mengurangi perhitungan aritmatika modular dengan modulus besar agar dapat melakukan perhitungan yang sama untuk setiap faktor modulus (Arief & Saputra, 2016).

Terdapat bilangan-bilangan $n_1, n_2, \dots, n_k$ adalah bilangan bulat positf yang berpasangan relatif prima. Contohnya: FPB$(n_i, n_j) = 1$ dimana $i \neq j$. Juga, $n = n_1, n_2, \dots, n_k$ dan $x_1, x_2, \dots, x_k$ adalah bilangan bulat. Maka sistem kongruen (Arief & Saputra, 2016):

$$x \equiv x_1 mod\ n_1,$$
$$x \equiv x_2 mod\ n_2,$$
...
$$x \equiv x_k mod\ n_k \quad (7)$$

Memiliki solusi yang simultan pada semua kongruen dan dua solusi apapun adalah saling kongruen modulo. Selain itu terdapat tepatnya satu solusi antara 0 dan *n*-1. Solusi unik dari kongruen simultan memenuhi $0 \leq x \leq n$ dapat dihitung dengan rumus (Arief & Saputra, 2016):

$$x = \left(\sum_{i=1}^{k} x_i r_i s_i\right) mod\ n$$
$$= (x_i r_i s_i + x_2 r_2 s_2 + \cdots + x_k r_k s_k) mod\ n \quad (8)$$

Dimana $r_i = \frac{n}{n_i} dan\ s_i = r_i^{-1} mod\ n_1$ untuk $i = 1, 2, \dots, k$.

Jika bilangan bulat $n_1, n_2, \dots, n_k$ adalah pasangan relatif prima, dan $n = n_1, n_2, \dots, n_k$, makaitu berlaku untuk semua bilangan bulat a, b. Dimana $a \equiv b$ mod n jika dan hanya jika $a \equiv b \, mod \, n_i$ untuk setiap $i = 1,2, \dots, k$ (Arief & Saputra, 2016).

Sebagai konsekuensi dai CRT, setiap bilangan bulat positif $a < n$ dapat direpresentasikan secara unik sebagai k-tuple $[a_i, a_2, \dots, a_k]$ dan sebaliknya. Dimana a_i menunjukkan sisa / residu $a \, mod \, n_i$ untuk masing-masing $i = 1, 2, \dots, k$. Konversi a ke sistem residu yang didefenisikan oleh $n_i, n_2, \dots, n_k$ dilakukan secara sederhana dengan reduksi modular $a \, mod \, n^i$. Konversi balik dari representasi sisa menjadi "angka-angka standar" adalah lebih sulit seperti yang dibutuhkan dalam kalkulasi pada rumus (Arief & Saputra, 2016).

Keuntungan dasar menggunakan *Chinese Remainder Theorem* adalah memungkinkan untuk membagi modulo eksponensial yang besar ke dalam dua eksponensial yang jauh lebih kecil, satu melebihi p dan satu melebihi q. Kedua modulo ini adalah faktor utama dari n yang diketahui (Arief & Saputra, 2016).

2.6.5 Algoritma RSA dan CRT (RSA-CRT)

Kriptosistem RSA dapat dimodifikasi menggunakan teorema CRT yang disebut dengan RSA CRT. Kriptosistem RSA CRT telah terbukti kira-kira empat kali lebih cepat dan lebih pendek dalam waktu komputasi dibandingkan kriptosistem RSA biasa (Arief & Saputra, 2016).

Algoritma RSA CRT dibagi menjadi tiga langkah:

1. Pembangkit Kunci RSA-CRT

RSA CRT pada dasarnya sama dengan RSA biasa, tetapi memanfaatkan teorema CRT untuk mengurangi ukuran bit eksponen dekripsi *d* dengan menyembunyikan *d* dalam sistem kongruen sehingga mempercepat waktu dekripsi. Di bawah ini adalah algortima pembangkit kunci RSA-CRT (Arief & Saputra, 2016):

a. Bangkitkan bilangan prima besar *p* dan *q*
b. Lihat rumus (1).
c. Lihat rumus (2).

d. Lihat rumus (3).

e. Lihat rumus (4).

f. dP = d mod (p-1) (9)

g. dQ = d mod (q-1) (10)

h. qInv = q^{-1} pada Z_p (11)

i. K_{publik} = (e, n), K_{privat} = (dP, dQ, qInv, p, q) (12)

2. Enkripsi RSA-CRT

Kunci *public* RSA-CRT sama dengan sistem RSA, yaitu (*e*, *n*) sehingga algoritma enkripsi tidak berubah. Artinya ia menggunakan fungsi eksponensial modular yaitu lihat pada rumus (5) (Arief & Saputra, 2016).

3. Dekripsi RSA-CRT

D adalah fungsi dekripsi

$D_{d,N}: C \rightarrow M$ (13)

Bahwasanya, $C \in C$ peta untuk $M \in M$, dengan kunci privat (*d*, *N*), sehingga $M \equiv C^d (mod\ N)$.

Penerima B melakukan hal berikut:

- Memperoleh kunci privat (*d*, *N*).
- Menerima *chipertext C* dari Pengirim A.
- Dekripsi-kan *chipertext C* dengan menggunakan rumus:
- $M = C^d\ mod\ N$ (14)

2.7 Steganografi

2.7.1 Pengertian dan Manfaat Steganografi

Teknik steganografi ini telah ada sejak 4000 tahun lalu di kota Menet Khufu, Mesir. Awalnya adalah penggunaan *hieroglyphic*, yakni menulis dengan simbol dalam bentuk gambar. Ahli tulis menggunakan tulisan Mesir kuno ini untuk menceritakan kehidupan tuannya. Tulisan tersebut dijadikan ide untuk membuat pesan rahasia saat ini. Oleh karena itu, tulisan Mesir kuno yang menggunakan gambar dianggap sebagai steganografi pertama di dunia (Syawal, Fikriansyah, &

Agani, 2016).Tidak hanya bangsa Mesir, tetapi bangsa lain juga telah menggunakan teknik steganografi di masa lalu.

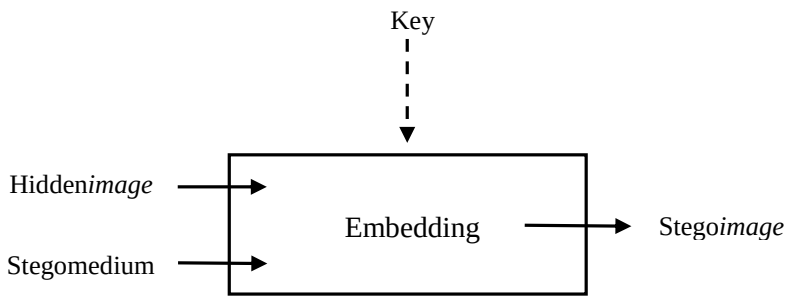
Steganografi merupakan seni untuk menyembunyikan suatu pesan dalam media digital agar orang lain tidak menyadari bahwa ada suatu pesan di dalam media tersebut. Steganografi berasal dari bahasa Yunani *steganos* (artinya "tersembunyi/terselubung") dan *graphein* ("menulis"), sehingga kurang lebih berarti "tulisan terselubung" (Andono, Sutojo, & Muliono, 2017).

Steganografi adalah pedang bermata dua yang dapat digunakan untuk alasan yang baik, tetapi juga dapat digunakan sebagai sarana kejahatan. Steganografi juga dapat digunakan sebagai cara untuk menyembunyikan informasi rahasia dan melindunginya dari pencurian dan akses yang tidak berhak untuk mengetahuinya. Steganografi juga dapat digunakan oleh teroris untuk berkomunikasi satu sama lain (Supardi, 2017).

2.7.2 Proses Steganografi

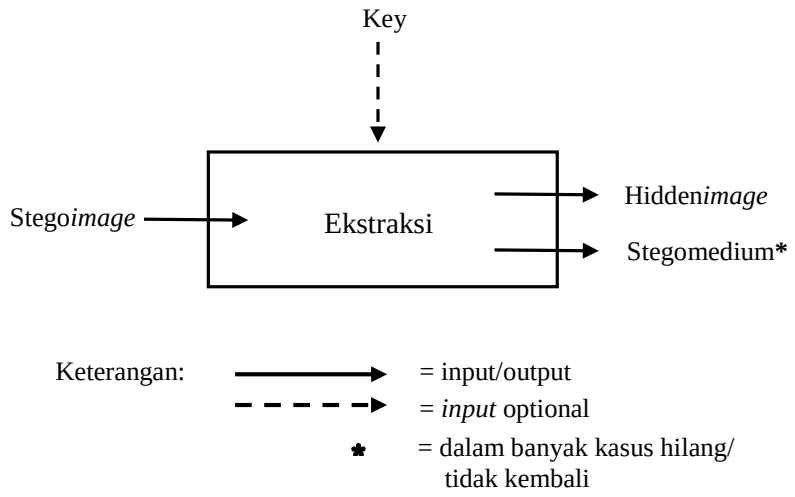
Secara umum, ada dua proses untuk steganografi, yaitu *embedding* dan *Ekstraksi*.

1. *Embedding* adalah proses menyembunyikan suatu pesan. Gambar 2.7, menunjukkan proses *embedding hiddenimage* yang hendak disembunyikan secara rahasia ke dalam *stegomedium* sebagai media penyimpanan, dengan memasukkan kunci tertentu (*key*), sehingga dihasilkan media dengan data tersembunyi di dalamnya (*stegoimage*).



Gambar 2.7 Embedding Citra

2. Ekstraksi, adalah proses untuk mengektraksi atau mengeluarkan pesan yang disembunyikan.



Gambar 2.8 Ekstraksi Citra

Pada Gambar 2.8 dilakukan proses ekstraksi pada *stegoimage* dengan memasukkan *key* yang sama sehingga didapatkan kembali *hiddenimage*. Kemudian pada kebanyakan teknik steganografi, ekstraksi pesan tidak akan mengembalikan *stegomedium* awal persis sama dengan *stegomedium* sesudah dilakukan ekstraksi, bahkan sebagian besar mengalami kehilangan.

2.7.3 Media Penyisipan

Media penyisipan adalah media untuk menyimpan pesan rahasia. Berikut adalah beberapa contoh media penyisipan pesan rahasia yang digunakan dalam teknik steganografi:

1. Teks

   Teks dari algoritma steganografi yang menggunakan teks sebagai media penyisipan biasanya digunakan pada teknik *Natural Language Processing* (NLP), sehingga teks yang disisipkan sebagai pesan rahasia tidak mencurigakan bagi orang yang melihatnya.

2. Audio

   Format ini biasanya dipilih karena format ini adalah file yang berukuran relatif besar. Sehingga dapat menampung sejumlah besar pesan rahasia.

3. Citra

   Format ini adalah salah satu format *file* yang banyak dipertukarkan di dunia *internet*. Alasan lainnya adalah karena banyaknya algoritma *steganography* yang tersedia untuk media penampung yang berupa format citra.

4. Video

   Format ini memang merupakan format dengan ukuran *file* yang relatif besar, namun jarang digunakan karena ukurannya terlalu besar. Akibatnya kurang praktis dan juga kurangnya algoritma yang mendukung format ini.

2.7.4 Kriteria Steganografi yang Bagus

Menyembunyikan data rahasia ke dalam media yang disisipkan akan mengubah kualitas media. Oleh karena itu, saat membuat metode penyembunyian pesan, kriteria berikut harus diperhatikan (Andono, Sutojo, & Muliono, 2017):

1. *Fidelity*. Kualitas citra penampung tidak banyak berubah. Setelah menambahkan data rahasia, citra steganografi masih terlihat bagus. Pengamat tidak menyadari bahwa ada data rahasia dalam citra.
2. *Robustness* (Ketangguhan). Data tersembunyi harus mampu menahan manipulasi yang dilakukan pada media penampung (seperti mengubah kontras, mempertajam, mengompresi, memutar/rotasi, perbesaran gambar, memotong (*cropping*), enkripsi, dll). Data yang disembunyikan tidak rusak saat dilakukan operasi pengolahan citra.
3. *Recovery* (Pulih). Data yang tersembunyi harus dapat diungkap kembali. Karena tujuan steganografi adalah menyembunyikan data, data rahasia di dalam citra penampung harus dapat diambil kembali.

2.7.5 Metode *Least Significant Bit* (LSB)

Berikut langkah-langkah penyisipan pesan menggunakan metode LSB: (Andono, Sutojo, & Muliono, 2017)

1. Ubah pesan menjadi bilangan biner 8-bit
2. Masukkan bit pesan ke dalam piksel citra sebagai berikut, dimulai dari MSB (*Most Significant Bit*):
    - *If* (bit pesan = ‘1’ dan piksel citra = ‘ganjil’) atau (bit pesan = ‘0’ dan piksel citra = ‘genap’), maka:
      Piksel citra (tetap)
    - *If* (bit pesan = ‘0’ dan piksel citra = ‘genap’), maka:
      Piksel citra baru = piksel citra lama + 1
    - *If* (bit pesan = ‘0’ dan piksel citra = ‘ganjil’), maka:
      Piksel citra baru = piksel citra lama – 1
3. Simpan citra baru sebagai *stegoimage*.

Langkah-langkah mengekstrak pesan dari metode LSB adalah sebagai berikut:

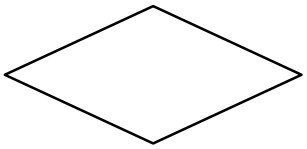
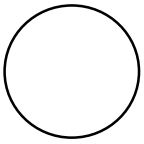
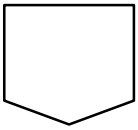
1. Ambil pesan dari *stegoimage*:
   Jika piksel stego = ‘Genap’, maka bit pesan = ‘0’.
   Jika piksel stego = ‘Ganjil’, maka bit pesan = ‘1’.
2. Urutkan bit-bit pesan dari MSB menjadi bilangan biner 8-bit.
3. Konversikan bilangan biner 8-bit ke bilangan desimal.

2.8 *Flowchart*

Untuk menggambarkan suatu algoritma yang terstruktur dan mudah dipahami oleh orang lain, terutama *programmer* yang bertangung jawab untuk mengimplementasikan program memerlukan diagram alir (*flowchart*). Sebuah *flowchart* menggambarkan urutan logis dari proses pemecahan masalah, sehingga *flowchart* adalah langkah-langkah penyelesaian masalah yang ditulis dalam simbol tertentu.

Tabel 2.1 merupakan simbol-simbol yang digunakan untuk menggambarkan algoritma dalam bentuk diagram alir dan kegunaan dari simbol-simbol yang bersangkutan (Sitorus, 2015).

Tabel 2.1 Simbol-Simbol *Flowchart*

No.	Simbol	Nama	Fungsi
1		Terminal	Menyatakan permulaan atau akhir suatu program
2		*Input/Output*	Menyatakan proses *input* atau *output* tanpa tergantung jenis peralatannya
3		*Process*	Menyatakan suatu tindakan (proses) yang dilakukan oleh computer
4		*Decision*	Menunjukkan suatu kondisi tertentu yang akan menghasilkan dua kemungkinan jawaban ya/tidak
5		*Connector*	Menyatakan sambungan dari proses ke proses lainnya dalam halaman yang sama
6		*Offline Connector*	Menyatakan sambungan dari proses ke proses lainnya dalam halaman yang berbeda
7		*Predefined Process*	Menyatakan penyediaan tempat penyimpanan suatu pengolahan untuk memberi harga awal
8		*Punched Card*	Menyatakan *input* berasal dari kartu atau *output* ditulis ke kartu

2.9 *Python*

Python adalah Bahasa pemrograman multiguna yang dibuat oleh Guido van Rossum pada tahun 1991. Bahasa ini dirancang untuk membuat kode mudah dibaca.

Python adalah Bahasa pemrograman interpretative multiguna yang diinterpretasikan dan dirancangan dengan berfokus pada tingkat keterbacaan kode. Python diklaim sebagai bahasa lengkap dengan sintaks kode yang sangat jelas dan fungsi perpustakaan standar yang besar dan komprehensif(Syahrudin & Kurniawan, 2018).

Python mendukung beberapa paradigma pemrograman, utamanya; namun tidak dibatasi; pada pemrograman berorientasi objek, pemrograman imperative, dan pemrograman fungsional. Salah satu fitur yang tersedia di *Python* adalah sebagai Bahasa pemrograman yang dinamis dilengkapi dengan manajemen memori otomatis. Seperti bahasa pemrograman dinamis lainnya, umumnya *Python* digunakan sebagai bahasa skrip, meskipun pada praktiknya penggunaan bahasa ini lebih luas mencakup konteks pemanfaatan yang umumnya tidak dilakukan dengan menggunakan Bahasa skrip. *Python* juga digunakan untuk berbagai keperluan platform sistem operasi dan pengembangan perangkat lunak (Syahrudin & Kurniawan, 2018).

2.10 Peneliti Lain Yang Terkait

Dalam penyusunan skripsi ini, penulis sedikit banyak terinspirasi dan mereferensi dari penelitian-penelitian sebelumnya yang berkaitan dengan latar belakang masalah pada skripsi ini. Tujuannya untuk memperkuat penalaran dan rasionalitas keterlibatan sejumlah variabel pada penelitian ini. Adapun penelitian yang berhubungan dengan skripsi ini antara lain yaitu:

Penelitian yang dilakukan oleh Niti Ravika Nasution, 2017 dengan judul "Kombinasi RSA-CRT dengan *Random* LSB untuk Keamanan Data di Kanwil Kementrian Agama Prov. Sumatera Utara" (Nasution, 2017). Penelitian ini menggunakan *Crytographic Algorithms Rivest Shamir Adleman Chinese Remainder Theoremi* (RSA-CRT) dan teknik *Random Least Significant Bits* (LSB). RSA-CRT untuk pengamanan pesan ke dalam media gambar. Hasil yang diperoleh

dari penelitian tersebut dikatakan bahwa penggunaan algoritma kriptografi RSA-CRT lebih baik dari RSA biasa dikarenakan waktu komputasi lebih cepat karena dapat direduksi dengan mengimplementasikan *Chinese Remainder Theorem* (CRT).

Selanjutnya penelitian dilakukan oleh Alyiza Dwi Ningtyas, 2017 yang berjudul "Implementasi Hybrid Cryptosystem Algoritma RSA-CRT Dan Algoritma RC4+ Dalam Pengamanan *File* Teks" (Ningtyas, 2017). Penelitian ini bertujuan untuk mengkombinasi RC4+ dengan RSA-CRT. Disebutkan bahwa untuk mengamankan kunci RC4+ menggunakan kunci *public* RSA-CRT dapat menjadikan kunci RC4+ berupa angka-angka yang sulit untuk diketahui, sehingga mampu meningkatkan tingkat keamanan kunci pesan (Ningtyas, 2017).

Penelitian selanjutnya dilakukan oleh Zaimah Panjaitan, Khairi Ibnuutama, dan M. Gilag Suryanata, 2019 dengan judul " Penggunaan *Chinese Remainder Theorem* (CRT) pada Algoritma RSA" (Panjaitan, Ibnutama, & Suryanata, 2019). Dalam penelitian ini dijelaskan bahwa *Chinese Remainder Theorem* (CRT) merupakan suatu teori matematis yang berfungsi untuk membantu menyederhanakan eksponensiasi modular yang berukuran besar. Algoritma RSA merupakan algoritma asimetris yang banyak digunakan untuk keamanan data. Algoritma RSA memiliki kekurangan yaitu sulitnya melakukan proses dekripsi pada algoritma ini yang disebabkan nilai eksponensiasi pada proses dekripsi relatif sangat besar. Karena adalanya kekurangan ini, dilakukan penggunaan CRT saat dekripsi pesan dengan algoritma RSA untuk membantu menyederhanakan nilai eksponen yang berukuran besar. Uji program menunjukkan bahwa tanpa penggunaan CRT, dekripsi pesan pada algoritma RSA tidak dapat dilakukan. Program secara otomatis memotong karakter *chipertext* sehingga hasil dekripsi tidak sama dengan pesan aslinya. Sebaliknya dengan CRT, nilai *chipertext* yang besar dapat dikembalikan sesuai pesan aslinya (Panjaitan, Ibnutama, & Suryanata, 2019).

Penelitian terakhir yang dilakukan oleh M. Farid Landriandani, 2020 dengan judul "Sistem Pengamanan Pesan Dengan Metode Kriptografi RSA-CRT dan Metode Steganografi *Linear Congruential Generator* Pada Media Citra

Digital" (Landriandani, 2020). Penelitian ini bertujuan untuk mengembangkan perangkat lunak dan menguji apakah proses dari penambahan CRT (*Chinese Remainder Theorem*) pada algoritma kriptograi RSA mempengaruhi 3 aspek pengujian saat dikombinasikan dengan steganografi *random* LSB. Maka dari hasil pengujian yang telah dilakukan dengan 3 aspek pengujian, penggunaan teorema CRT dalam algoritma RSA hanya berpengaruh pada proses dekripsi, dan tidak ada pengaruh pada proses enkripsi (Landriandani, 2020).

BAB III
METODE PENELITIAN

3.1 Waktu Dan Jadwal Pelaksanaan Penelitian

Adapun waktu dan jadwal pelaksanaan penelitian yaitu:

Tabel 3.1 Waktu dan jadwal pelaksanaan penelitian

No.	Waktu	Jadwal Penelitian				
		Mei	Juni	Juli	Agust	Sept
1.	Pengumpulan Data					
2.	Observasi					
3.	Analisis Data dan Perancangan Sistem					
4.	Pengujian Sistem					

3.2 Alat dan Bahan Penelitian

Pada saat penelitian, penulis membutuhkan beberapa alat dan bahan dalam melakukan penelitian untuk mendukung pengumpulan data dan penyelesaian penelitian yang dilakukan. Alat dan bahan yang digunakan dalam penelitian ini yaitu sebagao berikut.

3.2.1 Bahan Penelitian

Penelitian ini menggunakan bahan penelitian yaitu *file* video dengan format mp4 yang berdurasi dengan batas maksimal 20 Mb.

3.2.2 Alat Penelitian

Alat yang digunakan dalam penelitian ini terdiri dari perangkat keras dan perangkat lunak yaitu sebagai berikut.

Perangkat keras yang dibuat dikembangkan pada perangkat keras Laptop Acer Aspire 3 A314-21-4644, dengan spesifikasi:

a. CPU: AMD Dual-Core Processor A4-9120E
b. RAM: 4 GB DDR 4 Memory
c. Storage: 1000 GB HDD

Adapun perangkat lunak yang digunakan pada saat penelitian ini adalah sebagai berikut:

a. *Operating System Microsoft Windows* 10 64 bit
b. *Phyton*

3.3 Prosedur Kerja

Prosedur kerja dalam peelitian ini dilakukan untuk menguraikan semua tahapan-tahapan kegiatan yang dilaksanakan pada waktu penelitian agar sesuai dengan tujuan yang telah dibuat. Tahapan yang dilakukan adalah sebagai berikut:

3.3.1 Pengumpulan Data

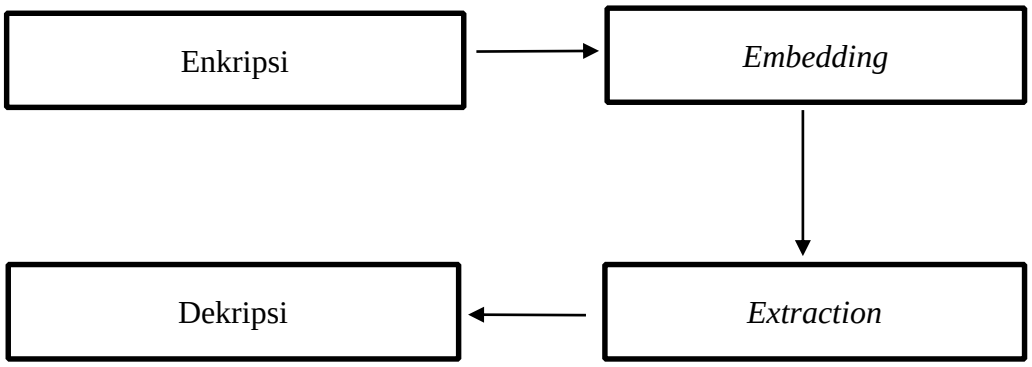
Teknik pengumpulan data yang akan digunakan adalah:

1. *Field Research* dilakukan dengan meneliti langsung ke objek yang akan diteliti dengan cara observasi. Observasi yaitu melakukan pengamatan secara langsung terhadap beberapa perangkat lunak untuk enkripsi pesan.
2. *Library Researc* (Penelitian Kepustakaan) yaitu dengan mencari referensi dari buku perpustakaan dan juga dari Internet, yang berhubungan dengan objek yang diteliti, dimana referensi tersebut dijadikan landasan teori dalam penyelesaian penelitian ini.
3. Pengumpulan data dilakukan dengan cara mengambil sebanyak 2 *file* video dengan format (*.mp4) dengan ketentuan maksimal 20 Mb.

3.3.2 Perencanaan

Dalam tahapan perencanaan akan disajikan langkah-langkah yang akan dilakukan dalam merealisasikan sistem yang dibuat. Diagram blok diperlukan agar

dapat mempresentasikan sistem secara umum. Diagram blok sistem yang dirancang pada penelitian memiliki tahapan-tahapan sebagai berikut:



Gambar 3.1 Diagram Blok Perencaan Sistem

Penelitian ini bertujuan untuk menerapkan pengamanan pesan ke dalam media video dengan kombinasi kriptografi metode RSA-CRT dengan steganografi LSB. Berdasarkan diagram perencanaan diatas, perangkat lunak akan terbagi menjadi 4 proses utama dalam penerapan kombinasi kriptografi dan steganografi, yakni:

1. Enkripsi *text* dengan algoritma RSA-CRT

   Pada proses ini dilakukan pembangkitan kunci publik untuk menerapkan algoritma RSA-CRT. Membangkitkan kunci publik dengan pasangan (*e*,*n*) dan kunci privat dengan pasangan (dP,dQ,qInv,p,q) kemudian dilakukan proses enkripsi dengan menggunakan kunci publik yang telah dibangkitkan sebelumnya, sehingga menghasilkan *chipertext*.
2. *Embedding Chipertext*

   Pada proses ini *chipertext* akan di sisipkan kedalam media penampung (video dengan format (.mp4)) dengan algoritma LSB.
3. *Extraction Chipertext*

   Proses *Extraction chipertext* dari media penampung dengan algoritma LSB. *Chipertext* dikeluarkan kembali dari *file* video (stego *file*) melalui proses

ekstraksi dengan menggunakan pembangkit bilangan acak yang sama pada saat penyisipan pesan. Hasil dari proses ini berupa *chipertext* dan *cover file*.

4. Dekripsi

   Selanjutnya pada proses dekripsi *chipertext* dengan algoritma RSA-CRT, dekripsi kunci *private* yang telah dibangkitkan sebelumnya akan digunakan lalu data asli berupa pesan teks akan diterima. Sehingga pesan acak dari *chipertext* kembali seperti semula dalam bentuk *plaintext* dengan demikian si penerima pesan memperoleh data atau informasi yang diinginkan dengan aman. Hasil dari proses ini berupa *plaintext* atau pesan asli.

3.3.3 Analisis Kebutuhan

Sebelum menuju tahap perancangan, hal yang dilakukan yaitu menganalisis kebutuhan. Analisis ini dibutuhkan untuk menentukan perangkat lunak seperti apa yang akan dihasilkan. Analisis kebutuhan pada penelitian ini adalah sebagai berikut.

3.3.3.1 Metode Analisis

Aplikasi untuk menerapkan kriptografi metode RSA-CRT kombinasi dengan steganografi metode LSB pada *file* video. Untuk melihat proses aplikasi yang mencakup *input* dan *output* dinyatakan dengan menggunakan *flowchart*.

3.3.3.2 Hasil Analisis

Dari data yang diperoleh setelah dilakukan proses analisis yang terdiri dari kebutuhan proses, kebutuhan *input* dan kebutuhan *output*, yaitu adalah sebagai berikut:

1. Analisis kebutuhan proses, kebutuhan proses dalam penerapan kombinasi kriptografi metode RSA-CRT dengan steganografi metode LSB pada *file* video yaitu proses dalam menyediakan *file* video format (*.mp4) dengan ukuran maksimum 20 Mb.

2. Analisis kebutuhan masukan, *input* atau masukan dalam aplikasi penyisipan pesan, yaitu sebagai berikut:
   a. Data video dimasukkan langsung oleh pengguna dengan ketentuan video dengan ukuran batas maksimum 20Mb.
   b. *Input* data rahasia yang digunakan oleh pengguna adalah teks.
   c. Melakukan proses enkripsi dengan metode RSA-CRT kemudian di sisipkan dengan metode LSB.
3. Analisis kebutuhan *output*, kebutuhan *output* yang diperoleh dari proses aplikasi penerapan steganografi pada citra digital dengan menggunakan metode *Chinese Remainder Theorem* adalah diperolehnya hasil *stegofile* berupa video dengan format (*.mp4) dan pesan rahasia yang telah disisipkan atau *plaintext* (pesan asli).

3.3.3.3 Kebutuhan Perangkat Lunak

Kebutuhan peerangkat lunak, digunakan untuk merancang sistem yang akan dibuat, menguji kinerja sistem serta menerapkan sistem yang akan dibuat dengan menggunakan Bahasa pemrograman. Perangkat lunak yang digunakan dalam penelitian ini yaitu, *operating system Microsoft Windows 10*, dan Bahasa pemrograman *Python*.

3.3.3.4 Kebutuhan Perangkat Keras

Kebutuhan akan perangkat keras digunakan untuk mendukung kinerja sistem yang akan dibuat. Perangkat keras yang digunakan berupa laptop Acer Aspire 3 A314-21-4644 *Processor* CPU AMD Dual-Core A4-9120E, RAM: 4GB DDR 4 *Memory* dan *Storage*: 1000 GB HDD.

3.3.3.5 Kebutuhan User (Pengguna)

Sistem ini dibangun untuk memberikan kontribusi keilmuan bagi para oengguna internet agar keamanan data dalam berkiri, pesan rahasia dapat lebih terjamin, dengan sistem yang mudah digunakan serta akan menambah teori

pengetahuan pengguna dalam bidang pengolahan citrs khususnya kriptografi dan steganografi.

3.3.4 Perancangan

Tahap perancangan yang dilakukan berfungsi agar sistem sesuai dengan kebutuhan fungsi. Perancangan juga dilakukan agar membantu pembuatan sistem dengan cepat dan membantu pengguna agar dapat menggunakan sistem dengan mudah. Tahap perancangan terdiri dari perancangan *flowchart* dan perancangan antar muka.

3.3.5 Pengujian

Tahap pengujian merupakan tahap untuk mengetahui proses sistem yang telah dirancang apakah sesuai dengan fungsi dan *output* nya. Pengujian dilakukan pada data video yang akan disisipkan pesan rahasia berupa teks dengan melalui proses enkripsi untuk membangkitkan kunci *public*, selanjutnya *embedding* untuk proses penyisipan *chipertext* ke dalam media video. Kemudian akan dilakukann *extraction chipertext* untuk mengeluarkam *chipertext* dari media video dan terakhir yaitu proses dekripsi untuk mengambil *plaintext* atau pesan asli yang telah disisipkan.

3.3.6 Penerapan/ Penggunaan

Penelitian ini digunakan dengan cara meng*input* informasi rahasia berupa teks kedalam video dengan format (*.mp4) pada sistem yang sudah tersedia dengan menggunakan kombinasi metode RSA-CRT Kriptografi dengan metode LCB Steganografi.

BAB IV
HASIL DAN PEMBAHASAN

4.1 Pembahasan

Setelah melalui tahap perancangan pada Bab III, maka selanjutnya pembahasan dan hasil, pada analisis data untuk kebutuhan sistem dalam menerapkan kombinasi metode kriptografi dengan steganografi. Dimana dengan adanya hasil dari pembahasan tersebut, maka akan diketahui apakah hasil yang diperoleh secara manual dengan sistem komputerisasi terdapat perbedaan atau tidak.

4.1.1 Analisis Data

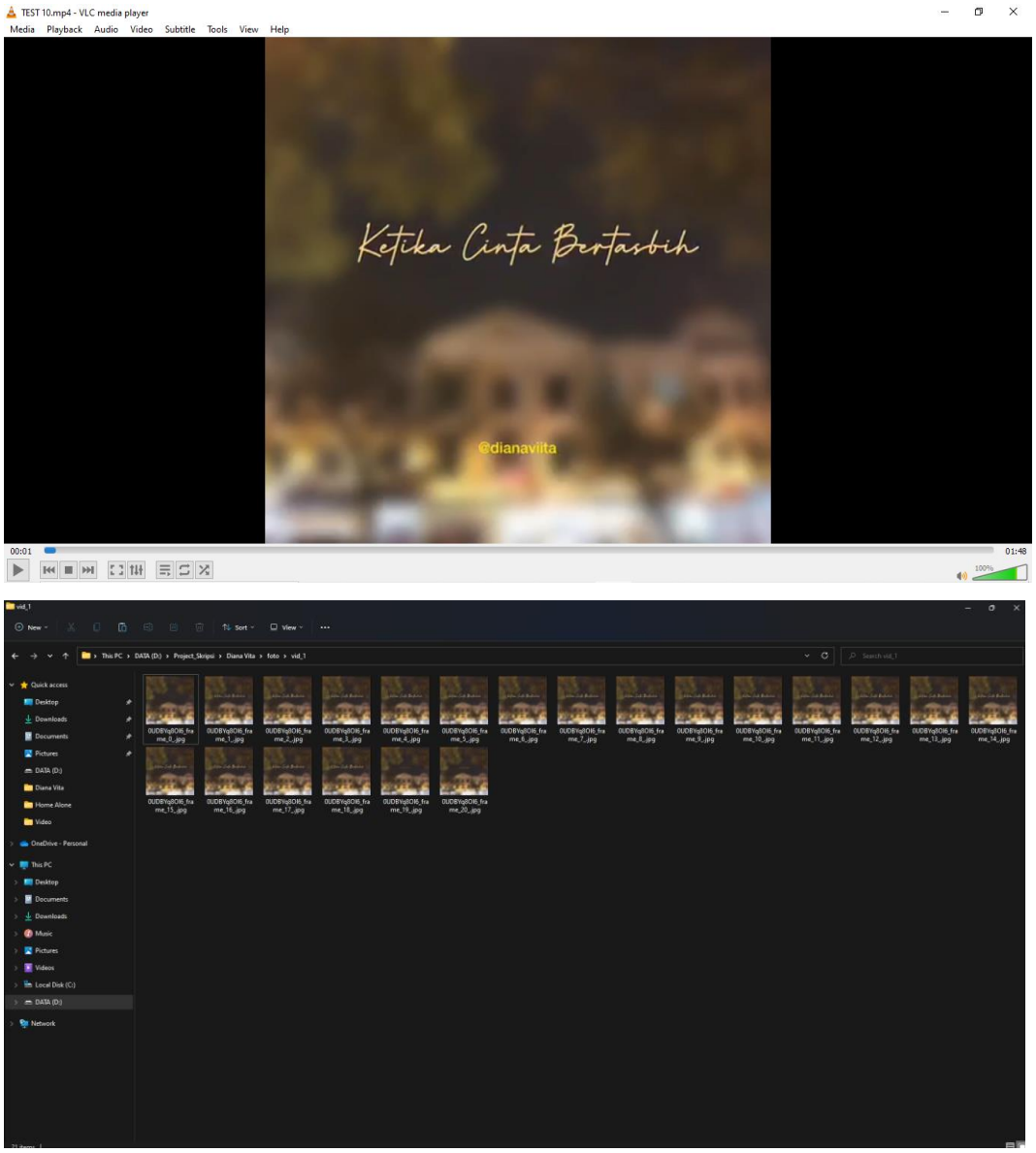
Setelah data didapatkan, maka selanjutnya menguraikan bagaimana cara pengolahan data tersebut. Tahapan analisis dimaksudkan untuk melakukan analisis terhadap data-data yang telah diperoleh yang selanjutnya akan dilakukan proses enkripsi dan ekstraksi pesan ke dalam media video berformat (.mp4) menerapkan kombinasi metode kriptografi RSA-CRT dengan metode steganografi LSB. Adapun beberapa tahapan dalam mengelola data yang diperoleh antara lain adalah sebagai berikut:

1. Menyediakan video dengan format (.mp4).
2. Mendapatkan 1 buah *frame* dari proses *framecutting* video.
3. Menerapkan metode kriptografi RSA-CRT untuk melakukan enkripsi dan dekripsi teks.
4. Melakukan *embedded* dan *extraction* menggunakan metode LSB.

4.1.2 Representasi Data

Data yang digunakan berupa video berdurasi 1 menit 48 detik dengan *frame rate* 5 fps dengan format (.mp4) yang digunakan sebagai media penampung. Data video akan dilakukan *framecutting* seperti pada Gambar 4.1, dalam kasus ini

peneliti hanya menggunakan 1 *frame* dari proses *framecutting* yang akan disisipi pesan.



Gambar 4.1 *Framecutting* dari Video

Pada gambar 4.2 merupakan salah satu citra yang telah dilakukan proses *framecutting*. Penulis menyebutnya dengan *frame*1. *Frame* 1 ini akan digunakan sebagai *sample* yang nantinya akan disisipi pesan.

46	40	40	40	38
46	40	40	40	40
46	40	40	40	40
46	40	40	40	40
46	40	40	40	40

Gambar 4.2 *Sampel frame* 1 sebagai data latih

Sample citra diatas merupakan salah satu *framecutting* diperoleh dari video yang memiliki nilai disetiap *pixel*, citra tersebut berjenis RGB yang memiliki rentang warna 0-255. *Sample* citra tersebut terdiri dari 5 baris dan 5 kolom yang akan digunakan untuk melakukan proses penyisipan pesan.

Sedangkan data pesan yang disembunyikan berupa teks yang terdiri dari karakter ASCII.

Pesan yang akan digunakan penulis adalah “GIH”. Kata “GIH” akan dikonversikan kedalam kode ASCII, sehingga akan di dapat kata “GIH” berupala kode ASCII. Untuk melihat perubahan pesan menjadi kode ASCII dapat dilihat di Tabel 4.1 dibawah ini:

Tabel 4.1 Data pesan dikonversi menjadi kode ASCII

Pesan	ASCII
G	71
I	73
H	72

4.1.3 Hasil Analisis Data

Proses enkripsi merupakan proses dimana *plaintext* (pesan asli) yang dikodekan menjadi *chipertext*. Pesan asli yang digunakan penulis adalah "GIH" yang telah di konversikan ke kode ASCII.

Pesan asli tersebut akan dienkripsi menggunakan algoritma RSA-CRT, namun terlebih dahulu melewati beberapa tahapan seperti berikut:

a. Pembangkit Kunci RSA-CRT

Melakukan enkripsi pesan dengan membangkitkan kunci publik terlebih dahulu menerapkan algoritma RSA-CRT.

Pada dasarnya RSA-CRT sama dengan RSA biasa tetapi memanfaatkan teorema CRT untuk memperpendek ukuran bit eksponen dekripsi d dengan cara menyembunyikan d pada sistem kongruen sehingga mempercepat waktu dekripsi. Berikut algoritma pembangkit kunci RSA-CRT:

- Pilih dua bilangan (*integer*) prima p dan q sembarang, dimana $p \neq q$.
  Misalnya: $p = 23$ dan $q = 13$ (nilai yang ditentukan penulis)
- Hitung nilai $N = p.q$, ($p \neq q$), bilangan N merupakan parameter keamanan, dimana panjang nilai N nya maka semakin sukar di pecahkan.
  $N = 23 \times 13 = 299$.
- Hitung $\varphi(N) = (p - 1)(q - 1)$
  $= (23 - 1)(13 - 1)$

= 264

- Bangkitkam secara acak kunci e dengan syarat:
  - $1 < e < \varphi(N)$
  - $\gcd(e.\varphi(N)) = 1$
  - e relatif prima (yang tidak merupakan faktor dari nilai $\varphi(N)$).

  Misalnya e = 529 (angka yang ditentukan penulis)

  $\gcd(e.\varphi(N)) = \gcd(529,264) = 1$ (**memenuhi**)
- hitung nilai d dengan syarat:
  - $(e,d) \bmod \varphi(N) = 1$
  - Dengan mencoba nilai d = 1, 2, 3, 4,.n sehingga memenuhi persamaan tersebut.
  - misalkan d = 793

    Keterangan: d = Angka yang ditentukan penulis
  - $(e.d) \bmod \varphi(N) = 1$

    $(529.793) \bmod 264 = 1$

    $419.497 \bmod 264 = 1$ (**memenuhi**)
- $dP = d \bmod (p - 1)$

  $= 793 \bmod (23 - 1)$

  $= 1$
- $dQ = d \bmod (q - 1)$

  $= 793 \bmod (13 - 1)$

  $= 1$
- $qInv = \frac{1+Kp}{q}$

  $= \frac{1+K.23}{13} (dimana\ K = 9)$

  $= \frac{1+(9.23)}{13}$

  $= \frac{208}{13}$

  $= 16$

Keterangan: K = bilangan bulat positif dengan tujuan hasil nilai qInv adalah bilangan bulat positif ketika dibagi dengan nilai q.

- Kpublik = (e, *N*) = (529, 299)
- Kprivat = (dP, dQ, qInv, *p*, *q*) = (1, 1, 16, 23, 13)

b. Enkripsi RSA-CRT

Kunci *public* RSA-CRT sama dengan sistem RSA yaitu (e, *N*) sehingga algoritma enkripsi tidak mengalami perubahan yaitu dengan menggunakan fungsi eksponensial modular. Dari rumus perhitungan enkripsi $c_i = M^e \bmod N$, maka dapat dihitung kode *chipertext* dari setiap pesan tersebut sebagai berikut:

Pesan yang akan dikirim *M*= GIH, dengan kode ASCII

G = 71

I = 73

H = 72

Plaintext P_1 = 71, maka nilai *chipertext* dari pesan dengan perhitungan:

$C = M^e \bmod N$

$= 71^{529} \bmod 299$

$= 71$

Plaintext P_2 = 73, maka nilai *chipertext* dari pesan dengan perhitungan:

$C = M^e \bmod N$

$=73^{529} \bmod 299$

$= 73$

Plaintext P_3 = 72, maka nilai *chipertext* dari pesan dengan perhitungan:

$C = M^e \bmod N$

$= 72^{529} \bmod 299$

$= 72$

Setelah mendapatkan semua kode *chipertext* maka dapat dirangkai seluruh kode yang menghasilkan *chipertext* (71 73 72).

Selanjutnya akan dilakukan proses *embedded* dengan menggunakan metode LSB menggunakan *plaintext* yang telah diubah ke *chipertext*.

c. Melakukan *embedded* dan *extraction* pesan menggunakan metode steganografi LSB

Setelah *chipertext* sudah didapatkan maka tahap selanjutnya adalah proses *embedded*, dimana *plaintext* yang telah diubah menjadi kode *chipertext* akan disisipkan kedalam media penampung.

Untuk melakukan proses *embedded*, terlebih dahulu menyiapkan *frame* yang akan dijadikan media menyisip pesan.

Dalam penelitian ini, penulis menggunakan 1 contoh *frame* untuk melakukan perhitungan dalam proses penyisipan pesan. Gambar 4.2 merupakan nila piksel 5x5 dari *frame* yang akan digunakan.

46	40	40	40	38
46	40	40	40	40
46	40	40	40	40
46	40	40	40	40
46	40	40	40	40

Gambar 4.3 Nilai piksel dari *Frame* data latih

Selanjutnya mengubah *chipertext* menjadi kode ASCII kode biner. Berikut adalah tabel perubahan *chipertext* ke biner:

Tabel 4.2 Perubahan *Chipertext* ke biner

Chipertext	**Biner**
71	1000111
73	1001001
72	1001000

Kemudian proses penyembunyian pesan menggunakan bit LSB adalah sebagai berikut:

a. Lakukan proses *embedded* (penyembunyian) dengan cara menyisipkan 1 bit pesan dengan aturan seperti berikut:
   - *If* (bit pesan = ‘1’ dan piksel citra = ‘ganjil’) atau (bit pesan = ‘0’ dan piksel citra = ‘genap’) maka:
     Piksel citra (tetap)
   - *If* (bit pesan = ‘1’ dan piksel citra = ‘genap’) maka:
     Piksel citra baru = piksel citra lama + 1
   - *If* (bit pesan = ‘0’ dan piksel citra = ‘ganjil’) maka:
     Piksel citra baru = piksel citra lama – 1

   Sehingga untuk *chipertext* 71 73 72= 1000111 1001001 1001000 adalah:

   - Bit pesan 1 paling kiri = ‘1’, piksel citra = 46 ‘genap’, maka piksel citra baru = 46+1 = 47
   - Bit pesan 2 = ‘0’, piksel citra = 40 ‘genap’, maka piksel citra baru tetap = 40
   - Bit pesan 3 = ‘0’, piksel citra = 40 ‘genap’, maka piksel citra baru tetap = 40
   - Bit pesan 4 = ‘0’, piksel citra = 40 ‘genap’, maka piksel citra baru tetap = 40
   - Bit pesan 5 = ‘1’, piksel citra = 38 ‘genap’, maka piksel citra baru = 38+1 = 39

- Bit pesan 6 = ‘1’, piksel citra = 46 ‘genap’, maka piksel citra baru = 46+1 = 47
- Bit pesan 7 = ‘1’, piksel citra = 40 ‘genap’, maka piksel citra baru = 40+1 = 41
- Bit pesan 8 = ‘1’, piksel citra = 40 ‘genap’, maka piksel citra baru = 40+1 = 41
- Bit pesan 9 = ‘0’, piksel citra = 40 ‘genap’, maka piksel citra baru tetap = 40
- Bit pesan 10 = ‘0’, piksel citra = 40 ‘genap’, maka piksel citra baru tetap = 40
- Bit pesan 11 = ‘1’, piksel citra = 46 ‘genap’, maka piksel citra baru = 46+1 = 47
- Bit pesan 12 = ‘0’, piksel citra = 40 ‘genap’, maka piksel citra baru tetap = 40
- Bit pesan 13 = ‘0’, piksel citra = 40 ‘genap’, maka piksel citra baru tetap = 40
- Bit pesan 14 = ‘1’, piksel citra = 40 ‘genap’, maka piksel citra baru = 40+1 = 41
- Bit pesan 15 = ‘1’, piksel citra = 40 ‘genap’, maka piksel citra baru = 40+1 = 41
- Bit pesan 16 = ‘0’, piksel citra = 46 ‘genap’, maka piksel citra baru tetap = 46
- Bit pesan 17 = ‘0’, piksel citra = 40 ‘genap’, maka piksel citra baru tetap = 40
- Bit pesan 18 = ‘1’, piksel citra = 40 ‘genap’, maka piksel citra baru = 40+1 = 41
- Bit pesan 19 = ‘0’, piksel citra = 40 ‘genap’, maka piksel citra baru tetap = 40
- Bit pesan 20 = ‘0’, piksel citra = 40 ‘genap’, maka piksel citra baru tetap = 40

- Bit pesan 21 = ‘0’, piksel citra = 46 ‘genap’, maka piksel citra baru tetap = 46

Begitu seterusnya hingga bit pesan terakhir. Berikut tampilan perubahan nilai piksel yang telah disisipi bit pesan biner sehingga menjadi citra baru (stego).

47	40	40	40	**39**
47	**41**	**41**	40	40
47	40	40	**41**	**41**
46	40	**41**	40	40
46	40	40	40	40

Gambar 4.4 Ilustrasi Perubahan nilai piksel setelah disisipi

b. Melakukan *extraxtion* (ekstraksi) pesan yang telah disisipi kedalam *frame* latih sebelumnya dengan cara:
   - Jika piksel citra stego = ‘genap’, maka bit pesan = ‘0’.
   - Jika piksel citra stego = ‘ganjil’, maka bit pesan = ‘1’.
   - Urutkan bit-bit pesan dari MSB, sehingga terbentuk bilangan biner.
   - Konversikan bilangan biner menjadi desimal.

   Sehingga diperoleh:
   - 47 (ganjil), maka bit pesan = 1
   - 40 (genap), maka bit pesan = 0
   - 40 (genap), maka bit pesan = 0
   - 40 (genap), maka bit pesan = 0
   - 39 (ganjil), maka bit pesan = 1
   - 47 (ganjil), maka bit pesan = 1
   - 41 (ganjil), maka bit pesan = 1
   - 41 (ganjil), maka bit pesan = 1

- 40 (genap), maka bit pesan = 0
- 40 (genap), maka bit pesan = 0
- 47 (ganjil), maka bit pesan = 1
- 40 (genap), maka bit pesan = 0
- 40 (genap), maka bit pesan = 0
- 41 (ganjil), maka bit pesan = 1
- 41 (ganjil), maka bit pesan = 1
- 46 (genap), maka bit pesan = 0
- 40 (genap), maka bit pesan = 0
- 41 (ganjil), maka bit pesan = 1
- 40 (genap), maka bit pesan = 0
- 40 (genap), maka bit pesan = 0
- 46 (genap), maka bit pesan = 0

Urutan bit pesan dimulai dari MSB = 1000111 1001001 1001000

Pesannya adalah = 71 73 72

d. Melakukan dekripsi *chipertext* untuk mendapatkan *plaintext* menggunakan metode kriptografi RSA-CRT

$M \equiv C^d \ (mod\ N)$

Dekripsi RSA-CRT seperti berikut:

$C_1 = 71 \rightarrow M_1 = 71^{793} \bmod 299 = 71$

$C_2 = 73 \rightarrow M_2 = 73^{793} \bmod 299 = 73$

$C_3 = 72 \rightarrow M_3 = 72^{793} \bmod 299 = 72$

Maka didapatilah hasil *plaintext* kembali yaitu 71 = G, 73 = I, 72 = H "GIH".

4.2 Perancangan

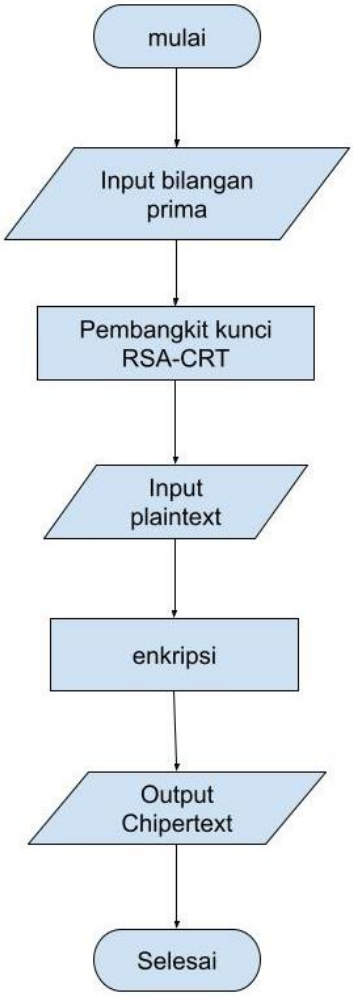
Berikut ini adalah peracangan yang dibuat penulis agar mempermudah *user* dalam menggunakannya. Perancangan terdiri dari perancangan *flowchart* metode Enkripsi, *Embedded*, Ekstraksi, *flowchart* sistem pengamanan pesan pada media

video, dan perancangan antarmuka sistem aplikasi perangkat lunak yang telah dilakukan.

4.2.1 Perancangan *Flowchart* Metode Enkripsi, *Embedded*, dan Ekstraksi

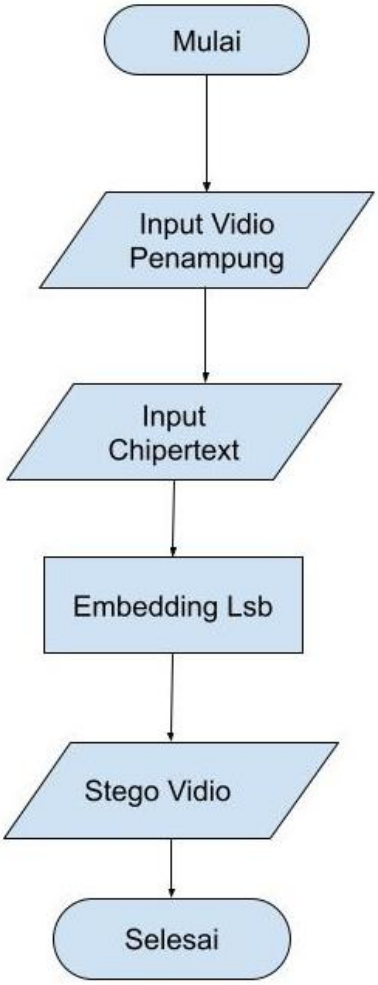
Berikut ini adalah perancangan *flowchart* metode Enkripsi, *Embedded*, dan Ekstraksi.

a. *Flowchart* Enkripsi



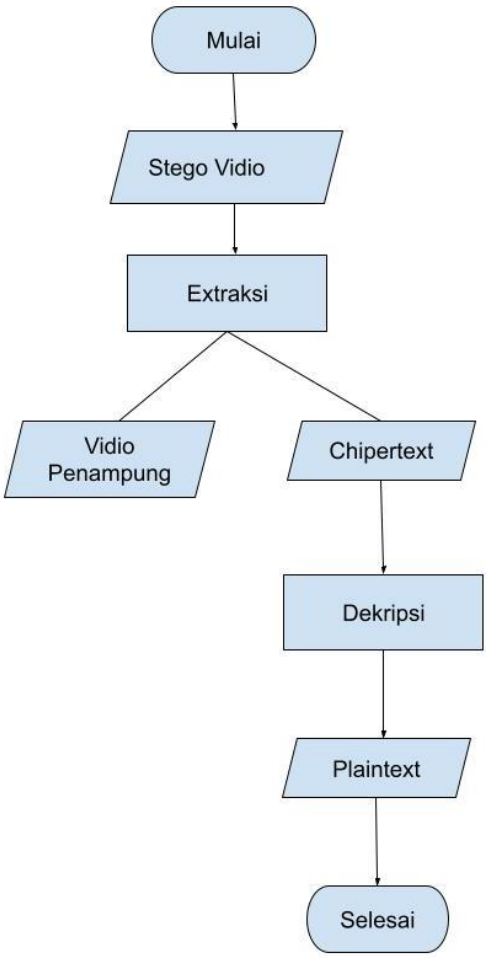
Gambar 4.5 *Flowchart* Metode Enkripsi

b. *Flowchart Embedded*



Gambar 4.6 *Flowchart* Metode *Embedded*

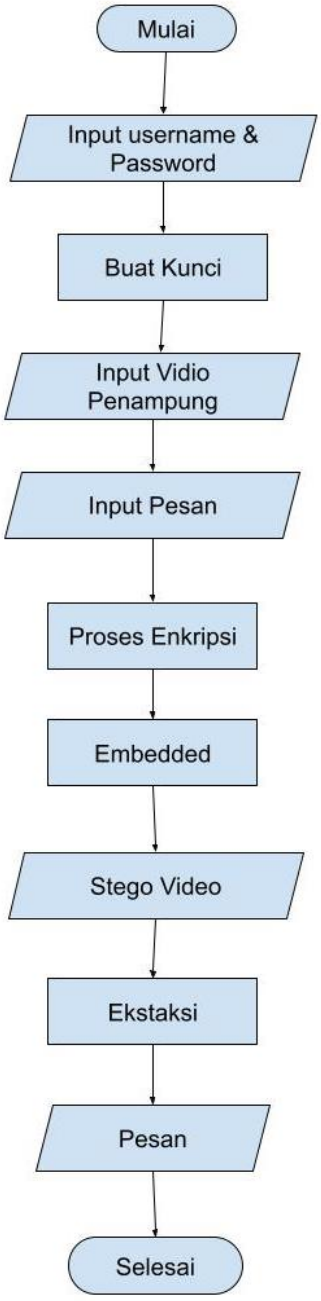
c. *Flowchart* Ekstraksi dan Dekripsi



Gambar 4.7 *Flowchart* Ekstraksi dan Dekripsi

4.2.2 Perancangan *Flowchart* Sistem Aplikasi

Berikut ini adalah perancangan *flowchart* sistem pengamanan pesan ke dalam media video.



Gambar 4.8 *Flowchart* sistem aplikasi

4.2.3 Perancangan Antar Muka

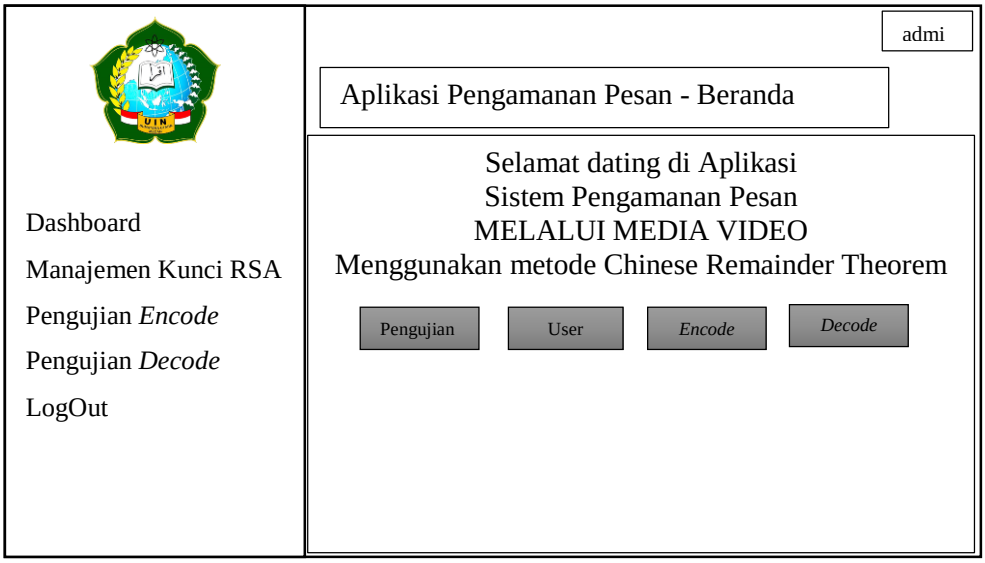
Sistem ini dirancang dengan menggunakan Bahasa pemrograman *Python*. Perancangan bertujuan untuk memudahkan pengguna (*user*) dalam menggunakan sistem yang telah dibuat. Perancangan terdiri dari, *form* utama, menu *login*, halaman *dashboard*, halaman pembuatan kunci, halaman pengujian *encode* dan *decode*.

a. *Form* Utama dan Menu *Login*

Gambar 4.9 Perancangan antar muka *Form* utama

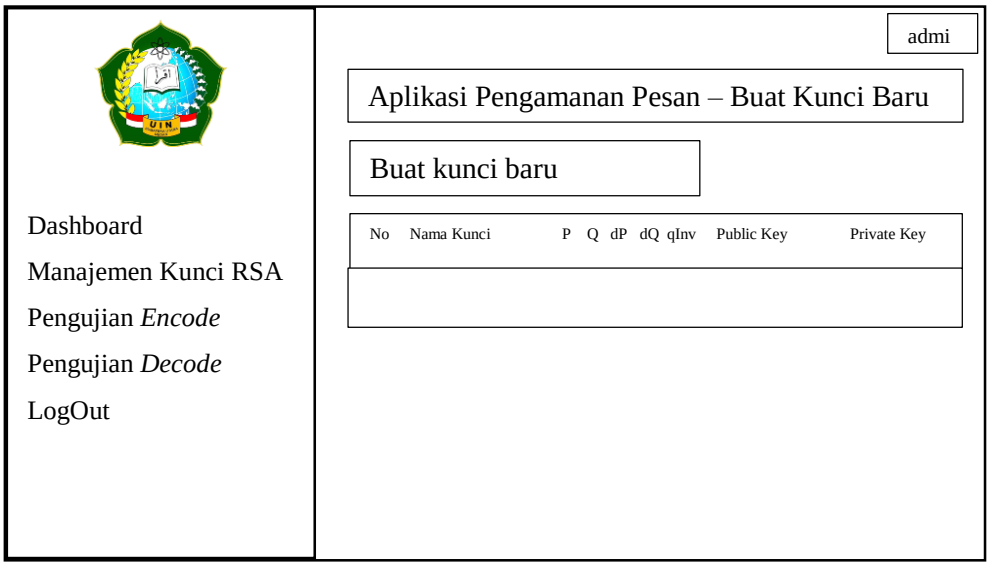
Gambar 4.10 Perancangan antar muka menu *login*

b. Halaman *Dashboard*



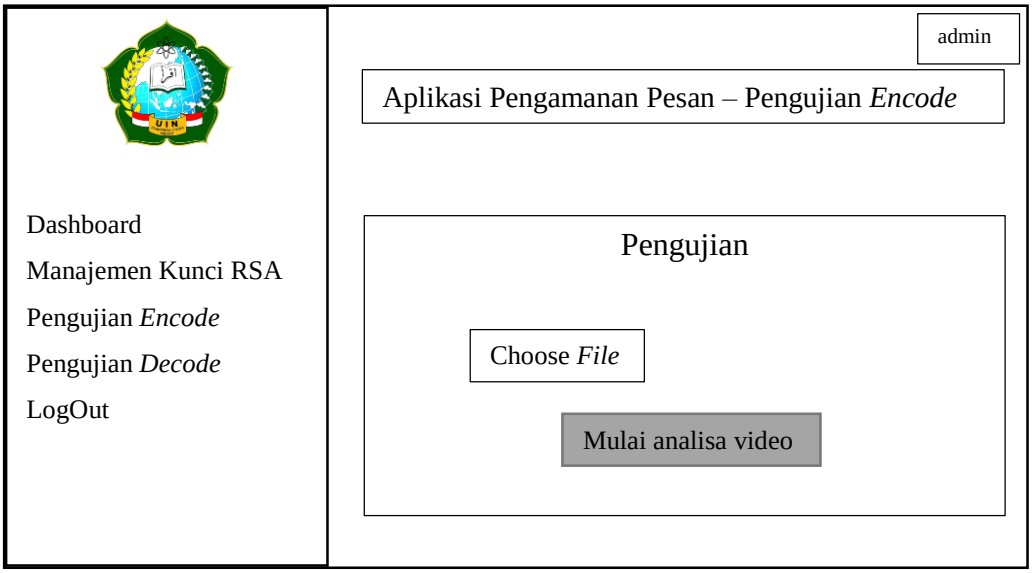
Gambar 4.11 Perancangan antar muka halaman *dashboard*

c. Halaman Pembuatan Kunci

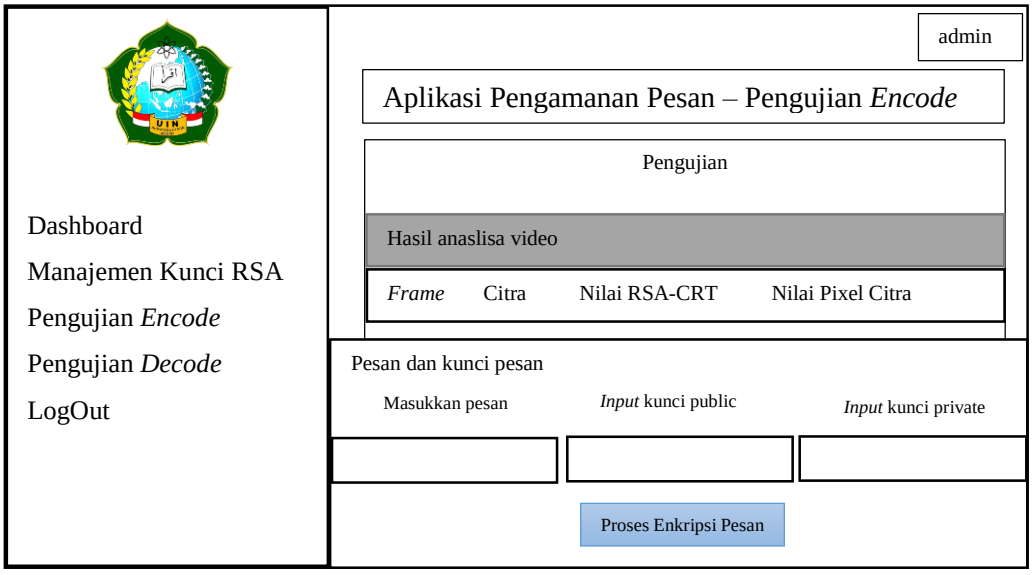


Gambar 4.12 Perancangan antar muka halaman pembuatan kunci

d. Halaman pengujian *Encode*

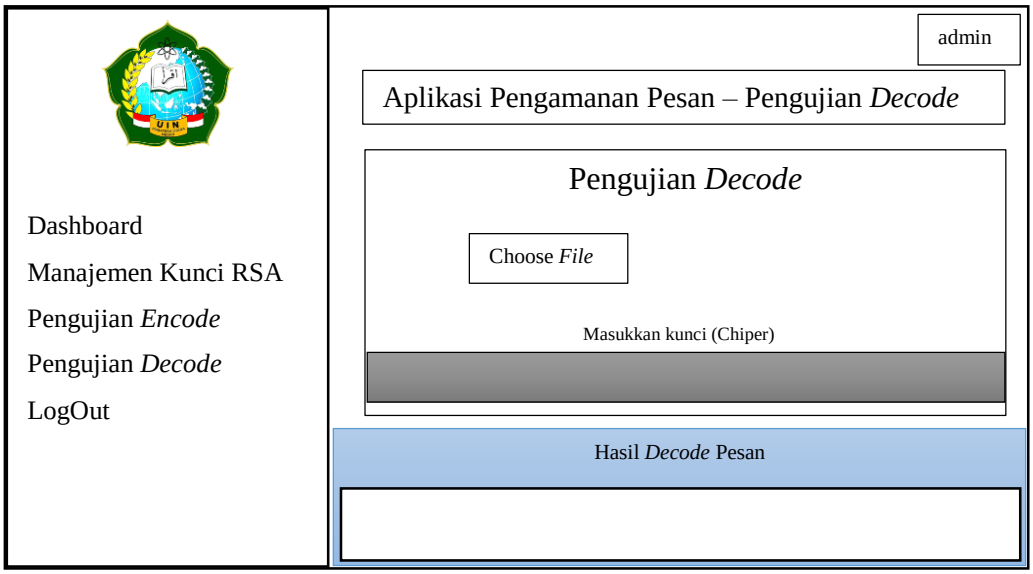


Gambar 4.13 Perancangan antar muka halaman pengujian *encode*



Gambar 4.14 Perancangan antar muka proses enkripsi pesan

e. Halaman pengujian *decode*



Gambar 4.15 Perancangan antar muka hasil dari proses dekripsi

4.3 Hasil dan Pengujian

Berdasarkan *sample* video yang sudah ada, maka akan dilakukan proses pengujian terhadap video dan *frame cut*. Pada tahap ini akan dilakukan pengujian terhadap objek video dengan format *file* (*.mp4) dan hasil dari proses pengujian menerapakan empat aspek yaitu *imperceptibility*, *fidelity*, *recovery*, dan *robustness*.

4.3.1 Pengujian

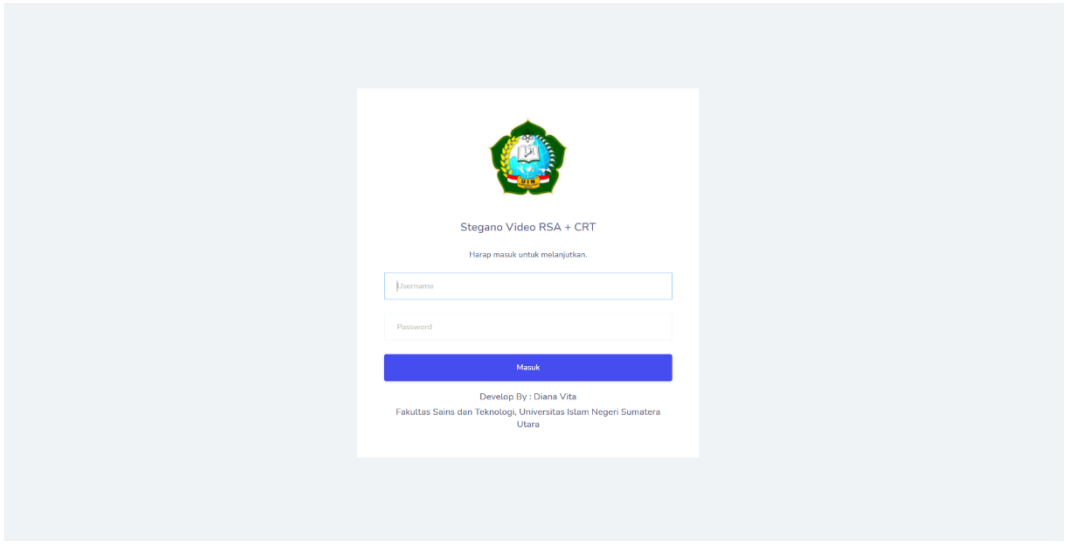
Penelitian ini menerapkan kriptografi metode RSA-CRT kombinasi dengan steganografi metode LSB pada *file* video.

Adapun proses pengujian video dapat dilihat pada gambar berikut:

a. Pengujian *Form* Utama dan Manu *Login*

Form utama digunakan untuk menampilkan *form* induk dari sistem yang didalamnya terdapat *integrase* antar *form* yang terhubung kedalam *form* utama. Tampilan *form* utama dapat dilihat pada gambar dibawah ini:

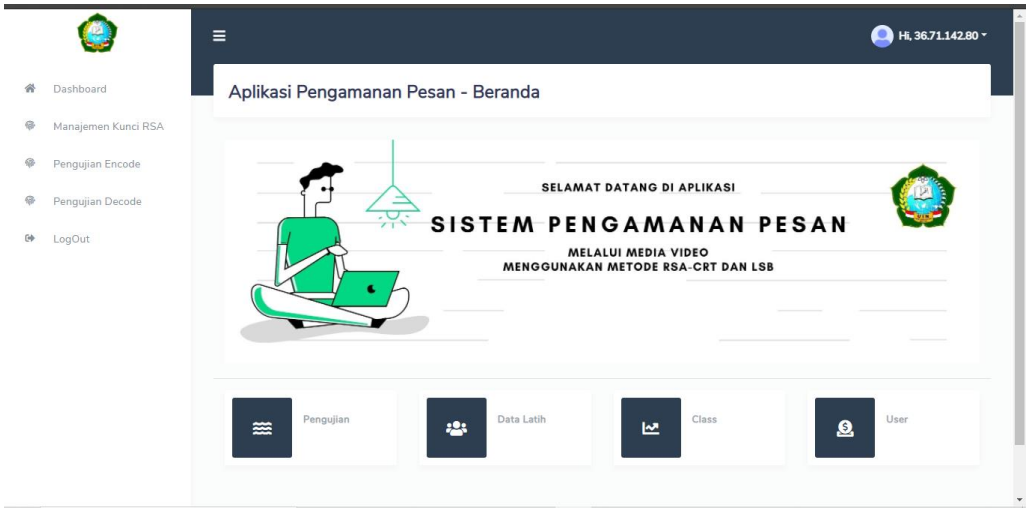
Gambar 4.16 *Form* Utama



Gambar 4.17 Menu *Login*

b. Pengujian Halaman *Dashboard*

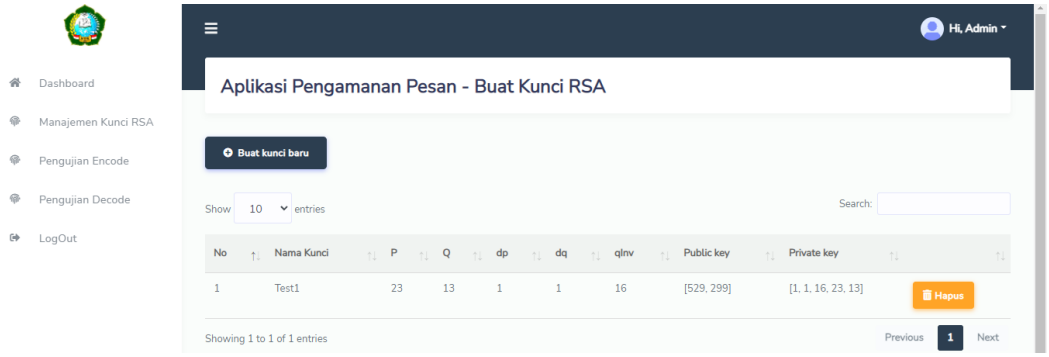
Terdapat beberapa menu yang memiliki fungsi yang berbeda, seperti menu membuat kunci, Pengujian *Encode*, *Decode*, dan lain sebagainya.



Gambar 4.18 Halaman *Dashboard*

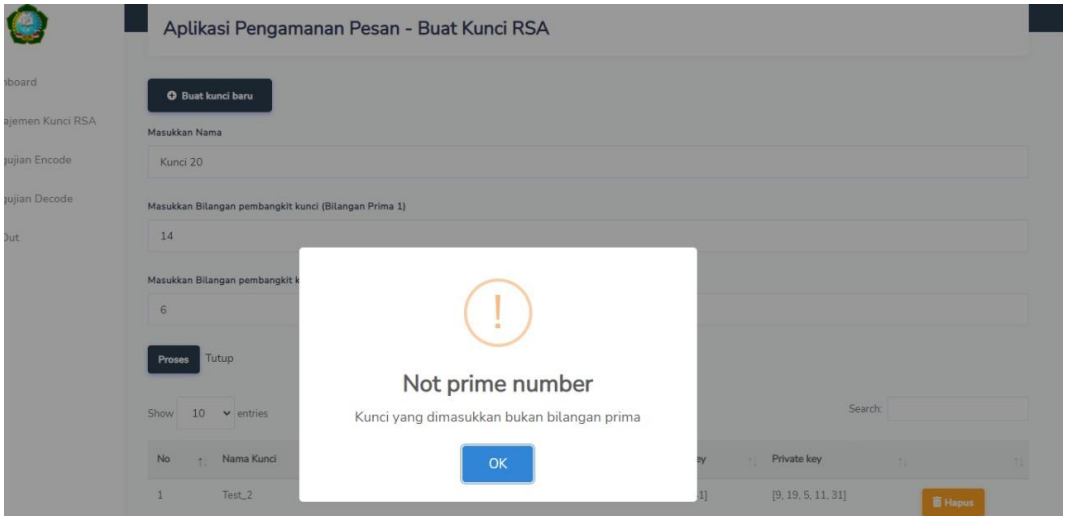
c. Pengujian Halaman Pembuatan Kunci

Pada halaman ini akan dilakukan pembuatan kunci dengan meng*input* 2 bilangan prima yang berbeda sehingga menghasilkan *public key* dan *private key* yang digunakan untuk proses enkripsi pengamanan pesan. Berikut tampilan halaman pembuatan kunci.



Gambar 4.19 Halaman Pembuatan Kunci

Pada proses ini apabila penginputan bilangan prima tidak sesuai, maka aplikasi akan memberikan notifikasi seperti pada gambar berikut.

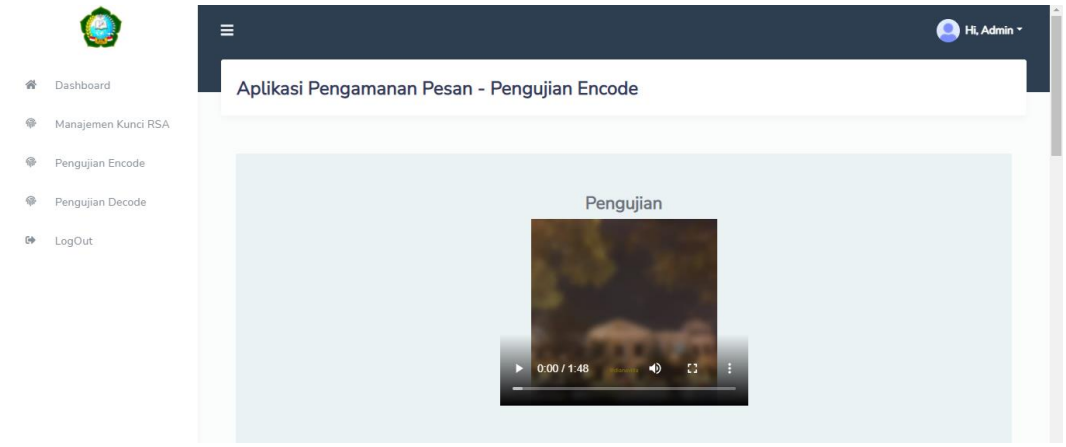


Gambar 4.20 Notifikasi penginputan bilangan prima

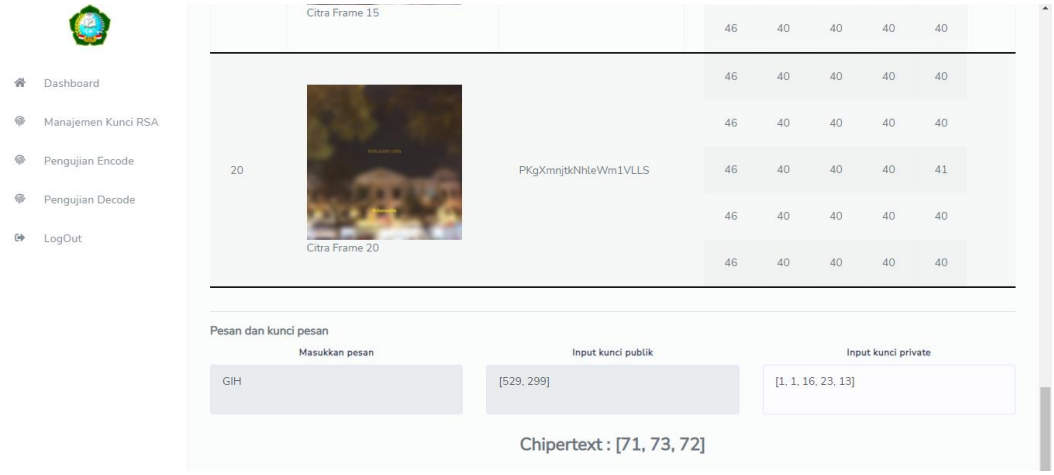
d. Halaman Pegujian *Encode* dan *Decode*

Pada halaman ini akan dilakukan pengujian *encode* dan *decode* pada *file* video yang telah di input. Proses *encode* dan *decode* dilakukan satu persatu terhadap video menggunakan aplikasi pengamanan pesan. Adapun proses pengujian *encode* dan *decode* masing-masing video dapat dilihat pada gambar berikut:

1. Pengujian Video 1



Gambar 4.21 Pengujian *encode* pada pengujian 1

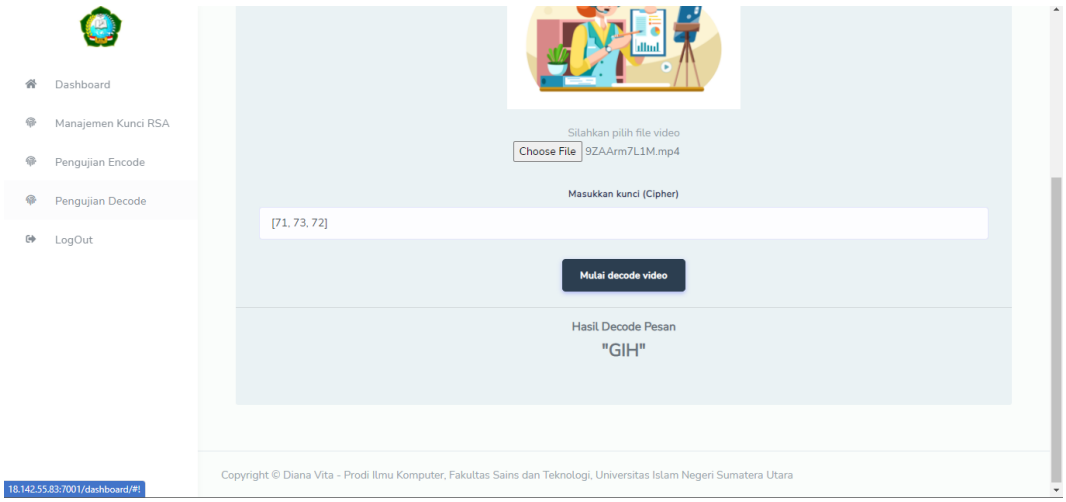


Gambar 4.22 Pengujian enkripsi pesan dan *chipertext* pada Pengujian 1

Pada proses pengujian *encode* video yang di *input* menggunakan video dengan ukuran 4,19 MB, bedurasi 1 menit 38 detik berformat (.mp4). Dalam melakukan proses *encode*, masukkan terlebih dahulu video yang akan diuji. Kemudian sistem akan memproses video sehingga hasil yang didapat berupa *framecutting* sebanyak 5 fps. Setelah itu proses pembuatan kunci dengan memasukkan 2 angka prima yang berbeda sehingga akan di dapat *public key* dan *private key*. Kemudian masukkan pesan yang akan disisipi bersamaan dengan kunci *public* yang telah didapatkan sebelumnya. Maka selanjutnya pesan akan di enkripsi oleh sistem. Pesan yang dimasukkan dalam pengujian pertama dengan panjang 1 kata.

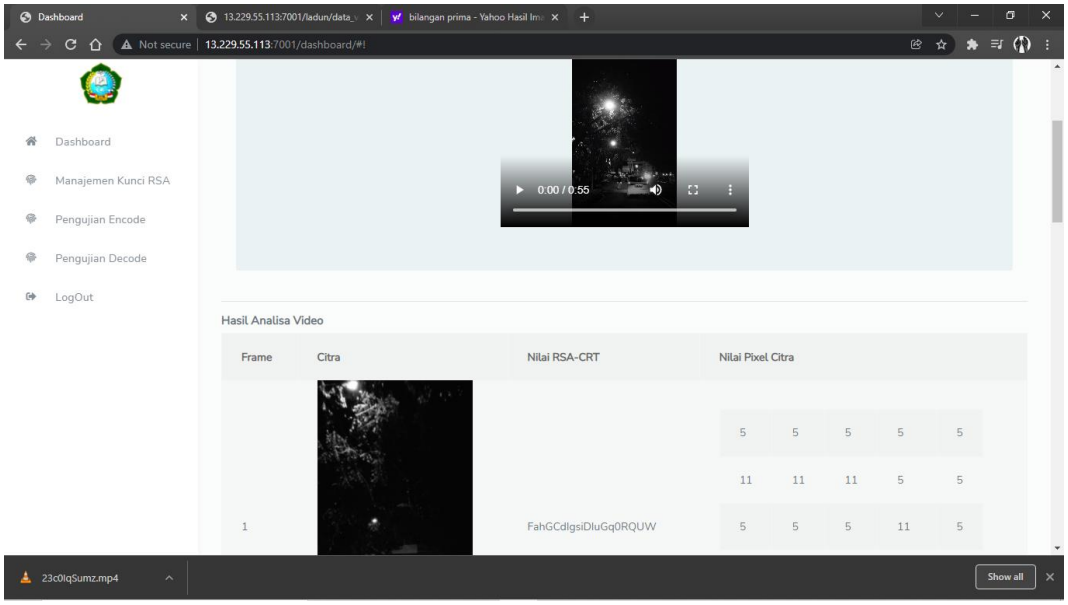
Selanjutnya akan dilakukan proses pengujian pengambilan kembali pesan yang telah disisipkan kedalam media video. Pesan yang telah dienkripsi akan menghasilkan stego *file* berupa video yang berisi pesan. Selanjutnya pesan dapat diambil kembali dengan menggunakan pengujian *decode*.

Apabila proses pengujian berhasil maka akan menampilkan pesan rahasia yang telah dimasukkan pada proses sebelumnya. Adapun hasil dari proses pengujian *decode* dapat dilihat pada gambar berikut:

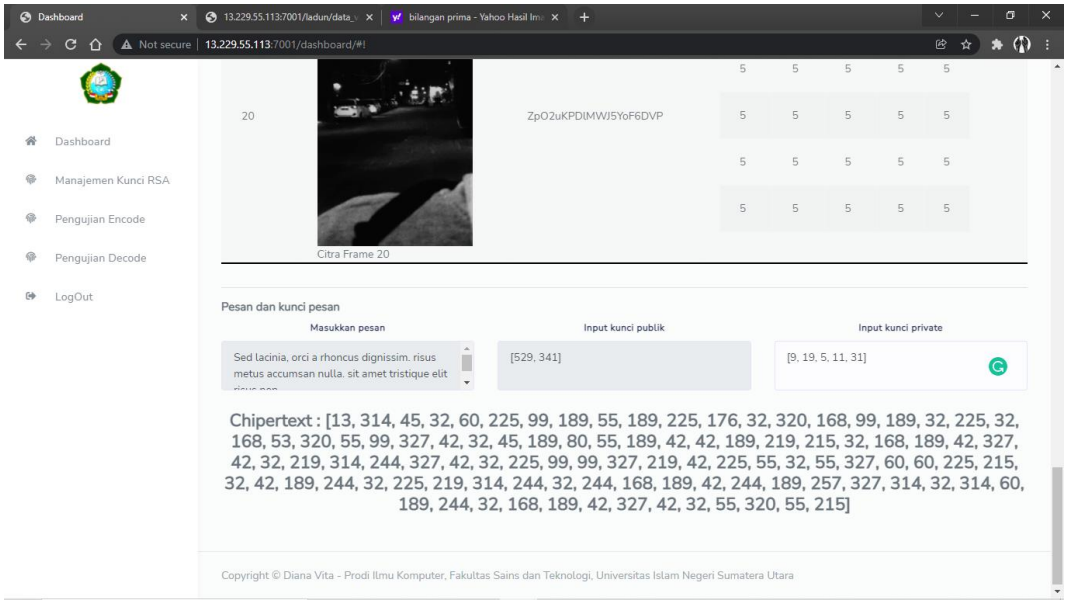


Gambar 4.23 Hasil *decode* pengujian 1

2. Pengujian Video 2



Gambar 4.24 Pengujian *encode* pada Pengujian 2

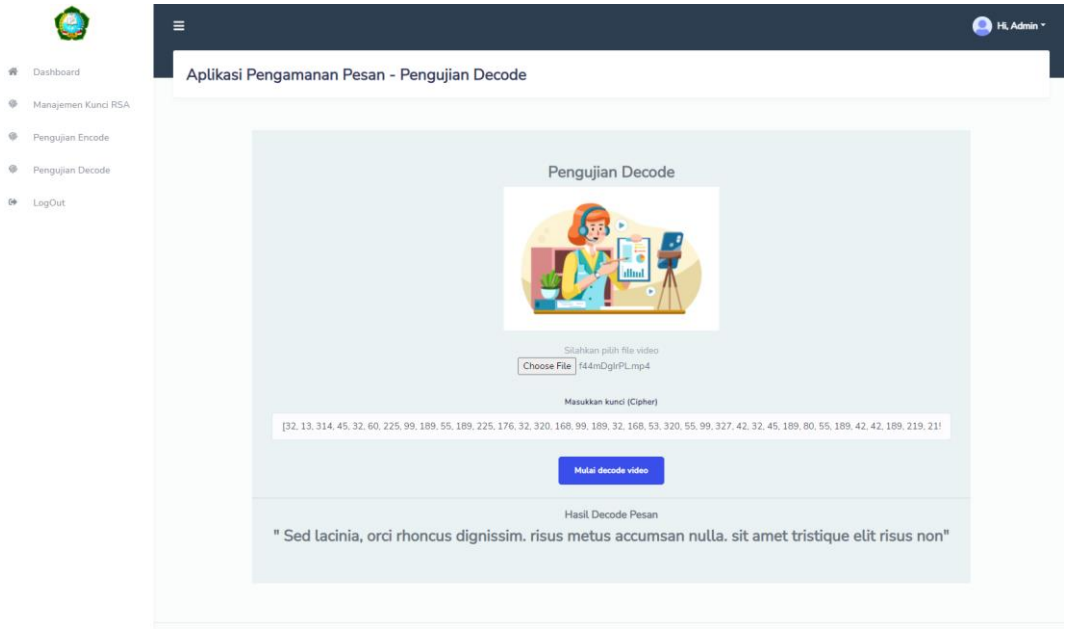


Gambar 4.25 Pengujian enkripsi pesan dan *chipertext* pada Pengujian 2

Pada proses pengujian *encode* video ke 2 yang di *input* menggunakan video dengan ukuran 1,95 Mb, berdurasi 9 detik dengan *frame* rate 5 fps berformat (.mp4). Dalam melakukan proses *encode*, masukkan terlebih dahulu video yang akan diuji. Kemudian sistem akan memproses video sehingga hasil yang didapat berupa *framecutting* sebanyak 5 fps. Setelah itu masukkan pesan yang akan disisipi bersamaan dengan kunci *public* RSA yang telah didapatkan sebelumnya. Maka selanjutnya pesan akan di enkripsi oleh sistem. Pesan yang dimasukkan dengan panjang 15 kata.

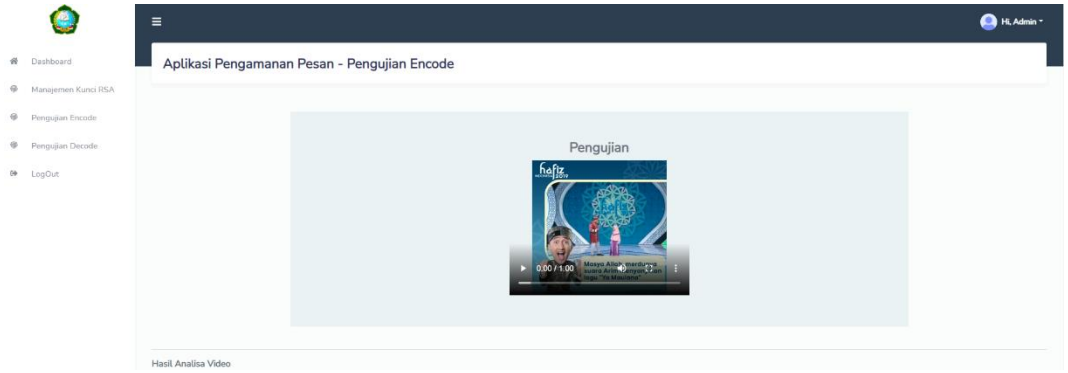
Selanjutnya akan dilakukan proses pengujian pengambilan kembali pesan yang telah disisipkan kedalam media video. Pesan yang telah dienkripsi akan menghasilkan stego *file* berupa video yang berisi pesan. Selanjutnya pesan dapat diambil kembali dengan menggunakan pengujian *decode*.

Apabila proses pengujian berhasil maka akan menampilkan pesan rahasia yang telah dimasukkan pada proses sebelumnya. Adapun hasil dari proses pengujian *decode* dapat dilihat pada gambar berikut:

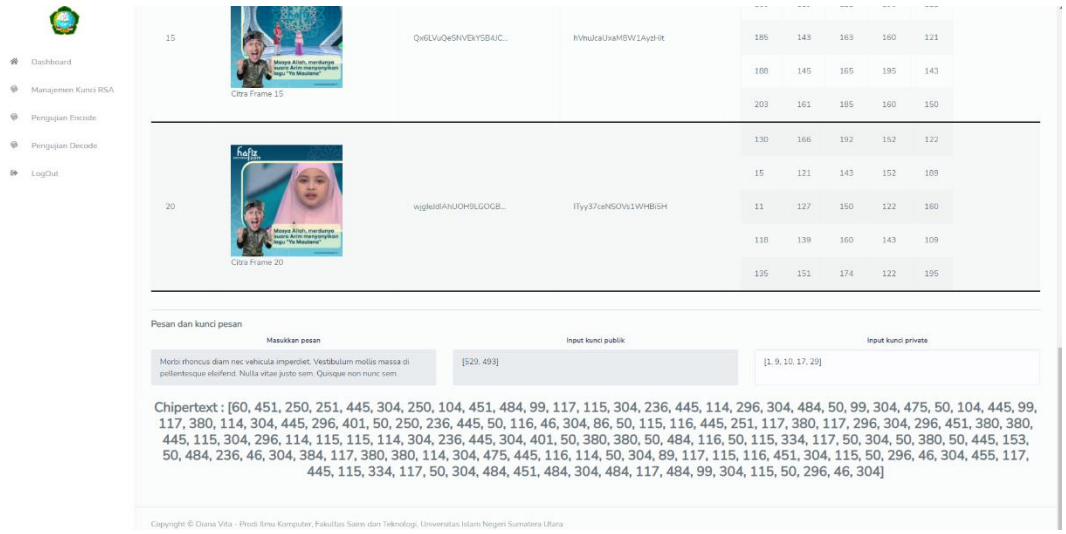


Gambar 4.26 Hasil *decode* pengujian 2

3. Pengujian Video 3



Gambar 4.27 Pengujian *encode* pengujian 3

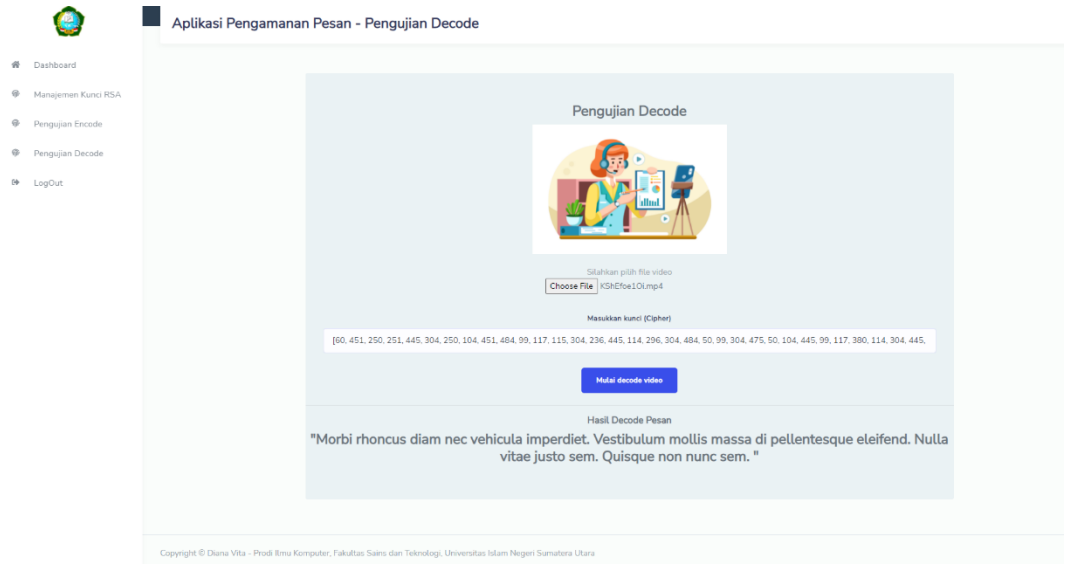


Gambar 4.28 Enkripsi pesan pengujian 3

Pada proses pengujian *encode* video yang di *input* menggunakan video dengan ukuran 33,4 MB, bedurasi 16 detik berformat (.mp4). Dalam melakukan proses *encode*, masukkan terlebih dahulu video yang akan diuji. Kemudian sistem akan memproses video sehingga hasil yang didapat berupa *framecutting* sebanyak 5 fps. Setelah itu proses pembuatan kunci dengan memasukkan 2 angka prima yang berbeda sehingga akan di dapat *public key* dan *private key*. Kemudian masukkan pesan yang akan disisipi bersamaan dengan kunci *public* yang telah didapatkan sebelumnya. Maka selanjutnya pesan akan di enkripsi oleh sistem. Pesan yang dimasukkan dalam pengujian pertama dengan panjang 1 kata.

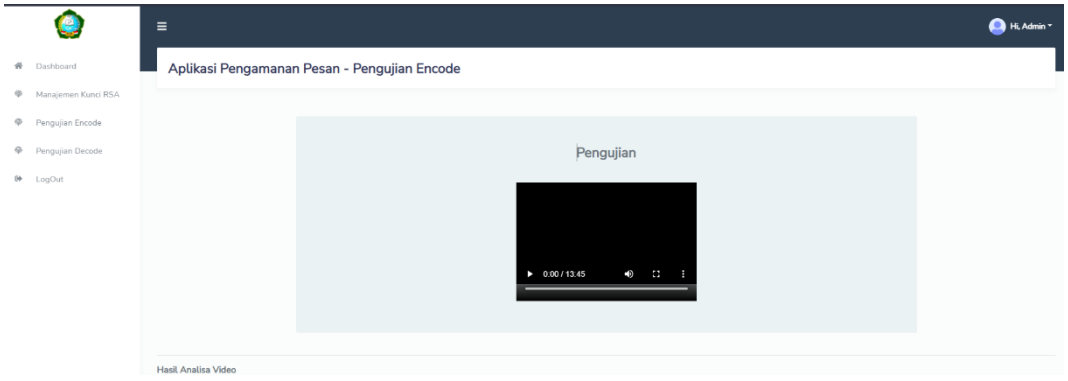
Selanjutnya akan dilakukan proses pengujian pengambilan kembali pesan yang telah disisipkan kedalam media video. Pesan yang telah dienkripsi akan menghasilkan stego *file* berupa video yang berisi pesan. Selanjutnya pesan dapat diambil kembali dengan menggunakan pengujian *decode*.

Apabila proses pengujian berhasil maka akan menampilkan pesan rahasia yang telah dimasukkan pada proses sebelumnya. Adapun hasil dari proses pengujian *decode* dapat dilihat pada gambar berikut:

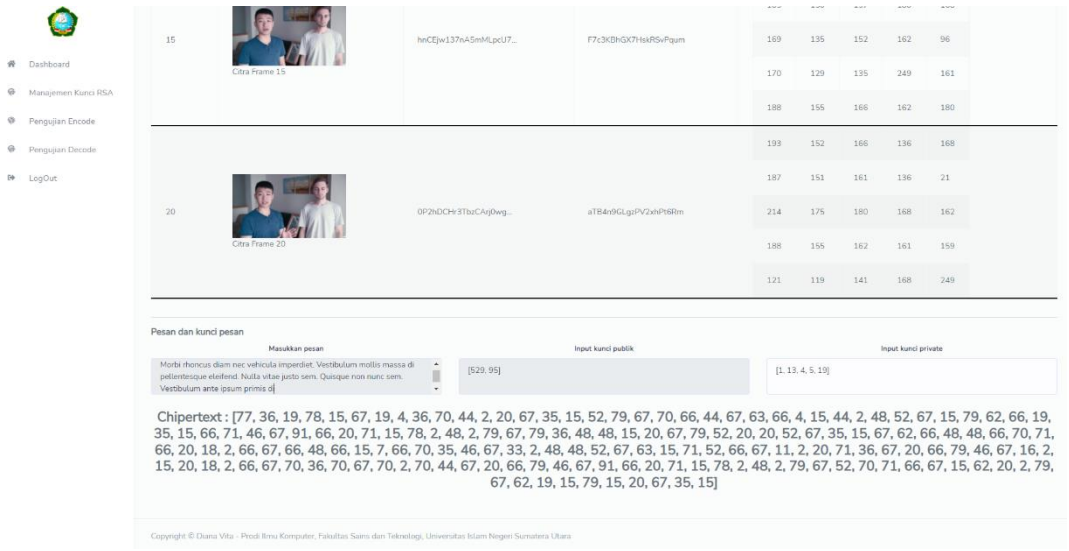


Gambar 4.29 Hasil *decode* pengujian 3

4. Pengujian Video 4



Gambar 4.30 Pengujin *encode* pengujian 4

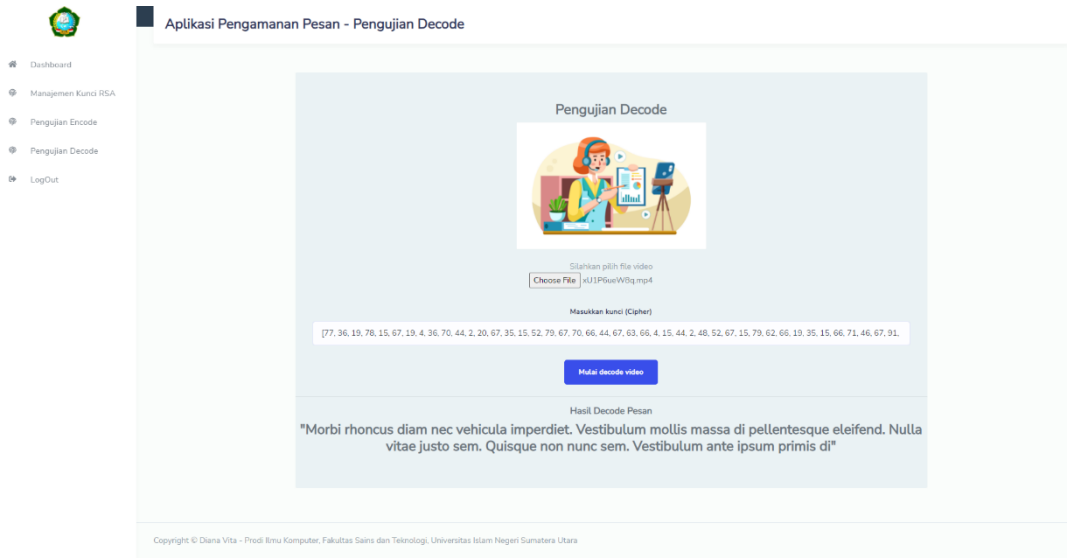


Gambar 4.31 Enkripsi pesan pengujian 4

Pada proses pengujian *encode* video yang di *input* menggunakan video dengan ukuran 39,5 MB, bedurasi 15 detik berformat (.mp4). Dalam melakukan proses *encode*, masukkan terlebih dahulu video yang akan diuji. Kemudian sistem akan memproses video sehingga hasil yang didapat berupa *framecutting* sebanyak 5 fps. Setelah itu proses pembuatan kunci dengan memasukkan 2 angka prima yang berbeda sehingga akan di dapat *public key* dan *private key*. Kemudian masukkan pesan yang akan disisipi bersamaan dengan kunci *public* yang telah didapatkan sebelumnya. Maka selanjutnya pesan akan di enkripsi oleh sistem. Pesan yang dimasukkan dalam pengujian pertama dengan panjang 1 kata.

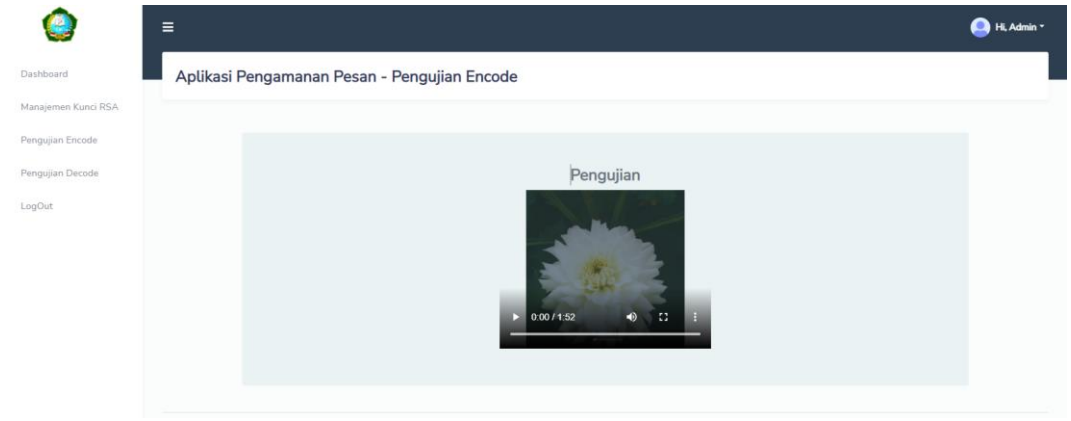
Selanjutnya akan dilakukan proses pengujian pengambilan kembali pesan yang telah disisipkan kedalam media video. Pesan yang telah dienkripsi akan menghasilkan stego *file* berupa video yang berisi pesan. Selanjutnya pesan dapat diambil kembali dengan menggunakan pengujian *decode*.

Apabila proses pengujian berhasil maka akan menampilkan pesan rahasia yang telah dimasukkan pada proses sebelumnya. Adapun hasil dari proses pengujian *decode* dapat dilihat pada gambar berikut:

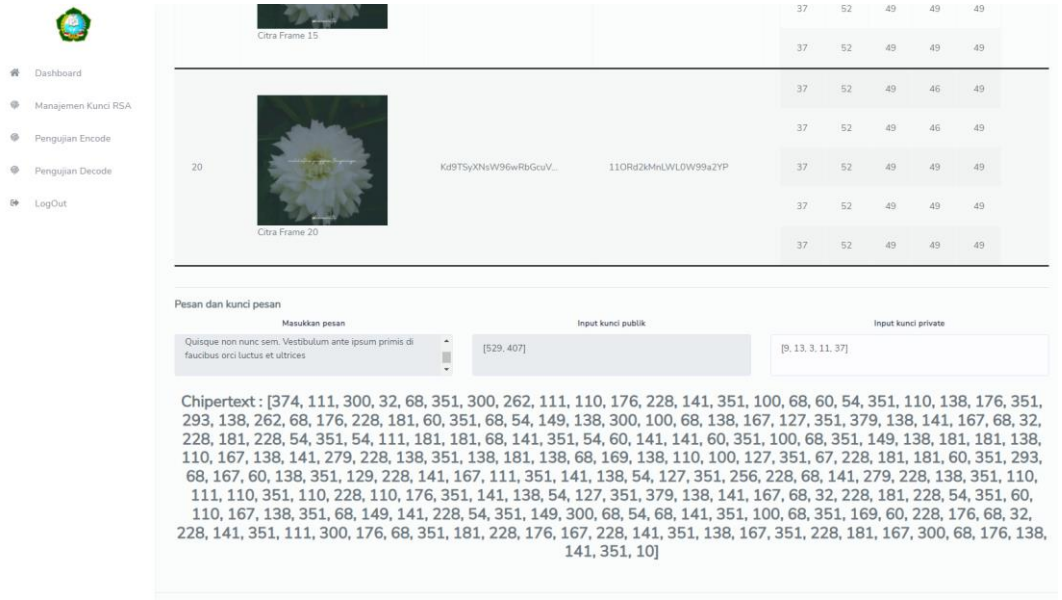


Gambar 4.32 Hasil *decode* pengujian 4

5. Pengujian video 5



Gambar 4.33 Pengujian *encode* pengujian 5



Gambar 4.34 Enkripsi pesan pengujian 5

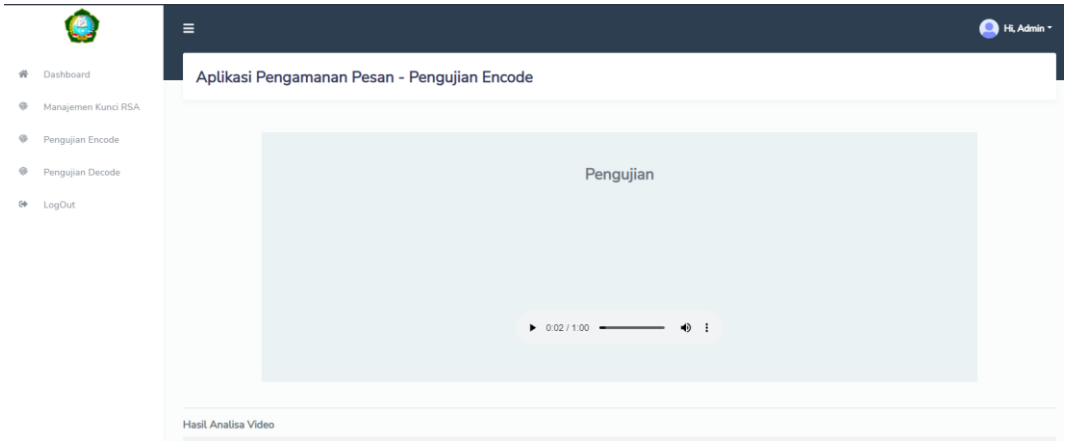
Pada proses pengujian *encode* video yang di *input* menggunakan video dengan ukuran 91,2 MB, bedurasi 1 menit 52 detik berformat (.mp4). Dalam melakukan proses *encode*, masukkan terlebih dahulu video yang akan diuji. Kemudian sistem akan memproses video sehingga hasil yang didapat berupa *framecutting* sebanyak 5 fps. Setelah itu proses pembuatan kunci dengan memasukkan 2 angka prima yang berbeda sehingga akan di dapat *public key* dan *private key*. Kemudian masukkan pesan yang akan disisipi bersamaan dengan kunci *public* yang telah didapatkan sebelumnya. Maka selanjutnya pesan akan di enkripsi oleh sistem. Pesan yang dimasukkan dalam pengujian pertama dengan panjang 1 kata.

Selanjutnya akan dilakukan proses pengujian pengambilan kembali pesan yang telah disisipkan kedalam media video. Pesan yang telah dienkripsi akan menghasilkan stego *file* berupa video yang berisi pesan. Selanjutnya pesan dapat diambil kembali dengan menggunakan pengujian *decode*.

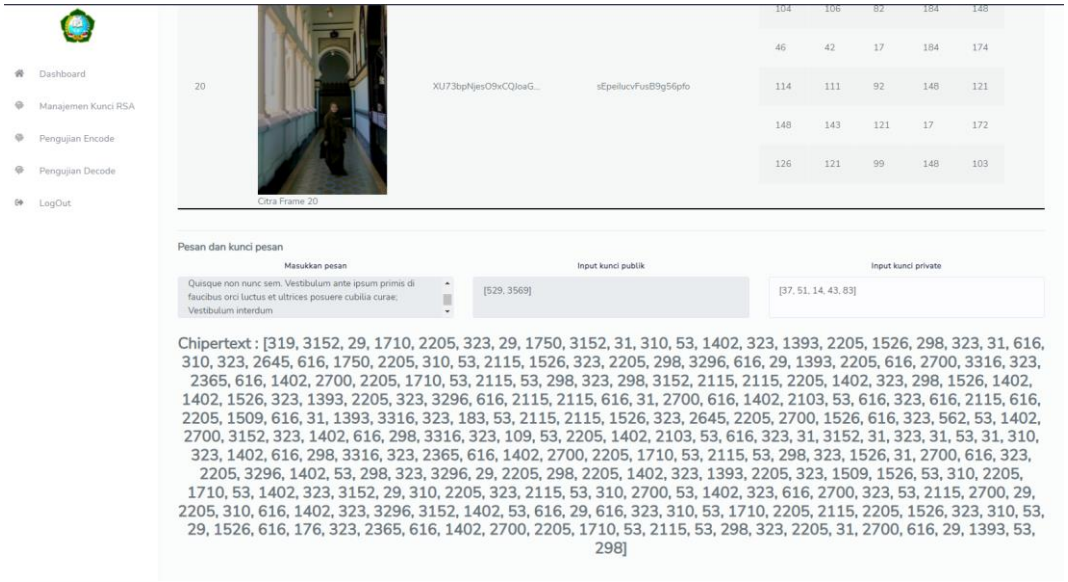
Apabila proses pengujian berhasil maka akan menampilkan pesan rahasia yang telah dimasukkan pada proses sebelumnya. Adapun hasil dari proses pengujian *decode* dapat dilihat pada gambar berikut:

Gambar 4.35 Hasil *decode* pengujian 5

6. Pengujian Video 6



Gambar 4.36 Pengujian *encode* pengujian 6

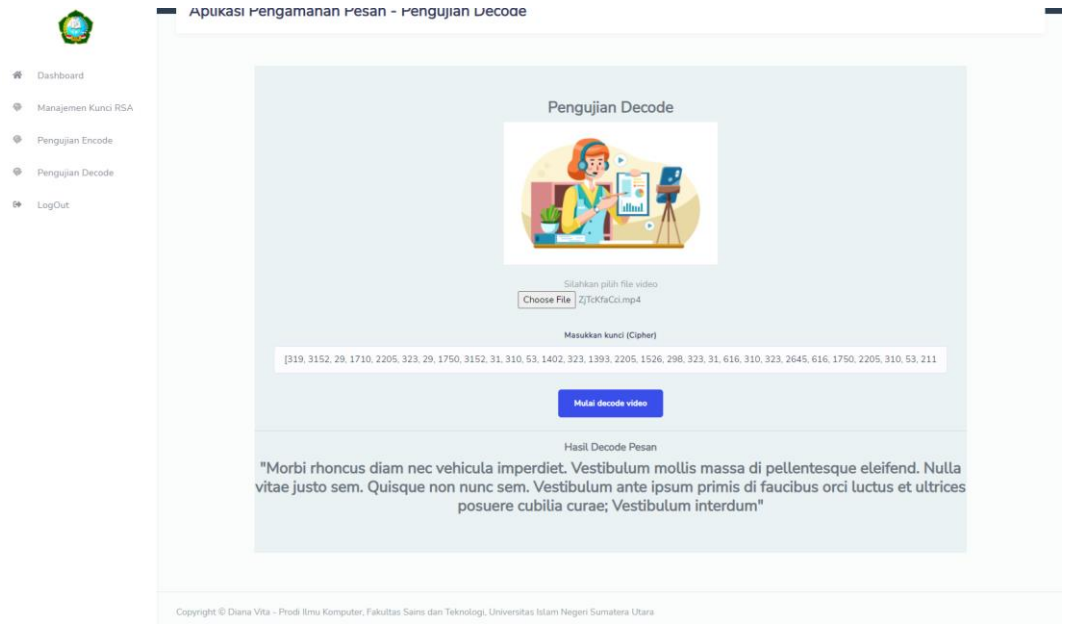


Gambar 4.37 Enkripsi pesan pengujian 6

Pada proses pengujian *encode* video yang di *input* menggunakan video dengan ukuran 81,4 MB, bedurasi 1 menit berformat (.mp4). Dalam melakukan proses *encode*, masukkan terlebih dahulu video yang akan diuji. Kemudian sistem akan memproses video sehingga hasil yang didapat berupa *framecutting* sebanyak 5 fps. Setelah itu proses pembuatan kunci dengan memasukkan 2 angka prima yang berbeda sehingga akan di dapat *public key* dan *private key*. Kemudian masukkan pesan yang akan disisipi bersamaan dengan kunci *public* yang telah didapatkan sebelumnya. Maka selanjutnya pesan akan di enkripsi oleh sistem. Pesan yang dimasukkan dalam pengujian pertama dengan panjang 1 kata.

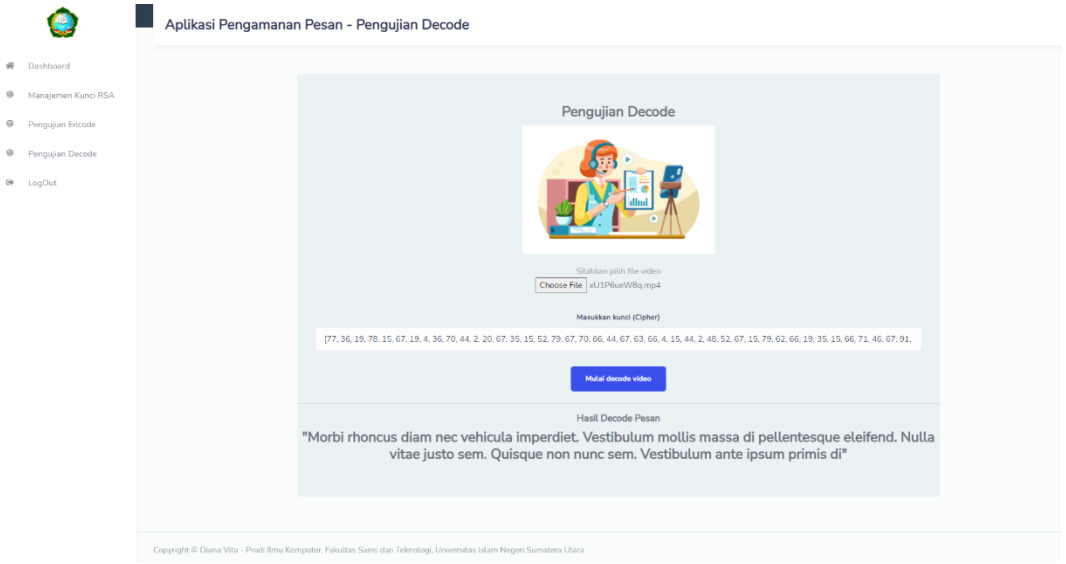
Selanjutnya akan dilakukan proses pengujian pengambilan kembali pesan yang telah disisipkan kedalam media video. Pesan yang telah dienkripsi akan menghasilkan stego *file* berupa video yang berisi pesan. Selanjutnya pesan dapat diambil kembali dengan menggunakan pengujian *decode*.

Apabila proses pengujian berhasil maka akan menampilkan pesan rahasia yang telah dimasukkan pada proses sebelumnya. Adapun hasil dari proses pengujian *decode* dapat dilihat pada gambar berikut:

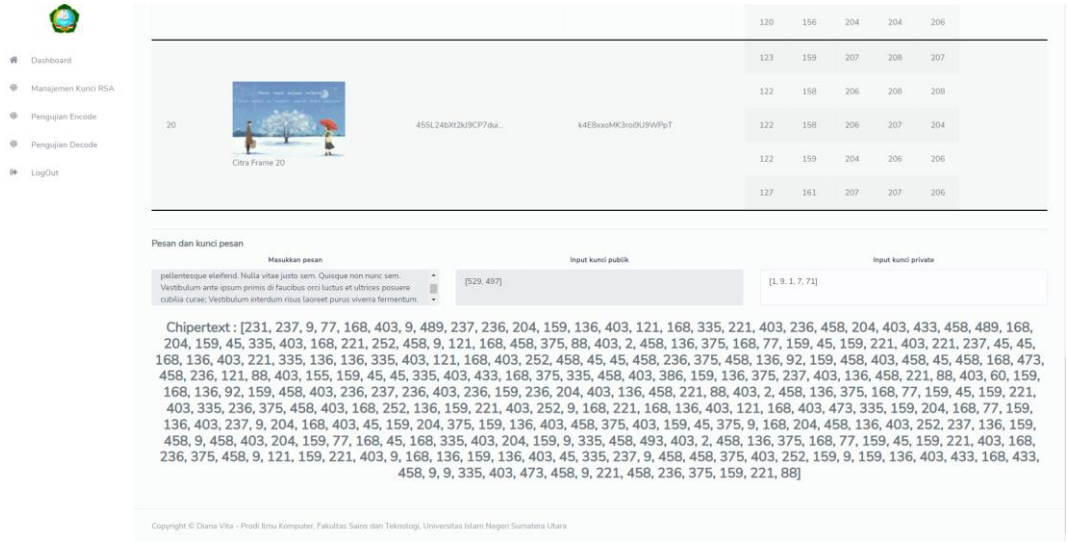


Gambar 4.38 Hasil *decode* pengujian 6

7. Pengujian video 7



Gambar 4.39 Pengujian *encode* pengujian 7

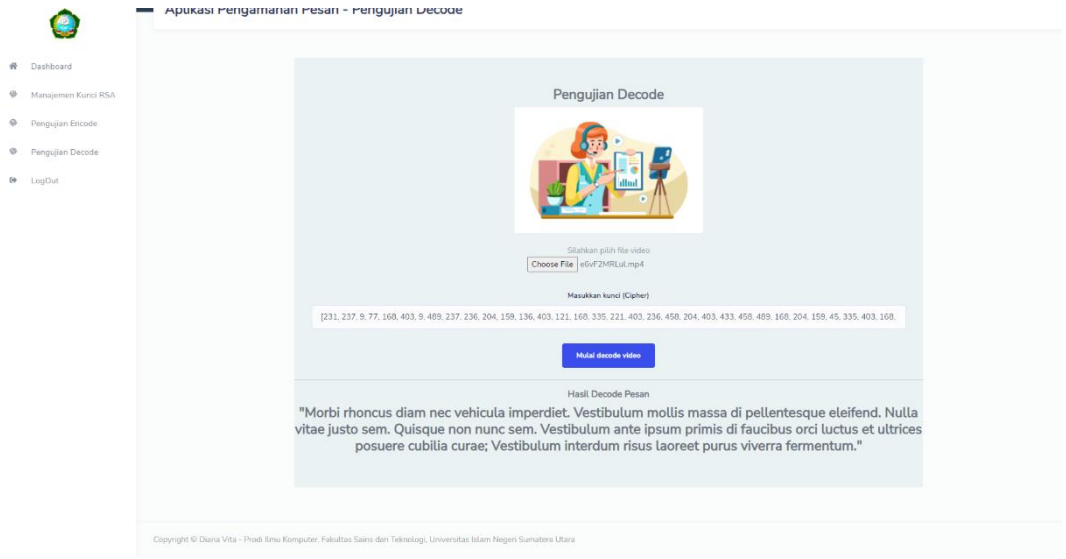


Gambar 4.40 Enkripsi pesan pengujian 7

Pada proses pengujian *encode* video yang di *input* menggunakan video dengan ukuran 22,1 MB, bedurasi 8 detik berformat (.mp4). Dalam melakukan proses *encode*, masukkan terlebih dahulu video yang akan diuji. Kemudian sistem akan memproses video sehingga hasil yang didapat berupa *framecutting* sebanyak 5 fps. Setelah itu proses pembuatan kunci dengan memasukkan 2 angka prima yang berbeda sehingga akan di dapat *public key* dan *private key*. Kemudian masukkan pesan yang akan disisipi bersamaan dengan kunci *public* yang telah didapatkan sebelumnya. Maka selanjutnya pesan akan di enkripsi oleh sistem. Pesan yang dimasukkan dalam pengujian pertama dengan panjang 1 kata.

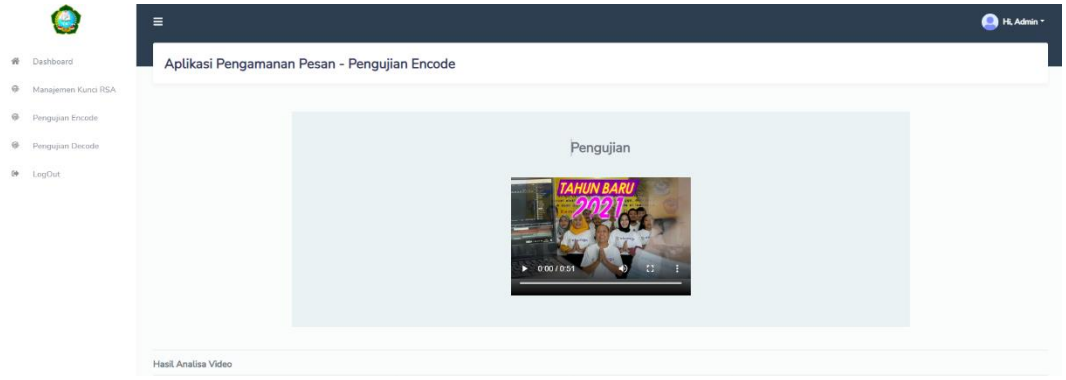
Selanjutnya akan dilakukan proses pengujian pengambilan kembali pesan yang telah disisipkan kedalam media video. Pesan yang telah dienkripsi akan menghasilkan stego *file* berupa video yang berisi pesan. Selanjutnya pesan dapat diambil kembali dengan menggunakan pengujian *decode*.

Apabila proses pengujian berhasil maka akan menampilkan pesan rahasia yang telah dimasukkan pada proses sebelumnya. Adapun hasil dari proses pengujian *decode* dapat dilihat pada gambar berikut:

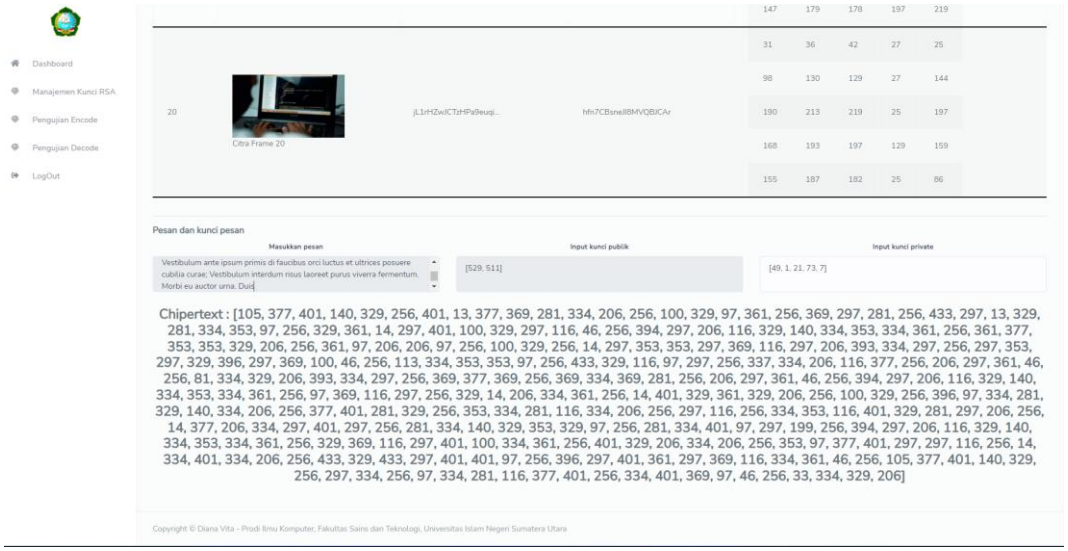


Gambar 4.41 Hasil *decode* pengujian 7

8. Pengujian video 8



Gambar 4.42 Pengujian *encode* pengujian 8

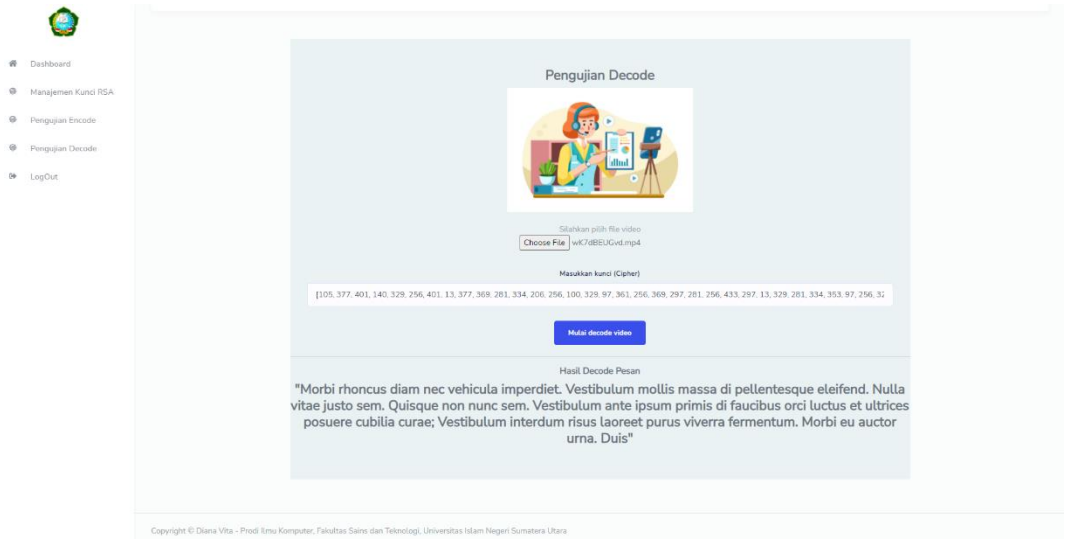


Gambar 4.43 Enkripsi pesan pengujian 8

Pada proses pengujian *encode* video yang di *input* menggunakan video dengan ukuran 1,14 MB, bedurasi 31 detik berformat (.mp4). Dalam melakukan proses *encode*, masukkan terlebih dahulu video yang akan diuji. Kemudian sistem akan memproses video sehingga hasil yang didapat berupa *framecutting* sebanyak 5 fps. Setelah itu proses pembuatan kunci dengan memasukkan 2 angka prima yang berbeda sehingga akan di dapat *public key* dan *private key*. Kemudian masukkan pesan yang akan disisipi bersamaan dengan kunci *public* yang telah didapatkan sebelumnya. Maka selanjutnya pesan akan di enkripsi oleh sistem. Pesan yang dimasukkan dalam pengujian pertama dengan panjang 1 kata.

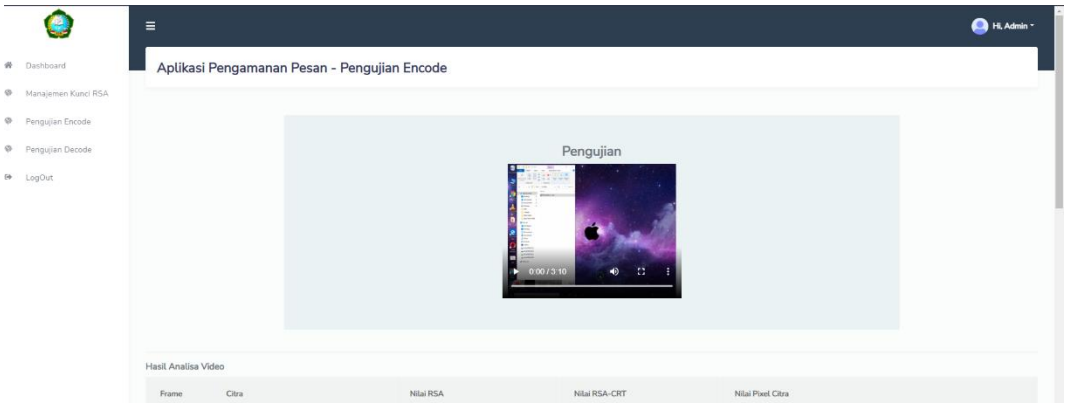
Selanjutnya akan dilakukan proses pengujian pengambilan kembali pesan yang telah disisipkan kedalam media video. Pesan yang telah dienkripsi akan menghasilkan stego *file* berupa video yang berisi pesan. Selanjutnya pesan dapat diambil kembali dengan menggunakan pengujian *decode*.

Apabila proses pengujian berhasil maka akan menampilkan pesan rahasia yang telah dimasukkan pada proses sebelumnya. Adapun hasil dari proses pengujian *decode* dapat dilihat pada gambar berikut:

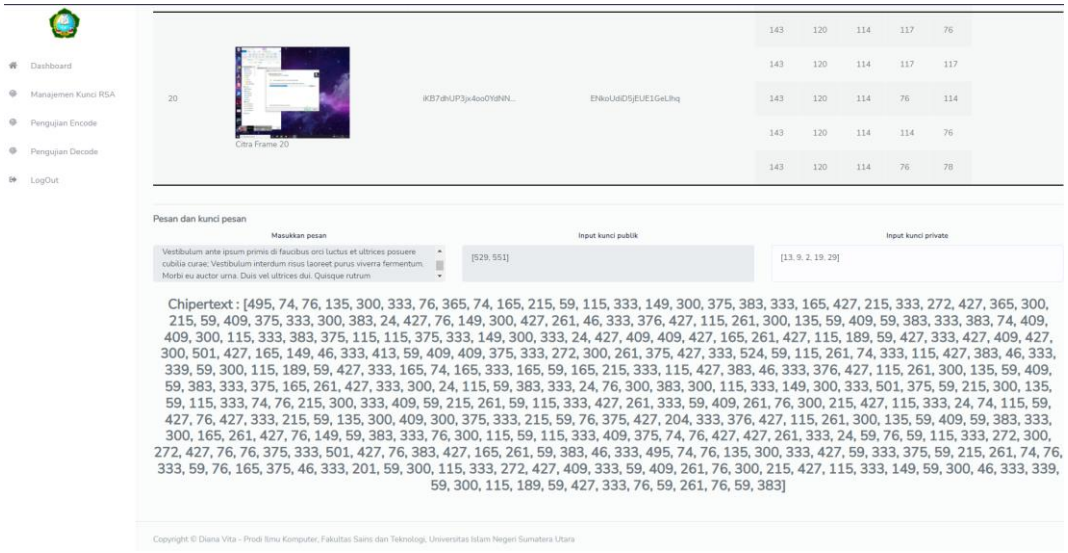


Gambar 4.44 Hasil *encode* pengujian 8

9. Pengujian video 9



Gambar 4.45 Pengujian *encode* pengujian 9

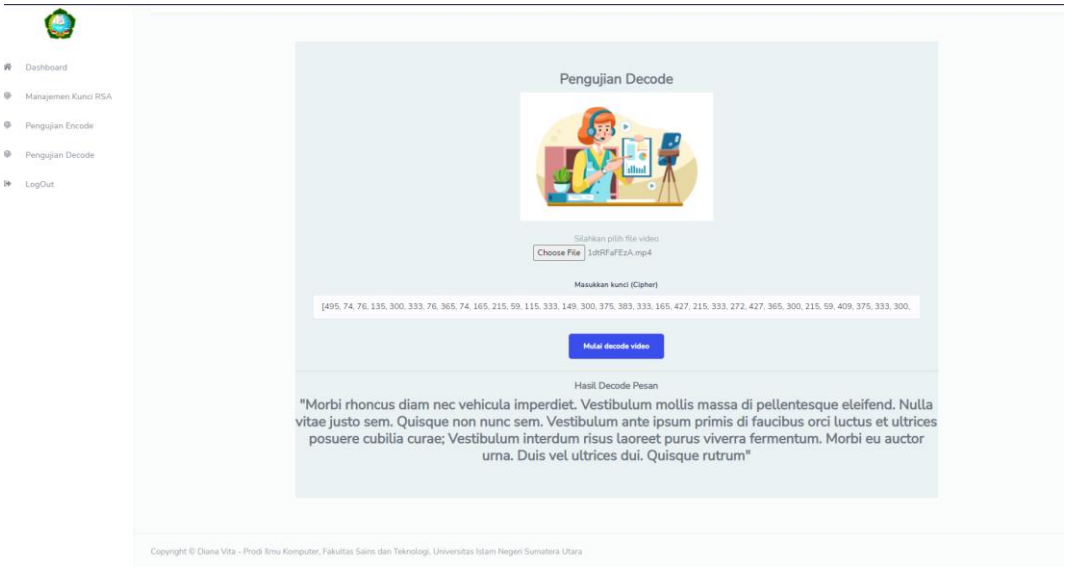


Gambar 4.46 Enkripsi pesan pengujian 9

Pada proses pengujian *encode* video yang di *input* menggunakan video dengan ukuran 98,6 MB, bedurasi 44 detik berformat (.mp4). Dalam melakukan proses *encode*, masukkan terlebih dahulu video yang akan diuji. Kemudian sistem akan memproses video sehingga hasil yang didapat berupa *framecutting* sebanyak 5 fps. Setelah itu proses pembuatan kunci dengan memasukkan 2 angka prima yang berbeda sehingga akan di dapat *public key* dan *private key*. Kemudian masukkan pesan yang akan disisipi bersamaan dengan kunci *public* yang telah didapatkan sebelumnya. Maka selanjutnya pesan akan di enkripsi oleh sistem. Pesan yang dimasukkan dalam pengujian pertama dengan panjang 1 kata.

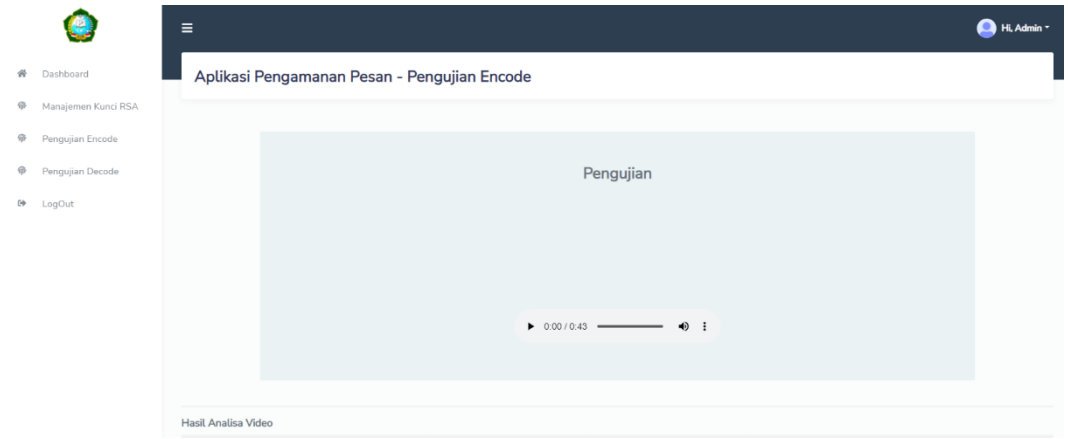
Selanjutnya akan dilakukan proses pengujian pengambilan kembali pesan yang telah disisipkan kedalam media video. Pesan yang telah dienkripsi akan menghasilkan stego *file* berupa video yang berisi pesan. Selanjutnya pesan dapat diambil kembali dengan menggunakan pengujian *decode*.

Apabila proses pengujian berhasil maka akan menampilkan pesan rahasia yang telah dimasukkan pada proses sebelumnya. Adapun hasil dari proses pengujian *decode* dapat dilihat pada gambar berikut:

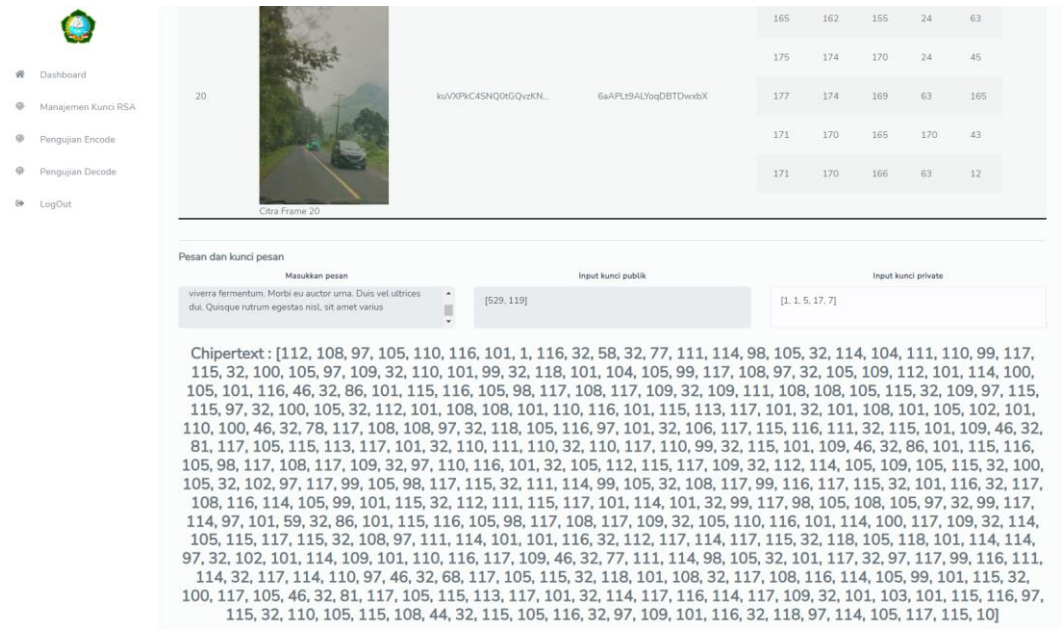


Gambar 4.47 Hasil *decode* pengujian 9

10. Pengujian video 10



Gambar 4.48 Pengujian *encode* pengujian 10

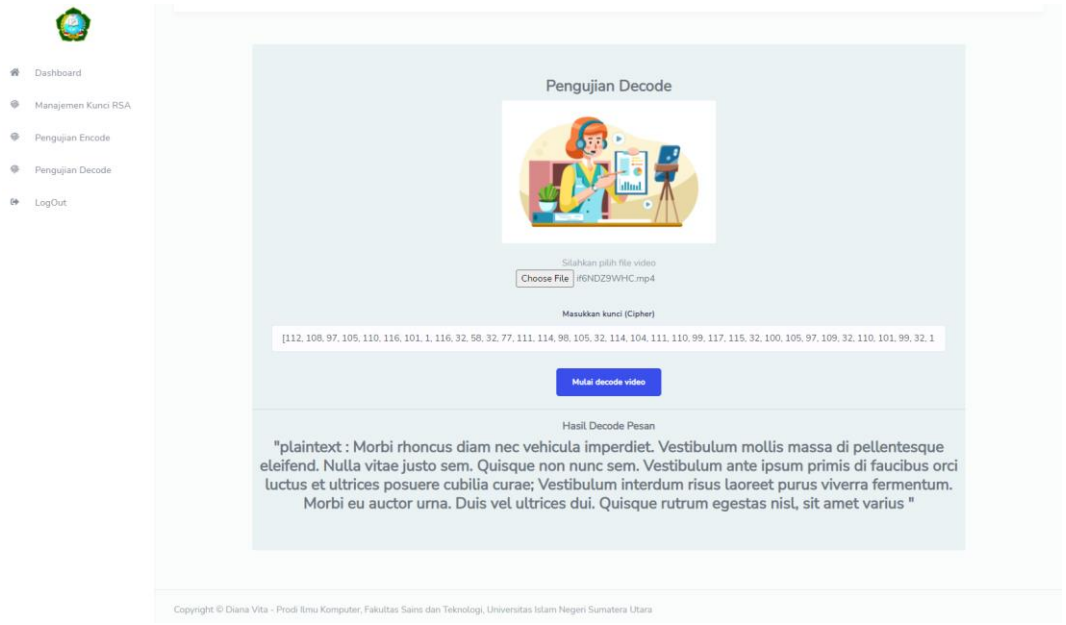


Gambar 4.49 Enkripsi pesan pengujian 10

Pada proses pengujian *encode* video yang di *input* menggunakan video dengan ukuran 59,8 MB, bedurasi 43 detik berformat (.mp4). Dalam melakukan proses *encode*, masukkan terlebih dahulu video yang akan diuji. Kemudian sistem akan memproses video sehingga hasil yang didapat berupa *framecutting* sebanyak 5 fps. Setelah itu proses pembuatan kunci dengan memasukkan 2 angka prima yang berbeda sehingga akan di dapat *public key* dan *private key*. Kemudian masukkan pesan yang akan disisipi bersamaan dengan kunci *public* yang telah didapatkan sebelumnya. Maka selanjutnya pesan akan di enkripsi oleh sistem. Pesan yang dimasukkan dalam pengujian pertama dengan panjang 1 kata.

Selanjutnya akan dilakukan proses pengujian pengambilan kembali pesan yang telah disisipkan kedalam media video. Pesan yang telah dienkripsi akan menghasilkan stego *file* berupa video yang berisi pesan. Selanjutnya pesan dapat diambil kembali dengan menggunakan pengujian *decode*.

Apabila proses pengujian berhasil maka akan menampilkan pesan rahasia yang telah dimasukkan pada proses sebelumnya. Adapun hasil dari proses pengujian *decode* dapat dilihat pada gambar berikut:



Gambar 4.50 Hasil *decode* pengujian 10

4.3.2 Hasil

Di bawah ini merupakan hasil dari proses pengujian terhadap masing-masing video dengan variasi ukuran dan panjang pesan.

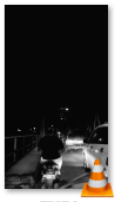
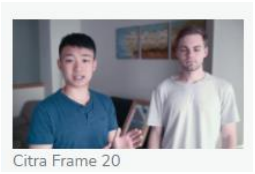
a. *Imperceptibility*

Dilakukan dengan membandingkan kualitas video hasil dengan video asli yang terlihat oleh *Human Visual System* (HVS). Video yang telah diuji memilki ukuran yang berbeda dan panjang pesan yang disisipi memiliki ukuran berbeda pula, sehingga dapat dilihat pengaruh dari masing-masing metode yang diterapkan.

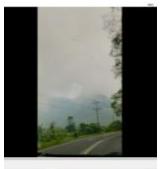
Tabel 4.3 Ukuran Citra Sebelum dan Setelah Disisipi Pesan

No.	Video	Ukuran Video Asli	Ukuran Stego Video	Jumlah Kata
1	 TEST 1	4,19 Mb	4,19 Mb	3

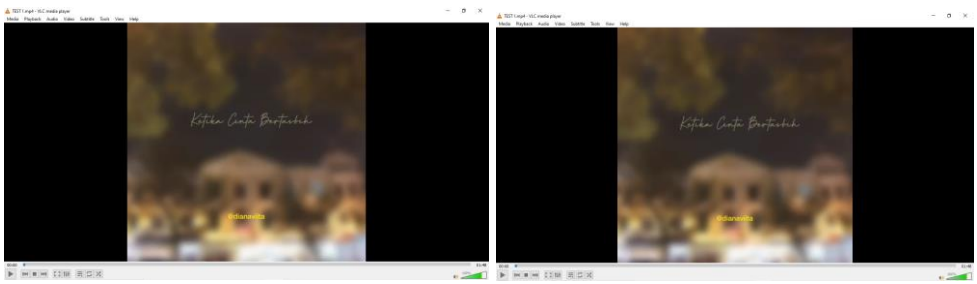
Tabel 4.4 Lanjutan Ukuran Citra Sebelum dan Setelah Disisipi Pesan

2		93,3 Mb	93,3 Mb	15
3		33,4 Mb	33,4 Mb	20
4		39,5 Mb	39,5 Mb	25
5		91,2 Mb	91,2 Mb	30
6		81,4 Mb	81,4 Mb	35
7		22,1 Mb	22,1 Mb	40
8		1,13 Mb	1,13 Mb	45

Tabel 4.5 Lanjutan Ukuran Citra Sebelum dan Sesudah Disisipi Pesan

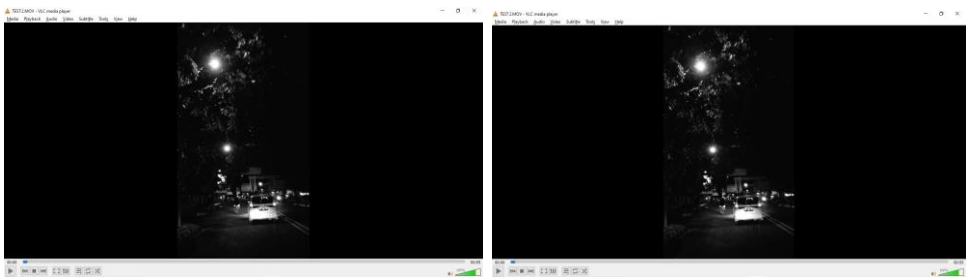
9		98,6 Mb	98,6 Mb	50
10		59,8 Mb	59,8 Mb	55

Dari tabel 4.3 sampai 4.5 dapat dilihat bahwa video asli tidak mengalami perubahan ukuran setelah mengalami proses steganografi. Kemudian seluruh video memiliki tampak yang sama dan tidak ada yang berubah jika dilihat secara *Human Visual System* (HVS) yang dapat dilihat pada gambar berikut:



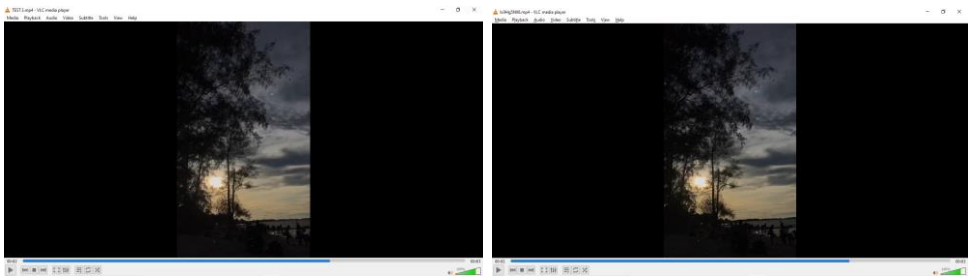
Video asli Stego Video

Gambar 4.51 Video Test 1 asli dan setelah disisipi pesan



Video asli Stego Video

Gambar 4.52 Video Test 2 asli dan setelah disisipi pesan



Video asli Stego Video

Gambar 4.53 Video test 3 asli dan setelah disisipi pesan

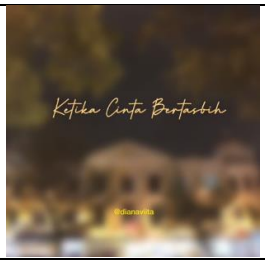
b. *Fidelity*

Dalam pengujian mutu citra pengukuran dilakukan menggunakan nilai *Mean Square Error* (MSE) dan *Peak Signal to Noise Ratio* (PSNR). Nilai tersebut didapat dari *stego image* hasil proses penyisipan *chipertext* ke dalam *cover image* untuk metode steganografi dikombinasikan dengan metode kriptografi.

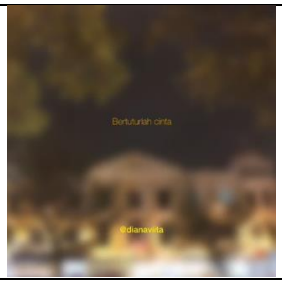
Pengujian dilakukan dengan menggunakan 5 sampel *frame* yang didapat dari 1 video berformat (*.mp4). Data hasil pengujian MSE dan PSNR dengan menggunakan 5 sampel *framecutting* dapat dilihat pada tabel berikut.

Tabel 4.6 Hasil Pengujian *Fidelity*

*Frame*_1				**MSE**	0,38
				PSNR	30,59

*Frame*_2				**MSE**	0,38
				PSNR	30,59

Tabel 4.7 Lanjutan Hasil Pengujian *Fidelity*

*Frame*_3		**MSE**	0,38
		PSNR	30,59
*Frame*_4		**MSE**	0,38
		PSNR	30,68
*Frame*_5		**MSE**	0,37
		PSNR	30,58

Dari tabel tersebut diperoleh bahwa nilai rata-rata PSNR dari seluruh *frame* adalah 30,61 Db dan nilai MSE 0,378.

c. Recovery

Pengujian ini dilakukan dengan melakukan proses *extraction* pada seluruh stego video. Uji *recoverable* dilakukan pada seluruh video asli dengan pesan yang jumlah kata yang bervariasi. Seluruh hasil pengujian menunjukkan bahwa *embedded message* bisa diperoleh kembali secara utuh melalui proses *extraction* yang dapat dilihat pada Tabel 4.4.

Tabel 4.8 Hasil Uji *Recovery* pada Stego Video

No.	Nama Video	*Recovery*	Jumlah Kata
1	Test 1.mp4	Berhasil	3
2	Test 2.mp4	Berhasil	15
3	Test 3.mp4	Berhasil	20
4	Test 4.mp4	Berhasil	25
5	Test 5.mp4	Berhasil	30
6	Test 6.mp4	Berhasil	35
7	Test 7.mp4	Berhasil	40
8	Test 8.mp4	Berhasil	45
9	Test 9.mp4	Berhasil	50
10	Test 10.mp4	Berhasil	55

Dari tabel tersebut diperoleh bahwa semua video yang melalui proses steganografi dapat diekstrak dan pesan yang didapat sesuai dengan pesan aslinya.

d. Robustness

Pengujian *robustness* atau ketahanan stego video terhadap proses manipulasi video, dilakukan penambahan *lighting* (pencahayaan). Kemudian pesan rahasia diekstrak dari stego video.

Seluruh hasil pengujian terhadap semua video menunjukkan bahwa proses *extraction* tidak menghasilkan pesan rahasia yang diinginkan, dapat dilihat pada Tabel 4.9.

Tabel. 4.9 Hasil Uji *Robustness*

No.	Nama Video	Robustness
1	Test 1.mp4	Gagal
2	Test 2.mp4	Gagal
3	Test 3.mp4	Gagal
4	Test 4.mp4	Gagal
5	Test 5.mp4	Gagal
6	Test 6.mp4	Gagal
7	Test 7.mp4	Gagal
8	Test 8.mp4	Gagal
9	Test 9.mp4	Gagal
10	Test 10.mp4	Gagal

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Pada bab sebelumnya, penulis telah melakukan analisis penelitian terhadap pengujian pengamanan pesan dengan metode kriptografi RSA-CRT kombinasi metode steganografi LSB pada media video. Dalam penelitian ini dapat ditarik kesimpulan bahwa:

1. Dengan adanya aplikasi ini, maka pesan yang dianggap penting dapat terjaga kerahasiaannya dari pihak yang tidak bertanggung jawab.
2. Metode kriptografi RSA-CRT kombinasi dengan metode steganografi LSB pada media video berhasil diimplementasikan dalam pengembangan aplikasi pengamanan pesan yang terdiri dari dua proses utama.
3. Berdasarkan hasil pengujian *imperceptibility* pada seluruh video, ukuran stego video tidak berubah atau tetap sama dengan video asli. Sehingga pada pengujian ini memenuhi aspek *imperceptibility*.
4. Pada pengujian *fidelity*, seluruh video memiliki tampak yang sama dan tidak ada yang berubah jika dilihat secara *Human Visual System* (HVS). Sedangkan berdasarkan nilai PSNR, seluruh video yang telah diuji memiliki nilai PSNR rata-rata 30,61 dB mengindikasikan kualitas yang baik, dari hasil PSNR tersebut dapat disimpulkan bahwa kombinasi kriptografi RSA-CRT dengan steganografi LSB merupakan metode yan cukup baik ditinjau dari kemiripan stego*file* yang dihasilkan karena memenuhi standar yaitu lebih dari 30-40 desibels.
5. Berdasarkan pengujian *recovery* pada seluruh stego video, pesan dapat diambil kembali secara utuh melalui proses *extraction*, sehingga memenuhi aspek *recovery*.
6. Berdasarkan pengujian robustness pada seluruh stego video, pesan tidak dapat diambil secara utuh setelah melalui proses penambahan *lighting* pada stego video, sehingga pada proses ini tidak memenuhi aspek *robustness*.

5.2 Saran

Selain menarik beberapa kesimpulan, dapat pula diajukan saran-saran untuk pengembangan sistem dan penelitian selanjutnya, yaitu: Pengembangan Aplikasi pengamanan pesan dengan metode kriptografi RSA CRT dan metode steganografi LSB disarankan dengan menambahkan metode steganografi *Linear Congruential Generator* dengan inputan beragam ekstensi *file*. Selain itu, untuk penelitian selanjutnya diharapkan dapat membangun aplikasi yang diterapkan pada perangkat Android, iOs atau yang lainnya. Menggunakan ukuran *framecutting* yang lebih besar dari maksimal 30 fps untuk video yang digunakan sebagai *cover file*.

DAFTAR PUSTAKA

A, M., & Painem. (2020). Pengamanan *File* Gambar Pada Media Video dengan Kriptografi Algoritma RSA dan Steganografi Algoritma End of *File* (EOF). *Jurnal Informatik*.

Abdullah, H. A., Abdulameeer, A. A., & Hussein, I. F. (2015). Audio Stgeganography and Security by using Cryptography. *i-manager's Journal on Information Technology, 4*, (4), 17-24. Retrieved from https://doi.org/10.26634/jit.4.4.3644

Ali, L. (2014). Analisis dan Implementasi Algoritma Exclusive OR dan *Least Significant Bit* untuk Penyisipan *File* Gambar Pada Gambar .

Andono, P., Sutojo, T., & Muliono. (2017). *Pengolahan Citra Digital.* Yogyakarta: ANDI.

Andrian. (2019). Perancangan Perangkat Lunak Kompresi Citra Menggunakan Transformasi Wavelet dan PCA. *ALGORITMA: Jurnal Ilmu Komputer dan Informatika, 3*(1), 1-8.

Anti, U., Kridalaksana, A. H., & Khairina, D. M. (2017). Steganografi Pada Video Menggunakan Metode *Least Significant Bit* (LSB) Dan End Of *File* (EOF). *Jurnal Informatika Mulawarman, 12*(2).

Arief, A., & Saputra, R. (2016). Implementasi Kriptografi Kunci Publlik dengan Algoritma RSA-CRT pada Aplikasi Instant Messaging. *Scientific Journal of Informatics*.

Ekodeck, S., & Ndoundam, R. (2016). PDF Steganography Based On Chinese Remainder Theorem. *Journal Of Information Security And Application*.

Firdaus, Y., & Suartana, M. (2020). Implementasi Steganografi Dengan Metode Pixel Value Differencin (PVD) pada Gambar JPG dan PNG. *Journal of Informastics and Computer Science, 01*(04).

Hidayatullah, P. (2017). *Pengolahan Citra Digital.* Bandung: Informatika Bandung.

Kromodimoeljo, S. (2010). *Teori & Aplikasi Kriptografi.* SPK IT Consulting.

Landriandani, M. F. (2020). Sistem Pengamanan Pesan Dengan Metode Kriptografi RSA-CRT dan Metode Steganografi Linear Congruental Generator Pada Media Citra Digital.

Malvi, A., & Painem. (2020). Pengamanan *File* Gambar pada Media Video dengan Kriptografi Algoritma RSA dan Steganografi Algoritma End of *File* (EOF).

Manullang, D. (2019). Penyisipan Pesan Ke Dalam *File* Video Menerapkan Metode Chinise Remainder Theorem. *KOMIK (Konferensi Nasional Teknologi Informasi dan Komputer), 03*(01).

Muis, A. (2011). Steganografi Metode Lesat Significant Bit Pada Citra Bitmap Dengan Teknik Kompres Data Dan Ekspansi Wadah.

Munir, R. (2019). *Kriptografi. 2.* Bandung: Informatika Bandung.

Nasution, N. (2017). Kombinasi RSA-CRT dengan Random LSB untuk Keamanan Data di Kanwil Kementrian Agama Prov. Sumatera Utara. *Jurnal Sistem Informasi*.

Ningtyas, A. (2017). Implementasi Hybrid Cryptosystem Algoritma RSA-CRT Dan Algoritma RC4_ Dalam Pengamanan *File* Teks.

Panjaitan, Z., Ibnutama, K., & Suryanata, M. (2019). Penggunaan Chinese Remainder Theorem (CRT) pada Algoritma RSA. *jurnal SAINTIKOM, 18*(1), 41-46.

Sinaga, A. S. (2018). Analisis Perbandingan *Least Significant Bit* + 1 Dan *Least Significant Bit*+Liniear Congruential Generator Pada Steganografi *File* Citra.

Sitorus, L. (2015). *Algoritma dan Pemrograman.* Yogyakarta: ANDI.

Soediro. (2018). Prinsip Keamanan, Privasi, Dan Etika Dalam Undang-Undang Informasi Dan Transaksi Elektronik Dalam Perspektif Hukum Islam. *Jurnal Kosmik Hukum, 18*(02).

Sriani, Triase, & Khairuna., d. (2017). Pendekomposisian Citra Digital Dengan Algoritma DWT. *01*(01), 35-39.

Supardi, R. (2017). Implementasi Metode Bit Matching Untuk Keamanan Pesan Teks Menggunakan Visual Basic Net. *Jurnal Teknologi Informasi, 1*(2), 197-210.

Suryansah, A., Habibi, R., & Awangga, R. (2020). *Penggunaan Face Recognition Untuk Akses Ruangan.* Bandung: Kreatif Industri Nusantara.

Syahrudin, A., & Kurniawan, T. (2018). *Input* dan *Output* Pada Bahasa Pemrograman *Python*. *Jurnal Dasar Pemrograman Python STMIK*.

Syawal, M., Fikriansyah, D., & Agani, N. (2016). Implementasi Teknik Steganografi Menggunakan Algoritma Vigenere Chiper Dan Metode LSB. *Jurnal TICOM, 4*(3), 91.

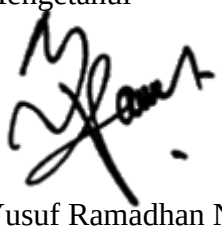
Tjolleng, A. (2017). *Pengantar Pemrograman MATLAB.* Jakarta: PT Elex Media Komputindo.

UJI COBA TINGKAT KEBERHASILAN

Nama Penguji: Yusuf Ramadhan Nasution, M.Kom

No.	Pengujian	Keterangan	
		Berhasil	Tidak Berhasil
1.	Login aplikasi	✓	
2.	Proses pembuatan kunci dengan menginput 2 bilangan prima	✓	
3.	Upload video sebagai uji encode	✓	
4.	Proses analisa video	✓	
5.	Pengujian enkripsi pesan dan chipertext pada video	✓	
6.	Video tersimpan sebagai Stego-Video	✓	
7.	Upload video sebagai uji decode	✓	
8.	Proses input chipertext	✓	
9.	Proses dekripsi pesan dari Stego-video	✓	

Mengetahui



Yusuf Ramadhan Nasution, M.Kom
NIB. 1100000075

LISTING PROGRAM

1. File urls.py

```
from django.contrib import admin
from django.urls import path

from home import views as home_app
from login import views as login_app
from main_app import views as main_app

urlpatterns = [
    path('', home_app.home_page),
    path('login/', login_app.login_page),
    path('login/proses', login_app.login_proses),
    path('dashboard/', main_app.main_dash),
    path('dashboard/beranda', main_app.beranda),
    path('dashboard/pengujian', main_app.pengujian),
    path('dashboard/pengujian/upload-video',
main_app.upload_video),
    path('dashboard/tes-enkripsi-rsa',
main_app.tes_enkripsi_rsa),
    path('dashboard/pengujian/proses-enkripsi',
main_app.proses_enkripsi),
    path('dashboard/buat-kunci-rsa',
main_app.buat_kunci_rsa),
    path('dashboard/pengujian-decode',
main_app.pengujian_decode),
    path('dashboard/proses-kunci-baru',
main_app.buat_kunci_baru),
    path('dashboard/proses-hapus-kunci',
main_app.proses_hapus_kunci),
    path('dashboard/proses-decode',
main_app.proses_decode)
```

```
        ]
```

2. File settings.py

```
    "
""
            Django settings for Diana_Django_Project project.

            Generated by 'django-admin startproject' using
        Django 3.1.4.

            For more information on this file, see
            https://docs.djangoproject.com/en/3.1/topics/set
        tings/

            For the full list of settings and their values,
        see
            https://docs.djangoproject.com/en/3.1/ref/settin
        gs/
            """

            from pathlib import Path

            # Build paths inside the project like this:
        BASE_DIR / 'subdir'.
            BASE_DIR                                    =
        Path(__file__).resolve().parent.parent



            # Quick-start development settings - unsuitable
        for production
            #                                           See
        https://docs.djangoproject.com/en/3.1/howto/deployment
        /checklist/

            # SECURITY WARNING: keep the secret key used in
        production secret!
```

```
      SECRET_KEY  =  'n8@^z$80v75xe@#vsx=iqb3=y(anz=q-
y(+$q%#c*8n8y6j$_3'

      # SECURITY WARNING: don't run with debug turned
on in production!
      DEBUG = True

      ALLOWED_HOSTS = ['128.199.226.166','127.0.0.1']



      # Application definition

      INSTALLED_APPS = [
          'django.contrib.admin',
          'django.contrib.auth',
          'django.contrib.contenttypes',
          'django.contrib.sessions',
          'django.contrib.messages',
          'django.contrib.staticfiles',
          'home',
          'login',
          'main_app'
      ]

      MIDDLEWARE = [

'django.middleware.security.SecurityMiddleware',

'django.contrib.sessions.middleware.SessionMiddleware'
,
          'django.middleware.common.CommonMiddleware',
          'django.middleware.csrf.CsrfViewMiddleware',

'django.contrib.auth.middleware.AuthenticationMiddlewa
re',
```

```
'django.contrib.messages.middleware.MessageMiddleware'
,
'django.middleware.clickjacking.XFrameOptionsMiddlewar
e',
      ]

      ROOT_URLCONF = 'Diana_Django_Project.urls'

      TEMPLATES = [
          {
              'BACKEND':
'django.template.backends.django.DjangoTemplates',
              'DIRS': ['bind'],
              'APP_DIRS': True,
              'OPTIONS': {
                  'context_processors': [

'django.template.context_processors.debug',

'django.template.context_processors.request',

'django.contrib.auth.context_processors.auth',

'django.contrib.messages.context_processors.messages',

'django_base_url.context_processors.base_url',
                  ],
              },
          },
      ]

      WSGI_APPLICATION                                =
'Diana_Django_Project.wsgi.application'
```

```
    # Database
    # https://docs.djangoproject.com/en/3.1/ref/settings/#databases

    DATABASES = {
        'default': {
            'ENGINE': 'django.db.backends.mysql',
            'NAME': 'dbs_diana',
            'USER' : 'root',
            'PASSWORD' : ''
        }
    }


    # Password validation
    # https://docs.djangoproject.com/en/3.1/ref/settings/#auth-password-validators

    AUTH_PASSWORD_VALIDATORS = [
        {
            'NAME': 'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',
        },
        {
            'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',
        },
        {
            'NAME': 'django.contrib.auth.password_validation.CommonPasswordValidator',
        },
        {
```

```
            'NAME':
'django.contrib.auth.password_validation.NumericPasswo
rdValidator',
        },
    ]



    # Internationalization
    #
https://docs.djangoproject.com/en/3.1/topics/i18n/

    LANGUAGE_CODE = 'en-us'

    TIME_ZONE = 'UTC'

    USE_I18N = True

    USE_L10N = True

    USE_TZ = True



    # Static files (CSS, JavaScript, Images)
    #
https://docs.djangoproject.com/en/3.1/howto/static-
files/

    BASE_URL="http://127.0.0.1:7001/"

    STATIC_URL = '/ladun/'

    STATICFILES_DIRS = [
        BASE_DIR / "ladun",
    ]
```

3. File view.py (core)

```
f
rom
django.
shortcu
ts
import
render
from django.views.decorators.csrf import
csrf_exempt
from django.http import JsonResponse
from django.core.files.storage import
FileSystemStorage
from django.utils.crypto import
get_random_string
from stegano import lsb
#from Crypto.PublicKey import RSA
#from Crypto.Cipher import PKCS1_OAEP
from PIL import Image

import binascii
import cv2
import math
import hashlib
import datetime
import random
from numpy import mod
import sympy
import time
import matplotlib.pyplot as plt

from .models import Encode_Pesan
from .models import Kunci_RSA
```

```
    # Create your views here.
    def main_dash(request):
        ip_address = request.META['REMOTE_ADDR']

        context = {
            'aplikasi' : '-',
            'developer' : 'Diana Vita',
            'ip_address' : ip_address
        }
        return                      render(request,
'dashboard/main.html', context)

    def beranda(request):
        context = {
            'status' : 'sukses'
        }
        return                      render(request,
'dashboard/beranda.html', context)

    def pengujian(request):
        context = {
            'status' : 'sukses'
        }
        return                      render(request,
'dashboard/pengujian.html', context)

    def pengujian_decode(request):
        context = {
            'status' : 'sukses'
        }
        return                      render(request,
'dashboard/pengujian-decode.html', context)

    def buat_kunci_rsa(request):
```

```
          dataKunciRsa                                =
Kunci_RSA.objects.all().values()
          context = {
              'status' : 'sukses',
              'kunciRsa' : dataKunciRsa
          }
          return   render(request,   'dashboard/buat-
kunci-rsa.html', context)

      @csrf_exempt
      def buat_kunci_baru(request):
          # {'teks':teks, 'kunci':kunci}
          teks = request.POST['teks']
          kunci = request.POST['kunci']
          kunci2 = request.POST['kunci2']



          # chiper = encrypt(teks, [e, n])


          #cek bilangan prima
          cek_prima = checkBilanganPrima(int(kunci))
          cek_prima2                                  =
checkBilanganPrima(int(kunci2))
          if(cek_prima == True or cek_prima2 == True):
              prima = False
              status = 'not_prime_number'
              hasil = ''
          else:
              prima = True
              # cek apakah nama kunci sudah ada
              cek_kunci                               =
Kunci_RSA.objects.filter(nama__contains=teks).count(
)
              if(cek_kunci < 1):
                  p = int(kunci)
```

```
                    q = int(kunci2)
                    n = p * q
                    phi = (p - 1) * (q - 1)
                    # nilai e di tentukan sendiri
                    e = 529
                    d = modular_inverse(e, phi)
                    dp = d % (p - 1)
                    dq = d % (q - 1)
                    qInv = modular_inverse(q, p)
                    public_key = [e, n]
                    p_key_s                          =
listToString(str(public_key))
                    private_key = [dp, dq, qInv, p, q]
                    priv_key_s                       =
listToString(str(private_key))
                    save_kunci                       =
Kunci_RSA.objects.create(nama=teks, p=p, q=q, dp=dp,
dq=dq,         q_inv=qInv,         public_key=p_key_s,
private_key=priv_key_s)
                    save_kunci.save()
                    status = 'success'
                else:
                    status = 'double_kunci'

            context = {
                'status' : status
            }
            return JsonResponse(context, safe=False)

        @csrf_exempt
        def proses_hapus_kunci(request):
            kdKunci = request.POST['kdKunci']

Kunci_RSA.objects.filter(kd_kunci__contains=kdKunci)
.delete()
```

```
            context = {
                'kdKunci' : kdKunci,
                'status' : 'sukses'
            }
            return JsonResponse(context, safe=False)


        @csrf_exempt
        def proses_decode(request):
            video = request.FILES['txtVideo']
            kunciRsa = request.POST['kunciRsa']
            video_name = video.name
            kd_pengujian  =  video_name.replace(".mp4",
"")
            # cek apakah video terdaftar
            total_v_data                                =
Encode_Pesan.objects.filter(kd_uji__contains=kd_peng
ujian).count()
            if(total_v_data < 1):
                status = 'no_video'
                pesan = ''
            else:
                # cek apakah kunci rsa cocok
                total_rsa_data                          =
Encode_Pesan.objects.filter(kd_uji__contains=kd_peng
ujian).filter(chiper_text__contains=kunciRsa).count(
)
                if(total_rsa_data < 1):
                    status = 'no_rsa_key'
                    pesan = ''
                else:
                    status = 'sukses'
                    data_decode                         =
Encode_Pesan.objects.filter(kd_uji__contains=kd_peng
ujian).filter(chiper_text__contains=kunciRsa).first(
)
                    pesan = data_decode.message_encode
```

```
            context = {
                'kd_pengujian' : kd_pengujian,
                'kunci_rsa' : kunciRsa,
                'status' : status,
                'pesan' : pesan
            }
            return JsonResponse(context, safe=False)

        @csrf_exempt
        def upload_video(request):
            count = 0
            kdPengujian = get_random_string(10)
            video = request.FILES['txtVideo']
            fs = FileSystemStorage()

fs.save("ladun/data_video_upload/"+kdPengujian+".mp4
", video)

fs.save("ladun/data_video_hash/"+kdPengujian+".mp4",
video)
            # alamat video yang sudah di upload
            videoPath                                   =
"ladun/data_video_upload/"+kdPengujian+".mp4"
            captureData = cv2.VideoCapture(videoPath)
            frameRate = captureData.get(5)
            x = 1
            pic_data = []
            while(captureData.isOpened()):
                idFrame = captureData.get(1)
                ret, frame = captureData.read()
                if(ret != True):
                    break
                if(idFrame  %  math.floor(frameRate)  ==
0):
```

```
                    filename                                =
"ladun/keras_proses/"+kdPengujian+"_frame_%d_.jpg"  %
count; count+=1
                    cv2.imwrite(filename, frame)
                    img = Image.open(filename)
                    colors = img.getpixel((320,240))
                    pic_data.append(colors)

            pesan = hidden_message(videoPath)
            #rsa
            newRsaF1 = generateRsa(kdPengujian)
            newRsaF5 = generateRsa(kdPengujian)
            newRsaF10 = generateRsa(kdPengujian)
            newRsaF15 = generateRsa(kdPengujian)
            newRsaF20 = generateRsa(kdPengujian)

            context = {
                'kdUji' : kdPengujian,
                'status' : 'sukses',
                'kunci' : pesan,
                'total_citra' : count,
                'rsaF1' : newRsaF1,
                'rsaF5' : newRsaF5,
                'rsaF10' : newRsaF10,
                'rsaF15' : newRsaF15,
                'rsaF20' : newRsaF20,
                'pic_data' : pic_data
            }
            return JsonResponse(context, safe=False)

        @csrf_exempt
        def tes_enkripsi_rsa(request):
            newRsa = generateRsa("Diana Vita")
```

```
            print(newRsa)
            context = {
                'status' : 'sukses',
                'newRsa' : newRsa
            }
            return JsonResponse(context, safe=False)

        @csrf_exempt
        def proses_enkripsi(request):
            kdUji = request.POST['kdUji']
            pesan = request.POST['pesan']
            kunciPublik = request.POST['kunci']
            kunciPrivate = request.POST['kunciPrivate']
            hash_key = request.POST['hashKey']

            now = datetime.datetime.now()
            total_kunci                                    =
Kunci_RSA.objects.filter(public_key__contains=kunciP
ublik).filter(private_key__contains=kunciPrivate).co
unt()
            if total_kunci > 0 :
                dataKunci                                  =
Kunci_RSA.objects.filter(public_key__contains=kunciP
ublik).filter(private_key__contains=kunciPrivate).fi
rst()
                # p = dataKunci.p
                # q = dataKunci.q
                # dp = dataKunci.dp
                # dq = dataKunci.dq
                key_pub_1   =   kunciPublik.replace("[",
"")
                key_pub_2 = key_pub_1.replace("]", "")
                key_pub_s = key_pub_2.split(", ")
                e = int(key_pub_s[0])
                n = int(key_pub_s[1])
```

```
            key_priv_1 = kunciPrivate.replace("[",
"")
            key_priv_2  =  key_priv_1.replace("]",
"")
            key_priv_s = key_priv_2.split(", ")
            enkripsiData = encrypt(pesan,[e,n])
            print(enkripsiData)
            # print(key_pub_s[0])
            status_kunci = 'sukses'
            save_encode                          =
Encode_Pesan.objects.create(kd_uji=kdUji,
nama_video="Pengujian Enkripsi", nama_pengujian="-",
rsa=key_pub_s,    rsa_crt=key_priv_s,    a_value=0,
c_value=0,   m_value=0,   chiper_text=enkripsiData,
crt_value=0,                  waktu_pengujian=now,
message_encode=pesan)
            save_encode.save()

            cipher                               =
listToString(str(enkripsiData))
        else:
            status_kunci = 'error'
            cipher = ''
        # Encode_Pesan
        context = {
            'status' : 'sukses',
            'kdUji' : kdUji,
            'status_kunci' : status_kunci,
            'chiper' : cipher
        }
        return JsonResponse(context, safe=False)

    def generateRsa(generator):
        pubRsaKey = get_random_string(20)
        privRsaKey = get_random_string(100)
        keyData = {
```

```
                'private' : pubRsaKey,
                'public' : privRsaKey
            }
            return keyData

        def hidden_message(filename):
            h = hashlib.sha1()
            with open(filename, 'rb') as file:
                chunk = 0
                while chunk != b'':
                    chunk = file.read(1024)
                    h.update(chunk)

            return h.hexdigest()

        def checkBilanganPrima(s):
            num = s
            # To take input from the user
            #num = int(input("Enter a number: "))
            # define a flag variable
            flag = False
            # prime numbers are greater than 1
            if num > 1:
                # check for factors
                for i in range(2, num):
                    if (num % i) == 0:
                        # if factor is found, set flag
to True
                        flag = True
                        # break out of loop
                        break
            return flag
```

```
        def to_ascii(text):
            ascii_values = [ord(character) for character
in text]
            return ascii_values

        def listToString(s):
            # initialize an empty string
            str1 = ""
            # traverse in the string
            for ele in s:
                str1 += ele
            # return string
            return str1


        def gcd(a, b):
            if b == 0:
                return a
            else:
                return gcd(b, a % b)


        def convert_to_int(text):
            converted = []
            for letter in text:
                converted.append(ord(letter) - 96)
            return converted


        def convert_to_ascii(text):
            converted = ''
            for number in text:
                converted = converted + chr(number + 96)
```

```
        return converted


    def choose_e(phi, n):
        print('Choosing e...')
        for e in range(2 ** 31, 2, -1):
            if gcd(e, phi) == 1 and gcd(e, n) == 1:
                return e


    def modular_inverse(a, m):  # modular inverse of
e modulo phi
        m0 = m
        y = 0
        x = 1

        if m == 1:
            return 0

        while a > 1:
            q = a // m
            t = m
            m = a % m
            a = t
            t = y
            y = x - q * y
            x = t
        if x < 0:
            x = x + m0

        return x
```

```
        def encrypt(text, public_key):
            key, n = public_key
            ctext = [pow(ord(char), key, n) for char in
text]
            return ctext


        def decrypt(ctext, private_key, d, n):
            m = []
            for x in ctext:
                c = pow(x,  d) % n
                m.append(c)
            return m


        # The CRT method of decryption is about four
times faster overall
        # Even though there are more steps in this
procedure,
        # the modular exponentiation to be carried out
uses much shorter exponents and so it is less expensive
in the end
        def CRT(p, q, dP, dQ, c):
            qInv = modular_inverse(q, p)
            m1 = pow(c, dP, p)
            m2 = pow(c, dQ, q)
            h = (qInv * (m1 - m2)) % p
            m = m2 + h * q
            return m
```

4. File pengujian.js

```
    //
Route
        var rToDecode = server + "dashboard/proses-
decode";
```

```
        // Inisialisasi
        var divPengujianDecode = new Vue({
            delimiters: ["[[", "]]"],
            el : '#divPengujianDecode',
            data : {
                videoField : false,
                titleForm : 'Pengujian Decode'
            },
            methods : {
                analisaVideoAtc : function()
                {
                    let           kunciRsa           =
document.querySelector('#txtKunciRsa').value;
                    if(kunciRsa === ''){
                        pesanUmumApp('warning',  'Input
Key', 'Harap input key ..');
                    }else{
                        $('#frmUpload').submit();
                    }
                }
            }
        });

        $('#frmUpload').on('submit', function(e){
            e.preventDefault();
            $.ajax({
                type : 'POST',
                enctype: 'multipart/form-data',
                url : rToDecode,
                data : new FormData(this),
                contentType : false,
                cache: false,
                processData: false,
```

```
            beforeSend: function(){

            },
            success : function(data){
                console.log(data);
                let status = data.status;
                if(status === 'no_video'){
                    pesanUmumApp('warning',     'No
hash video', 'Video tidak memiliki data / tidak ada
pesan');
                }else if(status === 'no_rsa_key'){
                    pesanUmumApp('warning',
'Invalid chiper key' , 'Kode cipher tidak sesuai');
                }else{
                    pesanUmumApp('success',
'Sukses', 'Pesan berhasil di encode');
                    let pesan = data.pesan;

document.querySelector("#divHasilDecode").innerHTML =
'<h4>"'+pesan+'"</h4>';

$("#divHasilDecodeVideo").show();
                    console.log(pesan);
                }
            }
        });

    });



    function detectVideo()
    {
        divPengujianDecode.videoField = true;
    }
```

```
function pesanUmumApp(icon, title, text)
{
  Swal.fire({
    icon : icon,
    title : title,
    text : text
  });
}
```

5. File pengjian.js

```
    /
/
route
        var rToUploadVideo = server +
    "dashboard/pengujian/upload-video";
        var rToProsesEnkripsi = server +
    "dashboard/pengujian/proses-enkripsi";

        var kdUjiGlobal = "";
        var hashFile = "";
        // vue object
        var divPengujian = new Vue({
            delimiters: ["[[", "]]"],
            el : '#divPengujian',
            data : {
                titleForm : 'Pengujian',
                videoField : false
            },
            methods : {
                analisaVideoAtc : function()
                {
                    if(this.videoField === false){
```

```
                    pesanUmumApp('warning',    'Pilih
video','Harap pilih video terlebih dahulu ..');
                }else{
                    $("#frmUpload").submit();
                }
            }
        }
    });

    // inisialisasi & fungsi
    $('#txtCapVideo').hide();
    $('#frmUpload').on('submit', function(e){
        e.preventDefault();
        $.ajax({
            type : 'POST',
            enctype: 'multipart/form-data',
            url : rToUploadVideo,
            data : new FormData(this),
            contentType : false,
            cache: false,
            processData: false,
            beforeSend: function(){
                $('#btnMulaiAnalisa').hide();
                $('#divStatusUji').show();
                $('#frmUpload').hide();
                $('#divLoading').show();
                $('#txtPreviewUpload').hide();
            },
            success : function(data){
                console.log(data);
                let kdUji = data.kdUji;
                hashFile = data.kunci;
                kdUjiGlobal = kdUji;
```

```
                let imgSrcFrame1 = server +
"ladun/keras_proses/"+kdUji+"_frame_1_.jpg";
                let imgSrcFrame5 = server +
"ladun/keras_proses/"+kdUji+"_frame_5_.jpg";
                let imgSrcFrame10 = server +
"ladun/keras_proses/"+kdUji+"_frame_10_.jpg";
                let imgSrcFrame15 = server +
"ladun/keras_proses/"+kdUji+"_frame_15_.jpg";
                let imgSrcFrame20 = server +
"ladun/keras_proses/"+kdUji+"_frame_20_.jpg";

document.querySelector('#imgFrame1').setAttribute('src
', imgSrcFrame1);

document.querySelector('#imgFrame5').setAttribute('src
', imgSrcFrame5);

document.querySelector('#imgFrame10').setAttribute('sr
c', imgSrcFrame10);

document.querySelector('#imgFrame15').setAttribute('sr
c', imgSrcFrame15);

document.querySelector('#imgFrame20').setAttribute('sr
c', imgSrcFrame20);
                // RSA render
                rsa_r_1 = data.rsaF1.public;
                rsa_r_5 = data.rsaF5.public;
                rsa_r_10 = data.rsaF10.public;
                rsa_r_15 = data.rsaF15.public;
                rsa_r_20 = data.rsaF20.public;

document.querySelector('#vRsaF1').innerHTML          =
rsa_r_1.substring(1, 20)+"...";

document.querySelector('#vRsaF5').innerHTML          =
rsa_r_5.substring(1, 20)+"...";

document.querySelector('#vRsaF10').innerHTML         =
rsa_r_10.substring(1, 20)+"...";
```

```
document.querySelector('#vRsaF15').innerHTML          =
rsa_r_15.substring(1, 20)+"...";

document.querySelector('#vRsaF20').innerHTML          =
rsa_r_20.substring(1, 20)+"...";

                // RSA CRT Render

document.querySelector('#vRsaCrtF1').innerHTML        =
data.rsaF1.private;

document.querySelector('#vRsaCrtF5').innerHTML        =
data.rsaF5.private;

document.querySelector('#vRsaCrtF10').innerHTML       =
data.rsaF10.private;

document.querySelector('#vRsaCrtF15').innerHTML       =
data.rsaF15.private;

document.querySelector('#vRsaCrtF20').innerHTML       =
data.rsaF20.private;

                let dataPic = data.pic_data;
                // pixel render  1
                let pix_f_1 = "<table>";
                pix_f_1         +=         "<tr><td>
"+dataPic[0][0]+"</td><td>   "+dataPic[0][1]+"</td><td>
"+dataPic[0][2]+"</td></tr>";
                pix_f_1         +=         "<tr><td>
"+dataPic[1][0]+"</td><td>   "+dataPic[1][1]+"</td><td>
"+dataPic[1][2]+"</td></tr>";
                pix_f_1         +=         "<tr><td>
"+dataPic[2][0]+"</td><td>   "+dataPic[2][1]+"</td><td>
"+dataPic[2][2]+"</td></tr>";
                pix_f_1         +=         "<tr><td>
"+dataPic[3][0]+"</td><td>   "+dataPic[3][1]+"</td><td>
"+dataPic[3][2]+"</td></tr>";
```

```
                    pix_f_1         +=          "<tr><td>
"+dataPic[4][0]+"</td><td>   "+dataPic[4][1]+"</td><td>
"+dataPic[4][2]+"</td></tr>";
                    pix_f_1 += "</table>";

document.querySelector("#pixF1").innerHTML = pix_f_1;

                    let pix_f_5 = "<table>";
                    pix_f_5         +=          "<tr><td>
"+dataPic[5][0]+"</td><td>   "+dataPic[5][1]+"</td><td>
"+dataPic[5][2]+"</td></tr>";
                    pix_f_5         +=          "<tr><td>
"+dataPic[6][0]+"</td><td>   "+dataPic[6][1]+"</td><td>
"+dataPic[6][2]+"</td></tr>";
                    pix_f_5         +=          "<tr><td>
"+dataPic[7][0]+"</td><td>   "+dataPic[7][1]+"</td><td>
"+dataPic[7][2]+"</td></tr>";
                    pix_f_5         +=          "<tr><td>
"+dataPic[8][0]+"</td><td>   "+dataPic[8][1]+"</td><td>
"+dataPic[8][2]+"</td></tr>";
                    pix_f_5         +=          "<tr><td>
"+dataPic[9][0]+"</td><td>   "+dataPic[9][1]+"</td><td>
"+dataPic[9][2]+"</td></tr>";
                    pix_f_5 += "</table>";

document.querySelector("#pixF5").innerHTML = pix_f_5;

                    let pix_f_10 = "<table>";
                    pix_f_10         +=         "<tr><td>
"+dataPic[10][0]+"</td><td>
"+dataPic[10][1]+"</td><td>
"+dataPic[10][2]+"</td></tr>";
                    pix_f_10         +=         "<tr><td>
"+dataPic[11][0]+"</td><td>
"+dataPic[11][1]+"</td><td>
"+dataPic[11][2]+"</td></tr>";
                    pix_f_10         +=         "<tr><td>
"+dataPic[12][0]+"</td><td>
"+dataPic[12][1]+"</td><td>
"+dataPic[12][2]+"</td></tr>";
```

```
                    pix_f_10         +=         "<tr><td>
"+dataPic[13][0]+"</td><td>
"+dataPic[13][1]+"</td><td>
"+dataPic[13][2]+"</td></tr>";
                    pix_f_10         +=         "<tr><td>
"+dataPic[14][0]+"</td><td>
"+dataPic[14][1]+"</td><td>
"+dataPic[14][2]+"</td></tr>";
                    pix_f_10 += "</table>";

document.querySelector("#pixF10").innerHTML = pix_f_10;

                    let pix_f_15 = "<table>";
                    pix_f_15         +=         "<tr><td>
"+dataPic[15][0]+"</td><td>
"+dataPic[15][1]+"</td><td>
"+dataPic[15][2]+"</td></tr>";
                    pix_f_15         +=         "<tr><td>
"+dataPic[16][0]+"</td><td>
"+dataPic[16][1]+"</td><td>
"+dataPic[16][2]+"</td></tr>";
                    pix_f_15         +=         "<tr><td>
"+dataPic[17][0]+"</td><td>
"+dataPic[17][1]+"</td><td>
"+dataPic[17][2]+"</td></tr>";
                    pix_f_15         +=         "<tr><td>
"+dataPic[18][0]+"</td><td>
"+dataPic[18][1]+"</td><td>
"+dataPic[18][2]+"</td></tr>";
                    pix_f_15         +=         "<tr><td>
"+dataPic[19][0]+"</td><td>
"+dataPic[19][1]+"</td><td>
"+dataPic[19][2]+"</td></tr>";
                    pix_f_15 += "</table>";

document.querySelector("#pixF15").innerHTML = pix_f_15;

                    let pix_f_20 = "<table>";
                    pix_f_20         +=         "<tr><td>
"+dataPic[20][0]+"</td><td>
```

```
"+dataPic[20][1]+"</td><td>
"+dataPic[20][2]+"</td></tr>";
                    pix_f_20         +=         "<tr><td>
"+dataPic[21][0]+"</td><td>
"+dataPic[21][1]+"</td><td>
"+dataPic[21][2]+"</td></tr>";
                    pix_f_20         +=         "<tr><td>
"+dataPic[22][0]+"</td><td>
"+dataPic[22][1]+"</td><td>
"+dataPic[22][2]+"</td></tr>";
                    pix_f_20         +=         "<tr><td>
"+dataPic[23][0]+"</td><td>
"+dataPic[23][1]+"</td><td>
"+dataPic[23][2]+"</td></tr>";
                    pix_f_20         +=         "<tr><td>
"+dataPic[24][0]+"</td><td>
"+dataPic[24][1]+"</td><td>
"+dataPic[24][2]+"</td></tr>";
                    pix_f_20 += "</table>";

document.querySelector("#pixF20").innerHTML = pix_f_20;



                    $('#divHasilAnalisaVideo').show();
                    $('#txtCapVideo').show();
                    $('#txtPreviewUpload').hide();
                    let   imgVideoUpload   =   server   +
"ladun/data_video_upload/"+kdUji+".mp4";

document.querySelector('#txtCapVideo').setAttribute('s
rc', imgVideoUpload);
                    pesanUmumApp('success',       'Sukses
analisa', "Berhasil menganalisa video");
                    $('#divStatusUji').hide();
                    $('#frmUpload').hide();
                    $('#divLoading').hide();
                }
            });
```

```
      });


      //
http://127.0.0.1:7001/ladun/dasbor/img/logo_uinsu.jpg
      document.querySelector('#btnEnkripsi').addEventL
istener('click', function(){
          let kdUji = kdUjiGlobal;
          let                  pesan                  =
document.querySelector('#txtPesan').value;
          let                  kunci                  =
document.querySelector('#txtKunci').value;
          let              kunciPrivate               =
document.querySelector("#txtKunciPrivate").value;


          let  ds  =  {   'kdUji':kdUji,  'pesan':pesan,
'kunci':kunci,                      'hashKey':hashFile,
'kunciPrivate':kunciPrivate}
          if(kdUji === '' || pesan === '' || kunci ===
''){
              pesanUmumApp('warning',   'Isi   field',
'Harap isi semua field!!!');
          }else{
              $.post(rToProsesEnkripsi,            ds,
function(data){
                  console.log(data);
                  let status_kunci = data.status_kunci;
                  if(status_kunci === 'error'){
                      pesanUmumApp('warning',     'Key
invalid',  'Kunci  RSA  tidak  dikenali,  harap  periksa
kembali');
                  }else{

document.querySelector('#txtPesan').setAttribute('disa
bled', 'disabled');

document.querySelector('#txtKunci').setAttribute('disa
bled', 'disabled');
                      Swal.fire({
                          title: "Sukses?",
```

```
                        text:     "Pesan     berhasil
disisipkan ke video, apakah ingin membuka/download video
hasil pemrosesan ... ?",
                        icon: "success",
                        showCancelButton: true,
                        confirmButtonColor:
"#3085d6",
                        cancelButtonColor: "#d33",
                        confirmButtonText: "Ya",
                        cancelButtonText: "Tidak",
                      }).then((result) => {
                        if (result.value) {
                            let kdUji = data.kdUji;
                            let urlVideo = server +
"ladun/data_video_hash/"+kdUji+".mp4";
                            window.open(urlVideo);

                        }
                      });

document.querySelector("#txtChiper").innerHTML       =
"<h4>Chipertext : "+data.chiper+"</h4>";
                      $("#divChiper").show();
                    $('#btnEnkripsi').hide();
                }

            });
        }

    });

    function detectVideo()
    {
        divPengujian.videoField = true;
    }
```

```
function pesanUmumApp(icon, title, text)
{
  Swal.fire({
    icon : icon,
    title : title,
    text : text
  });
}
```

6. File buatkuncirsa.js

```
    //
Route
          var rToCreateKey = server + "dashboard/proses-
        kunci-baru";
          var rToHapusKunci = server + "dashboard/proses-
        hapus-kunci";

          // Vue object
          var divMitra = new Vue({
              el : '#divMitra',
              data : {

              },
              methods : {
                  buatKunciBaruAtc : function()
                  {
                    $("#dFormBuatKunci").show();

        document.querySelector("#txtTeks").focus();
                        //          pesanUmumApp('success',
        'Sukses', 'Berhasil membuat kunci baru ... ');
                        //             $.post(rToCreateKey,
        function(data){
```

```
                //       divMain.titleApps = "Buat
Kunci RSA";
                //      renderMenu("dashboard/buat-
kunci-rsa");
                // });
            },
            hapusKunciAtc : function(kdKunci)
            {
                hapusKunci(kdKunci);
            },
            tutupFormAtc : function()
            {
              $("#dFormBuatKunci").hide();
            },
            prosesAtc : function(prosesAtc)
            {
              let              teks              =
document.querySelector("#txtTeks").value;
              let              kunci              =
document.querySelector("#txtKunci").value;
              let             kunci2             =
document.querySelector("#txtKunci2").value;
              let ds = {'teks':teks, 'kunci':kunci,
'kunci2':kunci2}
              console.log(ds);
              $.post(rToCreateKey,             ds,
function(data){
                let status = data.status;
                if(status === 'not_prime_number'){
                  pesanUmumApp('warning',      'Not
prime number', 'Kunci yang dimasukkan bukan bilangan
prima');
                }else         if(status         ===
'double_kunci'){
                  pesanUmumApp('warning',   'Double
record', 'Nama kunci sudah di gunakan');
                }else{
```

```
                    pesanUmumApp('success',
'Success', 'Sukses membuat kunci baru');
                    setTimeout(function(){
                      renderMenu("dashboard/buat-
kunci-rsa");
                    }, 2000);
                  }
                });
              }
          }
      });

      // Inisialisasi
      function hapusKunci(kdKunci)
      {
          let ds = {'kdKunci' : kdKunci}
          Swal.fire({
              title: "Hapus kunci?",
              text: "Jika kunci dihapus, video yang
terikat dengan kunci tidak akan bisa di decode.. Yakin
menghapus kunci ... ?",
              icon: "info",
              showCancelButton: true,
              confirmButtonColor: "#3085d6",
              cancelButtonColor: "#d33",
              confirmButtonText: "Ya",
              cancelButtonText: "Tidak",
            }).then((result) => {
              if (result.value) {
                 $.post(rToHapusKunci,            ds,
function(data){
                  console.log(data);
                  divMain.titleApps  =  "Buat  Kunci
RSA";
                  renderMenu("dashboard/buat-kunci-
rsa");
```

```
            });
          }
        });
}

function pesanUmumApp(icon, title, text)
{
  Swal.fire({
    icon : icon,
    title : title,
    text : text
  });
}

$("#tblDataKunci").dataTable();
```

KARTU BIMBINGAN

Buku Laporan Kegiatan Akademik Mahasiswa Fakultas SAINTEK UIN SU Medan

KARTU BIMBINGAN SKRIPSI

Semester Gasal/Genap Tahun Akademik /

Nama	: Dwi Via	Pembimbing I	: Dr. Muhammad Isa, S.S., M.Comp.Sc
NIM	: 0701163137	Pembimbing II	: Sriani, S.Kom., M.Kom.
Prog. Studi	: Ilmu Komputer	SK Pembimbing	:

Judul Skripsi :

PENGAMANAN METODE KRIPTOGRAFI RSA-CRT DAN METODE STEGANOGRAFI LSB PADA SISTEM PENGAMANAN PESAN DENGAN MEDIA VIDEO

PERT	PEMBIMBING I			PEMBIMBING II		
	Tgl.	Materi Bimbingan	Tanda Tangan	Tgl.	Materi Bimbingan	Tanda Tangan
I	24/11 2020	Pengiriman file bab 1, II, III		24/11 2020	Pengiriman file bab 1 Acc bab 1 lanjut bab	
II	27/11 2020	Acc bab I, II, III		7/12 2020 11/12 2020 16/12 2020	Kirim bab II & III Revisi bab II & III Revisi bab II & III	
III	12/12 2020	Acc Sempro		16/12 2020	Acc	
IV	13/01 2021	Revisi Objek Penelitian		14/01 2021	Bimbingan bab IV	
V	03/02 2021	Revisi bab 4		02/02 2021	Bimbingan bab 4	

VI	10/08 2021	Revisi Bab 4	Fitq	12/08 2021	bimbingan Bab 4	
VII	15/08 2021	bimbingan Bab 5	Fitq	16/08 2021	bimbingan Bab 5	
VIII	19/08 2021	Revisi Bab 5	Fitq	19/08 2021	Revisi Bab 5	
IX	23/08 2021	Revisi Abstrak	Fitq	24/08 2021	Revisi flowchart flowchart metode	
X	02/09 2021	Acc Sidang	Fitq	30/08 2021	Acc Sidang	

Medan, 20

An. Dekan

Ketua Jurusan/Program Studi

Rika Zuprita, M.Kom

NIP 198[illegible]

Catatan: Pada saat bimbingan, kartu ini harus diisi dan ditandatangani oleh pembimbing

DAFTAR RIWAYAT HIDUP

DATA DIRI

Nama	:	Diana Vita
NIM	:	0701163137
Tempat, Tanggal Lahir	:	Purwosari, 20 Agustus 1998
Jenis Kelamin	:	Perempuan
Alamat	:	Dusun Purwosari, Desa Bandar Tinggi
Kecamatan	:	Bilah Hulu
Kabupaten	:	Labuhanbatu
Agama	:	Islam
No.Hp	:	082272178783
Email	:	dvita059@gmail.com

NAMA ORANGTUA

Ayah	:	Supriadi
Ibu	:	Fitriati Ritonga

PENDIDIKAN FORMAL :

SD	:	SDN 115134 Janji Lobi
SMP	:	SMPN 2 Rantau Selatan
SMA	:	SMAN 3 Rantau Utara