PENYANDIAN CITRA MENGGUNAKAN ALGORITMA *4D PLAYFAIR CIPHER* DENGAN PEMBANGKITAN KUNCI MODIFIKASI *LINEAR FEEDBACK SHIFT REGISTER* (LFSR)

SKRIPSI

Oleh

Rivi Tri Rahayu

NIM 151810101020

JURUSAN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS JEMBER

2019

PENYANDIAN CITRA MENGGUNAKAN ALGORITMA *4D PLAYFAIR CIPHER* DENGAN PEMBANGKITAN KUNCI MODIFIKASI *LINEAR FEEDBACK SHIFT REGISTER* (LFSR)

SKRIPSI

Diajukan guna melengkapi dan memenuhi salah satu syarat untuk menyelesaikan Program Studi Matematika (S1) dan mencapai gelar Sarjana Sains

Oleh

Rivi Tri Rahayu

NIM 151810101020

JURUSAN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS JEMBER

2019

PERSEMBAHAN

Skripsi ini saya persembahkan untuk:

1. Kedua Orang tuaku tersayang, Ayahanda Khairi Mulyadi dan Ibunda Lutfiyah, serta kakak-kakakku Rivi Prima Setiawan dan Rivi Dwi Agus Maulida yang senantiasa memberi dukungan dan do'a;
2. Sahabat-sahabatku yang senantiasa memberikan motivasi dan do'a;
3. Guru-guru dan dosen dari sekolah dasar hingga perguruan tinggi;
4. Almamater Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

MOTTO

Banyak kegagalan dalam hidup ini dikarenakan orang-orang tidak menyadari betapa dekatnya mereka dengan keberhasilan saat mereka menyerah.
(Thomas Alva Edison)[1]

Jika kamu takut membuat kesalahan maka kamu tidak akan melakukan apapun.
(Asih)[2]

[1] Edison. T. A. dalam Nugroho, A. 2013. Pengaruh Motivasi Dan Minat Terhadap Prestasi Siswa Pada Mata Diklat Keselamatan Dan Kesehatan Kerja Di Smk Negeri 1 Sedayu. *Skripsi*.Yogyakarta: Program Studi Pendidikan Teknik Mesin Fakultas Teknik, Universitas Negeri Yogyakarta.

[2] Asih. 2015. Motivasi Belajar Siswa Di Smp Negeri 15 Yogyakarta. *Skripsi*. Yogyakarta: Program Studi Manajemen Pendidikan Jurusan Administrasi Pendidikan Fakultas Ilmu Pendidikan, Universitas Negeri Yogyakarta

PERNYATAAN

Saya yang bertanda tangan di bawah ini:

nama : Rivi Tri Rahayu

NIM : 151810101020

menyatakan dengan sesungguhnya bahwa skripsi yang berjudul "Penyandian Citra Menggunakan Algoritma *4D Playfair Cipher* dengan Pembangkitan Kunci Modifikasi *Linear Feedback Shift Register* (LFSR)" adalah benar-benar hasil karya sendiri, kecuali kutipan yang sudah saya sebutkan sumbernya, belum pernah diajukan pada institusi mana pun, dan 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, 21 Januari 2019

Yang menyatakan,

Rivi Tri Rahayu

NIM 151810101020

SKRIPSI

PENYANDIAN CITRA MENGGUNAKAN ALGORITMA *4D PLAYFAIR CIPHER* DENGAN PEMBANGKITAN KUNCI MODIFIKASI *LINEAR FEEDBACK SHIFT REGISTER* (LFSR)

Oleh:

Rivi Tri Rahayu

NIM. 151810101020

Pembimbing

Dosen Pembimbing Utama : Abduh Riski, S.Si., M.Si.

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

PENGESAHAN

Skripsi berjudul "Penyandian Citra Menggunakan Algoritma *4D Playfair Cipher* Dengan Pembangkitan Kunci Modifikasi *Linear Feedback Shift Register* (LFSR)" telah diuji dan disahkan pada:

hari, tanggal :

tempat : Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Jember

Tim Penguji:

Ketua,	Anggota I,
Abduh Riski, S.Si., M.Si.	Ahmad Kamsyakawuni, S.Si., M.Kom.
NIP 199004062015041001	NIP 197211291998021001
Anggota II,	Anggota III,
Kiswara Agung Santoso, S.Si., M.Kom.	Kusbudiono, S.Si., M.Si.
NIP 197209071998031003	NIP 197704302005011001

Mengesahkan
Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Jember

Drs. Sujito., Ph.D.
NIP 196102041987111001

RINGKASAN

Penyandian Citra Menggunakan Algoritma *4D Playfair Cipher* Dengan Pembangkitan Kunci Modifikasi *Linear Feedback Shift Register* (LFSR); Rivi Tri Rahayu, 151810101020; 2019: 70 Halaman; Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Perkembangan teknologi yang semakin cepat membuat semakin mudahnya seseorang mengirimkan suatu pesan kepada orang lain tetapi juga dapat membuat semakin mudahnya pihak ketiga menyabotase isi pesan tersebut maka dibutuhkan suatu teknik yang dinamakan kriptografi untuk mengamankan isi pesan. Kriptografi merupakan suatu ilmu untuk melindungi atau menyembunyikan pesan agar tidak diketahui oleh pihak ketiga dengan cara mengubah isi pesan asli menjadi kode–kode yang sulit dimengerti maknanya.

Pada penelitian ini membahas tentang peningkatan keamanan pada penyandian citra menggunakan algoritma *4D Playfair Cipher* dan pembangkitan matriks kuncinya menggunakan konsep algoritma *Linear Feedback Shift Register* (LFSR) yang telah dimodifikasi. Konsep algoritma *Linear Feedback Shift Register* (LFSR) yang digunakan yaitu dengan menambahkan langkah *Shift*, Logika XOR dan XNOR. Sedangkan Tedapat dua langkah modifikasi yang dilakukan pada LFSR yaitu dengan penambahan bilangan dan rotasi.

Proses enkripsi menggunakan *Playfair Cipher* dan *3D Playfair Cipher* tediri dari tiga tahap yaitu mensubstitusikan kunci kedalam matriks kunci, membagi piksel citra menjadi digram untuk *Playfair Cipher* dan trigram untuk *3D Playfair Cipher* serta proses enkripsi itu sendiri. Hasil penyandian citra menggunakan *Playfair Cipher* dan *3D Playfair Cipher* terlihat acak. Pada proses enkripsi menggunakan *4D Playfair Cipher*-Modifikasi LFSR terdiri dari empat tahap yaitu pembangkitan kunci menggunakan Modifikasi LFSR, kunci yang dihasilkan dari Modifikasi LFSR disubstitusikan kedalam matriks kunci algoritma *4D Playfair Cipher*, membagi piksel citra menjadi *quartets* dan proses enkripsi itu sendiri. Hasil dari proses enkripsi pada *4D Playfair* Cipher terlihat acak (tidak berpola)

dan rata. Proses dekripsi menggunakan *Playfair Cipher*, *3D Playfair Cipher* dan *4D Playfair Cipher*-Modifikasi LFSR berhasil mengembalikan *cipherimage* menjadi *plainimage*.

Berdasarkan data penelitian yang terdiri dari 4 citra yang telah diuji, hasil dari uji histogram menghasilkan histogram yang lebih seragam menggunakan *4D Playfair Cipher*-Modifikasi LFSR dibandingkan hasil histogram menggunakan *Playfair Cipher* dan *3D Playfair Cipher*, terlihat juga dari perhitungan X^2 bahwa hasil yang peroleh *4D Playfair Cipher*-Modifikasi LFSR lebih kecil dibandingkan perhitungan yang dihasilkan menggunakan *Playfair Cipher* dan *3D Playfair Cipher*. Hasil perhitungan NPCR yang diperoleh menggunakan *4D Playfair Cipher*-Modifikasi LFSR adalah sebesar 81,35% hingga 92,83% dan hasil perhitungan UACI diperoleh sebesar 22,84% hingga 31,52%. Sedangkan hasil perhitungan NPCR yang diperoleh menggunakan *Playfair Cipher* adalah sebesar 98,95% hingga 99,40% dan hasil perhitungan UACI diperoleh sebesar 21,76% hingga 32,01%. Hasil perhitungan NPCR yang diperoleh menggunakan *3D Playfair Cipher* adalah sebesar 99,14% hingga 99,34% dan hasil perhitungan UACI diperoleh sebesar 23,53% hingga 44,93%. Berdasarkan hasil yang telah diperoleh secara numerik nilai NPCR dan UACI menggunakan *Playfair Cipher* dan *3D Playfair Cipher* dapat dikatakan lebih baik dibandingkan dengan nilai NPCR dan UACI menggunakan *4D Playfair Cipher*-Modifikasi LFSR, tetapi secara visual hasil dari enkripsi *4D Playfair Cipher*-Modifikasi LFSR lebih baik dari *Playfair Cipher* karena citra yang dihasilkan oleh *4D Playfair Cipher*-Modifikasi LFSR acak dan lebih rata.

Berdasarkan perbandingan antara hasil perhitungan dari histogram, NPCR, UACI, dan *cipherimage* yang dihasilkan. Tingkat keamanan hasil penyandian citra menggunakan *4D Playfair Cipher* dengan pembangkitan kunci Algoritma Modifikasi LFSR lebih kuat dibandingkan dengan hasil penyandian citra menggunakan *Playfair Cipher* dan *3D Playfair Cipher*, dapat dilihat pada hasil histogram yang seragam, hasil perhitungan X^2 lebih kecil, serta *cipherimage* yang dihasilkan acak dan merata daripada menggunakan *Playfair Cipher* dan *3D Playfair Cipher.*

PRAKATA

Puji syukur ke hadirat Allah SWT. atas segala rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan skripsi yang berjudul "Penyandian Citra Menggunakan Algoritma *4D Playfair Cipher* Dengan Pembangkitan Kunci Modifikasi *Linear Feedback Shift Register* (LFSR)". Skripsi ini disusun untuk memenuhi salah satu syarat menyelesaikan pendidikan strata satu (S1) pada Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Penyusunan skripsi ini tidak lepas dari bantuan berbagai pihak. Oleh karena itu, penulis menyampaikan terima kasih kepada:

1. Abduh Riski S.Si., M.Si., dan Ahmad Kamsyakawuni, S.Si., M.Kom., selaku Dosen Pembimbing yang telah memberikan bimbingan dan bantuan dalam penyempurnaan skripsi ini;
2. Kiswara Agung Santoso, S.Si., M.Kom., dan Kusbudiono, S.Si., M.Si., selaku Dosen Penguji yang telah memberikan kritik dan saran yang membangun dalam penyempurnaan skripsi ini;
3. Dian Anggraeni, S.Si., M.Si., selaku selaku Dosen Pembimbing Akademik yang telah membimbing dalam pemilihan matakuliah;
4. Ayahanda Khairi Mulyadi dan Ibunda Lutfiyah, serta kakak-kakakku Rivi Prima Setiawan dan Rivi Dwi Agus Maulida yang telah memberikan dukungan dan doa;
5. Sepupuku Ayu Sajida Da'ad Arini yang telah memberikan motivasi dan do'a;
6. Seluruh teman-teman "SIGMA" 2015 yang tidak dapat disebutkan satu per satu telah memberikan motivasi serta dukungannya selama ini;
7. Sahabat-sahabat "Hura Squad" (Tammy, Ulfi dan Aulia), "Istri Sholeha" (Eriska, Intan, Mitha, Melati, Rika dan Nadiya), Anggia Retno Pawestri dan Amalia Putri Nur Habibah yang telah memberikan motivasi, dukungan dan pengalamannya selama ini;
8. Seluruh anggota pengurus HIMATIKA "Geokompstat" masa bhakti 2017 yang telah memberikan motivasi dan pengalamannya selama satu tahun;

9. Seluruh kakak tingkat dan adik tingkat yang telah memberikan dukungan dan pengalamannya selama ini;
10. Semua pihak yang tidak dapat disebutkan satu per satu.

Penulis menerima segala kritik dan saran yang bersifat membangun dari semua pihak demi kesempurnaan penulisan skripsi ini. Akhirnya penulis berharap, semoga skripsi ini dapat bermanfaat.

Jember, Januari 2019

Penulis

DAFTAR ISI

DAFTAR GAMBAR

Halaman

DAFTAR TABEL

Halaman

DAFTAR LAMPIRAN

BAB 1. PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi membuat semakin mudahnya seseorang mengirimkan pesan kepada orang lain dalam waktu singkat tanpa mempermasalahkan jarak antar keduanya. Kemudahan menggunakan teknologi membuat semakin banyak pula seseorang yang ingin menyabotase pesan yang akan diterima orang lain, seperti menambah atau mengurangi isi pesan. Oleh karena itu, diperlukannya suatu teknik yang dinamakan kriptografi untuk mengamankan isi pesan. Kriptografi merupakan suatu ilmu untuk melindungi atau menyembunyikan pesan agar tidak diketahui oleh pihak ketiga dengan cara mengubah isi pesan asli menjadi kode – kode yang sulit dimengerti maknanya. Algoritma kriptografi dibedakan menjadi dua, yaitu algoritma kriptografi klasik dan algoritma kriptografi modern. Salah satu perbedaan algoritma kriptografi klasik dan algoritma kriptografi modern adalah algoritma kriptografi klasik menggunakan metode substitusi dan transposisi sedangkan algoritma kriptografi modern memanfaatkan operasi komputer digital dalam proses enkripsi dan dekripsi (Donnaro, 2018).

Playfair Cipher adalah algoritma yang diciptakan oleh ilmuwan Inggris bernama Sir Carles Wheatstone pada tahun 1854. Pada awalnya algoritma ini hanya mendukung 26 huruf kapital dimana I dan J diperlakukan sama dan apabila pesan memiliki panjang huruf yang ganjil maka huruf X atau Z ditambahkan pada akhir pesan. Pada algoritma ini dibutuhkan dua huruf yang berpasangan dalam mengenkripsi dan mendekripsi pesan, dua huruf berpasangan dinamakan digram. Karena *Playfair Cipher* ini hanya menggunakan pasangan dua huruf sehingga pasangan yang akan terbentuk pada saat enkripsi dapat mudah dibaca maknanya. Kelemahan lain dari algoritma ini ialah algoritma ini hanya terbatas pada 26 huruf saja yang membuat proses dekripsi pesan tidak berbeda jauh dengan proses enkripsi (Nurkifli, 2014).

Singh (2015) mengusulkan variasi baru *Playfair Cipher* yang disebut dengan *3D Playfair Cipher* yang mendukung 64 karakter (terdiri dari huruf, angka, dan

beberapa simbol umum), apabila pesan memiliki panjang huruf bukan kelipatan tiga maka huruf X dan/atau Z ditambahkan pada akhir pesan. Pada algoritma ini dibutuhkan tiga huruf yang berpasangan dalam mengenkripsi dan mendekripsi pesan, tiga huruf berpasangan dinamakan trigram. Berdasarkan analisis keamanan pesan menggunakan *3D Playfair Cipher* lebih aman dibandingkan dengan *Playfair Cipher*.

Bhat (2017) di dalam jurnal penelitiannya membahas tentang algoritma *4D Playfair Cipher* dan *Linear Feedback Shift Register* (LFSR) dalam pengamanan informasi pesan. Pada penelitiannya menjelaskan bahwa berdasarkan analisis keamanan yang dilakukan, algoritma *4D Playfair Cipher* dan *Linear Feedback Shift Register* (LFSR) lebih aman terhadap serangan dibandingkan dengan versi sebelumnya yaitu *Playfair Cipher*.

Linear Feedback Shift Register (LFSR) merupakan salah satu algoritma yang dapat digunakan untuk membangkitkan deretan bilangan biner secara acak dalam pembuatan kunci pada kriptografi. LFSR membangkitkan deretan bilangan dengan menggunakan operasi XOR dan XNOR. LFSR adalah *shift register* yang bit masukannya merupakan fungsi umpan-balik dari bentuk sebelumnya. Modifikasi LFSR yang akan digunakan adalah penambahan langkah pada hasil keluaran LFSR, dimana hasil keluaran dirotasi kemudian hasil rotasi akan digunakan sebagai matriks kunci pada *4D Playfair Cipher*. Berdasarkan penjelasan diatas, penulis tertarik untuk melakukan penyandian citra menggunakan algoritma *4D Playfair Cipher* dengan pembangkitan kunci Modifikasi *Linear Feedback Shift Register* (LFSR).

1.2 Rumusan Masalah

Berdasarkan latar belakang tersebut, terdapat beberapa masalah yang dapat dirumuskan sebagai berikut:

a. Bagaimana proses pembangkitan kunci dengan Modifikasi *Linear Feedback Shift Register* (LFSR) ?
b. Bagaimana proses enkripsi dan dekripsi penyandian citra menggunakan algoritma *4D Playfair Cipher* dan Modifikasi *Linear Feedback Shift Register*

(LFSR) dengan penyandian citra menggunakan algoritma *Playfair Cipher* dan dan *3D Playfair Cipher*?

c. Bagaimana perbandingan tingkat keamanan hasil penyandian citra menggunakan algoritma *4D Playfair Cipher* dan Modifikasi *Linear Feedback Shift Register* (LFSR) dengan penyandian citra menggunakan algoritma *Playfair Cipher* dan dan *3D Playfair Cipher*?

1.3 Batasan Masalah

Batasan masalah pada penelitian ini adalah hasil XNOR dari setiap bit pada setiap kunci tidak boleh sama dengan 255.

1.4 Tujuan Penelitian

Berdasarkan rumusan masalah yang telah diberikan, diperoleh tujuan dari penelitian sebagai berikut:

a. Mengetahui proses pembangkitan kunci dengan Modifikasi *Linear Feedback Shift Register* (LFSR).
b. Mengetahui proses enkripsi dan dekripsi penyandian citra menggunakan algoritma *4D Playfair Cipher* dan Modifikasi *Linear Feedback Shift Register* (LFSR) dengan penyandian citra menggunakan algoritma *Playfair Cipher* dan *3D Playfair Cipher*.
c. Membandingkan tingkat keamanan hasil penyandian citra menggunakan algoritma *4D Playfair Cipher* dan Modifikasi *Linear Feedback Shift Register* (LFSR) dengan penyandian citra menggunakan algoritma *Playfair Cipher* dan dan *3D Playfair Cipher*.

1.5 Manfaat Penelitian

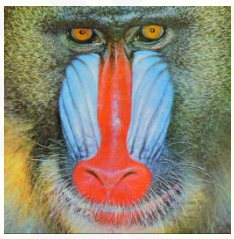
Penelitian yang akan dilakukan diharapkan dapat memberikan manfaat, antara lain :

a. Memberikan wawasan mengenai penyandian citra menggunakan algoritma *4D Playfair Cipher* dan Modifikasi *Linear Feedback Shift Register* (LFSR).
b. Memberikan motivasi penelitian lain untuk dapat melanjutkan tentang penyandian citra dengan menggunakan algoritma kriptografi yang berbeda.

BAB 2. TINJAUAN PUSTAKA

2.1 Citra

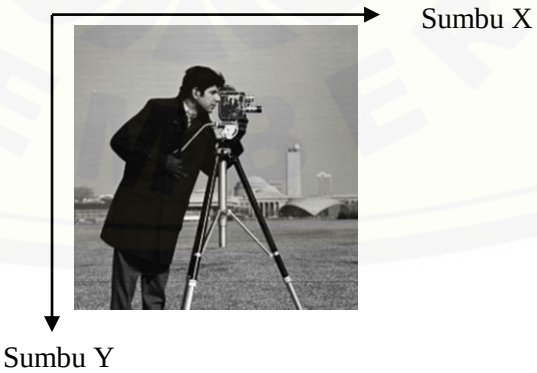
Citra merupakan gambar pada bidang dua dimensi. Citra terbagi menjadi dua antara lain, citra bersifat analog dan citra yang bersifat digital. Kualitas sebuah citra selalu dikaitkan dengan resolusi dalam intensitas warna. Resolusi citra menyatakan ukuran panjang kali lebar sebuah citra yang dinyatakan dalam satuan piksel. Semakin tinggi resolusi sebuah citra berarti semakin banyak jumlah piksel dan semakin tinggi kedalam intensitas berarti semakin banyak pula jumlah pikselnya. Contoh dari citra dapat dilihat pada Gambar 2.1 dan Gambar 2.2.



Gambar 2.1 Citra Baboon

Gambar 2.2 Citra Burung

Citra Berwarna dapat dilihat pada Gambar 2.1 dan dikenal sebagai citra spektral karena warna pada citra disusun atas tiga komponen warna yang disebut komponen RGB yaitu merah (*red*), hijau (*green*) dan biru (*blue*). Sedangkan Gambar 2.2 Citra Burung dinamakan Citra *Greyscale* karena warnanya ditentukan oleh suatu fungsi intensitas saja.



Gambar 2.3 Menentukan koordinat titik pada citra

Cara menentukan koordinat titik pada suatu citra dapat dilihat pada Gambar 2.3. Secara umum, fungsi intensitas cahaya dapat disimbolkan dengan $f(x, y)$

dimana (x, y) menyatakan koordinat pada bidang dan $f(x, y)$ menyatakan intensitas cahaya pada titik (x, y).

Citra akan dapat diolah menggunakan komputer, citra dapat dikatakan citra digital apabila suatu citra tersebut direpresentasikan secara numerik atau nilai-nilai diskrit melalui proses digitalisasi. Digitalisasi merupakan representasi citra dari fungsi kontinu menjadi fungsi diskrit. Secara umum, citra digital berbentuk persegi panjang dengan ukuran dimensi dapat dinyatakan sebagai tingggi × lebar atau lebar × panjang. Citra digital yang berukuran $N \times M$ dinyatakan dengan matriks berukusan 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 (x) dan indeks kolom (y) menyatakan koordinat suatu titik pada citra sedangkan $f(x, y)$ merupakan intensitas (derajat keabuan) ada titik (x, y). Masing-masing elemen pada citra digital (elemen pada matriks) disebut *pixel* (Munir, 2002).

2.2 Kode ASCII

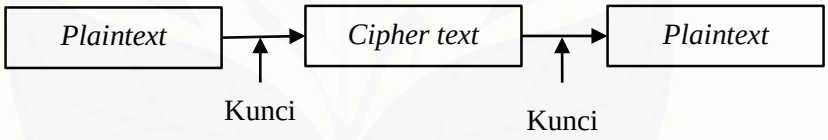
Kode ASCII (*American Standard Code for Information Interchange*) merupakan suatu standard internasional dalam kode huruf dan symbol seperti Hex dan Unicode tetapi ASCII lebih bersifat universal. Contohnya, 47 untuk karakter "/". Kode ASCII pada dasarnya memiliki komposisi bilangan biner sebanyak 8 bit. Dimulai dari 0000 0000 hingga 1111 1111. Total kombinasi yang dihasilkan pada Kode ASCII adalah 256, dimulai dari kode 0 hingga 255 dalam sistem bilangan Desimal.

Kode digital (biner) diterjemahkan oleh komputer secara per satuan *byte* (1 *byte* = 8 bit). Secara umum semua komputer saat ini mampu menerjemahkan kode-kode biner tersebut ke dalam suatu karakter tertentu. Kemampuan komputer ini dapat dimanfaatkan untuk mengubah setiap karakter yang dapat digunakan sebagai *cipherteks*. Hampir semua bahasa pemrograman sudah mampu menerjemahkan kode ASCII secara langsung. Misalnya program aplikasi Matlab

sudah mampu menerjemahkan 26 karakter alfabet secara langsung menjadi desimal dalam format hexadesimal (8 bit). Contohnya, nilai pada kode ASCII adalah 100 dalam desimal dan 01100100 dalam biner.

2.3 Kriptografi

Kriptografi berasal dari Bahasa Yunani yaitu *cryptos* yang berarti rahasia dan *graphein* yang berarti menulis, apabila kedua istilah digabungkan akan bermakna menulis rahasia. Kriptografi sendiri merupakan ilmu yang mempelajari teknik-teknik matematika yang berhubungan dengan aspek keamanan informasi, seperti kerahasiaan data, keabsahan data, integritas data, serta autentikasi data, namun tidak semua aspek keamanan informasi dapat diselesaikan dengan kriptografi. Terdapat beberapa istilah dalam kriptografi seperti enkripsi dan dekripsi. Enkripsi merupakan sebuah proses merubah pesan yang dapat dimengerti (*plaintext*) menjadi sebuah kode yang tidak dapat dimengerti (*ciphertext*) dan dekripsi merupakan sebuah proses kebalikan dari enkripsi. Proses enkripsi dan dekripsi dapat dilihat pada Gambar 2.4.



Gambar 2.4 Proses Enkripsi dan Dekripsi

2.3.1 Algoritma Kriptografi

Algoritma kriptografi merupakan langkah-langkah mengacak pesan dari orang-orang yang tidak berhak atas pesan tersebut. Algoritma kriptografi terdiri dari tiga fungsi dasar, yaitu :

a. Enkripsi

Enkripsi merupakan hal yang sangat diperlukan dalam keamanan data karena berfungsi untuk menjaga kerahasiaan suatu pesan melalui pengkodean. Pesan asli (*plaintext*) akan diubah menjadi pesan yang disandikan (*ciphertext*).

b. Dekripsi

Dekripsi kebalikan dari proses enkripsi, pesan yang disandikan (*ciphertext*) akan diubah kembali dalam bentuk pesan asli (*plaintext*).

c. Kunci

Kunci dibutuhkan untuk mengamankan pesan saat melakukan enkripsi dan dekripsi, dimana kunci dapat dibagi menjadi dua bagian yaitu kunci pribadi (*private key*) dan kunci umum (*public key*).

2.3.2 Jenis Kriptografi

Algoritma kriptografi dibedakan menjadi dua, yaitu algoritma kriptografi klasik dan algoritma kriptografi modern.

a. Algoritma Kriptografi Klasik

Algoritma kriptografi klasik digunakan sebelum era komputerisasi ada, sebagian besar manusia dulu menggunakan teknik kunci simetris. Terdapat dua metode yang digunakan pada algoritma kriptografi klasik yaitu metode substitusi dan metode transposisi. Metode substitusi adalah metode enkripsi dan dekripsi simetris yang mana dilakukan penggantian objek *plainteks* dengan objek lain dan menerapkan konsep korespodensi satu-satu untuk setiap objek *plainteks* yang disandikan. Sedangkan metode transposisi adalah metode enkripsi dan dekripsi yang mana dilakukan penggantian posisi objek-objek *plainteks* tanpa menggantikan objek *plainteks* tersebut, pembacaan matriks dilakukan dengan cara pembacaan kolom per kolom sesuai dengan kunci yang digunakan

b. Algoritma Kriptografi Modern

Algoritma Kriptografi Modern merupakan perkembangan dari algoritma kriptografi klasik. Algoritma ini memiliki tingkat kesulitan yang lebih sulit dibandingkan dengan algoritma kriptografi klasik dan kekuatan dari pengacakan pesan terdapat pada kuncinya. Algoritma kriptografi modern menggunakan pengolahan simbol biner karena berjalan mengikuti operasi komputer digital. Algoritma kriptografi modern menggunakan operasi logika seperti XOR, XNOR, AND dan OR dalam proses enkripsi dan dekripsinya.

(Donnaro, 2018).

2.4 Playfair Cipher

Playfair Cipher ditemukan oleh Sir Charles pada tahun 1854 dan digunakan pertama kali pada awal abad ke 20 untuk mengirim pesan rahasia antar markas yang ada di Inggris pada masa Perang Dunia I. *Playfair* menggunakan kunci dalam matriks 5×5 yang berisi 25 huruf alfabet dan mengganti huruf J menjadi huruf I yang ada didalam alfabet. *Playfair Cipher* merupakan salah satu kriptografi klasik yang penyandiannya menggunakan subtitusi. Pada algoritma ini dibutuhkan dua huruf yang berpasangan (digram) dalam mengenkripsi dan mendekripsi pesan. Kelemahan dari *Playfair Cipher* adalah terjadinya ambigu pada hasil dekripsi karena pada saat enkripsi, *playfair cipher* memiliki mekanisme mengganti huruf J dengan huruf I. Perlunya modifikasi *playfair cipher* yang dapat digunakan untuk memuat huruf kapital, huruf kecil, angka dan simbol. (Nurkifli, 2014).

Beberapa aturan dalam membuat sandi pada *playfair cipher*, yaitu:

a. Jika dua huruf *plaintext* berada pada satu baris kunci yang sama maka setiap huruf diganti dengan huruf yang berada disebelah kanannya.
b. Jika dua huruf *plaintext* berada pada satu kolom kunci yang sama maka setiap huruf diganti dengan huruf yang berada dibawahnya.
c. Jika huruf *plaintext* berada terbalik dengan tabel maka sandi yang dihasilkan akan dibaca terbalik, dengan kata lain yang semula dibacanya kiri ke kanan menjadi kanan ke kiri.
d. Jika dua huruf tidak pada baris yang sama atau kolom yang sama, maka huruf pertama diganti dengan huruf pada perpotongan baris huruf pertama dengan kolom huruf kedua. Huruf kedua diganti dengan huruf pada titik sudut keempat dari persegi panjang yang dibentuk dari tiga huruf yang digunakan.
e. Jika terdapat huruf yang ganda pada *plaintext* harus disisipkan huruf X atau Z dan apabila *plaintext* memiliki jumlah huruf yang ganjil maka harus disisipkan juga huruf X atau Z pada akhir *plaintext*.

(Nurkifli, 2014).

Contoh : Misal diketahui kunci “DUNIA” dan *plaintext* yaitu “MERDEKA NEGARAKU” maka dapat diselesaikan seperti pada tahap pengerjaan di bawah ini.

Tabel 2.1 Matriks 5 × 5 dengan kunci DUNIA

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

Langkah pertama, sisipkan huruf X diakhir *plaintext* kemudian ubah *plaintext* menjadi dua huruf berpasangan, maka akan menjadi seperti: ME RD EK AN EG AR AK UX.

Langkah selanjutnya dapat diselesaikan dengan menggunakan aturan pada *playfair cipher*, dapat dilihat pada langkah di bawah.

1. ME RD akan menjadi LF NP

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

2. EK AN akan menjadi CL ID

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

3. EG AR akan menjadi FB NT

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

4. AK UX akan menjadi UO NW

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

Sehingga didapatkan :
Plaintext : ME RD EK AN EG AR AK UX
Ciphertext : LF NP CL ID FB NT UO NW

2.5 3D Playfair Cipher

3D Playfair Cipher adalah pengembangan algoritma kriptografi klasik yang membutuhkan tiga huruf berpasangan (trigram) selama proses enkripsi dan dekripsi. Pada algoritma ini menggunakan formasi matriks kunci berukuran 4 × 4 × 4 yang berisi 26 huruf alfabet, 10 angka, dan beberapa simbol khusus. Pada algortima ini memiliki matriks kunci terdiri dari empat baris, empat kolom, dan empat tingkat. Kunci yang digunakan pada algoritma ini tidak boleh berulang (Singh dkk, 2015). Matriks kunci *3D Playfair Cipher* dapat dilihat pada Tabel 2.2.

Tabel 2.2 Matriks Kunci *3D Playfair Cipher*

TINGKAT 1			
0	1	2	3
4	5	6	7
8	9	A	B
C	D	E	F

TINGKAT 2			
G	H	I	J
K	L	M	N
O	P	Q	R
S	T	U	V

TINGKAT 3			
W	X	Y	Z
!	“	=	$
%	&	‘	(
)	*	+	,

TINGKAT 4			
-	.	/	:
;	<	=	>
?	@	[	\
]	^	_	|

2.5.1 Proses Enkripsi

Selama proses enkripsi, pesan akan dipecah menjadi trigram (pasangan yang tediri dari tiga huruf). Huruf tambahan X dan Z digunakan untuk memenuhi trigram, X atau Z ditambahkan jika tersisa satu tempat kosong pada pesan, X dan

Z ditambahkan jika terdapat dua tempat kosong. Contohnya, NEGARAKU akan dirubah menjadi {NEG}, {ARA} dan{KUX}. Penggantian huruf dalam trigram akan diganti oleh pesan yang sehubungan dengan posisi huruf dalam trigram di baris, kolom, tingkat dengan cara melingkar. Proses enkripsi menggunakan model *circular* dapat dilihat pada Tabel 2.3 (Singh dkk, 2015).

Tabel 2.3 Proses Enkripsi

Trigram *Plaintext*	Trigram *Plaintext*			Trigram *Ciphertext*
	Karakter -1	Karakter-2	Karakter-3	
Karakter-1	Baris	Kolom	Tingkat	Karakter-1
Karakter-2	Tingkat	Baris	Kolom	Karakter-2
Karakter-3	Kolom	Tingkat	Baris	Karakter-3

2.5.2 Proses Dekripsi

Penggantian huruf untuk tujuan dekripsi mengikuti mode *circular*, hanya urutannya berubah yaitu baris, tingkat, kolom huruf dalam trigram *plaintext* (Singh dkk, 2015). Proses Dekripsi dapat dilihat pada Tabel 2.4.

Tabel 2.4 Proses Dekripsi

Trigram *Ciphertext*	Trigram Ciphertext			Trigram *Plaintext*
	Karakter -1	Karakter-2	Karakter-3	
Karakter-1	Baris	Tingkat	Kolom	Karakter-1
Karakter-2	Kolom	Baris	Tingkat	Karakter-2
Karakter-3	Tingkat	Kolom	Baris	Karakter-3

Contoh : Misal diketahui kunci “DUNIA” dan *plaintext* yaitu “MERDEKA NEGARAKU” dapat diselesaikan seperti pada tahap pengerjaan di bawah ini.

Tabel 2.5 Matriks Kunci 4 × 4 × 4 dengan Kunci DUNIA

TINGKAT 1			
D	U	N	I
A	0	1	2
3	4	5	6
7	8	9	B

TINGKAT 2			
C	E	F	G
H	J	K	L
M	O	P	Q
R	S	T	V

TINGKAT 3			
W	X	Y	Z
!	“	=	$
%	&	‘	(
)	*	+	,

TINGKAT 4			
-	.	/	:
;	<	=	>
?	@	[	\
]	^	_	|

Langkah pertama, ubah *plaintext* menjadi tiga huruf berpasangan (trigram), sehingga akan menjadi seperti: MER DEK ANE GAR AKU.

Langkah selanjutnya dapat diselesaikan dengan menggunakan aturan pada 3D *Playfair Cipher*, dapat dilihat pada Tabel 2.6-Tabel 2.10.

Plaintext : MER DEK ANE GAR AKU

Tabel 2.6 Enkripsi dari MER

Trigram *Plaintext*	Trigram *Plaintext*			Trigram *Ciphertext*
	M	E	R	
M	Baris	Kolom	Tingkat	O
E	Tingkat	Baris	Kolom	C
R	Kolom	Tingkat	Baris	R

Tabel 2.7 Enkripsi dari DEK

Trigram *Plaintext*	Trigram *Plaintext*			Trigram *Ciphertext*
	D	E	K	
D	Baris	Kolom	Tingkat	E
E	Tingkat	Baris	Kolom	N
K	Kolom	Tingkat	Baris	H

Tabel 2.8 Enkripsi dari ANE

Trigram *Plaintext*	Trigram *Plaintext*			Trigram *Ciphertext*
	A	N	E	
A	Baris	Kolom	Tingkat	K
N	Tingkat	Baris	Kolom	U
E	Kolom	Tingkat	Baris	D

Tabel 2.9 Enkripsi dari GAR

Trigram *Plaintext*	Trigram *Plaintext*			Trigram *Ciphertext*
	G	A	R	
G	Baris	Kolom	Tingkat	C
A	Tingkat	Baris	Kolom	H
R	Kolom	Tingkat	Baris	B

Tabel 2.10 Enkripsi dari AKU

Trigram *Plaintext*	Trigram *Plaintext*			Trigram *Ciphertext*
	A	K	U	
A	Baris	Kolom	Tingkat	1
K	Tingkat	Baris	Kolom	0
U	Kolom	Tingkat	Baris	C

Sehingga didapatkan :
Plaintext : MER DEK ANE GAR AKU
Ciphertext : OCR ENH KUD CHB 1OC

2.6 *4D* Playfair Cipher

4D Playfair Cipher adalah pengembangan algoritma kriptografi klasik yang membutuhkan empat huruf berpasangan (*quartets*) selama proses enkripsi dan dekripsi. Pada algoritma ini menggunakan formasi matriks kunci berukuran 2 × 2 × 13 × 5 yang berisi 260 bilangan dari 0 sampai 259. Pada algoritma ini nilai 0 sampai 255 menampilkan bilangan yang dapat menyimpan seluruh nilai pada kode ASCII. Bilangan 256 sampai 258 digunakan atau disisipkan apabila jumlah huruf *plaintext* bukan kelipatan empat. Nilai 259 digunakan hanya selama proses substitusi. Matriks kunci *4D Playfair Cipher* terdiri dari dua *direction*, setiap *direction* memiliki dua *plane*, setiap *plane* memiliki 13 baris dan lima kolom. *4D Playfair Cipher* memiliki tiga langkah utama dalam proses enkripsi dan dekripsi. Tiga langkah utama, antara lain Formasi Matriks Kunci, Pesan sebelum proses, dan Pesan setelah proses (Bhat dkk, 2017).

2.6.1 Formasi Matriks Kunci

4D Playfair Cipher memiliki empat langkah utama dalam formasi matriks kunci, antara lain:

a. Buatlah kunci rahasia dengan barisan bilangan antara 0 sampai 259. Contohnya : (10 20 30 40 50 60 70 80).
b. Bilangan yang terdapat di dalam kunci rahasia tidak boleh berulang atau terdapat bilangan yang sama.

c. Masukkan kunci pada langkah sebelumnya kedalam matriks kunci berukuran 2 × 2 × 13 × 5 yang disediakan oleh *4D Playfair Cipher* dengan mengisi berurutan dimulai dari D1_P1, D1_P2, D2_P1 sampai D2_P2.

d. Sisipkan pada tempat yang tersisa dalam matriks dengan bilangan yang tidak terdapat didalam kunci dimulai dari bilangan 0 sampai 259 mengkuti aturan yang diberika pada langkah tiga (Bhat dkk, 2017).

Matriks kunci pada *4D Playfair Cipher* dapat dilihat pada Tabel 2.11.

Tabel 2.11 Matriks *4D Playfair Cipher*

D1_ P1				
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	26	27	28	29
30	31	32	33	34
35	36	37	38	39
40	41	42	43	44
45	46	47	48	49
50	51	52	53	54
55	56	57	58	59
60	61	62	63	64

D1_P2				
65	66	67	68	69
70	71	72	73	74
75	76	77	78	79
80	81	82	83	84
85	86	87	88	89
90	91	92	93	94
95	96	97	98	99
100	101	102	103	104
105	106	107	108	109
110	111	112	113	114
115	116	117	118	119
120	121	122	123	124
125	126	127	128	129

D2_P1				
130	131	132	133	134
135	136	137	138	139
140	141	142	143	144
145	146	147	148	149
150	151	152	153	154
155	156	157	158	159
160	161	162	163	164
165	166	167	168	169
170	171	172	173	174
175	176	177	178	179
180	181	182	183	184
185	186	187	188	189
190	191	192	193	194

D2_P2				
195	196	197	198	199
200	201	202	203	204
205	206	207	208	209
210	211	212	213	214
215	216	217	218	219
220	221	222	223	224
225	226	227	228	229
230	231	232	233	234
235	236	237	238	239
240	241	242	243	244
245	246	247	248	249
250	251	252	253	254
255	256	257	258	259

2.6.2 Pesan Sebelum Proses

Pesan sebelum proses berlangsung, *plaintext* akan dibagi kedalam empat bilangan berpasangan (*quartets*). Bilangan 256, 257, dan 258 adalah tiga huruf pengisi yang digunakan dalam beberapa aturan dasar. Terdapat beberapa aturan dasar dalam pembuatan empat bilangan berpasangan (*quartets*), antara lain:

a. Jika terdapat bilangan yang sama di dalam posisi bersebelahan, maka ditengah bilangan tersebut diisi bilangan 256. Contohnya, *plaintext* (10 20 30 30 40 50 60) dibagi kedalam *quartets* menjadi (10 20 30 256) dan (30 40 50 60).
b. Jika terdapat bilangan yang sama lebih dari dua di dalam posisi bersebelahan, maka bilangan 256 disisipkan diantara bilangan yang sama pertama dan 257 disisipkan diantara bilangan yang sama selanjutnya. Contohnya, (20 20 20 30 40 50) dibagai kedalam *quartets* menjadi (20 256 20 257) dan (20 30 40 50).
c. Jika jumlah huruf *plaintext* bukan kelipatan empat atau terdapat didalam huruf yang berpasangan jumlahnya kurang dari empat huruf maka akan disisipkan bilangan 256, 257, dan 258 diakhir *plaintext*. Contohnya, (10 20 30 40 50 60) dibagi kedalam *quartets* menjadi (10 20 30 40) dan (50 60 256 257) (Bhat dkk, 2017).

2.6.3 Proses Enkripsi

Tabel 2.12 Proses Enkripsi *4D Playfair Cipher*

Plaintext Quartet	*Plaintext Quartet*				*Ciphertext Quartet*
	Karakter-1	Karakter-2	Karakter-3	Karakter-4	
Karakter-1	R	C	D	P	Karakter-1
Karakter-2	P	R	C	D	Karakter-2
Karakter-3	D	P	R	C	Karakter-3
Karakter-4	C	D	P	R	Karakter-4

Proses enkripsi menggunakan metode *circular*, seperti pada Tabel 2.12. Pergantian huruf dalam *quartets* akan diganti oleh pesan sandi yang sehubungan dengan posisi yang terdapat dalam *quartets* pada baris (*row* / R), kolom (*coloumn*

/ C), arah (*direction* / D), dam kerangka (*plane* / P*)* dengan metode melingkar (Bhat dkk, 2017).

2.6.4 Proses Dekripsi

Penggantian huruf untuk tujuan dekripsi mengikuti mode *circular* seperti pada proses enkripsi, hanya urutannya berubah baris (*row* / R), kerangka (*plane* / P), arah (*direction* / D) dan kolom (*coloumn* / C) dengan metode melingkar (circular) (Bhat dkk, 2017). Proses Dekripsi dapat dilihat pada Tabel 2.13.

Tabel 2.13 Proses Enkripsi *4D Playfair Cipher*

Ciphertext Quartet	*Ciphertext Quartet*				*Plaintext Quartet*
	Karakter-1	Karakter-2	Karakter-3	Karakter-4	
Karakter-1	R	P	D	C	Karakter-1
Karakter-2	C	R	P	D	Karakter-2
Karakter-3	D	C	R	P	Karakter-3
Karakter-4	P	D	C	R	Karakter-4

Contoh : Misalkan kunci “16 9 199 6 10 7 21 17” dan *plaintext* yaitu “65 80 15 10 35 225 145 20” dapat diselesaikan seperti pada tahap pengerjaan di bawah ini.

Langkah pertama memasukkan kunci kedalam matriks kunci *4D Playfair Cipher* dapat dilihat pada Tabel 2.14.

Langkah kedua, ubah *plaintext* menjadi empat huruf berpasangan (*quartets*), maka akan menjadi seperti (65 80 15 10) dan (35 225 145 20).

Langkah selanjutnya dapat diselesaikan dengan menggunakan aturan pada *4D Playfair Cipher*, dapat dilihat pada Tabel 2.15 dan Tabel 2.16.

Tabel 2.14 Matriks dengan Kunci (16 9 199 6 10 7 21 17)

D1_ P1					D1_P2				
16	9	199	6	10	64	65	66	67	68
7	21	17	0	1	69	70	71	72	73
2	3	4	5	8	74	75	76	77	78
11	12	13	14	15	79	80	81	82	83
18	19	20	22	23	84	85	86	87	88
24	25	26	27	28	89	90	91	92	93
29	30	31	32	33	94	95	96	97	98
34	35	36	37	38	99	100	101	102	103
39	40	41	42	43	104	105	106	107	108
44	45	46	47	48	109	110	111	112	113
49	50	51	52	53	114	115	116	117	118
54	55	56	57	58	119	120	121	122	123
59	60	61	62	63	124	125	126	127	128
D2_P1					D2_P2				
129	130	131	132	133	194	195	196	197	198
134	135	136	137	138	200	201	202	203	204
139	140	141	142	143	205	206	207	208	209
144	145	146	147	148	210	211	212	213	214
149	150	151	152	153	215	216	217	218	219
154	155	156	157	158	220	221	222	223	224
159	160	161	162	163	225	226	227	228	229
164	165	166	167	168	230	231	232	233	234
169	170	171	172	173	235	236	237	238	239
174	175	176	177	178	240	241	242	243	244
179	180	181	182	183	245	246	247	248	249
184	185	186	187	188	250	251	252	253	254
189	190	191	192	193	255	256	257	258	259

Tabel 2.15 Enkripsi dari (65 80 15 10)

Plaintext Quartet	*Plaintext Quartet*				*Ciphertext Quartet*
	65	80	15	10	
65	R	C	D	P	9
80	P	R	C	D	15
15	D	P	R	C	15
10	C	D	P	R	9

Tabel 2.16 Enkripsi dari (35 225 145 20)

Plaintext Quartet	*Plaintext Quartet*				*Ciphertext Quartet*
	35	225	145	20	
35	R	C	D	P	164
225	P	R	C	D	30
145	D	P	R	C	81
20	C	D	P	R	150

Sehingga didapatkan :
Plaintext : 65 80 15 10 35 225 145 20
Ciphertext : 9 15 15 9 164 50 81 150

2.7 Linear Feedback Shift Register (LFSR)

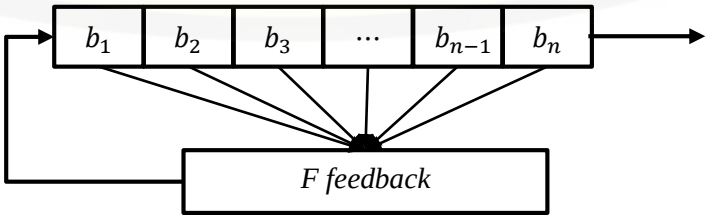
Linear Feedback Shift Register (LFSR) merupakan salah satu algoritma yang dapat digunakan untuk membangkitkan deretan bilangan biner secara acak dalam pembuatan kunci pada kriptografi. LFSR membangkitkan deretan bilangan dengan menggunakan operasi XOR dan XNOR. Pada proses XOR nilai awal bit yang ada pada register dengan panjang tertentu tergantung dari derajat polinomial yang digunakan. LFSR adalah *shift register* yang bit masukannya merupakan fungsi umpan-balik dari bentuk sebelumnya. Periode maksimum LFSR dengan n-bit memiliki rumus umum $2^n - 1$ (Pramudianto dan Rino, 2012).

Bentuk umum dari *Linear Feedback Shift Register* (LFSR) dapat didefinisikan, sebagai berikut:

$$r_{i+1}(b_0, b_1, b_2, \dots, b_{n-1}) = f(b_0, b_1, b_2, \dots, b_{n-1})$$

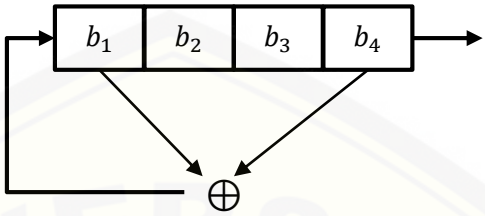
dimana f adalah fungsi *Linear Feedback Shift Register* (LFSR)

Pembangkitan fungsi umpan balik LFSR dapat dibangkitkan seperti pada ilustrasi Gambar 2.5.



Gambar 2.5 LFSR dengan n-bit

Contoh : Misal diketahui kunci sepanjang 4-bit yaitu 1100, kemudian kita akan membangkitkan kunci dengan algoritma LFSR dimana fungsi umpan-balik nya merupakan fungsi XOR bit ke-4 dan bit ke-1. Maka ilustrasi gambar dan pembangkitan bilangan akan terlihat pada gambar 2.6 dan tabel 2.17 di bawah ini.



Gambar 2.6 LFSR dengan 4-bit

Tabel 2.17 Perhitungan fungsi *feedback* LFSR dengan 4-bit

i	Isi Register				Output
0	1	1	0	0	
1	1	1	1	0	0
2	1	1	1	1	0
3	0	1	1	1	1
4	1	0	1	1	1
5	0	1	0	1	1
6	1	0	1	0	1
7	1	1	0	1	0
8	0	1	1	0	1
9	0	0	1	1	0
10	1	0	0	1	1
11	0	1	0	0	1
12	0	0	1	0	0
13	0	0	0	1	0
14	1	0	0	0	1
15	1	1	0	0	0

Pada tabel di atas, diperoleh output rangkaian bit dengan periode maksimum yaitu $2^4 - 1 = 16 - 1 = 15$ sebagai berikut :

1110 1111 0111 1011 0101 1010 1101 0110 0011 1001 0100 0010 0001 1000

dengan bit keluaran : 001111010110010.

Kemudian bit keluaran dikombinasikan 4-bit berurutan, seperti bit pada posisi 1 hingga 4, 2 hingga 5, 3 hingga 6 dan seterusnya, dari urutan bit keluaran di atas,

urutan bilangan acak yang dihasilkan adalah 3, 7, 15, 14, 13, 10, 5, 11, 6, 12, 10, 2.

Modifikasi *Linear Feedback Shift Register* (LFSR) merupakan pengembangan algoritma *Linear Feedback Shift Register* (LFSR) dimana pada modifikasinya urutan bilangan acak yang dihasilkan dari perhitungan *Linear Feedback Shift Register* (LFSR) selanjutnya dirotasi sejauh lima ke kiri. Sehingga dari contoh diatas urutan bilangan acak yang dihasilkan asalah 10, 5, 11, 6, 12, 10, 2, 3, 7, 15, 14, 13.

2.8 Logika XOR dan XNOR

XOR merupakan singkatan dari *Exclusive*-OR. Logika operasi XOR akan menghasilkan keluaran bernilai Benar jika dan hanya jika salah satu dari nilai input bernilai Benar. Jika kedua input bernilai Benar maka keluaran akan bernilai Salah (Putra, 2010). Pada operasi logika 0 merepresentasikan logika bernilai Salah dan 1 mempresentasikan logika bernilai Benar. Tabel 2.18 menunjukkan tabel kebenaran dari operasi XOR.

Tabel 2.18 Tabel Kebenaran Operasi XOR

A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

XNOR merupakan *invers* dari XOR. Jika pada XOR nilai Benar didapat jika dan hanya jika salah satu input saja yang bernilai Benar. Pada XNOR hasil Benar hanya didapat jika kedua nilai ini menunjukkan nilai yang sama (Benar-Benar atau Salah-Salah). Jika salah satu input saja yang bernilai Benar maka keluarannya akan bernilai Salah (Putra, 2010). Tabel 2.19 menunjukkan tabel kebenaran dari logika XNOR.

Tabel 2.19 Tabel Kebenaran Operasi XNOR

A	B	A XNOR B
0	0	1
0	1	0
1	0	0
1	1	1

2.9 Analisis Histogram

Analisis histogram merupakan salah satu analisis yang dapat digunakan untuk melihat apakah hasil dari proses enkripsi aman dan tahan terhadap serangan-serangan kriptanalisis. Di dalam bidang pengolahan citra histogram memperlihatkan distribusi nilai *pixel* di dalam sebuah citra. Histogram digunakan penyerang untuk melakukan kriptanalisis dengan memanfaatkan frekuensi kemunculan *pixel* di dalam histogram. Penyerang berharap nilai *pixel* yang sering muncul di dalam *plainimage* berkorelasi dengan nilai *pixel* yang sering muncul di dalam *cipherimage*. Histogram *cipherimage* dan *plainimage* seharusnya berbeda secara signifikan atau secara statistik tidak memiliki kemiripan. (Munir, 2012).

Analisis histogram yang digunakan adalah pengujian X^2 dari gambar yang telah terenkripsi. Persamaan X^2 dari gambar terenkripsi dari dimensi $m \times n$ sebagai berikut.

$$X^2 = \sum_{i=0}^{255} \frac{(v_i - v_0)^2}{v_0}$$

Dimana v_i adalah frekuensi yang diamati dari nilai piksel $i(0 \leq i \leq 255)$ dan v_0 merupakan frekuensi yang diharapkan dari nilai piksel i, jadi $v_0 = \frac{m \times n}{256}$. Sehingga semakin kecil hasil pengujian X^2 maka tingkat keseragaman dalam histogram semakin merata dan hasil dari enkripsi semakin aman dan berlaku sebaliknya (Boriga dkk, 2014).

2.10 Analisis Diferensial

Analisis Diferensial digunakan untuk menguji pengaruh perubahan setiap *pixel* pada citra yang terenkripsi. Terdapat dua indikator pengukuran umum yang digunakanpada analisis ini yaitu *Number of Pixel Change Rate* (NPCR) dan *Unified Avarage Changing Intensity* (UACI). Adapun perhitungan NPCR didefinisikan sebagai berikut:

$$NPCR = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{k=1}^{o} \frac{d_{i,j,k}}{T} \right) \times 100\%$$

dimana m, n dan o adalah lebar, tinggi, dan kanal. Kanal pada setiap jenis citra berbeda, citra GRB memiliki 3 kanal, citra *Grayscale* memiliki 1 kanal dan citra Hitam Putih memiliki 2 kanal. m, n dan o digunakan untuk menghitung T yang merupakan jumlah total *pixel* pada *cipherimage* sedangkan $d_{i,j,k}$ melambangkan derajat keabuan ditentukan sebagai berikut:

$$d_{i,j} = \begin{cases} 0, & jika\ c_{i,j,k}^{(1)} = c_{i,j,k}^{(2)} \\ 1, & jika\ c_{i,j,k}^{(1)} \neq c_{i,j,k}^{(2)} \end{cases}$$

dimana $c_{i,j,k}^{(1)}$ dan $c_{i,j,k}^{(2)}$ merupakan nilai derajat keabuan dari baris i, kolom j dan kanal k dari citra $c^{(1)}$ dan citra $c^{(2)}$. $c^{(1)}$ merupakan nilai *pixel* dari *plainimage* dan $c^{(2)}$ merupakan nilai *pixel* dari *cipherimage*

Sedangkan perhitungan UACI didefinisikan sebagai berikut :

$$UACI = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{k=1}^{o} \frac{\left| c_{i,j,k}^{(1)} - c_{i,j,k}^{(2)} \right|}{F \cdot T} \right) \times 100\%$$

dimana F menunjukkan nilai *pixel* terbesar yang kompatibel dengan format *chiperimage*. Secara teori, nilai minimum yang baik pada indikator NPCR adalah sebesar 99,6094% sedangkan untuk indikator UACI sebesar 33,4635% (Boriga dkk, 2014).

BAB 3. METODE PENELITIAN

3.1 Data Penelitian

Data yang digunakan pada penilitian ini adalah citra yang disebut sebagai *plainimage*. Berikut adalah citra-citra yang akan digunakan pada penelitian:

(a)

(b)

(c)

(d)

Gambar 3.1 (a) Lighthouse; (b) Citra Lighthouse Gray; (c) Citra Goldhill; (d) Citra Kucing

3.2 Langkah Penelitian

Adapun langkah–langkah pada penelitian kali ini adalah sebagai berikut:

a. Studi Literatur

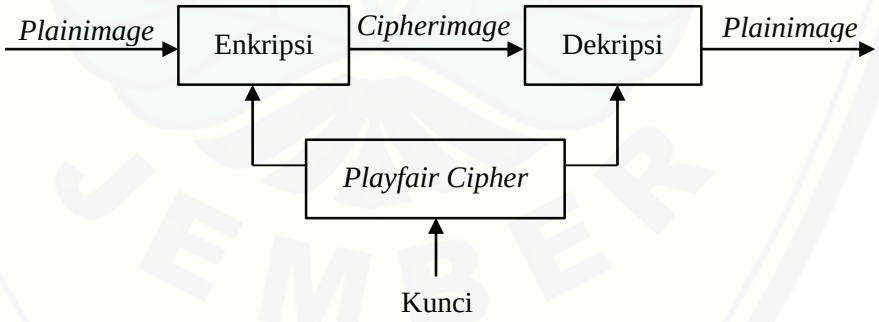
Studi literatur yang dilakukan pada penelitian ini diantaranya, dengan mempelajari referensi dari berbagai sumber, seperti jurnal penelitian, media internet, dan skripsi–skripsi terdahulu yang berkaitan tentang citra dan kriptografi terutama teori terkait dengan Algoritma *4D Playfair Cipher* dan Modifikasi *Linear Feedback Shift Register* (LFSR).

b. Percobaan Enkripsi dan Dekripsi Citra Menggunakan Algoritma *Playfair Cipher*

Pada tahap percobaan enkripsi dan dekripsi citra menggunakan algoritma *Playfair Cipher*, penulis melakukan penelitian dengan mencoba perhitungan secara manual dan menggunakan program. *Plainimage* dan *cipherimage* yang akan dienkripsi dan didekripsi adalah potongan *pixel* yang memiliki nilai–nilai derajat keabuan pada suatu citra digital. *Playfair Cipher* menggunakan satu buah kunci berukuran 8 karakter pada kode ASCII.

Langkah-langkah proses enkripsi dan dekripsi pada tahap ini adalah sebagai berikut:

1) Kunci dengan 8 karakter disubstitusikan kedalam ruang–ruang pada matriks kunci yang disediakan oleh algoritma *Playfair Cipher*.
2) *Plainimage* dienkripsi menggunakan algoritma *Playfair Cipher* dengan matriks kunci yang dihasilkan pada langkah pertama. Hasil keluaran pada tahap ini merupakan hasil akhir dari proses enkripsi yaitu berupa *cipherimage.*
3) *Cipherimage* didekripsi menggunakan algoritma *Playfair Cipher* dengan matriks kunci yang dihasilkan pada langkah pertama. Hasil keluaran pada tahap ini merupakan hasil akhir dari proses dekripsi yaitu berupa *plainimage.*



Gambar 3.2 Proses Enkripsi dan Dekripsi pada Algoritma *Playfair Cipher*

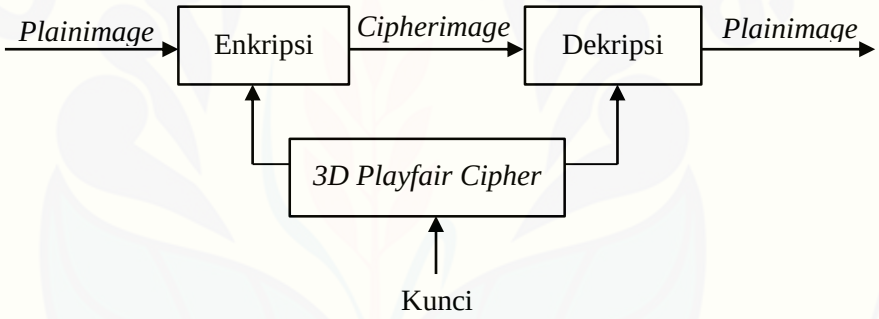
c. Percobaan Enkripsi dan Dekripsi Citra Menggunakan Algoritma *3D Playfair Cipher*

Pada tahap percobaan enkripsi dan dekripsi citra menggunakan algoritma *3D Playfair Cipher*, penulis melakukan penelitian dengan mencoba perhitungan secara manual dan menggunakan program. *Plainimage* dan *cipherimage* yang akan dienkripsi dan didekripsi adalah potongan *pixel* yang memiliki nilai–nilai

derajat keabuan pada suatu citra digital. *3D Playfair Cipher* menggunakan satu buah kunci berukuran 8 karakter pada kode ASCII.

Langkah-langkah proses enkripsi dan dekripsi pada tahap ini adalah sebagai berikut:

1) Kunci dengan 8 karakter disubstitusikan kedalam ruang–ruang pada matriks kunci yang disediakan oleh algoritma *Playfair Cipher*.
2) *Plainimage* dienkripsi menggunakan algoritma *3D Playfair Cipher* dengan matriks kunci yang dihasilkan pada langkah pertama. Hasil keluaran pada tahap ini merupakan hasil akhir dari proses enkripsi yaitu berupa *cipherimage*.
3) *Cipherimage* didekripsi menggunakan algoritma *3D Playfair Cipher* dengan matriks kunci yang dihasilkan pada langkah pertama. Hasil keluaran pada tahap ini merupakan hasil akhir dari proses dekripsi yaitu berupa *plainimage.*



Gambar 3.3 Proses Enkripsi dan Dekripsi pada Algoritma *3D Playfair Cipher*

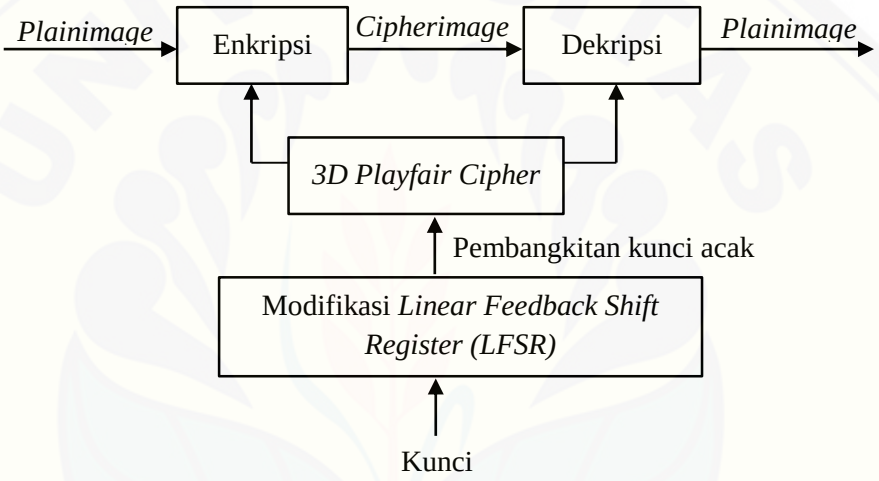
d. Percobaan Enkripsi dan Dekripsi Citra Menggunakan Algoritma *4D Playfair Cipher* dan Modifikasi *Linear Feedback Shift Register* (LFSR)

Plainimage dan *cipherimage* yang akan dienkripsi dan didekripsi adalah potongan *pixel* yang memiliki nilai–nilai derajat keabuan pada suatu citra digital. Penggabungan algoritma *4D Playfair Cipher* dan Modifikasi *Linear Feedback Shift Register* (LFSR) menggunakan satu buah kunci berukuran 8 karakter pada kode ASCII.

Langkah-langkah proses enkripsi dan dekripsi pada tahap ini adalah sebagai berikut:

1) Kunci dengan 8 karakter yang digunakan diolah terlebih dahulu menggunakan algoritma Modifikasi *Linear Feedback Shift Register* (LFSR).

2) Hasil pembangkitan acak kunci yang baru dari algoritma Modifikasi *Linear Feedback Shift Register* (LFSR) kemudian dimasukkan kedalam ruang–ruang pada matriks kunci yang disediakan oleh algoritma *4D Playfair Cipher*.
3) *Plainimage* dienkripsi menggunakan algoritma *4D Playfair Cipher* dengan matriks kunci yang dihasilkan pada langkah pertama. Hasil keluaran pada tahap ini merupakan hasil akhir dari proses enkripsi yaitu berupa *cipherimage*.
4) *Cipherimage* didekripsi menggunakan algoritma *4D Playfair Cipher* dengan matriks kunci yang dihasilkan pada langkah pertama. Hasil keluaran pada tahap ini merupakan hasil akhir dari proses dekripsi yaitu berupa *plainimage*.



Gambar 3.4 Proses Enkripsi dan Dekripsi pada Algoritma *4D Playfair Cipher* dan Modifikasi *Linear Feedback Shift Register* (LFSR)

e. Pembuatan Program Aplikasi Enkripsi dan Dekripsi Citra

Program yang digunakan untuk penelitian ini adalah Matlab R2015b dengan menuliskan *script* ke dalam program tersebut.

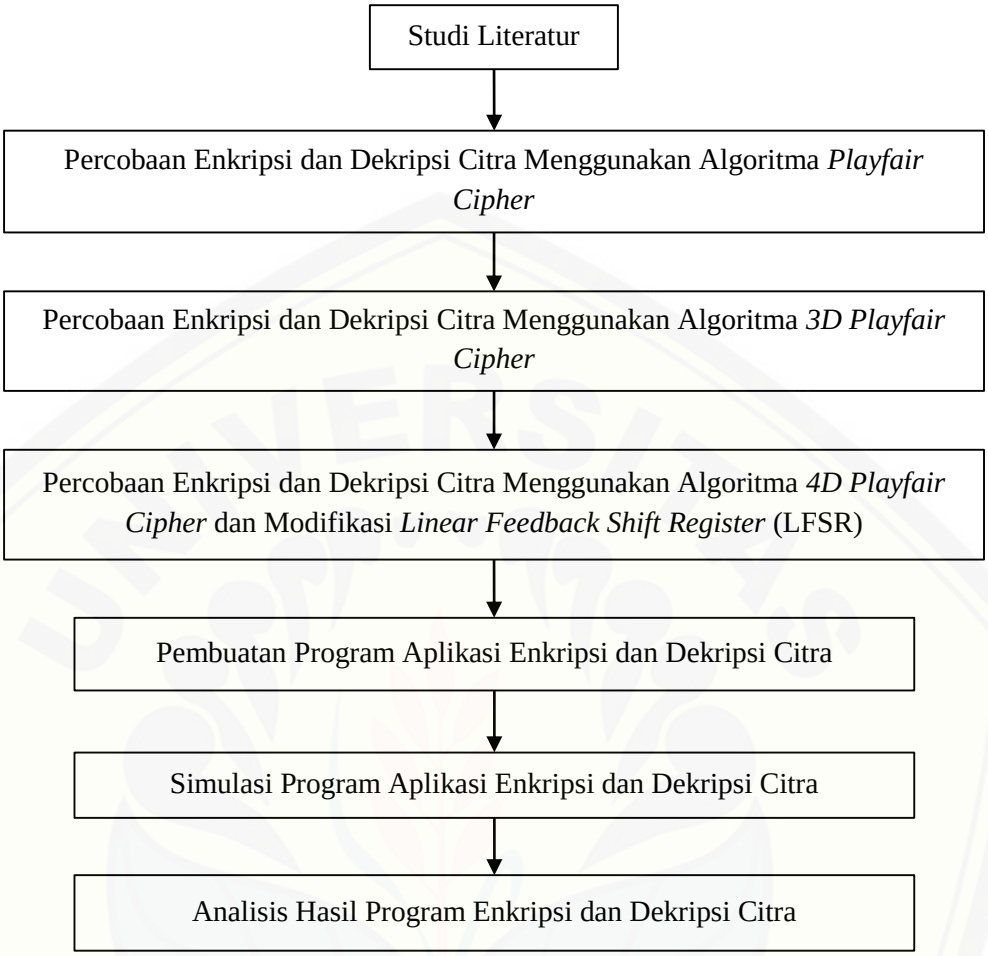
f. Simulasi Program Aplikasi Enkripsi dan Dekripsi Citra

Simulasi program dilakukan dengan menguji coba program yang telah dibuat pada aplikasi Matlab R2015b agar dapat dilakukan analisis hasil enkripsi dan dekripsi.

g. Analisis Hasil Program Enkripsi dan Dekripsi Citra

Pada tahap hasil akhir dari proses enkripsi citra pada program simulasi Matlab R2015b akan dianalisis menggunakan analisis histogram dan analisis diferensial.

Berikut ini adalah skema langkah–langkah pada penelitian:



Gambar 3.5 Skema Langkah–Langkah Penelitian

BAB 5. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan, didapat beberapa kesimpulan sebagai berikut:

a. Pembangkitan kunci menggunakan Modifikasi *Linear Feedback Shift Register* (LFSR) dapat menghasilkan deretan bilangan secara acak tanpa terdapat perulangan kunci pada deretan bilangan.
b. Proses enkripsi menggunakan algoritma *4D Playfair Cipher* dan Modifikasi *Linear Feedback Shift Register* (LFSR) menghasilkan *cipherimage* berbeda dari citra asli secara visual dan proses dekripsi dapat mengembalikan *cipherimage* kedalam citra aslinya. Proses enkripsi menggunakan algoritma *Playfair Cipher* maupun *3D Playfair Cipher* menghasilkan *cipherimage* berbeda dengan citra asli secara visual dan proses dekripsi dapat mengembalikan *cipherimage* kedalam citra aslinya.
c. Berdasarkan perbandingan antara hasil perhitungan dari histogram, NPCR, UACI, dan *cipherimage* yang dihasilkan. Tingkat keamanan hasil penyandian citra menggunakan *4D Playfair Cipher* dengan pembangkitan kunci Algoritma Modifikasi LFSR lebih kuat dibandingkan dengan hasil penyandian citra menggunakan *Playfair Cipher* dan *3D Playfair Cipher*, dapat dilihat pada hasil histogram yang seragam, hasil perhitungan X^2 lebih kecil, serta *cipherimage* yang dihasilkan acak dan merata daripada menggunakan *Playfair Cipher* dan *3D Playfair Cipher*.

5.2 Saran

Adapun saran yang perlu diperhatikan untuk penelitian lebih lanjut adalah Peneliti selanjutnya dapat menerapkan algoritma *Non-Linear Feedback Shift Register* (NLFSR) ataupun algoritma kriptografi modern lainnyauntuk dikombinasikan dengan *4D Playfair Cipher* atau algoritma kriptografi klasik lainnya.

DAFTAR PUSTAKA

Bhat, K., D. Mahto., dan D. K. Yadav. 2017. A Novel Approach to Information Four Dimensional (4D) Playfair Cipher Fused With Linear Feedback Shift Register. *Indian Journal of Computer Science and Engineering (IJCSE).* 8 (1):15-32.

Boriga, R. E., A.C. Dascalescu, dan A.V. Diaconu. 2014. A New Fast Image Encryption Scheme Based on 2D Chaotic Maps. *IAENG International Journal of Computer (IJSC).* 41(4):1-10.

Donnaro, D. P. 2018. Implementasi Teknik Playfair Cipher untuk Penyembunyian Teks Terenkripsi pada Citra dengan Metode End of File. *Skripsi*. Jember: Universitas Jember.

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

Munir, R. 2012. Analisis Keamanan Algoritma Enkripsi Citra Digital Menggunakan Kombinasi Dua Chaos Map dan Penerapan Teknik Selektif. *Juti*. 10(2):89–95.

Nurkifli, E. H. 2014. Modifikasi Algoritma Playfair dan Menggabungkan dengan Linear Feedback Shift Register (LFSR). *SENTIKA*. ISSN: 2089-9813.366-271.

Pramudianto, A. D. Dan Rino. 2012. Penggunaan Polinomial untuk Stream Key Generator pada Algorita Stream Ciphers Berbasis Feedback Shift Register. *Seminar Nasional Matematika dan Pendidikan Matematika*. ISBN : 978-979-16353-8-7. Yogyakarta: Universitas Negeri Yogyakarta.

Putra, D. 2010. Pengolahan Citra Digital. Yogyakarta:Andi Offset.

Singh, S., R. Jain., P. Deep. dan S Agarwal. 2015. Developing Mobile Messege Security Application Using 3D Playfair Cipher Algorithm. *International Coference on Advances in Computer Engineering and Applications (ICACEA).*838 – 841.

LAMPIRAN

LAMPIRAN A. Skrip Program Enkripsi dan Dekripsi pada MATLAB R2015b

a. Skrip program pembangkitan Kunci

```
function Key=kunci(knci)
n=length(knci);
c=digit2biner((knci(1)));
for i=2:n
    b=digit2biner((knci(i)));
    c=sor(c,b);
end
p=0;t=0;
for i=1:255
    x8=c(8);
    for j=6:-1:4
        s=~xor(x8,c(j));
        x8=s;
    end
    s=~xor(s,1);
    key(i)=c(8);
    c=[s c(1:7)];
end
for i=1:255
    if i<=248
        Key(i)=biner2digit(key(i:i+7));
    else
        p=p+1;
        Key(i)=biner2digit([key(i:255) key(1:p)]);
    end
end
Key=[Key 255:259];
Key=[Key(64:260) Key(1:63)];
```

b. Skrip program pada proses enkripsi *4D Playfair Cipher*

```
clc
knci=double(get(handles.edit2,'string'));
[m n o]=size(handles.data);
if o>1
    Cip(:,:,1)=(enkrip(knci,handles.data(:,:,1)));
    Cip(:,:,2)=(enkrip(knci,handles.data(:,:,2)));
    Cip(:,:,3)=(enkrip(knci,handles.data(:,:,3)));
    C=uint16(Cip);
    an_h1=an_hist(C(:,:,1));
    an_h2=an_hist(C(:,:,2));
    an_h3=an_hist(C(:,:,3));
    ANH=(an_h1+an_h2+an_h3)/3;
    uac=uaci(handles.data,Cip);
    npc=npcr(handles.data,Cip);
    set(handles.text6,'string',sprintf('Analisis Histogram :
%.2f',ANH));
```

```
    set(handles.text7,'string',[sprintf('NPCR : %.2f',npc) '
%']);
    set(handles.text8,'string',[sprintf('UACI : %.2f',uac) '
%']);
    dlmwrite('simpan.txt', C(:,:,1), 'delimiter',' ')
    dlmwrite('simpan1.txt', C(:,:,2), 'delimiter',' ')
    dlmwrite('simpan2.txt', C(:,:,3), 'delimiter',' ')
else
    Cip(:,:,1)=(enkrip(knci,handles.data(:,:,1)));
    C=uint16(Cip);
    ANH=an_hist(C(:,:,1));
    uac=uaci(handles.data,Cip);
    npc=npcr(handles.data,Cip);
    set(handles.text6,'string',sprintf('Analisis Histogram :
%6.2f',ANH));
    set(handles.text7,'string',[sprintf('NPCR : %6.2f',npc) '
%']);
    set(handles.text8,'string',[sprintf('UACI : %6.2f',uac) '
%']);
    dlmwrite('simpan.txt', C(:,:,1), 'delimiter',' ')
end
    Cip=uint8(Cip);
    guidata(hObject,handles);
    axes(handles.axes3);
    imshow(Cip);
    guidata(hObject,handles);
    axes(handles.axes4);
    imhist(Cip(:,:,1));
    set(handles.figure1,'userdata',Cip);
```

c. Skrip program pada proses enkripsi *3D Playfair Cipher*

```
if o>1
    %3dplyfair
    Cip3d(:,:,1)=(en_plyfair3d(knci,handles.data(:,:,1)));
    Cip3d(:,:,2)=(en_plyfair3d(knci,handles.data(:,:,2)));
    Cip3d(:,:,3)=(en_plyfair3d(knci,handles.data(:,:,3)));
    C3d=uint8(Cip3d);
    an_h1=an_hist(Cip3d(:,:,1));
    an_h2=an_hist(Cip3d(:,:,2));
    an_h3=an_hist(Cip3d(:,:,3));
    ANH=(an_h1+an_h2+an_h3)/3;
    uac=uaci(handles.data,C3d);
    npc=npcr(handles.data,C3d);
else
    Cip3d(:,:,1)=(en_plyfair3d(knci,handles.data(:,:,1)));
    C3d=uint8(Cip3d);
    ANH=an_hist(Cip3d(:,:,1));
    uac=uaci(handles.data,C3d);
    npc=npcr(handles.data,C3d);
end
    set(handles.text16,'string',sprintf('%6.2f',ANH));
    set(handles.text10,'string',[sprintf('NPCR : %6.2f',npc) '
%']);
    set(handles.text11,'string',[sprintf('UACI : %6.2f',uac) '
%']);
```

```
    Cip3d=uint8(Cip3d);
    guidata(hObject,handles);
    axes(handles.axes6);
    imshow(Cip3d);
    guidata(hObject,handles);
    axes(handles.axes5);
    imhist(Cip3d(:,:,1));
    set(handles.uipanel1,'userdata',Cip3d);
```

d. Skrip program pada proses enkripsi *Playfair Cipher*

```
if o>1
    %2dplyfair
    Cip2d(:,:,1)=(en_2d(knci,handles.data(:,:,1)));
    Cip2d(:,:,2)=(en_2d(knci,handles.data(:,:,2)));
    Cip2d(:,:,3)=(en_2d(knci,handles.data(:,:,3)));
    C2d=uint8(Cip);
    an_h1=an_hist(C2d(:,:,1));
    an_h2=an_hist(C2d(:,:,2));
    an_h3=an_hist(C2d(:,:,3));
    ANH=(an_h1+an_h2+an_h3)/3;
    uac=uaci(handles.data,C2d);
    npc=npcr(handles.data,C2d);
else
    Cip2d(:,:,1)=(en_2d(knci,handles.data(:,:,1)));
    C2d=uint8(Cip);
    ANH=an_hist(C2d(:,:,1));
    uac=uaci(handles.data,C2d);
    npc=npcr(handles.data,C2d);
end
    set(handles.text12,'string',sprintf('Analisis Histogram :
%6.2f',ANH));
    set(handles.text13,'string',[sprintf('NPCR : %6.2f',npc) '
%']);
    set(handles.text14,'string',[sprintf('UACI : %6.2f',uac) '
%']);
    Cip2d=uint8(Cip2d);
    guidata(hObject,handles);
    axes(handles.axes8);
    imshow(Cip2d);
    guidata(hObject,handles);
    axes(handles.axes7);
    imhist(Cip2d(:,:,1));
    set(handles.uipanel2,'userdata',Cip2d);
```

e. Skrip program pada proses dekripsi *4D Playfair Cipher*

```
if o>1
        A(:,:,1)=dlmread('simpan.txt');
        A(:,:,2)=dlmread('simpan1.txt');
        A(:,:,3)=dlmread('simpan2.txt');
        Cip(:,:,1)=(dekrip(knci,A(:,:,1)));
        Cip(:,:,2)=(dekrip(knci,A(:,:,2)));
        Cip(:,:,3)=(dekrip(knci,A(:,:,3)));
```

```
        Cip=uint8(Cip);
        an_h1=an_hist(Cip(:,:,1));
        an_h2=an_hist(Cip(:,:,2));
        an_h3=an_hist(Cip(:,:,3));
        ANH=(an_h1+an_h2+an_h3)/3;
        uac=uaci(handles.data,Cip);
        npc=npcr(handles.data,Cip);
    else
        A(:,:,1)=dlmread('simpan.txt');
        Cip(:,:,1)=(dekrip(knci,A(:,:,1)));
        Cip=uint8(Cip);
        ANH=an_hist(Cip(:,:,1));
        uac=uaci(handles.data,Cip);
        npc=npcr(handles.data,Cip);
    end
    set(handles.text2,'string',sprintf('Analisis Histogram :
%.2f',ANH));
    set(handles.text3,'string',[sprintf('NPCR : %.2f',npc) '
%']);
    set(handles.text4,'string',[sprintf('UACI : %.2f',uac) '
%']);
    Cip=uint8(Cip);
    guidata(hObject,handles);
    axes(handles.axes3);
    imshow(Cip);
    guidata(hObject,handles);
    axes(handles.axes4);
    imhist(Cip(:,:,1));
    set(handles.figure1,'userdata',Cip);
```

f. Skrip program pada proses dekripsi *3D Playfair Cipher*

```
%3dplyfair
if o>1
    Cip3d(:,:,1)=(de_plyfair3d(knci,handles.data(:,:,1)));
    Cip3d(:,:,2)=(de_plyfair3d(knci,handles.data(:,:,2)));
    Cip3d(:,:,3)=(de_plyfair3d(knci,handles.data(:,:,3)));
    C3d=uint8(Cip3d);
    an_h1=an_hist(C3d(:,:,1));
    an_h2=an_hist(C3d(:,:,2));
    an_h3=an_hist(C3d(:,:,3));
    ANH=(an_h1+an_h2+an_h3)/3;
    uac=uaci(handles.data,C3d);
    npc=npcr(handles.data,C3d);
else
    Cip3d(:,:,1)=(de_plyfair3d(knci,handles.data(:,:,1)));
    C3d=uint8(Cip3d);
    ANH=an_hist(C3d(:,:,1));
    uac=uaci(handles.data,C3d);
    npc=npcr(handles.data,C3d);
end
set(handles.text2,'string',sprintf('Analisis Histogram :
 %.2f',ANH));
set(handles.text3,'string',[sprintf('NPCR : %.2f',npc) ' %']);
set(handles.text4,'string',[sprintf('UACI : %.2f',uac) ' %']);
Cip3d=uint8(Cip3d);
```

```
guidata(hObject,handles);
axes(handles.axes3);
imshow(Cip3d);
guidata(hObject,handles);
axes(handles.axes4);
imhist(Cip3d(:,:,1));
set(handles.figure1,'userdata',Cip3d);
```

g. Skrip program pada proses dekripsi *Playfair Cipher*

```
%2dplyfair
 if o>1
     Cip2d(:,:,1)=(de_2d(knci,handles.data(:,:,1)));
     Cip2d(:,:,2)=(de_2d(knci,handles.data(:,:,2)));
     Cip2d(:,:,3)=(de_2d(knci,handles.data(:,:,3)));
     C2d=uint8(Cip2d);
     an_h1=an_hist(C2d(:,:,1));
     an_h2=an_hist(C2d(:,:,2));
     an_h3=an_hist(C2d(:,:,3));
     ANH=(an_h1+an_h2+an_h3)/3;
     uac=uaci(handles.data,C2d);
     npc=npcr(handles.data,C2d);
 else
     Cip2d(:,:,1)=(de_2d(knci,handles.data(:,:,1)));
     C2d=uint8(Cip2d);
     ANH=an_hist(C2d(:,:,1));
     uac=uaci(handles.data,C2d);
     npc=npcr(handles.data,C2d);
 end
 set(handles.text2,'string',sprintf('Analisis Histogram :
%.2f',ANH));
 set(handles.text3,'string',[sprintf('NPCR : %.2f',npc) ' %']);
 set(handles.text4,'string',[sprintf('UACI : %.2f',uac) ' %']);
 Cip2d=uint8(Cip2d);
 guidata(hObject,handles);
 axes(handles.axes3);
 imshow(Cip2d);
 guidata(hObject,handles);
 axes(handles.axes4);
 imhist(Cip2d(:,:,1));
 set(handles.figure1,'userdata',Cip2d);
```

LAMPIRAN B. Kode ASCII

DEC	OCT	HEX	BIN	Symbol
0	000	00	00000000	NUL
1	001	01	00000001	SOH
2	002	02	00000010	STX
3	003	03	00000011	ETX
4	004	04	00000100	EOT
5	005	05	00000101	ENQ
6	006	06	00000110	ACK
7	007	07	00000111	BEL
8	010	08	00001000	BS
9	011	09	00001001	HT
10	012	0A	00001010	LF
11	013	0B	00001011	VT
12	014	0C	00001100	FF
13	015	0D	00001101	CR
14	016	0E	00001110	SO
15	017	0F	00001111	SI
16	020	10	00010000	DLE
17	021	11	00010001	DC1
18	022	12	00010010	DC2
19	023	13	00010011	DC3
20	024	14	00010100	DC4
21	025	15	00010101	NAK
22	026	16	00010110	SYN
23	027	17	00010111	ETB
24	030	18	00011000	CAN
25	031	19	00011001	EM
26	032	1A	00011010	SUB
27	033	1B	00011011	ESC
28	034	1C	00011100	FS
29	035	1D	00011101	GS
30	036	1E	00011110	RS
31	037	1F	00011111	US
32	040	20	00100000	
33	041	21	00100001	!
34	042	22	00100010	Ŗ
35	043	23	00100011	#
36	044	24	00100100	$
37	045	25	00100101	%
38	046	26	00100110	&
39	047	27	00100111	Ř
40	050	28	00101000	(
41	051	29	00101001	)
42	052	2A	00101010	*

DEC	OCT	HEX	BIN	Symbol
43	053	2B	00101011	+
44	054	2C	00101100	,
45	055	2D	00101101	Ŕ
46	056	2E	00101110	.
47	057	2F	00101111	/
48	060	30	00110000	0
49	061	31	00110001	1
50	062	32	00110010	2
51	063	33	00110011	3
52	064	34	00110100	4
53	065	35	00110101	5
54	066	36	00110110	6
55	067	37	00110111	7
56	070	38	00111000	8
57	071	39	00111001	9
58	072	3A	00111010	:
59	073	3B	00111011	;
60	074	3C	00111100	<
61	075	3D	00111101	=
62	076	3E	00111110	>
63	077	3F	00111111	?
64	100	40	01000000	@
65	101	41	01000001	A
66	102	42	01000010	B
67	103	43	01000011	C
68	104	44	01000100	D
69	105	45	01000101	E
70	106	46	01000110	F
71	107	47	01000111	G
72	110	48	01001000	H
73	111	49	01001001	I
74	112	4A	01001010	J
75	113	4B	01001011	K
76	114	4C	01001100	L
77	115	4D	01001101	M
78	116	4E	01001110	N
79	117	4F	01001111	O
80	120	50	01010000	P
81	121	51	01010001	Q
82	122	52	01010010	R
83	123	53	01010011	S
84	124	54	01010100	T
85	125	55	01010101	U

DEC	OCT	HEX	BIN	Symbol
86	126	56	01010110	V
87	127	57	01010111	W
88	130	58	01011000	X
89	131	59	01011001	Y
90	132	5A	01011010	Z
91	133	5B	01011011	[
92	134	5C	01011100	\
93	135	5D	01011101	]
94	136	5E	01011110	^
95	137	5F	01011111	_
96	140	60	01100000	`
97	141	61	01100001	a
98	142	62	01100010	b
99	143	63	01100011	c
100	144	64	01100100	d
101	145	65	01100101	e
102	146	66	01100110	f
103	147	67	01100111	g
104	150	68	01101000	h
105	151	69	01101001	i
106	152	6A	01101010	j
107	153	6B	01101011	k
108	154	6C	01101100	l
109	155	6D	01101101	m
110	156	6E	01101110	n
111	157	6F	01101111	o
112	160	70	01110000	p
113	161	71	01110001	q
114	162	72	01110010	r
115	163	73	01110011	s
116	164	74	01110100	t
117	165	75	01110101	u
118	166	76	01110110	v
119	167	77	01110111	w
120	170	78	01111000	x
121	171	79	01111001	y
122	172	7A	01111010	z
123	173	7B	01111011	{
124	174	7C	01111100	|
125	175	7D	01111101	}
126	176	7E	01111110	~
127	177	7F	01111111	
128	200	80	10000000	€
129	201	81	10000001	
130	202	82	10000010	‚

DEC	OCT	HEX	BIN	Symbol
131	203	83	10000011	Ƒ
132	204	84	10000100	„
133	205	85	10000101	…
134	206	86	10000110	†
135	207	87	10000111	‡
136	210	88	10001000	ˆ
137	211	89	10001001	‰
138	212	8A	10001010	Š
139	213	8B	10001011	‹
140	214	8C	10001100	Œ
141	215	8D	10001101	
142	216	8E	10001110	Ţ
143	217	8F	10001111	
144	220	90	10010000	
145	221	91	10010001	Ř
146	222	92	10010010	Ř
147	223	93	10010011	Ŗ
148	224	94	10010100	Ŗ
149	225	95	10010101	•
150	226	96	10010110	Ŕ
151	227	97	10010111	Ŕ
152	230	98	10011000	˜
153	231	99	10011001	™
154	232	9A	10011010	Š
155	233	9B	10011011	›
156	234	9C	10011100	œ
157	235	9D	10011101	
158	236	9E	10011110	ţ
159	237	9F	10011111	Ÿ
160	240	A0	10100000	
161	241	A1	10100001	¡
162	242	A2	10100010	¢
163	243	A3	10100011	£
164	244	A4	10100100	¤
165	245	A5	10100101	¥
166	246	A6	10100110	¦
167	247	A7	10100111	§
168	250	A8	10101000	¨
169	251	A9	10101001	©
170	252	AA	10101010	ª
171	253	AB	10101011	«
172	254	AC	10101100	¬
173	255	AD	10101101	
174	256	AE	10101110	®
175	257	AF	10101111	¯

DEC	OCT	HEX	BIN	Symbol
176	260	B0	10110000	°
177	261	B1	10110001	±
178	262	B2	10110010	²
179	263	B3	10110011	³
180	264	B4	10110100	´
181	265	B5	10110101	µ
182	266	B6	10110110	¶
183	267	B7	10110111	·
184	270	B8	10111000	¸
185	271	B9	10111001	¹
186	272	BA	10111010	º
187	273	BB	10111011	»
188	274	BC	10111100	¼
189	275	BD	10111101	½
190	276	BE	10111110	¾
191	277	BF	10111111	¿
192	300	C0	11000000	À
193	301	C1	11000001	Á
194	302	C2	11000010	Â
195	303	C3	11000011	Ã
196	304	C4	11000100	Ä
197	305	C5	11000101	Å
198	306	C6	11000110	Æ
199	307	C7	11000111	Ç
200	310	C8	11001000	È
201	311	C9	11001001	É
202	312	CA	11001010	Ê
203	313	CB	11001011	Ë
204	314	CC	11001100	Ì
205	315	CD	11001101	Í
206	316	CE	11001110	Î
207	317	CF	11001111	Ï
208	320	D0	11010000	Đ
209	321	D1	11010001	Ñ
210	322	D2	11010010	Ò
211	323	D3	11010011	Ó
212	324	D4	11010100	Ô
213	325	D5	11010101	Õ
214	326	D6	11010110	Ö
215	327	D7	11010111	×
216	330	D8	11011000	Ø

DEC	OCT	HEX	BIN	Symbol
217	331	D9	11011001	Ù
218	332	DA	11011010	Ú
219	333	DB	11011011	Û
220	334	DC	11011100	Ü
221	335	DD	11011101	Ý
222	336	DE	11011110	Þ
223	337	DF	11011111	ß
224	340	E0	11100000	à
225	341	E1	11100001	á
226	342	E2	11100010	â
227	343	E3	11100011	ã
228	344	E4	11100100	ä
229	345	E5	11100101	å
230	346	E6	11100110	æ
231	347	E7	11100111	ç
232	350	E8	11101000	è
233	351	E9	11101001	é
234	352	EA	11101010	ê
235	353	EB	11101011	ë
236	354	EC	11101100	ì
237	355	ED	11101101	í
238	356	EE	11101110	î
239	357	EF	11101111	ï
240	360	F0	11110000	ð
241	361	F1	11110001	ñ
242	362	F2	11110010	ò
243	363	F3	11110011	ó
244	364	F4	11110100	ô
245	365	F5	11110101	õ
246	366	F6	11110110	ö
247	367	F7	11110111	÷
248	370	F8	11111000	ø
249	371	F9	11111001	ù
250	372	FA	11111010	ú
251	373	FB	11111011	û
252	374	FC	11111100	ü
253	375	FD	11111101	ý
254	376	FE	11111110	þ
255	377	FF	11111111	ÿ

LAMPIRAN C. Tabel Polinomial Primitif

Bits (n)	**Polinomial Primitif**	**Periode ($2^n - 1$)**
2	$x^2 + x + 1$	3
3	$x^3 + x^2 + 1$	7
4	$x^4 + x^3 + 1$	15
5	$x^5 + x^4 + 1$	31
6	$x^6 + x^5 + 1$	63
7	$x^7 + x^6 + 1$	127
8	$x^8 + x^6 + x^5 + x^4 + 1$	255
9	$x^9 + x^5 + 1$	511
10	$x^{10} + x^7 + 1$	1023
11	$x^{11} + x^9 + 1$	2047
12	$x^{12} + x^{11} + x^{10} + x^4 + 1$	4095
13	$x^{13} + x^{12} + x^{11} + x^8 + 1$	8191
14	$x^{14} + x^{13} + x^{12} + x^2 + 1$	16383
15	$x^{15} + x^{14} + 1$	32767
16	$x^{16} + x^{14} + x^{13} + x^{11} + 1$	65535
17	$x^{17} + x^{14} + 1$	131071
18	$x^{18} + x^{11} + 1$	262143
19	$x^{19} + x^{18} + x^{17} + x^{14} + 1$	524287
dst	dst	dst