ENKRIPSI CITRA RGB DENGAN ALGORITMA *VEGENERE CIPHER* DAN ALGORITMA SPICA-XB *CIPHER* DENGAN KUNCI BERKELANJUTAN

SKRIPSI

Oleh:
Risky Damayanti
NIM 131810101043

JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS JEMBER
2017

ENKRIPSI CITRA RGB DENGAN ALGORITMA *VEGENERE CIPHER* DAN ALGORITMA SPICA-XB *CIPHER* DENGAN KUNCI BERKELANJUTAN

SKRIPSI

diajukan guna memenuhi tugas akhir dan memenuhi salah satu syarat
untuk menyelesaikan Program Studi Matematika (S1)
dan mencapai gelar Sarjana Sains

Oleh:
Risky Damayanti
NIM 131810101043

JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS JEMBER
2017

PERSEMBAHAN

Skripsi ini saya persembahkan untuk:

1. ALLAH SWT yang telah memberikan semua kemudahan dan kesempurnaan dalam kehidupan ini;
2. Kedua orang tua saya, Ayahanda Paimin dan Ibunda Badiatun Nisa yang senantiasa memberikan semangat, dorongan moril maupun materi, kasih sayang serta do'a tiada henti;
3. Seluruh guru dan dosen, sejak taman kanak-kanak hingga perguruan tinggi yang telah memberi banyak ilmu dan membimbing dengan tulus;
4. Sahabat saya, Yulia Nurul Farida, Dewintha Bunga PCW, Riska Ayu yang telah menjadi sahabat terbaik selama kuliah, terima kasih atas motivasi-motivasinya, semangat kalian dan candaan kalian;
5. Teman-teman "ATLAS", dan semua pihak yang telah memberikan dukungan serta motivasi, sehingga skripsi ini bisa terselesaikan;
6. Almamater tercinta Jurusan Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

MOTTO

Siapapun yang menempuh suatu jalan untuk mendapatkan ilmu, maka Allah akan memberikan kemudahan jalannya menuju syurga. *)

Sesuatu yang belum dikerjakan, seringkali tampak mustahil; kita baru yakin kalau kita telah berhasil melakukannya dengan baik. **)

*) H.R Muslim

**) Evelyn Underhill

PERNYATAAN

Saya yang bertanda tangan di bawah ini:

Nama : Risky Damayanti

NIM : 131810101043

menyatakan dengan sesungguhnya bahwa skripsi yang berjudul " Enkripsi Citra RGB dengan Algoritma *Vegenere Cipher* dan Algoritma SPICA-XB *Cipher* dengan Kunci Berkelanjutan" adalah benar-benar hasil karya sendiri, kecuali kutipan yang sudah saya sebutkan sumbernya, belum pernah diajukan dalam institusi manapun dan juga bukan karya jiplakan. Saya bertanggung jawab atas keabsahan dan kebenaran isinya sesuai dengan sikap ilmiah yang harus dijunjung tinggi.

Demikian pernyataan ini saya buat dengan sebenarnya, tanpa ada tekanan dan paksaan dari pihak manapun serta bersedia mendapat sanksi akademik jika ternyata di kemudian hari pernyataan ini tidak benar.

Jember, Juni 2017

Yang menyatakan,

Risky Damayanti

NIM 131810101043

SKRIPSI

ENKRIPSI CITRA RGB DENGAN ALGORITMA *VEGENERE CIPHER* DAN ALGORITMA SPICA-XB *CIPHER* DENGAN KUNCI BERKELANJUTAN

Oleh:
Risky Damayanti
NIM 131810101043

Pembimbing :

Dosen Pembimbing Utama : Ahmad Kamsyakawuni, S.Si., M.Kom.

Dosen Pembimbing Anggota : M. Ziaul Arif, S.Si., M.Sc.

PENGESAHAN

Skripsi berjudul "Enkripsi Citra RGB dengan Algoritma *Vegenere Cipher* dan Algoritma SPICA-XB *Cipher* dengan Kunci Berkelanjutan" telah diuji dan disahkan pada :

hari, tanggal :

tempat : Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Tim Penguji :

Ketua,	Anggota I,
Ahmad Kamsyakawuni, S.Si., M.Kom.	M. Ziaul Arif, S.Si., M.Sc.
NIP. 197211291998021001	NIP. 198501112008121002
Anggota II,	Anggota III,
Drs. Rusli Hidayat, M.Sc.	Dr. Firdaus Ubaidillah, S.Si., M.Si.
NIP. 196610121993031001	NIP. 197006061998031003

Mengesahkan

Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam

Universitas Jember

Drs. Sujito, Ph.D.

NIP. 196102041987111001

RINGKASAN

Enkripsi Citra RGB dengan Algoritma *Vegenere Cipher* dan Algoritma Spica-Xb *Cipher* dengan Kunci Berkelanjutan; Risky Damayanti, 131810101043; 2017; 62 Halaman; Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Internet merupakan salah satu perkembangan dari teknologi informasi dan komunikasi yang digunakan sebagai sebuah sarana pertukaran informasi dapat memberikan kerugian bagi penggunanya. Salah satu kerugian adanya penyadapan informasi sehingga informasi yang rahasia dapat diketahui oleh orang yang tidak memiliki hak untuk mengakses informasi tersebut.

Keamanan informasi didapatkan salah satunya dengan menerapkan teknik kriptografi pada informasi. Kriptografi adalah ilmu dan seni mengubah informasi untuk membuatnya aman dan kebal dari serangan. Kriptografi klasik yakni algoritma yang digunakan dalam kriptografi yang bersifat sederhana, sedangkan kriptografi modern bersifat kompleks. Salah satu algoritma yang digunakan dalam penelitian ini adalah *Vigenere Cipher*. Namun, untuk melindungi suatu data yang besar seperti citra, algoritma *Vigenere Cipher* tidaklah aman karena pada saat data berbentuk teks dapat dipecahkan dan merupakan algoritma klasik. Untuk mengatasi kelemahan ini, maka akan dilakukan pengenkripsian dua kali dengan menggunakan algoritma SPICA-XB *Cipher* serta pemodifikasian kunci sehingga kunci berkelanjutan. Algoritma SPICA-XB *Cipher* merupakan modifikasi dari algoritma *Caesar Cipher* yang termasuk dalam algoritma modern. Penggunaan kunci berkelanjutan akan mempengaruhi proses enkripsi dengan input dibagi perblok-blok sehingga tiap blok kunci berbeda.

Data yang digunakan dalam penelitian ini adalah data berupa citra berwarna (citra RGB) yang digunakan sebagai *plainimage*. Kunci dari citra tersebut kemudian akan dienkripsi menggunakan algoritma *Vigenere Cipher* sebagai kunci berkelanjutan. Setelah itu citra akan dilakukan proses enkripsi dengan algoritma SPICA-XB *Cipher* dan algoritma *Vigenere Cipher*. Hasil dari enkripsi ini akan

dihasilkan sebuah *cipherimage* yang tidak lagi mengandung informasi dari *plainimage* yang ada.

Analisis keamanan dari algoritma yang diajukan menunjukkan bahwa algoritma aman dari serangan analisis histogram dan analisis diferensial.

PRAKATA

Puji syukur ke hadirat Allah SWT yang telah melimpahkan rahmat, taufik dan karuniaNya sehingga skripsi yang berjudul "Enkripsi Citra RGB dengan Algoritma *Vegenere Cipher* dan Algoritma SPICA-XB *Cipher* dengan Kunci Berkelanjutan" dapat terselesaikan. Skripsi ini disusun untuk memenuhi salah satu syarat dalam menyelesaikan pendidikan strata 1 (S1) di Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Penyusunan skripsi ini tidak terlepas dari bantuan berbagai pihak, baik secara langsung maupun tidak langsung. Oleh karena itu, penulis menyampaikan terima kasih kepada:

1. Bapak Ahmad Kamsyakawuni, S.Si., M.Kom. selaku Dosen Pembimbing Utama dan Bapak M. Ziaul Arif, S.Si., M.Sc. selaku Dosen Pembimbing Anggota yang telah memberikan bimbingan dan bantuan dalam penyusunan skripsi ini;
2. Bapak Drs. Rusli Hidayat, M.Sc. selaku dosen penguji I dan Bapak Dr. Firdaus Ubaidillah, S.Si., M.Si. selaku dosen penguji II yang telah memberikan kritik dan saran yang membangun untuk penyempurnaan skripsi ini;
3. Ika Hesti Agustin, S.Si., M.Si selaku Dosen Pembimbing Akademik yang telah membimbing dalam pemilihan mata kuliah;
4. Seluruh dosen dan karyawan Jurusan Matematika Fakultas MIPA Universitas Jember yang telah memberikan ilmu serta membantu selama proses perkuliahan berlangsung;
5. Keluarga yang selalu memberikan doa tiada henti dan dukungan baik lahir maupun batin serta semua pihak yang membantu terselesaikannya skripsi ini.

Penulis menyadari bahwa dalam menyusun skripsi ini masih terdapat kekurangan baik isi maupun susunannya. Oleh sebab itu, penulis mengharapkan saran dan kritik demi penyempurnaan skripsi ini. Akhirnya penulis berharap semoga skripsi ini dapat memberi manfaat bagi pembaca.

Jember, Juni 2017

Penulis

DAFTAR ISI

Halaman

HALAMAN JUDUL i
PERSEMBAHAN ii
HALAMAN MOTTO iii
PERNYATAAN iv
HALAMAN PEMBIMBINGAN v
PENGESAHAN vi
RINGKASAN vii
PRAKATA ix
DAFTAR ISI x
DAFTAR GAMBAR xii
DAFTAR TABEL xiii
DAFTAR LAMPIRAN xiv
BAB 1. PENDAHULUAN 1
1.1 Latar Belakang 1
1.2 Rumusan Masalah 3
1.3 Tujuan Penelitian 3
1.4 Manfaat Penelitian 4
BAB 2. TINJAUAN PUSTAKA 5
2.1 Kriptografi 5
2.2 Algoritma Kriptografi 6
2.3 Citra 7
2.3.1 Citra Hitam-putih dan Citra Warna 8
2.3.2 Digitalisasi Citra 9
2.4 Sistem Basis pada Bilangan 10
2.5 Algoritma *Vigenere Cipher* 13
2.6 Algoritma *Caesar Cipher* 16
2.7 Algoritma SPICA-XB *Cipher* 17
2.8 Analisis Keamanan 23

2.8.1 Analisis Histogram Derajat Keabuan 24
2.8.2 Analisis Diferensial..... 24
BAB 3. METODE PENELITIAN..... 25
3.1 Data Penelitian 25
3.2 Langkah-langkah Penelitian..... 25
BAB 4. HASIL DAN PEMBAHASAN..... 31
4.1 Hasil..... 31
4.1.1 Pembagian Nilai Derajat Keabuan *Plainimage* 32
4.1.2 Enkripsi dengan Algoritma *Vigenere Cipher* sebagai Kunci Berkelanjutan..... 32
4.1.3 Enkripsi dengan Algoritma SPICA-XB *Cipher*..... 34
4.1.4 Enkripsi Kembali dengan algoritma *Vigenere Cipher*..... 37
4.1.5 Pembagian Nilai Derajat Keabuan *Cipherimage*..... 38
4.1.6 Dekripsi dengan Algoritma *Vigenere Cipher* 39
4.1.7 Dekripsi Kembali dengan Algoritma SPICA-XB *Cipher*..... 41
4.1.8 Enkripsi dengan algoritma *Vigenere Cipher* sebagai Kunci Berkelanjutan..... 44
4.1.9 Analisis Keamanan 45
4.1.10 Aplikasi Program 47
4.2 Pembahasan..... 53
4.2.1 Proses Enkripsi 53
4.2.2 Proses Dekripsi 54
4.2.3 Analisis Keamanan 54
BAB 5. KESIMPULAN DAN SARAN..... 55
5.1 Kesimpulan..... 55
5.2 Saran 55
DAFTAR PUSTAKA..... 56
LAMPIRAN..... 57

DAFTAR GAMBAR

Halaman

2.1 Diagram Aliran Proses Enkripsi dan Deskripsi 5
2.2 Menentukan Koordinat Titik pada Citra 8
2.3 Menentukan Kanal RGB pada Citra RGB 9
2.4 Bujursangkar *Vegenere* 14
2.5 Langkah-langkah Enkrispi SPICA-XB *Cipher* 19
2.6 Langkah-langkah Dekrispi SPICA-XB *Cipher* 23
3.1 Citra Buah 25
3.2 Proses Enkripsi Kanal R 27
3.3 Proses Dekripsi Kanal R 29
3.4 *Flowchart* Langkah-langkah Penelitian 30
4.1 Hasil Proses Enkripsi 31
4.2 Analisis Histogram 46
4.3 Analisis dengan Diferensial 46
4.4 Aplikasi Program 47
4.5 Tampilan Enkripsi pada Aplikasi Program 49
4.6 Proses Enkripsi dengan Input Citra *Grayscale* 50
4.7 Proses Enkripsi dengan Input Citra Hitam 50
4.8 Hasil Dekripsi dengan Kunci Enkripsi 51
4.9 Proses Dekripsi Citra *Grayscale* 52
4.10 Proses Dekripsi Citra Hitam 53

DAFTAR TABEL

Halaman

2.1 Operasi Penjumlahan pada Bilangan Biner 12
2.2 Operasi Pengurangan pada Bilangan Biner 12
2.3 Operasi XOR pada Bilangan Biner 12
2.4 Konveksi Karakter *Alphabet Plaintext* dan Kunci Sebelum Dienkripsi 14
4.1 Potongan *Pixel Plainimage* Citra Buah 31
4.2 Blok *Pixel Plainimage* Citra Buah 32
4.3 k_2 dan k_3 dari Bp_1 dan Bp_2 33
4.4 sp_1 dan sp_2 dari Bp_1 dan Bp_2 36
4.5 Bc_2 dan Bc_3 dari Bp_1 dan Bp_2 38
4.6 Blok *Pixel Cipherimage* Citra Buah 39
4.7 sp_1 dan sp_2 dari Bc_1 dan Bc_2 40
4.8 Bp_1 dan Bp_2 dari Bc_1 dan Bc_2 43
4.9 k_2 dan k_3 dari Bc_1 dan Bc_2 45

DAFTAR LAMPIRAN

Halaman

A. Matriks Derajat Keabuan dari Gambar 4.1(a) 57
B. Matriks Derajat Keabuan dari Gambar 4.1(b) 60

BAB 1. PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi informasi dan komunikasi yang pesat menimbulkan masalah keamanan informasi. Internet sebagai sebuah sarana pertukaran informasi memberikan keuntungan dan kerugian bagi penggunanya, salah satu hal yang merugikan adanya penyadapan informasi sehingga informasi yang rahasia dapat diketahui oleh orang yang tidak memiliki hak untuk mengakses informasi tersebut.

Keamanan informasi didapatkan salah satunya dengan menerapkan teknik kriptografi pada informasi. Kriptografi adalah ilmu dan seni mengubah informasi untuk membuatnya aman dan kebal dari serangan. Terdapat dua faktor utama dalam teknik kriptografi yaitu enkripsi dan dekripsi. Enkripsi atau penyandian merupakan proses pengubahan informasi agar tidak terbaca. Hasil dari enkripsi berupa informasi yang disandikan atau *ciphertext*. *Ciphertext* dapat diambil informasinya dengan cara membalik sandi tersebut menggunakan algoritma kriptografi yang sama. Proses pembalikan sandi sehingga didapatkan informasi yang nyata ini biasa disebut proses dekripsi.

Secara umum, algoritma yang digunakan dalam kriptografi dapat terbagi ke dalam dua macam, yaitu algoritma kriptografi klasik dan algoritma kriptografi modern. Algoritma kriptografi klasik biasanya adalah algoritma penyembunyian teks yang bersifat sederhana, berbasis pada pemrosesan per-karakter dan dapat dilakukan tanpa menggunakan komputer. Sedangkan algoritma kriptografi modern adalah algoritma kriptografi yang menggunakan algoritma kompleks dan menggunakan pengolahan berbasis bit dalam proses enkripsi pesannya.

Algoritma yang umum digunakan dalam kriptografi sekarang ini adalah algoritma kriptografi modern yang lebih kuat dan aman. Algoritma kriptografi modern yang berkembang sekarang ini berkat algoritma kriptografi klasik yang

lebih dulu digunakan dan kemudian dikembangkan serta disesuaikan kebutuhan dan perkembangan zaman.

Vigenere Cipher dan *Caesar Cipher* merupakan algoritma kriptografi klasik. *Vigenere Cipher* menggunakan teknik *cipher alphabet* – majemuk (*polyalpabetic substitution cipher*) dan merupakan salah satu dari yang terbaik. *Vigenere Cipher* menggunakan bujursangkar *Vigenere* untuk melakukan enkripsi. Setiap baris di dalam bujursangkar menyatakan huruf-huruf *ciphertext* yang diperoleh dengan *Caesar Cipher.* Pada pertengahan abad XIX *Vigenere Cipher* berhasil dipecahkan dengan metode Babbage-Kasiski (Munir, 2004).

C*aesar Cipher* adalah salah satu algoritma kriptografi klasik yang terkenal. Metode yang digunakan dalam algoritma ini yaitu metode substitusi, dimana pesan yang berupa karakter (*plaintext*) disandikan dengan cara disubstitusi dengan karakter lain dengan pola selisih tertentu dalam susunan abjad (*alphabet*) maupun susunan karakter ASCII. Pesan yang dienkripsi menggunakan algoritma *Caesar Cipher* masih mudah dipecahkan karena jumlah kuncinya sangat sedikit yaitu hanya ada 26 kunci (Munir, 2004).

Islami (2013) telah memodifikasi Algoritma *Caesar Cipher* dengan menggabungkan sifat dari algoritma *Caesar Cipher* dengan beberapa sifat dari algoritma kriptografi modern yang menghasilkan algoritma SPICA-XB *Cipher*. Tekniknya menggunakkan algoritma *Caesar Cipher* layaknya memutar sebuah roda (*spinning*). Algoritma SPICA-XB *Cipher* bekerja dengan melakukan proses substitusi dengan kunci yang berputar. Pada tahun 2011, Andhika telah memodifikasi Algoritma *Vigenere Cipher* dengan menggunakan Algoritma *Caesar Cipher* dan enkripsi berlanjut untuk pembentukan *key*-nya. Input yang digunakan berupa pesan, pesan akan dibagi menjadi blok sepanjang banyaknya karakter kunci sehingga kunci untuk blok selanjutnya berasal dari blok sebelumnya.

Pada penelitian ini, penulis ingin mengajukan suatu metode baru dalam mengenkripsi data digital yang berupa citra RGB. Untuk mengatasi kelemahan dari algoritma *Vigenere Cipher*, penulis akan melakukan enkripsi dengan algoritma *Vigenere Cipher* dan SPICA-XB *Cipher* dengan kunci berkelanjutan.

Dari penelitian ini, penulis berharap bahwa metode yang diajukan ini akan memberikan tingkat keamanan yang lebih tinggi dari penelitian-penelitian sebelumnya.

1.2 Rumusan Masalah

Berdasarkan latar belakang tersebut, maka rumusan masalah pada penelitian ini adalah:

a. Bagaimana mengenkripsi citra RGB menggunakan algoritma *Vigenere cipher* dan algoritma SPICA-XB *cipher* dengan kunci berkelanjutan?
b. Bagaimana mendekripsi citra RGB yang telah dienkripsi menggunakan algoritma *Vigenere cipher* dan algoritma SPICA-XB *cipher* dengan kunci berkelanjutan?
c. Bagaimana analisis keamanan menggunakan analisis histogram dan analisis diferensial pada *ciphertext* yang dihasikan dari enkripsi menggunakan algoritma *Vigenere cipher* dan algoritma SPICA-XB *cipher* dengan kunci berkelanjutan?

1.3 Tujuan Penelitian

Adapun tujuan pada penelitian ini yaitu:

a. Mengenkripsi citra RGB menggunakan algoritma *Vigenere cipher* dengan algoritma SPICA-XB *cipher* dengan kunci berkelanjutan.
b. Mendekripsi citra RGB yang telah dienkripsi menggunakan algoritma *Vigenere cipher* dan algoritma SPICA-XB *cipher* dengan kunci berkelanjutan.
c. Menganalisis keamanan menggunakan analisis histogram dan analisis diferensial pada *ciphertext* yang dihasikan dari enkripsi menggunakan algoritma *Vigenere cipher* dan algoritma SPICA-XB *cipher* dengan kunci berkelanjutan.

1.4 Manfaat Penelitian

Manfaat yang diharapkan pada penelitian ini adalah:

a. Mengetahui metode serta proses untuk mengenkripsi sekaligus mendekripsi citra RGB menggunakan algoritma *Vigenere cipher* dengan algoritma SPICA-XB *cipher* dengan kunci berkelanjutan.
b. Mengetahui hasil dari proses enkripsi serta dekripsi dari metode yang diajukan.
c. Mampu menganalisis keamanan mengunakan analisis histogram dan analisis diferensial dari metode yang diajukan.

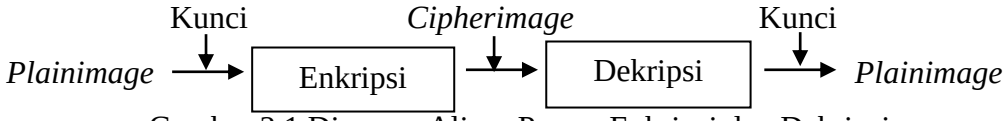
BAB 2. TINJAUAN PUSTAKA

2.1 Kriptografi

Kriptografi berasal dari dua kata dari bahasa Yunani, yaitu *cryptós* (rahasia) dan *gráphein* (tulisan). Oleh sebab itu, dapat diartikan bahwa kriptografi mempelajari tentang tulisan rahasia atau tulisan tersembunyi. Secara garis besar, ilmu kriptografi mempelajari teknik untuk menyembunyikan, melindungi dan mengamankan suatu informasi dengan cara membuat suatu bentuk baru (sandi) sehingga sulit untuk dipahami dan dibaca oleh indera visual seseorang (Munir, 2006).

Kriptografi pada saat ini dapat didefinisikan sebagai ilmu dan seni untuk menjaga keamanan pesan. Kriptografi mentransformasikan data asli (*plaintext*) ke dalam bentuk data sandi (*ciphertext*), jika data citra asli (*plainimage*) ke dalam bentuk data citra sandi (*cipherimage*). Proses pengubahan suatu *plaintext* ke dalam *ciphertext* dan *plainimage* ke dalam *cipherimage* disebut dengan proses *Encipherment* atau *encryption* (enkripsi). Sedangkan proses mentransformasikan kembali *ciphertext* ke dalam bentuk *plaintext* dan *cipherimage* ke dalam bentuk *plainimage* disebut dengan proses *Decipherment* atau *decryption* (dekripsi). Dalam melakukan proses enkripsi dan dekripsi diperlukan sebuah kunci (*key*) dan suatu algoritma. Algoritma tersebut merupakan suatu fungsi matematika yang digunakan untuk mengenkripsi dan mendekripsi data. Sedangkan kunci merupakan suatu deretan bit yang diperlukan untuk mengontrol jalannya algoritma.

Diagram aliran dari proses enkripsi dan dekripsi pada Gambar 2.1 sebagai berikut:



Gambar 2.1 Diagram Aliran Proses Enkripsi dan Dekripsi

2.2 Algoritma Kriptografi

Algoritma dalam kriptografi berguna untuk memproses enkripsi dan dekripsi. Algoritma dikenal juga sebagai *cipher*. Menurut perkembangannya, algoritma kriptografi dibagi menjadi 2 jenis, yakni:

a. Algoritma Kriptografi Klasik

Algoritma kriptografi klasik merupakan algoritma kriptografi pertama yang berkembang dan tidak menggunakan bantuan komputerisasi sebagai teknik proses enkripsi dan dekripsi karena masih berbasis karakter. Penggunaannya menggunakan teknik kunci simetris dan biasanya hanya menggunakan satu kunci saja. Serta, metode penyembunyian pesannya menggunakan teknik substitusi, transposisi atau keduanya (Sadikin, 2012).

Contoh dari Algoritma Kriptografi klasik adalah *Cipher* Transposisi (*Transposition Cipher*) dan *Cipher* Substitusi (*Substitution Cipher*). *Cipher* Substitusi yang berkembang saat ini sering juga disebut sebagai *Caesar Cipher* (Munir, 2004).

b. Algoritma Kriptografi Modern

Algoritma kriptografi modern merupakan perkembangan dari algoritma kriptografi klasik. Modifikasi dari metode transposisi dan substitusi kunci dari algoritma klasik merupakan salah satu alasan dikembangkannya algoritma ini. Jadi, algoritma ini memiliki tingkat algoritma yang lebih kompleks daripada algoritma kriptografi klasik (Sadikin, 2012).

Sedangkan, menurut kunci yang membangun algoritmanya, algoritma kriptografi dibagi menjadi 2 jenis juga, yakni:

a. Algoritma Kunci Simetris

Istilah lain dari algoritma kunci simetris adalah algoritma kunci tunggal. Dikatakan simetris karena menggunakan kunci (*key*) yang sama dalam proses enkripsi mapun dekripsi. Sering juga diistilahkan sebagai algoritma kunci rahasia karena kunci (*key*) dalam algoritma ini bersifat rahasia (Prayudi, 2005).

b. Algoritma Kunci Asimetris

Perbedaan yang mendasar dengan algoritma kunci simetris adalah pada algoritma ini kunci (*key*) yang digunakan tidak sama, artinya kunci untuk enkripsi

dan kunci dekripsi berbeda satu sama lain.

Kunci yang digunakan untuk proses enkripsi diistilahkan sebagai kunci publik (*public key*), sedangkan kunci yang digunakan untuk proses dekripsi diistilahkan sebagai kunci rahasia (*private key*). *Public key* pada proses enkripsi yang bersifat umum untuk dipublikasikan membuat algoritma ini sering disebut sebagai Algortima Kunci Umum (Prayudi, 2005).

2.3 Citra

Secara harafiah, citra (*image*) adalah gambar pada bidang dwimatra (dua dimensi). Ditinjau dari sudut pandang matematis, citra merupakan fungsi menerus (*continue*) dari intensitas cahaya pada bidang dwimatra. Sumber cahaya menerangi objek, objek memantulkan kembali sebagian dari berkas cahaya tersebut. Pantulan cahaya ini ditangkap oleh alat-alat optik, misalnya pada mata manusia, kamera, pemindai (*scanner*), dan sebagainya, sehingga bayangan objek yang disebut citra dapat terekam.

Menurut (Murni, 1992) citra sebagai keluaran dari suatu sistem perekam data dapat bersifat :

a. optik berupa foto,
b. analog berupa sinyal video seperti gambar pada monitor televisi,
c. digital yang dapat langsung disimpan pada suatu pita magnetik.

Citra terdiri atas dua macam yaitu citra diam (*still images*) dan citra bergerak (*moving images*). Citra diam adalah citra tunggal yang tidak bergerak, sementara citra bergerak adalah rangkaian citra diam yang ditampilkan secara beruntun sehingga memberi kesan pada mata kita sebagai citra yang bergerak. Setiap citra dalam rangkaian itu biasa disebut dengan *frame.*

Secara matematis, fungsi intensitas cahaya pada suatu citra di bidang dwimatra disimbolkan dengan $f(x,y)$, dimana :

(x,y) : koordinat pada bidang dwimatra

$f(x,y)$: intensitas cahaya (*brightness*) pada titik (x,y)

Pada Gambar 2.2 terlihat cara untuk menentukan koordinat titik pada suatu citra. Intensitas f dari gambar hitam putih pada titik (x,y) disebut sebagai derajat

keabuan (*gray level*), sedangkan citranya disebut sebagai citra hitam-putih (*grayscale image*) atau citra monokrom (*monochrome image*). Derajat keabuan memiliki rentang nilai dari l_{min} sampai l_{max} atau $l_{min} < f < l_{max}$. Selang (l_{min}, l_{max}) disebut sebagai skala keabuan.

Gambar 2.2 Menentukan Koordinat Titik pada Citra
(Sumber : Murni, 1992)

Selang (l_{min}, l_{max}) atau selang [0, L] yang berarti nilai intensitas 0 menyatakan hitam sedangkan intensitas L menyatakan putih sehingga nilai intensitas antara 0 sampai L bergeser dari hitam ke putih.

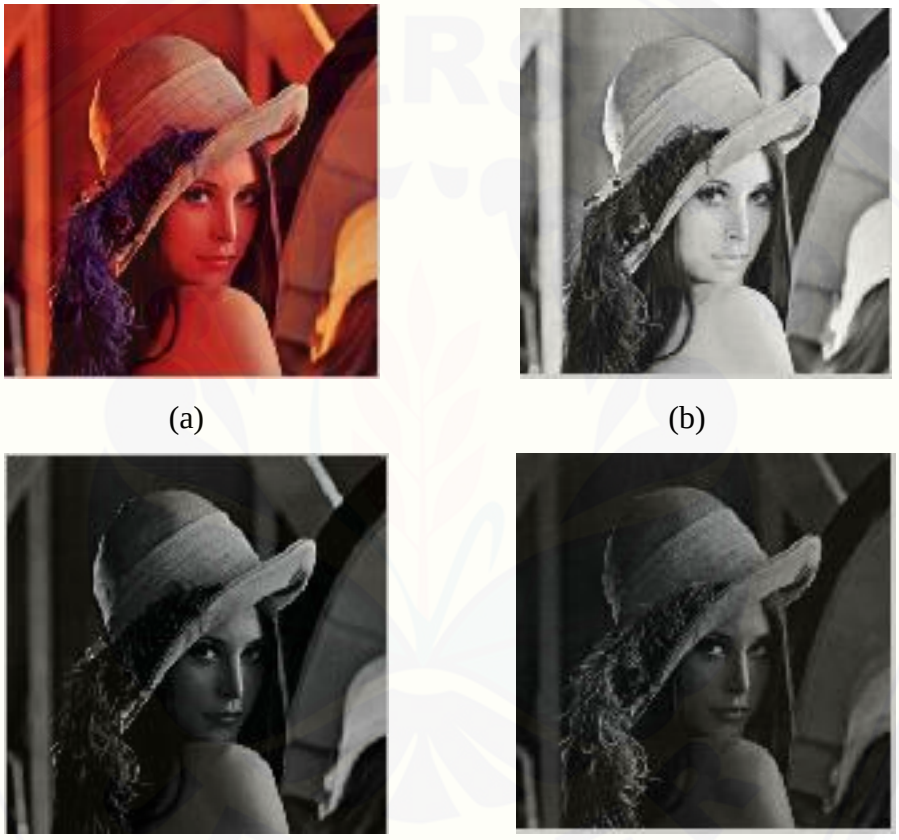
2.3.1 Citra Hitam-putih dan Citra Warna

Citra hitam-putih disebut sebagai citra satu kanal, yaitu warnanya hanya ditentukan oleh satu fungsi intensitas. Pada citra hitam-putih dengan 256 *level* yang memiliki skala keabuan dari 0 sampai 255 atau [0, 255] sehingga nilai antara 0 sampai 255 menyatakan warna keabuan yang terletak antara hitam dan putih.

Citra berwarna (*color images*) disebut sebagai citra spektral, yaitu warna pada citra disusun oleh tiga kanal yang menyatakan komponen warna disebut sebagai komponen *RGB*, yaitu merah (*red*), hijau (*green*), dan biru (*blue*). Intensitas suatu titik pada citra berwarna merupakan kombinasi dari tiga intensitas, yakni :

a. derajat keabuan merah ($f_r(x,y)$),
b. derajat keabuan hijau ($f_g(x,y)$) dan
c. derajat keabuan biru ($f_b(x,y)$) (Munir, 2002).

Gambar 2.3(a) menunjukkan gambar asli dari citra berwarna, Gambar 2.3(b) menunjukkan intensitas derajat keabuan merah ($f_r(x,y)$), Gambar 2.3(c) menunjukkan intensitas derajat keabuan hijau ($f_g(x,y)$), dan Gambar 2.3(d) menunjukkan intensitas derajat keabuan biru ($f_b(x,y)$),



(a) (b)

(c) (d)

(a) Citra RGB; (b) Kanal *Red*; (c) Kanal *Green*; (d) Kanal *Blue*

Gambar 2.3 Menentukan Kanal RGB pada Citra RGB

2.3.2 Digitalisasi Citra

Agar suatu citra dapat diolah dengan komputer digital, suatu citra harus direpresentasikan secara numerik dengan nilai-nilai diskrit. Representasi citra dari fungsi kontinu menjadi nilai-nilai diskrit disebut sebagai proses digitalisasi. Citra yang dihasilkan inilah yang disebut sebagai citra digital (*digital image*). Pada

umumnya, citra digital berbentuk persegi panjang, dan dimensi ukurannya dinyatakan dengan tinggi x lebar atau lebar x panjang.

Citra digital berukuran N x M lazim dinyatakan dengan matriks yang berukuran N baris dan M kolom sebagai berikut :

$$f(x,y) = \begin{bmatrix} f(1,1) & \cdots & f(1,M) \\ \vdots & \ddots & \vdots \\ f(N,1) & \cdots & f(N,M) \end{bmatrix}$$

indeks baris (i) dan indeks kolom (j) menyatakan suatu koordinat titik pada citra, dan $f(i,j)$ merupakan intensitas (derajat keabuan) pada titik (i,j). Masing-masing elemen pada citra digital (elemen matriks) disebut dengan *pixel* atau *pel*. Jadi citra yang berukuran N x M mempunyai NM buah *pixel* (Dulimarta, 1997).

2.4 Sistem Basis pada Bilangan

Berdasarkan basisnya bilangan dibagi menjadi bilangan desimal, bilangan biner, dan bilangan hexadesimal. Bilangan desimal merupakan bilangan yang memiliki basis 10, yaitu : 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Bilangan biner merupakan bilangan yang memiliki basis 2, yaitu : 0 dan 1. Sementara bilangan hexadesimal memiliki basis 16, yaitu : 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F (Price & Peselnick, 1987).

Bilangan-bilangan tersebut dapat dikonversi menjadi satu lainnya. Berikut merupakan beberapa langkah untuk melakukan konversi bilangan-bilangan tersebut.

a. Konversi bilangan desimal ke bilangan biner

Metode yang dilakukan dengan proses pembagian. Bilangan desimal dibagi dengan 2, pembacaan nilai akhir pembagian dan urutan sisa hasil pembagian adalah bentuk bilangan biner dari nilai desimal.

Contoh :

Bilangan desimal 50_{10}

50_{10} : 2 = 25 sisa 0

25_{10} : 2 = 12 sisa 1

12_{10} : 2 = 6 sisa 0

6_{10} : 2 = 3 sisa 0

3_{10} : 2 = 1 sisa 1

Maka bilangan biner dari 50_{10} adalah 110010_2

b. Konversi bilangan biner ke bilangan desimal

Sistem bilangan biner masing-masing digitnya disebut bit (*binary digit*) yang hanya mempunyai dua nilai koefisien yaitu 0 dan 1. Konversi dilakukan menggunakan persamaan (2.1).

$$\text{dec} = \sum_{k=1}^{n} B_k 2^{(n-k)} \qquad (2.1)$$

dimana,

B_k = bilangan-bilangan yang menyusun suatu biner

n = panjang biner

Contoh :

Bilangan biner 110010_2

diketahui n = 6, maka

$$\text{dec} = \sum_{1}^{6} B_k 2^{(6-1)} = (1)2^5 + (1)2^4 + (0)2^3 + (0)2^2 + (1)2^1 + (0)2^0$$

$$= 32 + 16 + 0 + 0 + 2 + 0 = 50$$

Maka bilangan desimal dari 110010_2 adalah 50_{10}

c. Konversi biner ke hexadesimal

Setiap 4 digit bilangan biner akan dikonversi menjadi 1 digit bilangan hexadesimal karena 4 digit bilangan biner bisa dikonversi menjadi 16 macam bilangan hexadesimal.

Contoh :

Bilangan biner 01011010_2

0101 = 5

1010 = A

Maka bilangan hexadesimal dari 01011010_2 adalah 5A

d. Konversi hexadesimal ke biner

Setiap 1 digit bilangan hexadesimal akan dikonversi menjadi 4 digit bilangan biner.

Contoh :

Bilangan hexadesimal 5A

5 = 0101

A = 1010

Maka bilangan biner dari 5A adalah 01011010_2

Pada sistem bilangan digital, dikenal juga operasi-operasi bilangan seperti penjumlahan, pengurangan dan *exclusive-OR* (XOR) (Yohandri,2013).

Tabel 2.1 menunjukkan hasil dari operasi penjumlahan pada bilangan biner. Tabel 2.2 menunjukkan hasil dari operasi pengurangan pada bilangan biner. Tabel 2.3 menunjukkan hasil dari operasi XOR pada bilangan biner.

Tabel 2.1 Operasi Penjumlahan pada Bilangan Biner

Bilangan 1	Bilangan 2	Bilangan 1 ⊕ Bilangan 2
0	0	0
0	1	1
1	0	1
1	1	0

Contoh :

Jumlahkan 011 dengan 001

$$\begin{array}{ccc} 0 & 1 & 1 \\ 0 & 0 & 1 \\ \hline 1 & 0 & 0 \end{array} +$$

Tabel 2.2 Operasi Pengurangan pada Bilangan Biner

Bilangan 1	Bilangan 2	Bilangan 1 ⊕ Bilangan 2
0	0	0
0	1	1
1	0	1
1	1	0

Contoh :

Kurangi 100 dengan 011

$$\begin{array}{ccc} 1 & 0 & 0 \\ 0 & 1 & 1 \\ \hline 0 & 0 & 1 \end{array} -$$

Tabel 2.3 Operasi XOR pada Bilangan Biner

Bilangan 1	Bilangan 2	Bilangan 1 ⊕ Bilangan 2
0	0	0
0	1	1
1	0	1
1	1	0

2.5 Algoritma *Vigenere Cipher*

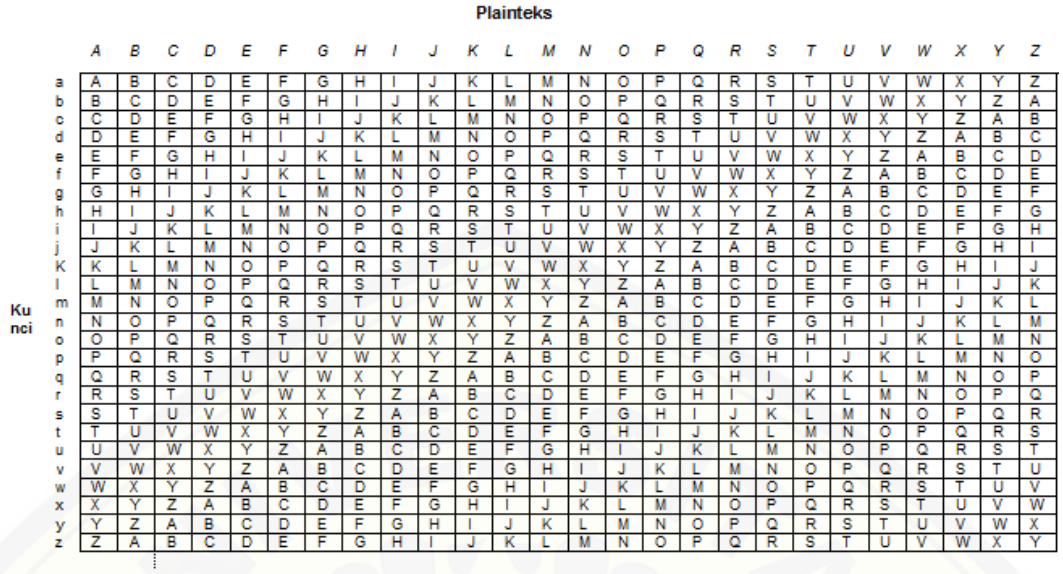
Algoritma *Vigenere Cipher* termasuk ke dalam *cipher* abjad-majemuk (*polyalpabetic substitution cipher*). Dipublikasikan oleh diplomat (sekaligus seorang kriptologis) Perancis, Blaise de Vigenere pada abad XVI (tahun 1586). Tetapi sebenarnya Giovan Batista Belaso telah menggambarkannya pertama kali pada tahun 1553 seperti ditulis di dalam bukunya *La Cifra del Sig. Giova Batista Belaso*. Algoritma tersebut baru dikenal luas 200 tahun kemudian yang oleh penemunya *cipher* tersebut kemudian dinamakan *Vigenere Cipher*. *Cipher* ini berhasil dipecahkan oleh Babbage dan Kasiski pada pertengahan Abad XIX.

Vigenere Cipher digunakan oleh Tentara Konfederasi (*Confederate Army*) pada Perang Sipil Amerika (*American Civil war*). Perang Sipil terjadi setelah *Vigenere Cipher* berhasil dipecahkan. *Vigenere Cipher* menggunakan bujursangkar *Vigenere* untuk melakukan enkripsi. Setiap baris di dalam bujursangkar menyatakan huruf-huruf *ciphertext* yang diperoleh dengan *Caesar Cipher* (Munir, 2004).

Secara matematis, proses enkripsi dari algoritma *Vigenere Cipher* menggunakan persamaan berikut:

$$c_i(p_i) = (p_i + k_i) \bmod 26 \qquad (2.2)$$

dimana c_i adalah *ciphertext* hasil enkripsi, p_i adalah *plaintext*, dan k_i adalah kunci (*key*) yang digunakan dalam proses enkripsi maupun dekripsi, dimana kunci harus berupa bilangan bulat positif. *Ciphertext* tersebut didapatkan dengan cara menggeser karakter dari *plaintext* (p_i) sejauh kunci (k_i). Fungsi dari modulo tersebut untuk membatasi karakter yang dipakai agar sesuai dengan jumlah karakter dalam *alphabet* yaitu 26. Pergeseran karakter menggunakan bujursangkar *Vigenere* pada Gambar 2.4 sebagai berikut:



Gambar 2.4 Bujursangkar *Vigenere*

(Sumber: Munir, 2004).

Jika panjang kunci lebih pendek daripada panjang plainteks, maka kunci diulang secara periodik. Misalkan panjang kunci = 20, maka 20 karakter pertama dienkripsi dengan persamaan (2.2), setiap karakter ke-*i* menggunakan kunci k_i. Untuk 20 karakter berikutnya, kembali menggunakan pola enkripsi yang sama. Tabel karakter yang sesuai dengan persamaan (2.2) adalah sebagai berikut:

Tabel 2.4 Konversi Karakter *Alphabet Plaintext* dan Kunci Sebelum Dienkripsi

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

Contoh:

Plaintext = THISPLAINTEXT; kunci = sony

Penyelesaian:

Menyamakan panjang *plaintext* dengan kunci, kunci diulang secara periodik.

Plaintext : T H I S P L A I N T E X T

Desimal : s o n y s o n y s o n y s

Dengan menggunakan Tabel 2.4, maka didapatkan hasil konversi *plaintext* tersebut dan kunci, yaitu:

Plaintext :	T	H	I	S	P	L	A	I	N	T	E	X	T
Desimal :	19	7	8	18	15	11	0	8	13	19	4	23	19

Kunci :	s	o	n	y	s	o	n	y	s	o	n	y	s
Desimal :	18	14	13	24	18	14	13	24	18	14	13	24	18

Proses selanjutnya yaitu menghitung nilai *ciphertext* menggunakan persamaan (2.2) dengan kunci seperti pada tabel di atas:

a. karakter T pada c_1 dengan kunci s akan mendapatkan *ciphertext*:
   $E(T) = (19+18) \bmod 26 = 11$
b. karakter H pada c_2 dengan kunci o akan mendapatkan *ciphertext*:
   $E(H) = (7+14) \bmod 26 = 21$
c. karakter I pada c_3 dengan kunci n akan mendapatkan *ciphertext*:
   $E(I) = (8+13) \bmod 26 = 21$
d. karakter S pada c_4 dengan kunci y akan mendapatkan *ciphertext*:
   $E(S) = (18+24) \bmod 26 = 16$
e. karakter P pada c_5 dengan kunci s akan mendapatkan *ciphertext*:
   $E(P) = (15+18) \bmod 26 = 7$
f. karakter L pada c_6 dengan kunci o akan mendapatkan *ciphertext*:
   $E(L) = (11+14) \bmod 26 = 25$
g. karakter A pada c_7 dengan kunci n akan mendapatkan *ciphertext*:
   $E(A) = (0+13) \bmod 26 = 13$
h. karakter I pada c_8 dengan kunci y akan mendapatkan *ciphertext*:
   $E(I) = (8+24) \bmod 26 = 6$
i. karakter N pada c_9 dengan kunci s akan mendapatkan *ciphertext*:
   $E(N) = (13+18) \bmod 26 = 5$
j. karakter T pada c_{10} dengan kunci o akan mendapatkan *ciphertext*:
   $E(T) = (19+14) \bmod 26 = 7$
k. karakter E pada c_{11} dengan kunci n akan mendapatkan *ciphertext*:
   $E(E) = (4+13) \bmod 26 = 17$
l. karakter X pada c_{12} dengan kunci y akan mendapatkan *ciphertext*:
   $E(X) = (23+24) \bmod 26 = 21$
m. karakter T pada c_{13} dengan kunci s akan mendapatkan *ciphertext*:

$E(T) = (19+18) \bmod 26 = 11$

Nilai *ciphertext* tersebut kemudian dikonversi lagi ke bentuk *alphabet* menggunakan Tabel 2.4, sehingga akan didapatkan:

Desimal : 11 21 21 16 7 25 13 6 5 7 17 21 11
ciphertext : L V V Q H Z N G F H R V L

Hasil *ciphertext* dari *plaintext* THISPLAINTEXT, adalah LVVQHZNGFHRVL

Huruf yang sama tidak selalu dienkripsi menjadi huruf *ciphertext* yang sama pula. Dari contoh diatas, huruf *plaintext* T dapat dienkripsi menjadi L atau H, dan huruf *ciphertext* V dapat merepresentasikan huruf *plaintext* H, I, dan X. Hal di atas merupakan karakteristik dari *cipher* abjad-majemuk yaitu setiap huruf *ciphertext* dapat memiliki kemungkinan banyak huruf *plaintext*.

Ciphertext yang didapatkan saat proses enkripsi dapat dikembalikan lagi ke dalam bentuk informasi awal (*plaintext*). Proses yang digunakan diistilahkan sebagai proses dekripsi. Metode yang digunakan tetap sama yaitu metode substitusi. Perbedaannya adalah persamaan umum yang dipakai, yang mana tanda pada nilai *key* yang digunakan bernilai negatif (–). Persamaan (2.2) menjadi seperti berikut:

$$p_i = D(c_i) = \begin{cases} (ci - ki) \bmod 26 & ,(ci - ki) \geq 0 \\ ((ci - ki) + 26) \bmod 26 & ,(ci - ki) < 0 \end{cases} \quad (2.3)$$

(Munir, 2004).

2.6 Algoritma *Caesar Cipher*

Algoritma substitusi atau yang lebih dikenal dengan *Caesar Cipher* merupakan algoritma kriptografi klasik dan termasuk kedalam algoritma kunci simetris karena hanya terdapat memakai satu kunci yang sama untuk proses enkripsi dan dekripsi suatu informasi. Algoritma ini pertama kali dikenalkan dan dipakai pada pemerintahan kaisar Romawi, Julius Caesar, untuk menyandikan pesan yang ia kirim ke salah satu gubernurnya. Oleh sebab itu, algoritma ini terkenal dengan istilah *Caesar Cipher*.

Algoritma *Caesar Cipher* merupakan algoritma yang cukup sederhana tetapi sering digunakan oleh berbagai pihak. Metode yang digunakan dalam algoritma ini yaitu metode substitusi, dimana pesan yang berupa karakter (*plaintext*) disandikan dengan cara disubstitusi dengan karakter lain dengan pola selisih tertentu dalam susunan abjad (*alphabet*) maupun susunan karakter ASCII. Hasil pergantian semua karakter-karakter dalam pesan tersebut akan mendapatkan suatu pesan baru yang telah disandikan (*ciphertext*) dan kunci (*key*) dari algoritma ini adalah pola (selisih) substitusi yang digunakan untuk mengenkripsi pesan tersebut (Munir, 2004).

Secara matematis, proses enkripsi dari algoritma *Caesar Cipher* menggunakan persamaan berikut:

$$c_i = E(p_i) = (p_i + k) \bmod 26 \qquad (2.4)$$

c_i adalah *ciphertext* hasil enkripsi, p_i adalah *plaintext*, dan k adalah kunci (*key*) yang digunakan dalam proses enkripsi maupun dekripsi, dimana kunci harus berupa bilangan bulat positif. *Ciphertext* tersebut didapatkan dengan cara menggeser karakter dari *plaintext* (p_i) sejauh kunci (k). Fungsi dari modulo tersebut untuk membatasi karakter yang dipakai agar sesuai dengan jumlah karakter dalam *alphabet* yaitu 26.

2.7 Algoritma SPICA-XB *Cipher*

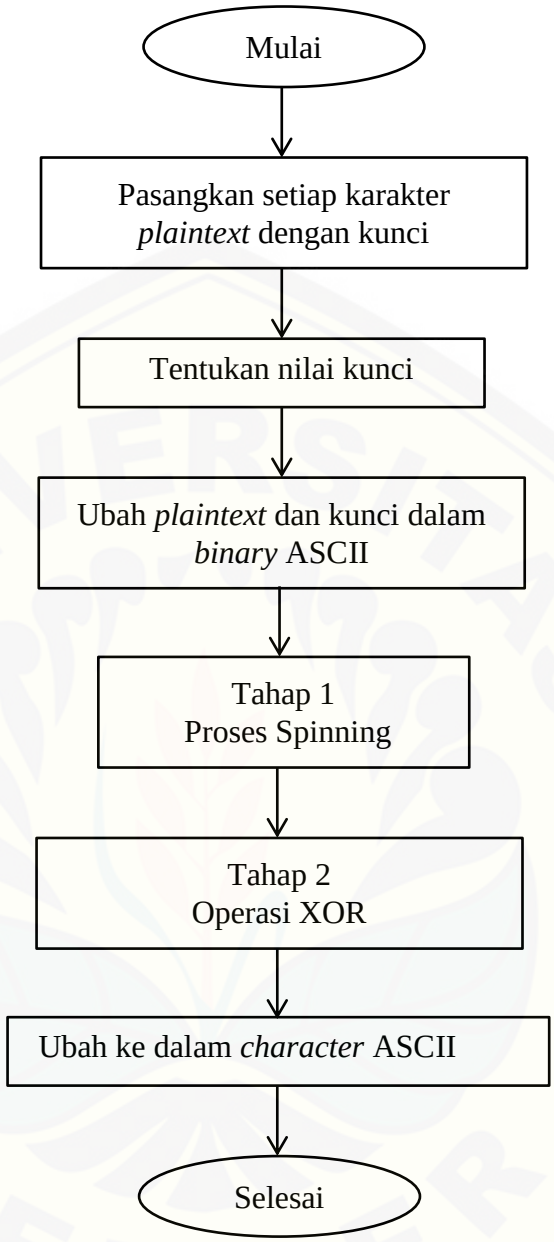
Algoritma SPICA-XB *Chiper* adalah algoritma yang menggabungkan sifat dari algoritma *Caesar Cipher* dengan beberapa sifat dari algoritma kriptografi modern. Asumsikan algoritma *Caesar Cipher* layaknya memutar sebuah roda (*spinning*). Berbeda dengan algoritma *Caesar Cipher* biasa, algoritma SPICA-XB *Cipher* bekerja dengan melakukan proses substitusi dengan kunci yang berputar. Perputaran tersebut akan sangat bergantung dari kunci dasar yang ditentukan oleh pembuat pesan sebelumnya (Islami, 2013).

Selain itu, jika dalam algoritma *Caesar Cipher* biasa proses substitusi akan dilakukan begitu saja (misalkan mengganti A dengan J), dalam algoritma SPICA-XB *Cipher* ini proses enkripsi akan dilakukan dengan terlebih dahulu merubah huruf-huruf yang ada kedalam bentuk *binary* ASCII 8 bit. Proses enkripsi

dilakukan dengan melakukan operasi penjumlahan lalu XOR antara *plaintext* dengan kunci. Berikut rincian algoritma SPICA-XB *Cipher* secara umum:

a. Pembuat pesan menentukan pesan serta kunci yang akan digunakan, panjang kunci adalah bebas namun merupakan kombinasi huruf-huruf dalam *alphabet*.
b. Selanjutnya setiap huruf dalam *plaintext* akan dipasangkan dengan huruf kunci pasangannya. Jika panjang kunci kurang dari panjang *plaintext*, akan dilakukan proses pengulangan kunci.
c. Setiap huruf dalam *plaintext* dan kunci akan diubah ke dalam bentuk *binary* ASCII berukuran 8 bit.
d. Sebagai persiapan dalam tahapan enkripsi awal, seluruh karakter dalam kunci akan diubah menjadi nilai angka yang berhubungan. Misalkan A akan diberi nilai 0, B diberi nilai 1, dan seterusnya. Hal ini akan sangat berpengaruh dalam proses *spinning* selama proses enkripsi.
e. Dalam tahapan proses enkripsi pertama, roda enkripsi akan dimulai dengan kondisi A=A. Selanjutnya proses enkripsi akan dilakukan karakter per karakter dengan sebelumnya dilakukan perputaran roda senilai dengan nilai kunci. Perputaran roda dilakukan berlawanan dengan arah jarum jam. Misalkan jika kunci yang saat itu aktif adalah huruf C, maka roda akan diputar sejauh tiga karakter berlawanan dengan arah jarum jam. Pasca perputaran, huruf *plaintext* akan memiliki pasangan kunci baru (selanjutnya akan disebut *middle key*). Karakter *plaintext* yang saat itu akan dienkripsi akan dilakukan operasi penjumlahan dengan *middle key* dalam bentuk *binary* ASCII 8 bit. Proses ini akan terus diulang hingga seluruh karakter terenkripsi.
f. Setelah tahapan enkripsi pertama dilakukan, selanjutnya dilakukan proses enkripsi kedua, yakni operasi XOR antara hasil enkripsi pertama dengan *binary* ASCII dari kunci awal.
g. Setelah didapat hasil enkripsi tahap kedua yakni *ciphertext*, setiap *binary* ASCII *ciphertext* akan diubah ke dalam bentuk *character* ASCII (Islami, 2013).

Gambar 2.5 menunjukkan langkah-langkah pada SPICA-XB *Cipher* pada proses enkripsi.



Gambar 2.5 Langkah-langkah Enkripsi SPICA-XB *Cipher*

Contoh :

Plaintext = SELAMAT PAGI

Kunci = BERMAIN

Penyelesaian:

1. Memasangkan setiap karakter *plaintext* dengan kunci yang berhubungan:

plaintext	:	S	E	L	A	M	A	T	P	A	G	I
Kunci	:	B	E	R	M	A	I	N	B	E	R	M

2. Merubah *plaintext* dan kunci ke dalam bentuk *binary* ASCII:

Plaintext :

S=01010011; E=01000101; L=01001100; A=01000001; M=01001101; A=01000001; T=01010100; P=01010000; A=01000001; G=01000111 I=01001001

Kunci :

B=01000010; E=01000101; R=01010010; M=01001101; A=01000001; I=01001001; N= 01001110

3. Menentukan nilai dari kunci

Kunci :	B	E	R	M	A	I	N
Nilai :	1	4	16	12	0	8	13

4. Proses enkripsi tahap 1

Roda Awal :

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S

T	U	V	W	X	Y	Z
T	U	V	W	X	Y	Z

Proses enkripsi huruf S, kunci yang berpasangan adalah huruf B. Huruf B ini memiliki nilai 1, karena itu roda akan berputar sejauh 1 karakter berlawanan dengan arah jarum jam (dalam bentuk tabel ini berarti pergeseran ke arah kiri). Sehingga roda akan menjadi roda yang akan disebut roda temp1, seperti berikut:

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
B	C	D	E	F	G	H	I	J	K	L	M	N	0	P	Q	R	S	T

T	U	V	W	X	Y	Z
U	V	W	X	Y	Z	A

Berdasarkan pada tabel di atas, maka dalam enkripsi tahap 1 dilakukan proses enkripsi antara huruf S dengan huruf T. Proses yang dilakukan adalah seperti berikut:

a. Merubah huruf S dan huruf T yang merupakan *middle key* ke dalam bentuk *binary* ASCII, sehingga memberi hasil:

01010011 dan 01010100

b. Melakukan operasi penjumlahan antara kedua nilai ASCII tersebut, sehingga diperoleh:

10100111

Nilai ini merupakan nilai enkripsi pertama untuk huruf S dari kata SELAMAT.

Proses selanjutnya adalah melakukan proses enkripsi untuk huruf E. Seperti yang sudah diperoleh sebelumnya, huruf E ini akan berpasangan dengan huruf kunci E juga (dari kata BERMAIN). Karena huruf E memiliki nilai empat, maka roda akan berputar sejauh 4 karakter berlawanan dengan arah jarum jam. Perputaran dilakukan berdasarkan pada roda pada kondisi temp1, sehingga diperoleh hasil sebagai berikut:

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
E	F	G	H	I	J	K	L	M	N	0	P	Q	R	S	T	U	V	W

T	U	V	W	X	Y	Z
X	Y	Z	A	B	C	D

Dapat dilihat pada “roda” di atas, bahwa dalam kondisi kali ini huruf E berpasangan dengan huruf I, sehingga proses enkripsi yang terjadi adalah:

01000101 + 01001001 = 10001110

Proses ini akan terus diulang untuk seluruh karakter yang terdapat di dalam *plaintext*.

5. Proses enkripsi tahap 2

Proses enkripsi tahap kedua yang dilakukan adalah dengan melakukan proses XOR antara hasil enkripsi pada enkripsi pertama, dengan kunci awal dalam bentuk *binary* ASCII. Enkripsi kata “SELAMAT PAGI” dengan menggunakan kunci “BERMAIN” akan dilakukan seperti berikut:

a. Dalam proses enkripsi yang sebelumnya telah diperoleh hasil (sebagai contoh proses enkripsi hanya akan dilakukan untuk huruf S dan E saja demi kesederhanaan) seperti berikut:

10100111 10001110

b. Karena kunci yang digunakan adalah kata “BERMAIN” dan dalam tahap sebelumnya sudah pernah ditentukan bahwa huruf S dan E

berpadanan dengan huruf B dan E, maka hasil enkripsi di atas akan dilakukan operasi XOR dengan huruf B dan E sehingga diperoleh hasil sebagai berikut:

10100111 10001111

XOR

01000010 01000101

=

11100101 11001010

Sehingga secara keseluruhan, kalimat SELAMAT PAGI dengan kunci BERMAIN menggunakan algoritma SPICA-XB, maka akan diperoleh hasil *ciphertext*:

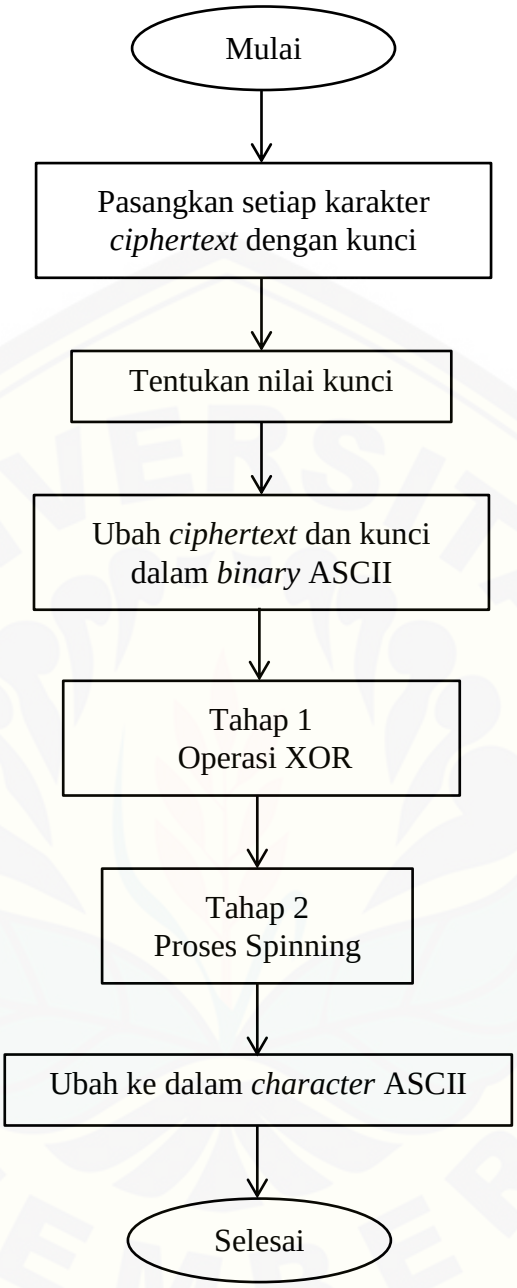
11100101 11001011 11111011 11000011 11011011 11000011 11111011 11100011 11000011 11001101 11010011

6. Mengubah *ciphertext* dalam *character* ASCII berdasarkan tabel Kode ASCII yang sudah ditentukan.

11100101 = å	11000011 = Ã	11111011 = û	11001101 = Í
11001011 = Ë	11011011 = Û	11100011 = ã	11010011 = Ó
11111011 = û	11000011 = Ã	11000011 = Ã	

Hasil *ciphertext* dari *plaintext* SELAMAT PAGI, adalah åËûÃÛÃûãÃÍÓ.

Ciphertext yang didapatkan saat proses enkripsi dapat dikembalikan lagi ke dalam bentuk informasi awal (*plaintext*). Proses yang digunakan diistilahkan sebagai proses dekripsi. Proses dekripsi dilakukan dengan melakukan operasi XOR lalu penjumlahan antara *ciphertext* dengan kunci. Gambar 2.6 menunjukkan langkah-langkah pada SPICA-XB *Cipher* pada proses dekripsi.



Gambar 2.6 Langkah-langkah Dekripsi SPICA-XB *Cipher*

2.8 Analisis Keamanan

Dalam proses perlindungan data pada sebuah gambar, terdapat beberapa metode yang dapat digunakan untuk menganalisis keamanan dari suatu algoritma yang digunakan. Berikut ini merupakan beberapa analisis keamanan yang ada.

2.8.1 Analisis Histogram Derajat Keabuan

Histogram dapat mencerminkan informasi dari penyebaran nilai *pixel* pada sebuah citra. Histogram dari sebuah *cipherimage* haruslah tersebar secara merata agar mampu aman dari serangan statistik, dan jika tidak maka nantinya histogram akan memberikan informasi yang dapat digunakan untuk mendeduksi *plainimage* (Behnia dkk., 2007).

2.8.2 Analisis Diferensial

Analisis Diferensial untuk menentukan perbedaan dari dua buah citra, maka dapat ditentukan dengan menghitung nilai dari *number of pixels change rate* (*NPCR*).

$$NPCR = \frac{\sum_{i,j} D(i,j)}{W \times H} \times 100\% \qquad (2.5)$$

dimana W dan H melambangkan lebar dan tinggi citra. $D(i,j)$ ditentukan sebagai berikut.

$$D(i,j) = \begin{cases} 0, & C(i,j) = C'(i,j) \\ 1, & C(i,j) \neq C'(i,j) \end{cases}$$

dimana $C(i,j)$ dan $C'(i,j)$ masing-masing menunjukkan nilai derajat keabuan dari baris *i* dan kolom *j* dari citra *C* dan *C'* (Akhavan dkk., 2011).

BAB 3. METODE PENELITIAN

3.1 Data Penelitian

Data yang penulis gunakan dalam penelitian ini adalah citra RGB yang berlaku sebagai *plainimage*. Gambar 3.1 merupakan citra RGB yang digunakan sebagai *plainimage.* Citra berdimensi 250×201 *pixels*.

Gambar 3.1 Citra Buah

3.2 Langkah-langkah Penelitian

Secara sistematis, langkah-langkah penelitian yang dilakukan dalam penelitian ini adalah sebagai berikut:

a. Studi Literatur

   Tahap ini dilakukan sebagai pembelajaran mengenai teori-teori yang dipakai sebagai acuan penelitian. Teori yang dipakai dalam penelitian ini adalah teknik kriptografi dengan menggunakan dua algoritma yakni algoritma *Vigenere Cipher* dan SPICA-XB *Cipher* dengan kunci berkelanjutan untuk mengenkripsi dan mendekripsi citra RGB.

b. Analisa Data

   Langkah-langkah berikut ini akan diterapkan pada seluruh kanal dari citra RGB yang menjadi *plainimage*. Masing-masing kanal *Red*, *Green*, dan *Blue* dilakukan enkripsi dan dekripsi.

1. Proses Enkripsi
   a) Tentukan nilai derajat setiap *pixel* pada masing-masing kanal *plainimage*.
   b) Pembagian nilai derajat keabuan pada setiap *pixel*.

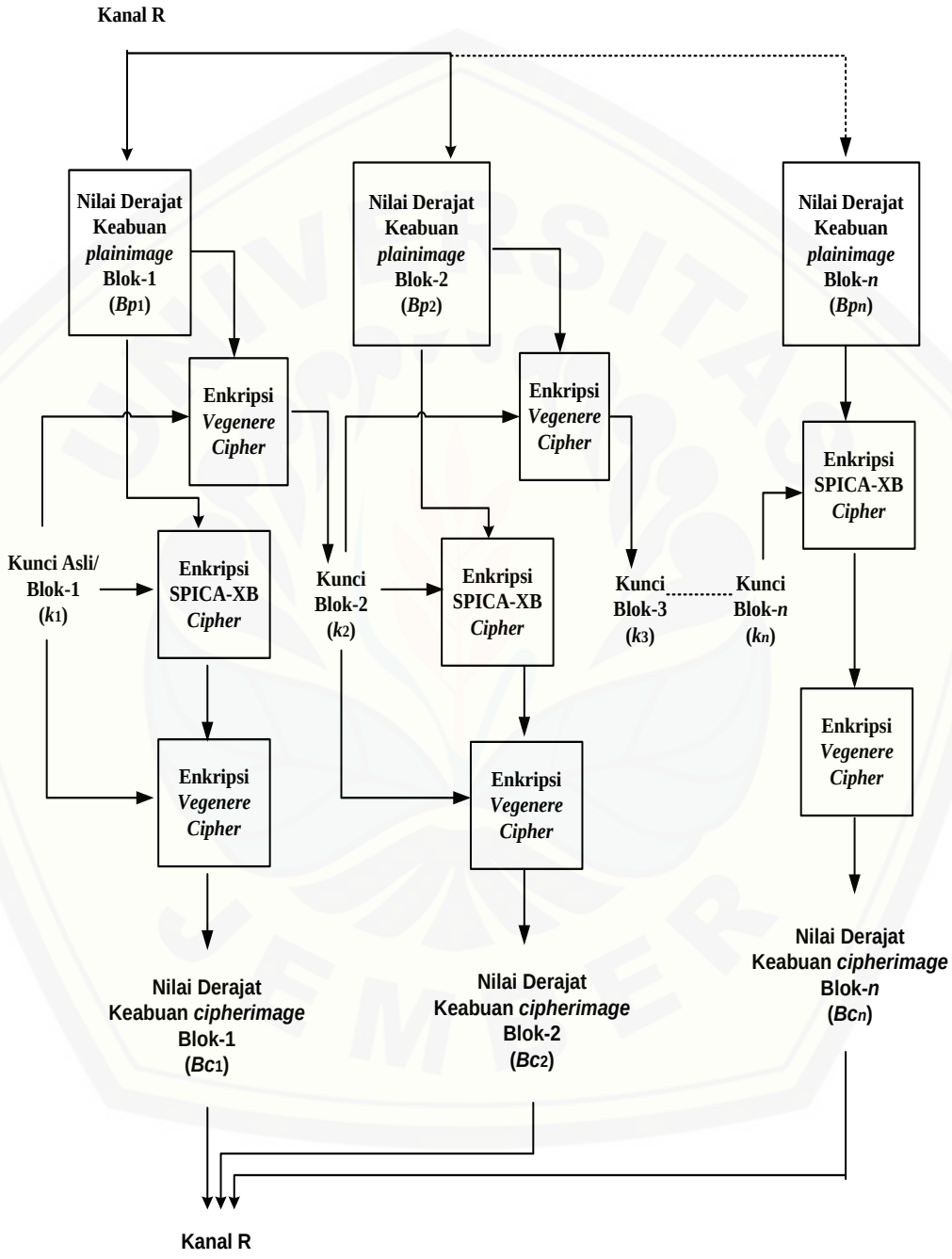
      Nilai derajat keabuan dibagi menjadi blok-blok dengan panjang blok sama dengan panjang kunci yang digunakan sehingga menghasilkan Bp_1 hingga Bp_n, dimana n yaitu banyak blok pada pembagian nilai derajat keabuan. Panjang kunci akan menyesuaikan dengan panjang nilai derajat keabuan.
   c) Enkripsi nilai derajat keabuan yang telah dibagi sebagai kunci berkelanjutan dengan algoritma *Vigenere Cipher*

      Setiap Bp_1 hingga Bp_{n-1} dienkrispsikan dengan algoritma *Vigenere Cipher* pada persamaan (2.2) untuk menghasilkan kunci k_2 sampai k_n. Hasil enkripsi Bp_1 yaitu k_2 digunakan sebagai kunci pada Bp_2 dan berkelanjutan pada hasil enkripsi Bp_{n-1} yaitu k_n yang digunakan untuk kunci pada Bp_n. Sedangkan kunci untuk Bp_1 menggunakan kunci asli (k_1). Kunci awal berbentuk *character* ASCII sehingga diubah dahulu menjadi desimal ASCII.
   d) Enkripsi nilai derajat keabuan yang telah dibagi dengan algoritma SPICA-XB *Cipher*

      Bp_1 hingga Bp_n dienkripsikan menggunakan algoritma SPICA-XB *Cipher* seperti langkah yang telah dijelaskan pada Gambar 2.5 dengan kunci yang telah ditentukan sebelumnya, sehingga didapatkan hasil yakni sp_1 hingga $sp_{10.050}$.
   e) Enkripsi kembali dengan algoritma *Vigenere Cipher*

      Hasil enkripsi algoritma SPICA-XB *Cipher* yakni sp_1 hingga $sp_{10.050}$ dienkripsikan lagi dengan menggunakan algoritma *Vigenere Cipher* pada persamaan (2.2) dengan kunci yang sama dengan pengenkripsian algoritma SPICA-XB *Cipher* yang akan menjadi hasil akhir dari proses enkripsi yakni Bc_1 hingga $Bc_{10.050}$.

Proses enkripsi dilakukan seperti pada Gambar 3.2. Proses tersebut akan diterapkan dengan menggunakan kanal *Red* dan diterapkan pula pada kanal *Green* dan kanal *Blue*.



.Gambar 3.2 Proses Enkripsi Kanal R

2. Proses Dekripsi
   a) Tentukan nilai derajat setiap *pixel* pada masing-masing kanal *cipherimage*.
   b) Pembagian nilai derajat keabuan pada setiap *pixel*.

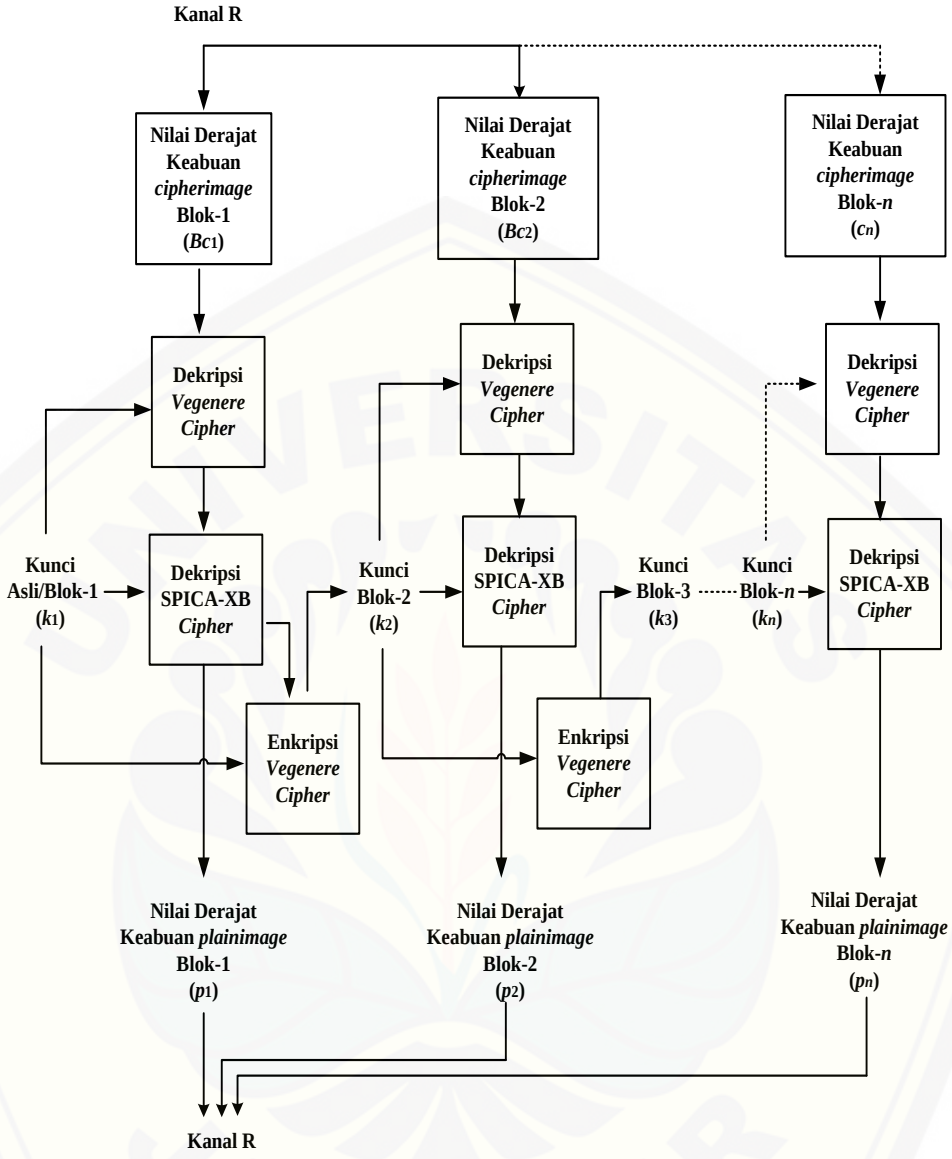
      Nilai derajat keabuan dibagi menjadi blok-blok dengan panjang blok sama dengan panjang kunci yang digunakan sehingga menghasilkan Bc_1 hingga Bc_n, dimana n yaitu banyak blok pada pembagian nilai derajat keabuan.
   c) Dekripsi nilai derajat keabuan yang telah dibagi dengan algoritma *Vigenere Cipher*

      Setiap Bc_1 hingga Bc_n didekripsikan dengan algoritma *Vigenere Cipher* pada persamaan (2.3) dengan kunci Bc_1 menggunakan kunci asli, sehingga didapatkan hasil yakni sp_1 hingga $sp_{10.050}$.
   d) Dekripsi nilai derajat keabuan dengan algoritma SPICA-XB *Cipher*

      Hasil dekripsi algoritma *Vigenere Cipher* yakni sp_1 hingga $sp_{10.050}$ didekripsikan lagi dengan menggunakan algoritma SPICA-XB *Cipher* seperti langkah yang telah dijelaskan pada Gambar 2.6 dengan kunci yang sama dengan pendekripsian algoritma *Vigenere Cipher*, sehingga didapatkan hasil yakni Bp_1 hingga $Bp_{10.050}$.
   e) Enkripsi nilai derajat keabuan dengan algoritma *Vigenere Cipher* untuk menghasilkan kunci berkelanjutan

      Setiap hasil Bc_1 hingga Bc_{n-1} yang telah didekripsi dua kali yakni Bp_1 hingga $Bp_{10.050}$ dienkripsikan dengan algoritma *Vigenere Cipher* pada persamaan (2.2) untuk menghasilkan kunci k_2 hingga k_n. Hasil enkripsi Bp_1 yaitu k_2 digunakan sebagai kunci pada Bc_2 dan berkelanjutan sampai hasil enkripsi Bp_{n-1} yaitu k_n yang digunakan untuk kunci pada Bc_n. Sedangkan kunci untuk Bc_1 menggunakan kunci asli (k_1).

Proses dekripsi dilakukan seperti pada Gambar 3.3. Proses tersebut akan diterapkan dengan menggunakan kanal *Red* dan diterapkan pula pada kanal *Green* dan kanal *Blue*.



Gambar 3.3 Proses Dekripsi Kanal R

c. Perancangan Program

Perancangan program dilakukan dengan menggunakan *software* Matlab 2009a. Desain program menggunakan aplikasi GUI (*Graphic User Interface*) yang ada dalam Matlab untuk merancang form *layout* dan desain interaksi program agar lebih menarik.

d. Pembuatan Program

Pembuatan program didasarkan pada konsep enkripsi dan dekripsi algoritma *Vigenere Cipher* dan SPICA-XB *Cipher* dengan kunci berkelanjutan.

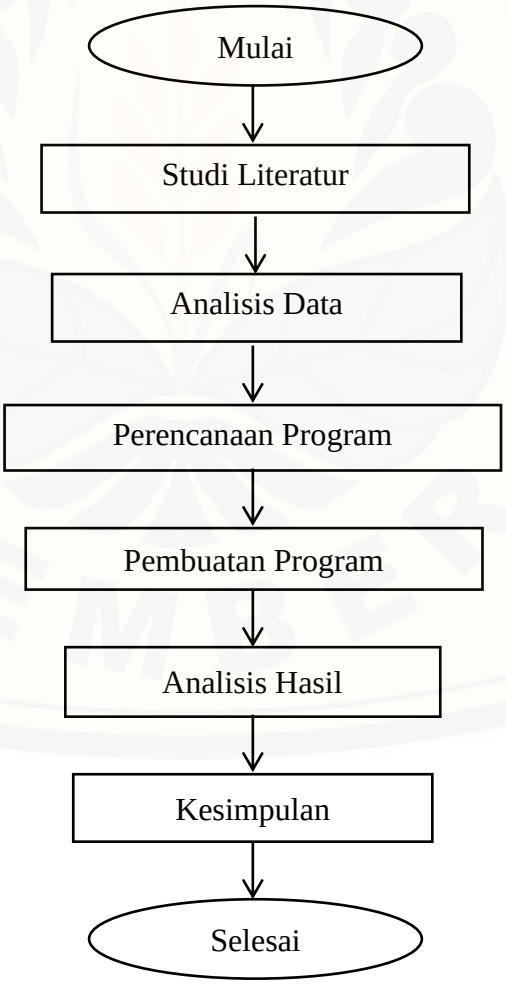
e. Analisis Hasil

Tahap pengujian dan menganalisa program yang telah dibuat agar sesuai dengan konsep teori yang digunakan dan agar sesuai dengan hasil yang diinginkan. Tahap ini dilakukan juga analisis keamanan pada hasil enkripsi data citra menggunakan analisis histogram derajat keabuan, dan analisis diferensial.

f. Kesimpulan

Mengambil kesimpulan dari penelitian yang telah dilakukan, yaitu dengan menganalisis proses enkripsi dan dekripsi algoritma *Vigenere Cipher* dan SPICA-XB *Cipher* dengan kunci berkelanjutan.

Flowchart dari langkah-langkah penelitian yang dilakukan dapat diamati pada Gambar 3.4.



Gambar 3.4 *Flowchart* Langkah-langkah Penelitian

BAB 5. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan penelitian yang telah dilakukan, maka dapat diambil beberapa kesimpulan sebagai berikut.

a. Proses enkripsi citra RGB dengan algoritma *Vigenere Cipher* dan algoritma SPICA-XB *Cipher* dengan kunci berkelanjutan dapat dilakukan dan dapat memberikan keamanan bagi data karena proses ini menghasilkan *cipherimage* yang tidak mengandung informasi dari *plainimage*.
b. Proses dekripsi citra RGB dapat dilakukan dengan baik jika kunci diketahui. Proses dekripsi mampu mengembalikan *cipherimage* menjadi *plainimage* tanpa adanya informasi yang hilang.
c. Metode enkripsi yang diajukan memiliki tingkat keamanan yang baik karena tidak dapat diserang menggunakan analisis histogram dan analisis diferensial.

5.2 Saran

Saran yang diberikan untuk penelitian selanjutnya yakni :

a. Menggabungkan proses enkripsi algoritma *Vigenere Cipher* dan algoritma SPICA-XB *Cipher* sehingga tidak perlu dua kali proses enkripsi untuk memperpendek waktu berjalannya proses enkripsi maupun dekripsi.
b. Menerapkan kunci berkelanjutan pada algoritma enkripsi klasik ataupun modern yang lainnya seperti Hill, Playfair, DES, AES, dsb.

DAFTAR PUSTAKA

Akhavan, A., A. Samsudin, & A. Akhsani. 2011. A symmetric image encryption scheme based on combination of nonlinear chaotic maps. *Journal of the Franklin Institute* 348 (8): 1797-1813.

Behnia, S., A. Akhsani, S. Ahadpour, H. Mahmodi, & A. Akhavan. 2007. A fast chaotic encryption scheme based on piecewise nonlinear chaotic maps. *Physics Letters A* 366 (4-5): 391-396.

Dulimarta, H.S. 1997. *Diktat Kuliah Pengolahan Citra*. Bandung: Jurusan Teknik Informatika Institut Teknologi Bandung.

Islami, Rizal P. 2013. Modifikasi Algoritma Caesar Cipher Menjadi SPICA-XB (Spinning Caesar dengan XOR Binary). http://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2012-2013/Makalah12013/Makalah1_Kripto2013-005.pdf. [Diakses pada 26 November 2016].

Murni, A. 1992. *Pengantar Pengolahan Citra*. Jakarta: PT Elex Media Komputindo.

Munir, R. 2002. *Pengolahan Citra Digital*. Bandung: Departemen Teknik Informatika ITB.

Munir, R. 2004. *Algoritma Kriptografi Klasik*. Bandung: Departemen Teknik Informatika ITB.

Munir, R. 2006. *Diktat Kuliah IF5054 Kriptografi*. Jakarta: Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika.

Prayudi. 2005. Studi Analisis Algoritma Rivest Code 6 (RC6) Dalam Enkripsi/Dekripsi Data. *Seminar Nasional Aplikasi Teknologi Informasi2005* (*SNATI 2005*).

Price, W. T., & C. Peselnick. 1987. *Elements of Data Processing Mathematics*. 3rd ed. USA: Harcourt.

Sadikin, R. 2012. *Kriptografi untuk Keamanan Jaringan dan Implementasinya dalam Bahasa Java*. Yogyakarta: Penerbit Andi.

Yohandri. 2013. *Elektronika Digital.* Padang: Universitas Padang.

LAMPIRAN

LAMPIRAN A. Matriks Derajat Keabuan dari Gambar 4.1(a)
250x201 *pixels*
Kanal *Red*

105	**155**	**129**	**143**	**101**	**124**	**104**	**80**	**73**	**72**	119	209	...	30
174	150	96	128	102	106	108	134	92	176	220	186	...	43
151	136	109	139	140	150	164	184	190	184	167	152	...	20
134	162	180	186	187	194	192	180	182	221	238	213	...	102
180	190	206	203	202	194	190	186	196	190	200	226	...	146
203	191	188	179	187	197	194	196	200	205	199	202	...	147
204	194	189	174	173	197	184	185	208	197	194	215	...	117
202	189	193	195	180	187	176	205	189	212	205	191	...	48
204	182	180	192	190	186	188	187	195	210	204	195	...	72
194	182	190	201	193	186	190	193	200	216	210	194	...	139
195	185	194	201	192	184	187	183	196	214	209	189	...	144
208	195	194	194	187	184	183	173	192	207	202	185	...	175
204	200	201	194	176	177	186	180	184	195	190	185	...	195
192	200	211	197	172	170	184	185	173	182	180	183	...	206
195	201	208	198	177	173	167	149	157	170	171	176	...	205
210	199	195	189	184	178	147	99	148	165	169	168	...	192
198	196	203	169	180	169	143	142	177	194	179	164	...	195
184	186	198	191	154	163	191	186	196	192	179	178	...	161
169	203	180	177	150	158	191	214	199	189	184	189	...	164
171	196	161	146	147	154	165	190	184	190	190	190	...	204
199	177	193	176	171	190	208	198	187	194	194	192	...	229
145	142	179	159	170	183	193	202	205	196	195	200	...	211
118	155	167	156	174	176	177	203	208	193	200	209	...	188
149	171	171	188	163	173	193	181	193	190	206	213	...	185
194	168	179	208	207	195	188	176	188	209	212	204	...	183
201	214	221	213	199	182	174	180	206	212	206	196	...	185
168	188	182	175	182	170	160	178	212	204	198	194	...	187
199	191	180	189	196	180	172	190	203	188	192	199	...	188
202	181	184	195	179	166	174	183	196	179	182	189	...	182
198	180	187	194	187	197	207	190	200	184	175	172	...	172
195	186	181	187	191	194	188	177	198	193	178	172	...	171
165	175	173	178	176	149	142	174	182	189	182	178	...	174
154	150	162	175	162	139	148	177	193	175	183	155	...	170
117	141	170	184	181	178	186	199	175	163	176	151	...	152
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱	⋮
183	187	198	203	196	184	188	199	202	210	207	200	...	147

250x201 *pixels*
Kanal *Green*

78	**131**	**108**	**129**	**90**	**122**	**105**	**86**	**80**	**78**	...	31
148	127	76	112	92	100	107	136	95	178	...	44
129	114		122	127	141	157	178	187	181	...	19
111	141	159	166	168	175	175	163	167	206	...	101
156	166	182	176	172	164	158	154	163	158	...	144
173	159	153	142	143	151	143	145	149	154	...	143
161	149	139	120	112	130	112	111	134	123		112
143	129	128	124	99	103	84	108	92	115	...	42
126	101	96	100	92	81	77	73	79	94	...	128
98	84	87	93	79	67	67	66	70	86	...	130
81	71	75	80	66	56	54	48	61	79	...	160
81	68	67	66	56	51	48	38	57	72	...	128
71	67	69	62	45	46	54	51	54	66	...	139
52	62	75	65	41	43	60	62	52	61	...	157
53	61	72	66	51	53	51	35	45	58	...	161
74	67	65	66	67	65	37	0	38	55	...	56
75	77	86	57	73	64	38	35	68	83	...	72
69	73	90	87	53	62	89	80	85	81	...	66
56	92	74	74	49	57	89	108	88	77	...	89
58	85	55	45	46	54	63	81	72	73	...	101
88	69	87	75	71	90	104	90	73	75	...	107
35	35	74	57	70	83	89	92	89	76	...	98
8	48	62	54	74	74	71	93	89	71	...	69
38	61	66	86	61	69	85	70	73	65	...	67
81	57	72	103	102	89	78	60	66	82	...	79
85	101	110	106	93	73	62	61	81	82	...	65
47	70	69	64	71	60	45	57	85	72	...	59
75	69	62	73	83	65	52	67	73	56	...	36
73	54	62	75	61	49	52	57	65	46	...	3
65	48	60	72	66	77	83	64	69	51	...	21
60	52	52	60	69	72	64	49	65	58	...	50
36	48	50	57	58	31	24	53	58	64	...	43
43	42	54	69	56	33	43	71	87	69	...	34
10	34	63	78	75	72	83	96	73	61	...	45
41	49	52	57	79	97	82	50	62	52	...	57
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱	⋮
52	40	27	28	38	46	46	43	45	47	...	104

250x201 *pixels*
Kanal *Blue*

31	**83**	**63**	**82**	**45**	**74**	**61**	**40**	**36**	**34**	77	164	...	17
97	77	26	63	43	52	59	87	48	131	173	136	...	28
71	56	32	66	72	86	103	126	136	128	109	91	...	1
44	76	96	105	109	117	119	107	112	151	167	140	...	99
84	94	112	109	108	102	99	97	109	101	112	136	...	81
100	86	85	74	82	91	88	90	96	101	94	99	...	121
92	81	76	58	57	77	64	64	89	78	74	98	...	116
83	69	72	72	52	59	45	73	57	80	76	63	...	83
78	54	52	61	57	51	50	49	56	71	68	59	...	10
60	49	56	65	55	47	51	51	57	73	70	55	...	30
55	45	53	61	51	43	45	42	57	73	68	47	...	96
62	51	52	53	46	44	44	34	54	69	64	47	...	101
56	52	56	50	35	38	49	46	52	61	49	50	...	131
39	49	61	52	31	34	52	55	44	53	52	51	...	102
39	46	58	53	37	39	38	24	33	46	38	46	...	114
62	54	52	51	50	49	20	0	23	40	20	40	...	134
68	69	77	45	57	45	19	15	48	64	19	35	...	141
66	67	80	74	35	42	67	58	65	61	67	48	...	46
50	83	61	59	31	37	67	86	68	55	67	56	...	68
50	74	41	27	26	31	40	58	50	53	40	56	...	79
77	56	71	55	48	66	79	64	49	53	79	55	...	84
22	19	55	35	46	57	62	67	64	52	62	59	...	75
0	30	43	32	50	49	45	66	65	48	45	66	...	47
21	44	45	63	38	44	59	43	49	43	59	70	...	47
65	40	52	81	80	65	53	37	45	63	53	60	...	63
70	85	93	86	71	50	40	39	61	66	40	52	...	52
36	56	53	45	51	37	24	38	68	59	24	53	...	52
65	58	50	58	65	46	35	51	60	44	35	59	...	33
67	45	51	61	47	32	37	43	55	39	37	52	...	6
60	43	53	61	55	63	71	52	61	46	71	38	...	27
57	49	47	53	58	61	54	40	60	54	54	40	...	61
31	42	43	48	46	19	12	42	50	58	12	53	...	54
32	30	41	55	40	17	24	55	73	56	24	42	...	42
2	24	53	65	61	58	66	79	58	46	66	38	...	49
44	51	51	53	72	90	73	40	51	41	73	33	...	57
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱	⋮
70	55	40	35	40	40	34	25	21	33	29	22	...	59

LAMPIRAN B. Matriks Derajat Keabuan dari Gambar 4.1(b)

250x201 *pixels*

Kanal *Red*

72	**156**	**49**	**14**	**249**	**42**	**193**	**105**	**74**	**17**	...	240
248	86	195	47	157	234	216	117	203	9	...	118
32	94	246	69	78	226	53	221	229	155	...	28
135	108	60	90	118	140	17	233	201	152	...	152
145	185	186	202	185	202	108	213	161	43	...	56
113	205	118	197	231	127	215	49	198	180	...	90
91	20	177	170	70	238	202	64	171	212	...	10
123	235	125	248	70	134	147	35	157	9	...	66
103	13	129	52	143	29	217	2	251	39	...	66
140	106	69	144	191	105	47	47	59	179	...	107
45	251	251	236	138	22	24	123	71	28	...	166
128	47	238	230	89	121	21	113	234	124	...	152
91	212	231	222	208	178	86	71	27	10	...	203
103	252	122	4	24	77	64	149	186	14	...	35
128	219	21	227	177	250	216	77	59	208	...	229
118	141	228	121	151	162	158	36	167	139	...	125
12	101	98	19	145	105	150	190	155	247	...	30
139	140	116	179	245	7	68	13	226	1	...	141
108	102	139	90	70	185	233	164	230	149	...	0
141	147	227	110	76	139	43	120	198	242	...	214
63	203	243	71	228	12	39	152	10	42	...	222
69	228	98	75	24	34	6	61	94	205	...	103
5	182	32	4	217	28	84	245	183	227	...	122
121	221	184	20	113	3	250	252	95	199	...	13
204	31	7	89	81	9	179	183	250	53	...	80
5	69	65	164	92	248	151	123	115	142	...	84
61	162	164	154	221	75	194	62	42	126	...	30
189	68	129	252	118	120	124	148	18	8	...	227
9	34	206	49	208	102	147	63	49	28	...	172
85	196	33	227	99	138	189	115	132	136	...	84
93	58	108	194	191	120	146	53	99	30	...	36
16	8	8	146	247	174	120	85	144	124	...	162
30	29	99	79	108	114	70	97	55	133	...	179
93	238	4	237	146	141	241	193	124	123	...	237
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱	⋮
150	192	139	170	115	220	211	164	48	64	...	169

250x201 *pixels*
Kanal *Green*

35	**164**	**94**	**32**	**12**	**217**	**115**	**111**	**251**	**152**	...	146
179	49	164	7	29	35	124	103	221	246	...	27
213	172	6	187	64	75	199	140	82	209	...	103
225	166	228	161	124	136	176	169	33	131	...	135
111	172	7	216	239	79	110	47	141	225	...	173
95	126	232	42	244	105	51	166	29	166	...	215
16	130	241	222	93	245	78	31	126	136	...	252
101	110	225	31	51	193	208	22	204	119	...	244
198	139	166	84	61	16	3	87	255	87	...	169
105	22	185	200	176	88	241	5	129	54	...	60
91	230	207	51	185	98	96	8	182	188	...	105
134	112	254	184	205	157	67	103	91	3	...	249
0	49	34	126	144	209	51	126	120	158	...	119
40	77	89	224	119	228	224	118	216	67	...	20
188	107	153	149	247	109	149	62	174	19	...	213
111	44	39	245	70	222	117	74	183	91	...	4
209	229	202	120	251	245	204	217	92	193	...	97
248	120	196	253	76	4	224	94	0	18	...	251
27	29	242	3	219	244	36	48	247	62	...	74
237	255	174	190	162	73	10	170	192	208	...	69
200	0	162	124	150	214	166	145	253	42	...	151
136	4	161	161	219	117	32	29	236	152	...	160
252	176	70	97	125	86	119	167	130	128	...	118
238	65	197	58	27	201	8	30	6	17	...	180
84	11	172	138	111	96	203	55	157	195	...	90
180	16	207	155	161	170	104	49	231	84	...	197
99	187	121	74	182	174	246	4	234	163	...	153
131	249	38	79	1	178	22	97	100	202	...	111
68	24	188	188	199	210	88	243	163	48	...	215
84	88	23	82	222	139	216	118	248	10	...	255
132	117	95	33	230	74	199	2	184	252	...	226
113	207	168	249	203	130	13	105	183	102	...	245
211	38	20	11	60	199	191	23	154	36	...	16
97	102	149	128	184	129	137	201	196	24	...	43
94	21	122	9	104	30	117	199	12	66	...	192
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱	⋮
42	228	249	4	221	79	80	54	104	154	...	18

250x201 *pixels*
Kanal *Blue*

110	**212**	**107**	**207**	**193**	**150**	**142**	**233**	**26**	**114**	...	254
96	153	97	168	250	205	176	201	59	203	...	13
67	137	90	251	119	91	217	99	98	226	...	45
185	22	249	49	184	63	170	141	14	71	...	178
150	118	163	233	186	22	23	230	202	36	...	220
20	243	133	19	163	81	75	90	129	72	...	149
24	115	90	87	250	16	60	186	134	160	...	219
215	75	97	49	32	121	148	193	41	37	...	223
154	245	112	131	28	91	241	131	227	174	...	75
49	82	227	72	168	133	168	148	35	202	...	36
41	116	181	208	106	153	31	184	29	12	...	5
66	126	177	177	181	133	129	23	83	187	...	55
18	185	84	16	175	75	239	129	217	167	...	42
145	157	119	190	93	223	118	28	163	55	...	189
233	220	150	97	218	70	127	137	199	156	...	225
246	243	190	84	32	211	37	174	57	121	...	238
49	33	113	236	105	240	164	145	156	200	...	191
212	66	148	44	183	201	110	164	211	6	...	57
78	255	233	116	90	199	100	212	78	194	...	189
133	97	225	27	174	84	197	209	62	21	...	163
206	89	92	131	112	12	251	11	146	90	...	198
176	255	86	69	140	92	72	67	216	102	...	149
85	132	90	199	80	88	119	112	165	183	...	50
50	101	62	25	98	33	16	69	99	101	...	107
199	100	111	91	159	87	121	114	140	163	...	17
253	206	225	148	156	135	26	119	77	34	...	17
20	22	43	230	81	232	145	126	244	185	...	110
70	35	78	5	141	165	248	15	83	89	...	248
244	63	59	25	225	157	64	115	130	249	...	216
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱	⋮
103	81	156	128	84	132	204	98	67	186	...	160