PERANCANGAN APLIKASI KRIPTOGRAFI *IMAGE* MENGGUNAKAN METODE *ADVANCED ENCRYPTION STANDARD (AES)*

SKRIPSI

Diajukan untuk Memenuhi Salah Satu Persyaratan dalam Memperoleh Gelar Sarjana Matematika (S.Mat)

OLEH
NUR AFIFAH
NIM. H72214019

PROGRAM STUDI MATEMATIKA
JURUSAN SAINS
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI SUNAN AMPEL SURABAYA
SURABAYA

2018

PERNYATAAN KEASLIAN

Yang bertandatangan dibawah ini:

Nama : Nur Afifah

NIM : H72214019

Program Studi : Matematika

Angkatan : 2014

Menyatakan bahwa saya tidak melakukan plagiat dalam penulisan skripsi saya yang berjudul: **Perancangan Aplikasi Kriptografi *Image* Menggunakan Metode *Advanced Encryption Standard (AES).*** Apabila suatu saat nanti terbukti saya melakukan tindakan plagiat, maka saya akan menerima sanksi yang telah ditetapkan. Demikian pernyataan keaslian ini saya buat dengan sebenar-benarnya.

Surabaya, 3 Agustus 2018

Nur Afifah

NIM. H72214019

LEMBAR PENGESAHAN

PERANCANGAN APLIKASI KRIPTOGRAFI *IMAGE* MENGGUNAKAN METODE *ADVANCED ENCRYPTION STANDARD (AES)*

Disusun oleh

Nur Afifah
NIM.H72214019

Telah dipertahankan di depan Dewan Penguji
Pada Tanggal 18 Juli 2018
Dan dinyatakan telah memenuhi syarat untuk memperoleh gelar
Sarjana Matematika (S.Mat)

Dewan Penguji

Penguji I

Aris Fanani, M.Kom
NIP. 198701272014031002

Penguji II

Yuniar Farida, M.T
NIP. 197905272014032002

Penguji III

Wika Dianita Utami, M.Sc
NIP.199206102018012003

Penguji IV

Putroue Keumala Intan, M.Si
NIP.198805282018012001

Mengesahkan
Dekan Fakultas Sains dan teknologi
UIN Sunan Ampel Surabaya

Dr. Eni Purwati, M.Ag
NIP.196512211990022001

KEMENTERIAN AGAMA
UNIVERSITAS ISLAM NEGERI SUNAN AMPEL SURABAYA
PERPUSTAKAAN
Jl. Jend. A. Yani 117 Surabaya 60237 Telp. 031-8431972 Fax.031-8413300
E-Mail: perpus@uinsby.ac.id

LEMBAR PERNYATAAN PERSETUJUAN PUBLIKASI KARYA ILMIAH UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademika UIN Sunan Ampel Surabaya, yang bertanda tangan di bawah ini, saya:

Nama : NUR AFIFAH
NIM : H72214019
Fakultas/Jurusan : SAINTEK / MATEMATIKA
E-mail address : affahazfec22@gmail.com

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Perpustakaan UIN Sunan Ampel Surabaya, Hak Bebas Royalti Non-Eksklusif atas karya ilmiah :
☑ Skripsi ☐ Tesis ☐ Desertasi ☐ Lain-lain (.....)
yang berjudul :

PERANCANGAN APLIKASI KRIPTOGRAFI IMAGE MENGGUNAKAN METODE ADVANCED ENCRYPTION STANDARD (AES)

beserta perangkat yang diperlukan (bila ada). Dengan Hak Bebas Royalti Non-Ekslusif ini Perpustakaan UIN Sunan Ampel Surabaya berhak menyimpan, mengalih-media/format-kan, mengelolanya dalam bentuk pangkalan data (database), mendistribusikannya, dan menampilkan/mempublikasikannya di Internet atau media lain secara ***fulltext*** untuk kepentingan akademis tanpa perlu meminta ijin dari saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan atau penerbit yang bersangkutan.

Saya bersedia untuk menanggung secara pribadi, tanpa melibatkan pihak Perpustakaan UIN Sunan Ampel Surabaya, segala bentuk tuntutan hukum yang timbul atas pelanggaran Hak Cipta dalam karya ilmiah saya ini.

Demikian pernyataan ini yang saya buat dengan sebenarnya.

Surabaya, 31 Juli 2018

Penulis

(NUR AFIFAH)

PERANCANGAN KRIPTOGRAFI IMAGE MENGGUNAKAN METODE ADVANCED ENCRYPTION STANDARD (AES)

ABSTRAK

Perkembangan teknologi memberikan dampak yang signifikan terhadap kemajuan di segala bidang kehidupan, baik dampak positif maupun negatif. Diantara dampak negatif yaitu tindak kejahatan seperti pencurian, pembobolan dan lain-lain. Dalam bidang komunikasi keamananan suatu informasi merupakan hal yang sangat penting. Kerahasiaan data menjadi prioritas sehingga untuk menjaga keamanan suatu informasi diperlukan suatu aplikasi yang dapat mengenkripsikan data tersebut. Salah satu ilmu yang digunakan untuk membuat aplikasi keamanan suatu data adalah kriptografi. Dalam kriptografi ada banyak metode yang bisa digunakan untuk mengenkripsikan suatu data namun metode yang paling baik yaitu metode *Advanced Encripton Standart* (AES). Sehingga pada pembuatan aplikasi kriptografi ini menggunakan metode tersebut dan data yang digunakan berupa citra. Dalam proses pembuatan aplikasi dengan metode ini melalui beberapa tahapan diantaranya yiatu proses enkripsi, dekripsi, pembangkitan kunci dan pengujian terhadap metode yang digunakan. Pengujian tersebut berupa *crop, blur*, dan *enhancement* yang dilakukan pada citra *ciphertext*. Hasil dari enkripsi dan dekripsi yang dilakukan diketahui bahwa metode ini berhasil diaplikasikan terhadap citra. Sedangkan pada pengujian serangan yang dilakukan didapatkan bahwa metode ini tahan terhadap serangan *cropping*, namun tidak tahan terhadap serangan *blurring* dan *enhancement.*

Kata Kunci *: Kriptografi, Citra, Advanced Encryption Standard, Serangan. Enkripsi Dekripisi*

DESIGN OF IMAGE CRYPTOGRAPHY USING ADVANCED ENCRYPTION STANDARD (AES) METHOD

ABSTRACT

Development of technology gives significant impact on progress at all field of life, both positive and negative. Among the negative impacts is crime such as theft, burglary and others. In communication security, information is very important. Data confidentiality is a priority so that to maintain the security of an information is needed an application that can encrypt the data. One of the sciences used to design a data security application is cryptography. In cryptography, there are many methods can be used to encrypt a data but the best method is *Advanced Encryption Standard* method (AES). There are many types of AES can be used but the most effective is AES-128. So the aim of this study is to design image cryptographic application using the AES-128 method. Process of design applications with this method through several stages including such as process of encryption, decryption, generation of keys and testing of the methods used. The attacks test is given by *crop, blur,* and *enhancement* to the *ciphertext* image. The result of encryption and decryption known that this method successfully applied to the image. While on the attack test found that this method is resistant to *cropping* attacks, but not resistant to *blurring* and *enhancement* attacks.

Keywords*: Cryptography, Image, Advanced Encryption Standard, Attack, Encryption and Decryption.*

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

BAB I

PENDAHULUAN

A. Latar Belakang

Perkembangan teknologi merupakan salah satu aspek yang sangat penting dalam kehidupan manusia, termasuk kemajuan teknologi komputer yang dapat menyelesaikan semua pekerjaan. Bahkan hal serumit apapun dapat diselesaikan dengan cepat dan mudah oleh komputer. Perkembangan teknologi tersebut tidak terlepas dari perkembangan ilmu matematika karena setiap pembuatan sebuah teknologi baru selalu dihitung secara matematis menggunakan matematika. Namun dengan kecanggihan teknologi sekarang segala sesuatu yang menjadi rahasia dapat ditemukan dengan mudah hanya melalui kerja komputer. Hal ini yang sangat disayangkan. Hal ini juga merupakan dampak negatif dari perkembangan teknologi.

Dalam menemukan suatu data atau informasi yang dirahasiakan membuat orang yang ingin mengetahui menempuh segala cara untuk menemukannya, baik dengan cara membobol, mencuri, atau bahkan menyadap. Tidak jarang orang melakukan kejahatan-kejahatan seperti itu demi mendapatkan informasi tersebut. Beberapa contoh kasus pencurian data yang dialami oleh perusahaan besar seperti kasus pencurian *password LinkedIn* (Rahmat, 2016), pencurian data *Gmail* (Darmawan, 2011) dan masih banyak lagi.

Dalam perspektif agama, tindak kejahatan seperti itu merupakan suatu hal yang dilarang. Sebagaimana dalil yang terdapat pada Alqur'an surat An Nisa' ayat 135 yang berbunyi : *"wahai orang-orang yang beriman, jadilah kamu orang yang*

benar-benar menegakkan keadilan, menjadi saksi karena Allah biarpun terhadap dirimu sendiri atau ibu bapak dan kaum kerabatmu. Jika ia kaya atau miskin, maka Allah lebih tahu kemaslahatannya… maka sesungguhnya Allah adalah Maha Mengetahui segala apa yang kamu kerjakan". Dalam dalil tersebut menjelaskan tentang larangan kita untuk berbuat kriminalitas, baik pencurian, penipuan ataupun yang lain. Dalam dalil lain juga jelas disebutkan *" laki-laki yang mencuri dan perempuan yang mencuri, potonglah tangan keduanya (sebagai) pembalasan bagi apa yang mereka kerjakan dan sebagai siksaan dari Allah. Dan Allah Maha Perkasa lagi Maha Bijaksana" (QS. Al Maidah :38).* Dimana dalil kedua merupakan salah satu yang menjelaskan mengenai hukuman yang diperoleh akibat dari tindak kejahatan yaitu mencuri.

Selain dilarang oleh agama, hal ini juga termasuk pelanggaran terhadap hak cipta yang terdapat dalam UU No.28 tahun 2014. Karena informasi yang didapatkan tidak menggunakan izin dari pemilik atau pembuatnya. Selain itu Indonesia juga mengatur tentang hukum pencurian data dalam UU ITE tahun 2008 pasal 3.

Kriptografi merupakan studi matematika komputasi yang mempunyai hubungan dengan keamanan informasi seperti integritas data, keaslian entitas dan keaslian data. Kriptografi menggunakan berbagai macam teknik dalam upaya untuk mengamankan data. Dalam algoritma kriptografi terdapat dua jenis yaitu algoritma kriptografi simetris dan algoritma kriptografi asimetris. Algoritma simetris disebut juga algoritma konvensional, algoritma ini menggunakan kunci yang sama untuk enkripsi dan deskripsi. Sedangkan algoritma asimetris menggunakan kunci yang berbeda dalam proses enkripsi dan deskripsi (Munir, 2006).

Algoritma AES merupakan salah satu algoritma kriptografi simetris yang beroperasi pada sekumpulan *byte* data atau per blok. Algoritma kriptografi Rijndael yang didesain oleh Vincent Rijmen dan John Daemen ini adalah pengembangan dari algoritma DES (*Data Encryption Standard*), karena algoritma DES dianggap sudah tidak aman lagi karena dengan perangkat keras khusus kuncinya dapat ditemukan dalam beberapa hari saja (NIST, 1998). Algoritma AES berorientasi pada penyandi blok (*block cipher*) yang memproses blok data 128-bit dengan panjang kunci 128-bit, 192-bit, atau 256-bit. Pada tahun 2006, AES merupakan salah satu algoritma terpopuler yang digunakan dalam kriptografi kunci simetrik (Munir, 2006).

Beberapa algoritma kriptografi simetris lainnya yaitu Blowfish, DES, IDEA, RC4, RC5, Triple DES, Twofish, dan Serpent. Diantara algoritma tersebut memiliki kelemahan dan kelebihan yang disajikan dalam tabel berikut :

Table 1.1 Perbandingan Algoritma Kriptografi Simetri

Chiper	Pembuat	Panjang Kunci	Keterangan
Blowfish	Bruce Schneier	1-448 Bit	Old And Slow
DES	IBM	56 Bit	Too Weak To Use Now
IDEA	Massey Dan Xuejia	128 Bit	Good, But Patented
RC4	Ronald Rivest	1-2048 Bit	Caution : Some Keys Are Weak
RC5	Ronald Rivest	128-256 Bit	Good, But Patented
Rijndael/AES	Daemen Dan Rijmen	128-256 Bit	Best Choice
Serpent	Anderson, Biham, Knudsen	128-256 Bit	Very Strong
Triple DES	IBM	168 Bit	Second Best Choice
Twofish	Bruce Schneier	128-256 Bit	Very Strong, Widely Used

Sumber : Rinaldi Munir, 2006

Dari perbandingan di atas maka pada perancangan aplikasi untuk kriptografi ini menggunakan algoritma AES atau Rijndael. Dalam pengimplementasian algoritma ini pada citra digital sebelumnya sudah pernah dilakukan diantaranya Rancang Bangun Aplikasi Enkripsi dan Deskripsi Citra Digital Menggunakan Algoritma Rijndael Berbasis Java SE yang menghasilkan teknik pengenkripsian dan pendekripsian dengan nilai akurasi 100% (Palilianto, 2012), *Image Encryption And Descryption Using AES Algorithm* dapat melakukan enkripsi dan dekripsi sebuah citra (Roshni Padate, Aamna Patel, 2015), *Effective AES Implementation* yang menghasilkan algoritma AES yang paling baik adalah menggunakan 128-bit (Arya, 2016). Implementasi Algoritma Rijndael untuk enkripsi dan dekripsi pada citra digital yang menghasilkan bahwa kecepatan proses enkripsi dan dekripsi bergantung pada panjang kunci yang digunakan (Bendi, 2012).

Menurut Schneier, Algoritma yang terbaik adalah algoritma yang telah dipublikasikan dan telah diserang oleh para kriptografer dan kriptanalis terbaik dunia dan hingga kini belum berhasil dipecahkan (Munir, 2006). Serangan itu sendiri ada tiga macam yaitu serangan berdasarkan keterlibatan penyerang, banyaknya informasi yang diketahui oleh kriptanalis, dan teknik yang digunakan. Salah satu serangan berdasarkan banyaknya informasi diantaranya adalah *ciphertext only attack* dimana serangan ini dilakukan dari *ciphertext* yang ada. Kriptanalis terebut melakukan penyadapan melalui serangan yang diberikan terhadap *ciphertext* untuk menemukan kunci seperti mencoba semua kemungkinan kunci secara *exhaustive search*, menggunakan teknik analisis dan masih banyak lagi.

Pada beberapa penelitian diatas tidak dilakukan pengujian hasil enkripsi dan dekripsi dari serangan sehingga pada penelitian ini akan mengembangkan aplikasi enkripsi dan dekripsi sebuah citra atau *image* dengan pengujian ketahanan metode enkripsi dan dekripsi tersebut dari serangan seperti *crop*, *blur*, *Enhancement*, dan lain sebagainya.

B. Rumusan Masalah

Berdasarkan latar belakang permasalahan sebelumnya, maka pokok permasalahan yang dikaji dalam penelitian ini adalah sebagai berikut.

1. Apakah metode *Advanced Encryption Standart* (AES) dapat digunakan untuk enkripsi dan dekripsi sebuah citra atau *image*?
2. Bagaimana hasil pengujian terhadap citra hasil enkripsi dan dekripsi tersebut terhadap serangan ?

C. Tujuan Penelitian

Tujuan dari penelitian ini adalah sebagai berikut.

1. Untuk mengetahui *enkripsi* sebuah data *image* menggunakan metode *Advanced Encrypton Standart (AES)*.
2. Untuk mengetahui hasil pengujian terhadap citra hasil enkripsi dan dekripsi tersebut terhadap serangan.

D. Batasan Masalah

Berikut ini adalah batasan masalah yang ditetapkan oleh peneliti :

1. Data yang digunakan untuk pengujian sistem adalah citra atau *image grayscale*.
2. Ukuran blok data yang digunakan adalah 128 bit atau 16 *byte*.

3. Ukuran pada citra atau *image* adalah 32 x 32 piksel.
4. Serangan yang digunakan untuk pengujian sistem meliputi *crop*, *blur*, dan *enhancement*.

E. Manfaat Penelitian

Manfaat toritis dari penelitian ini adalah sebagai berikut.

1. Bagi penulis, dapat menerapkan dan mengembangkan ilmu keamanan data digital dari sebuah citra atau *image* dan juga dapat menambah pengetahuan terapan dan mengaplikasikan ilmu kriptografi yang telah didapat semasa kuliah dengan kondisi yang ada di lapangan.
2. Bagi akademik, dapat mengetahui kemampuan mahasiswanya dalam menguasai teori-teori dalam ilmu kriptografi, keamanan data digital, dan dapat dijadikan sebagai bahan evaluasi.
3. Bagi pengguna, aplikasi kriptografi *image* ini dapat dijadikan sebagai salah satu media alternatif untuk mengamankan pengiriman sebuah dokumen atau pesan *image* sehingga dapat diterima oleh *receiver* dalam keadaan tetap rahasia.

F. Sistematika Penulisan

Penulisan pada penelitian ini disusun degan sistematika yang terdiri dari beberapa bab atau subbab sebagaimana berikut:

Bab I Pendahuluan

Pada bab ini berisi pemaparan tentang latar belakang masalah dalam penelitian, rumusan masalah, batasan masalah, tujuan dan manfaat penelitian, dan sistematika penulisan.

Bab II Tinjauan Pustaka

Pada bab ini berisikan tentang landasan teori yang digunakan dalam penyusunan dan penulisan diantaranya yaitu membahas pengertian kriptografi, algoritma AES dan citra digital.

Bab III Metode Penelitian

Bab ini berisi mengenai metodologi yang digunakan oleh penulis dan rancangan sistem penelitian yang dilakukan.

Bab IV Hasil dan Pembahasan

Bab ini menjelaskan hasil dari proses enkripsi dan dekripsi serta pengujian citra yang telah dienkripsikan terhadap serangan yang telah ditentukan.

Bab V Kesimpulan dan Saran

Pada bab ini akan berisi kesimpulan dari pembahasan pada bab-bab sebelumnya serta saran-saran yang bersifat membangun dan dapat mengembangkan sistem yang telah diteliti.

BAB II

TINJAUAN PUSTAKA

Pada bab ini akan dipaparkan mengenai teori-teori yang digunakan dan berkaitan dengan penelitian yang dilakukan. Diantaranya adalah penjelasan tentang ilmu kriptografi, metode *Advanced Encryption Standard* (AES) beserta algoritmanya, dan pengertian citra.

A. Kriptografi

Kriptografi berasal dari bahasa Yunani, menurut bahasa dibagi menjadi dua *kripto* dan *graphia*, *kripto* berarti *secret* (rahasia) dan *graphia* berarti *writing* (tulisan) (Munir, 2006). Menurut teminologinya kriptografi adalah mengenai teknik enkripsi dimana data diacak mengggunakan suatu kunci enkripsi menjadi sesuatu yang sulit dibaca oleh seseorang yang tidak memiliki kunci dekripsi (Kromodimoeljo, 2009).

Implementasi dari ilmu kriptografi sangat banyak dan bisa kita temui dalam kehidupan sehari-hari, seperti *Automatic Teller Machine* (ATM), penggunaan ATM untuk *banking*, bahkan mulai meningkat menjadi *Internet Banking* dan lain sebagainya. Menurut catatan sejarah, kriptografi sudah digunakan oleh bangsa Mesir sejak 4000 tahun yang lalu oleh raja-raja Mesir pada saat perang untuk mengirimkan pesan rahasia kepada panglima perangnya melalui kurir-kurinya. Orang yang melakukan penyandian ini disebut kriptografer, sedangkan orang yang mendalami ilmu dan seni dalam membuka atau memecahkan suatu algoritma kriptografi tanpa harus mengetahui kuncinya disebut kriptanalis (Munir, 2006).

Seiring dengan perkembangan teknologi, algoritma kriptografi pun mulai berubah menuju ke arah algoritma kriptografi yang lebih rumit dan kompleks. Kriptografi mau tidak mau harus diakui mempunyai peranan yang paling penting dalam peperangan sehingga algoritma kriptografi berkembang cukup pesat pada saat Perang Dunia I dan Perang Dunia II (Munir, 2006). Menurut catatan sejarah, terdapat beberapa algoritma kriptografi yang pernah digunakan dalam peperangan, diantaranya adalah ADFVGX yang dipakai oleh Jerman pada Perang Dunia I, Sigaba/M-134 yang digunakan oleh Amerika Serikat pada Perang Dunia II, Typex oleh Inggris, dan Purple oleh Jepang. Selain itu Jerman juga mempunyai mesin legendaris yang dipakai untuk memecahkan sandi yang dikirim oleh pihak musuh dalam peperangan yaitu Enigma (Kromodimoeljo, 2009).

Algoritma kriptografi yang baik ditentukan oleh kerumitan dalam mengolah data atau pesan yang akan disampaikan dan memenuhi 4 persyaratan berikut (Munir, 2006):

- Kerahasiaan. Pesan (*plaintext*) hanya dapat dibaca oleh pihak yang memiliki kewenangan.
- Autentikasi. Pengirim pesan harus dapat diidentifikasi dengan pasti, penyusup harus dipastikan tidak bisa berpura-pura menjadi orang lain.
- Integritas. Penerima pesan harus dapat memastikan bahwa pesan yang dia terima tidak dimodifikasi ketika sedang dalam proses transmisi data.
- *Non-Repudiation*. Pengirim pesan harus tidak bisa menyangkal pesan yang dia kirimkan.

Kriptografi pada dasarnya terdiri dari dua proses, yaitu proses enkripsi dan proses dekripsi. Proses enkripsi adalah proses penyandian pesan yang umum menjadi pesan rahasia atau yang disebut dengan *ciphertext*. *Ciphertext* inilah yang nantinya akan dikirimkan melalui saluran komunikasi terbuka. Pada saat *ciphertext* diterima oleh penerima pesan, selanjutnya pesan rahasia tersebut diubah lagi menjadi pesan terbuka melalui proses dekripsi sehingga pesan tersebut dapat dibaca kembali oleh penerima pesan. *Plaintext* ini dapat berupa tulisan, foto, atau video yang berbentuk data biner. Ada beberapa istilah penting yang harus diketahui dalam kriptografi (Munir, 2006) :

1. Pesan, *Plaintext* dan *Ciphertext.*

   Pesan adalah data atau informasi yang dapat dibaca dan dimengerti maknanya. Nama lain untuk pesan adalah *plaintext*. Agar pesan tidak bisa dimengerti maknanya oleh pihak lain, maka pesan tersebut perlu disandikan ke bentuk lain yang tidak dapat dipahami sehingga pesan hanya bisa dibaca oleh pihak yang bersangkutan. Bentuk pesan yang tersandi inilah yang disebut *ciphertext.*

2. Pengirim dan Penerima

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

3. Enkripsi dan Dekripsi

   Enkripsi merupakan proses menyandikan *plaintext* menjadi *ciphertext*. Sedangkan proses mengembalikan *ciphertext* menjadi *plaintext* semula dinamakan dekripsi.

4. *Cipher*

   Algoritma kriptografi disebut juga *cipher* yaitu aturan untuk *enciphering* dan *deciphering*, atau fungsi matematika yang digunakan untuk enkripsi dan dekripsi. Konsep matematis yang m algoritma menjadi dasar dari algoritma kriptografi ini adalah relasi antara dua buah himpunan yaitu himpunan yang berisi elemen-elemen *plaintext* dan himpunan yang berisi *ciphertext*. Sedangkan enkripsi dan dekripsi adalah fungsi yang memetakan elemen-elemen antara kedua himpunan tersebut.

5. Sistem Kriptografi

   Sistem kriptografi merupakan kumpulan yang terdiri dari algoritma kriptografi, semua *plaintext* dan *ciphertext* yang mungkin dan kuncinya.

6. Penyadap

   Penyadap adalah orang yang berusaha mencoba untuk menangkap pesan selama proses transmisi dengan tujuan mendapatkan informasi sebanyak-banyaknya mengenai sistem kriptografi yang digunakan untuk berkomunikasi dengan maksud untuk memecahkan *ciphertext*.

7. Kriptanalisis dan Kriptologi

   Kriptanalisis (*cryptanalysis*) adalah ilmu dan seni untuk memecahkan *ciphertext* menjadi *plaintext* tanpa mengetahui kunci yang digunakan.

Pelakunya disebut kriptanalis. Kriptologi adalah studi mengenai kriptografi dan kriptanalisis.

B. Metode Kriptografi

Dalam menjaga kerahasiaan data, kriptografi mentransformasikan data jelas (*plaintext*) ke dalam bentuk data sandi (*ciphertext*) yang tidak dapat dikenali. *Ciphertext* inilah yang kemudian dikirimkan oleh pengirim (*sender*) kepada penerima (*receiver)*. Setelah sampai di penerima, *ciphertext* tersebut ditranformasikan kembali ke dalam bentuk *plaintext* agar dapat dibaca.

Proses tranformasi dari *plaintext* menjadi *ciphertext* disebut proses *Encipherment* atau enkripsi (*encryption*), sedangkan proses mentransformasikan kembali *ciphertext* menjadi *plaintext* disebut proses dekripsi (*decryption*). Dalam melakukan proses mengenkripsi dan mendekripsi data, Kriptografi menggunakan suatu algoritma (*cipher*) dan kunci (*key)*. *Cipher* adalah fungsi matematika yang digunakan untuk mengenkripsi dan mendekripsi data. Sedangkan kunci merupakan sederetan bit yang diperlukan untuk mengenkripsi dan mendekripsi data.

Algoritma kriptografi adalah algoritma yang berfungsi untuk melakukan tujuan dari ilmu kriptografi itu sendiri yaitu penyandian. Secara umum berdasarkan kesamaan kuncinya, algoritma penyandian dibedakan menjadi 2 yaitu :

1. Algoritma Kunci Simetris

Dalam *symetric cryptosystem* ini, kunci yang digunakan untuk proses enkripsi dan dekripsi pada prinsipnya identik, tetapi satu buah kunci dapat pula diturunkan dari kunci yang lainnya. Kunci-kunci ini harus dirahasiakan. Oleh karena itulah sistem ini sering disebut sebagai *secret-key ciphersystem*.

Contoh dari sistem ini adalah *Data Encryption Standard* (DES), Blowfish, IDEA.

Kriptografi *secret key* seringkali disebut sebagai kriptografi konvensional atau kriptografi simetris (*Symmetric Cryptography)* dimana proses dekripsi adalah kebalikan dari proses enkripsi dan menggunakan kunci yang sama.

Kriptografi simetris dapat dibagi menjadi dua, yaitu penyandian blok dan penyandian alir. Penyandian blok bekerja pada suatu data yang terkelompok menjadi blok-blok data atau kelompok data dengan panjang data yang telah ditentukan. Pada penyandian blok, data yang masuk akan dipecah-pecah menjadi blok data yang telah ditentukan ukurannya. Penyandian alir bekerja pada suatu data bit tunggal atau terkadang dalam satu *byte*. Jadi format data yang mengalami proses enkripsi dan dekripsi adalah berupa aliran bit-bit data.

Algoritma yang digunakan pada saat ini kebanyakan bekerja untuk penyandian blok karena kebanyakan proses pengiriman data pada saat ini menggunakan blok-blok data yang telah ditentukan ukurannya untuk kemudian dikirim melalui saluran komunikasi.

2. Algoritma Kunci Asimetris.

Algoritma Asimetris atau sering disebut algoritma *public key*, penggunaan kunci dalam algoritma ini adalah, kunci yang dipakai dalam proses enkripsi berbeda dengan kunci yang dipakai pada proses dekripsi, jadi jumlah kunci enkripsi tidak sama dengan kunci dekripsi.

Ada 2 jenis kunci di algoritma ini, yaitu

a. Kunci publik adalah kunci yang digunakan untuk melakukan proses enkripsi data bersifat terbuka sehingga setiap orang bisa mengetahui kuncinya.
b. Kunci privat adalah kunci yang digunakan untuk melakukan proses dekripsi data. Kunci ini disebut privat karena 1 kunci privat hanya dimiliki oleh 1 orang saja. Kunci privat sering juga disebut kunci rahasia. Kunci ini digunakan untuk membuka pesan yang telah dirahasiakan.

Istilah kunci rahasia dalam algoritma simetris tersebut digunakan untuk menyatakan kunci enkripsi dan dekripsi, sementara pada algoritma asimetris yang digunakan adalah kunci privat, karena kunci publik tidak dirahasiakan.

C. Advanced Encryption Standard (AES)

Advanced Encryption Standard (AES) merupakan salah satu algoritma kriptografi yang dapat digunakan untuk mengamankan data. Algoritma AES ini bekerja pada blok *ciphertext* simetrik yang dapat mengenkripsi (*encipher*) dan dekripsi (*decipher*) informasi. Algoritma AES ini menggunakan kunci kriptografi 128, 192, dan 256 bits untuk mengenkripsi dan dekripsi data. Maka dari itu algoritma ini dikenal dikenal sebagai AES-128, AES-192, dan AES-256. Algoritma ini juga mempunyai nama lain yiatu algoritma *Rijndael*. Karena algoritma ini dibuat oleh *Rijndael*.

AES (*Advanced Encryption Standart)* merupakan pengembangan dari algoritma enkripsi standar DES (*Data Encryption Standart*) yang masa berlakunya

dianggap telah usai karena faktor keamanan. Kecepatan komputer yang sangat pesat dianggap sangat membahayakan DES, sehingga pada tanggal 2 Maret tahun 2001 ditetapkanlah algoritma baru Rijndael sebagai AES. Kriteria pemilihan AES didasarkan pada 3 kriteria utama yaitu: keamanan, harga, dan karakteristik algoritma beserta implementasinya. Keamanan merupakan faktor terpenting dalam evaluasi (minimal seaman *Triple* DES), yang meliputi ketahanan terhadap semua analisis sandi yang telah diketahui dan diharapkan dapat menghadapi analisis sandi yang belum diketahui. Di samping itu, AES juga harus dapat digunakan secara bebas tanpa harus membayar royalti dan juga murah untuk diimplementasikan pada smart card yang memiliki ukuran memori kecil. AES juga harus efisien dan cepat (minimal secepat *Triple* DES) dijalankan dalam berbagai mesin 8 bit hingga 64 bit, dan berbagai perangkat lunak. DES menggunakan stuktur Feistel yang memiliki kelebihan bahwa struktur enkripsi dan dekripsinya sama, meskipun menggunakan fungsi f yang tidak invertibel. Kelemahan Feistel yang utama adalah bahwa pada setiap ronde, hanya setengah data yang diolah. Sedangkan AES menggunakan struktur SPN (*Substitution Permutation Network*) yang memiliki derajat paralelisme yang lebih besar, sehingga diharapkan lebih cepat dari pada Feistel.

Kelemahan SPN pada umumnya (termasuk pada Rijndael) adalah berbedanya struktur enkripsi dan dekripsi sehingga diperlukan dua algoritma yang berbeda untuk enkripsi dan dekripsi. Dan tentu pula tingkat keamanan enkripsi dan dekripsinya menjadi berbeda. AES memiliki blok masukan dan keluaran serta kunci 128 bit. Untuk tingkat keamanan yang lebih tinggi, AES dapat menggunakan kunci 192 dan 256 bit. Setiap masukan 128 bit *plaintext* dimasukkan ke dalam *State* yang

berbentuk bujur sangkar berukuran 4×4 *byte*. *State* ini di-XOR dengan *key* dan selanjutnya diolah 10 kali dengan subtitusi-transformasi linear-*Addkey*. Dan di akhir diperoleh *ciphertext*.

Berikut ini adalah operasi Rijndael (AES) yang menggunakan 128 bit kunci:

- Ekspansi kunci utama (dari 128 bit menjadi 1408 bit)
- Pencampuran *subkey*
- Ulang dari i=1 sampai i=10 Transformasi : *Byte*Sub (subtitusi per *byte*) *ShiftRow* (Pergeseren *byte* perbaris) *MixColumns* (Operasi perkalian GF(2^8) per kolom) (Publications, 2001)
- Pencampuran *subkey* (dengan XOR)
- Transformasi : *Byte*Sub dan *ShiftRow*
- Pencampuran *subkey*.

1. Representasi data

Input dan output dari algoritma AES ini terdiri dari urutan data sebesar 128 bit urutan data yang sudah terbentuk dalam satu kelompok 128 bit tersebut yang selanjutnya disebut sebagai blok data atau *plaintext* yang kemudian akan di lakuakan proses enkripsi menjadi *ciphertext*. *Cipher key* dari AES bergantung pada panjang bit yang digunakan.

Urutan dari bit diberi beri nomor urut mulai dari 0 sampai dengan n-1 dimana n adalah nomor urutan. Urutan data 8 bit secara beruntun disebut sebagai *byte*, dimana *byte* ini adalah unit dasar dari operasi yang akan dilakukan pada blok data.

Dalam algoritma AES data sepanjang 128 bit akan dibagi menjadi *array byte* dimana setiap *array byte* ini terdiri dari 8 bit data input yang saling berurutan. *Array byte* ini direpresentasikan dalam bentuk:

$$a_0 a_1 a_2 \dots a_{15} \quad (2.1)$$

Dimana:

$a_0 = \{input_0, input_1, \dots, input_7\}$

$a_1 = \{input_8, input_9, \dots, input_{15}\}$

$a_{15} = \{input_{120}, input_{121}, \dots, input_{127}\}$

$a_n = \{input_{8n}, input_{8n+1}, \dots, input_{8n+7}\}$

Operasi algoritma AES dilakukan pada sebuah *state* dimana *state* sendiri adalah seuah *array byte* dua dimensi. Setiap *state* pasti mempunyai jumlah baris yang tetap yaitu 4, sedangkan jumlah kolom tergantung pada besarnya blok data. Baris pada *state* mempunyai indeks nomor *row* (r) dimana $0 \leq r < 4$. Sedangkan kolom mempunyai indeks *column* (c) dimana $0 \leq c < Nb$. Nb adalah besarnya blok data dibagi dengan 32.

Pada saat permulaan input bit awalnya akan disusun menjadi suatu *array byte* diana panjang dari *array byte* yaitu 8 bit data. *Array byte* inilah yang kemudian akan dimasukkan atau di kopi kedalam *state* dengan urutan (Fachrurrozi, Muhammad Farid, 2006) :

$$s[r, c] = in[r + 4c] \ \ untuk\ 0 \leq r < 4\ dan\ 0 \leq c < Nb \quad (2.2)$$

Sedangkan dari *state* yang akan dikopi ke output dengan urutan:

$$out[r + 4c] = s[r, c] \ \ untuk\ 0 \leq r < 4\ dan\ 0 \leq c < Nb \quad (2.3)$$

2. Operasi aljabar

Dalam metode AES merupakan proses operasi matematika. Setiap tahapan atau langkah transformasinya melibatkan *State*, akan tetapi unit dasar operasi AES adalah *byte* dan setiap *byte* merupakan elemen *finite field* $GF(2^8)$yang didefinisikan oleh operasi penjumlahan dan perkalian. $GF(2^8)$ Merupakan *finite field* yang merupakan sebuah himpunan dengan modulo 8.

Elemen *finite field* adalah elemen dari *field* yang memiliki sifat *ring* komutatif. Dalam hal ini untuk semua *finite field* yang memiliki p^n dimana p merupakan bilangan prima dan n merupakan bilangan bulat $n \geq 1$ sama dinotasikan dengan $GF(p^n)$. Oleh karena itu, elemen $GF(2^8)$ juga merupakan *ring* komutatif yang memiliki sifat-sifat berikut (Wagstaff, 2003):

- Grup (G , +), Grup sendiri memenuhi sifat tertutup pada operasi penjumlahan, asosiatif pada operasi penjumlahan, memiliki elemen identitas, mempunyai invers, dan grup abelian
- *Ring* (G , + , *), *Ring* juga memenuhi sifat tertutup pada operasi perkalian, asosiatif pada operasi perkalian, dan distributif.

3. Field GF(2^8)

Elemen dari *finite field* bisa direpresentasikan dengan beberapa cara yang berbeda. Bisa dengan polinomial, bit, maupun heksadesimal. Untuk semua pangkat n adalah bilangan prima memiliki satu *finite field* (Fachrurrozi, Muhammad Farid, 2006). Sehingga $GF(2^8)$ dan ASCII merupakan isomorfisme atau homomorfisme yang memiliki fungsi satu-satu. Sedangkan homomorfisme adalah fungsi dari *ring* ke *ring* lain yang

mempunyai sifat $f(a+b) = f(a) + f(b)\ dan\ f(ab) = f(a)f(b)$ (Wagstaff, 2003). Jadi ketika elemen tersebut direpresentasikan maka akan memiliki pengaruh yang kuat dalam implementasi yang kompleks. Dalam hal ini akan direpresentasikan *ring* atas polinomial. Jika b merupakan suatu nilai dari 0 atau 1 maka terbentuk ukuran *byte* dari urutan bit $b_7 + b_6 + b_5 + b_4 + b_3 + b_2 + b_1 + b_0$ (koefisien *binary*) sehingga dapat dituliskan pada persamaan 2.4

$$b(x) = b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0 \qquad (2.4)$$

Dari persamaan diatas dapat diketahui bahwa pangkat tertinggi dari polinomial GF(2^8) adalah x^7.

a. Penjumlahan

Penjumlahan dari dua elemen *finite field* dapat didefinisikan dengan operasi XOR (penjumlahan 2 elemen dengan modulo 2) per bit. Sehingga penyederhanaannya merupakan operasi identik. Misal $30 \oplus d4 =$

00110000

<u>11010100</u> $\oplus$

<u>11100100</u>

$(x^5 + x^4) + (x^7 + x^6 + x^4 + x^2) = (x^7 + x^6 + x^5 + x^2)$

Bentuk biner dari $(x^7 + x^6 + x^5 + x^2)$ adalah 11100100 dan dalam heksadesimal diperoleh $e4$. Semua kondisi yang penting dalam menyelesaikan operasi di atas merupakan bagian dari grup abelian.

b. Perkalian

Perkalian elemen GF(2^8) adalah perkalian yang direpresentasikan dalam polinomial dengan modulo polinomial m(x) yang *irreducible* dari polinomial pangkat 8 (Vincent, 2003). *Irreducible* adalah polinomial yang hanya mempunyai faktor 1 dan bilangan itu sendiri.

$$m(x) = x^8 + x^4 + x^3 + x + 1 \qquad (2.5)$$

Atau 11B dalam bentuk heksadesimal dan bentuk desimal adalah 283.

Contoh :

$57 \cdot 84$

$= [01010111] \cdot [10000100]$

$= (x^6 + x^4 + x^2 + x + 1)\ (x^7 + x^2)\ mod$

$(x^8 + x^4 + x^3 + x + 1)$

$= (\ x^{13} + x^{11} + x^9 + x^8 + x^7 + x^8 + x^6 + x^4 + x^3 + x^2\ )mod$

$(x^8 + x^4 + x^3 + x + 1)$

$= (x^{13} + x^{11} + x^9 + x^7 + x^6 + x^4 + x^3 + x^2\ )\ mod$

$(x^8 + x^4 + x^3 + x + 1)$

$= (x^{13} + x^{11} + x^9 + x^8 + x^7 + x^6 + x^2 + x + 1)\ mod$

$(x^8 + x^4 + x^3 + x + 1)$

$= x^{13} + x^{11} + x^9 + x^7 + x^6 + x^4 + x^3 + x^2 + 1$

Dari bentuk polinomial mod 8 tersebut dirubah ke biner menjadi 11011101 yang dalam bentuk heksadesimalnya adalah dd.

c. Perkalian dengan variabel x

Jika dituliskan perkalian $b(x)$ sebagai berikut:

$$x \cdot b(x) = b_8x^8 + b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0$$

Perkalian $x \cdot b(x)$ dapat diwujudkan sebagai *leftshift* atau persegseran bit ke kiri yang diikuti oleh XOR kondisional dengan (1b), jika b_8 =1 maka XOR dilakukan, namun jika b_8 =0 maka tidak dilakukan XOR. *Exclusive* OR kondisional tersebut tidak lain adalah operasi modulo dengan $m(x)$. Serangkaian *left shift* yang disusul operasi XOR tersebut dapat digunakan untuk perkalian antara elemen *finite field*. Operasi $x \cdot b(x)$ dinotasikan sebagai *xtime* (Vincent, 2003). Sebagai contoh:

$$'57' \cdot '13' = 'FE'$$

$$'57' \cdot '02' = \ x\ time\ (57) = 'AE'$$

$$'57' \cdot '04' = x\ time\ (AE) = '47'$$

$$'57' \cdot '08' = x\ time\ (47) = '8E'$$

$$'57' \cdot '10' = x\ time\ (8E) = '07'$$

$$'57' \cdot '13' = '^{57'} \cdot ('01' \oplus' 02' \oplus' 10') =' 57' \oplus' AE' \oplus' 07' = 'FE'$$

3. Koefisien polinom pada GF(2^8)

Direpresentasikan polinomial yang didefinisikan dengan koefisien GF(2^8) sebagai persamaan berikut:

$$a(x) = a_3x^3 + a_2x^2 + a_1x + a_0 \qquad (2.6)$$

Sehingga didapat bentuk koefisien sebagai $[a_3, a_2, a_1, a_0]$. Polinomial ini berbeda dengan polinomial pada *finite field* yang sebelumnya sebagai

polinomial koefisien biner. Pada polinomial ini akan dioperasikan perkalian dengan polinomial yang berbeda, akan tetapi bentuk polinomialnya sama yaitu berderajat 4.

$$b(x) = b_3x^3 + b_2x^2 + b_1x + b_0 \quad (2.7)$$

Definisi dari kedua polinomial diatas dioperasikan sebagai XOR antara perasamaan tersebut. Operasi XOR ini berkoresponden antara pangkat pada variabel x dapat dilihat pada persamaan berikut:

$$a(x) + b(x) = (a_3 \oplus b_3)x^3 + (a_2 \oplus b_2)x^2 + (a_1 \oplus b_1)x + (a_0 \oplus b_0) \quad (2.8)$$

Didefnisikan $a(x) + b(x) = c(x)$ sehingga menghasilkan Persamaan:

$$c(x) = c_6x^6 + c_5x^5 + c_4x^4 + c_3x^3 + c_2x^2 + c_1x + c_0 \quad (2.9)$$

didapat dengan cara:

$c_0 = a_0 \cdot b_0$

$c_1 = a_1 \cdot b_0 \oplus a_0 \cdot b_1$

$c_2 = a_2 \cdot b_0 \oplus a_1 \cdot b_1 \oplus a_0 \cdot b_2$

$c_3 = a_3 \cdot b_0 \oplus a_2 \cdot b_1 \oplus a_1 \cdot b_2 \oplus a_0 \cdot b_3$

$c_4 = a_3 \cdot b_1 \oplus a_2 \cdot b_2 \oplus a_1 \cdot b_3$

$c_5 = a_3 \cdot b_2 \oplus a_2 \cdot b_3$

$c_6 = a_3 \cdot b_3$

Hasil dari $c(x)$ diatas belum dalam bantuk 4 *byte*, maka selanjutnya yaitu $c(x)$ di modularkan dengan polinomial berderajat 4. Pada algoritma AES diberikan polinomial $x^4 + 1$. Menjadi:

$$x^i \, mod(x^4 + 1) = x^{i \, mod \, 4}$$

Operasi modulo dari $a(x)$ dan $b(x)$ menghasilkan sebuah $d(x)$ yang direpresentasikan pada Persamaan 2.12.

$$d(x) = d_3x^3 + d_2x^2 + d_1x + d_0 \tag{2.10}$$

dengan hasil:

$$\begin{aligned} d_0 &= a_0 \cdot b_0 \oplus a_3 \cdot b_1 \oplus a_2 \cdot b_2 \oplus a_1 \cdot b_3 \\ d_1 &= a_1 \cdot b_0 \oplus a_0 \cdot b_1 \oplus a_3 \cdot b_2 \oplus a_2 \cdot b_3 \\ d_2 &= a_2 \cdot b_0 \oplus a_1 \cdot b_1 \oplus a_0 \cdot b_2 \oplus a_3 \cdot b_3 \\ d_3 &= a_3 \cdot b_0 \oplus a_2 \cdot b_1 \oplus a_1 \cdot b_2 \oplus a_0 \cdot b_3 \end{aligned} \tag{2.11}$$

$$\begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{bmatrix} = \begin{bmatrix} a_0 & a_3 & a_2 & a_1 \\ a_1 & a_0 & a_3 & a_2 \\ a_2 & a_1 & a_0 & a_3 \\ a_3 & a_2 & a_1 & a_0 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} \tag{2.12}$$

4. Algoritma AES

Algoritma kriptografi bernama Rijndael yang didesain oleh Vincent Rijmen dan John Daemen asal Belgia keluar sebagai pemenang kontes algoritma kriptografi pengganti DES yang diadakan oleh NIST (*National Institutes of Standards and Technology*) milik pemerintah Amerika Serikat pada 26 November 2001. Algoritma Rijndael inilah yang kemudian dikenal dengan *Advanced Encryption Standard* (AES). Setelah mengalami beberapa proses standardisasi oleh NIST, Rijndael kemudian diadopsi menjadi standar algoritma kriptografi secara resmi pada 22 Mei 2002. Pada 2006, AES merupakan salah satu algoritma terpopuler yang digunakan dalam kriptografi kunci simetrik. AES ini merupakan algoritma *block cipher* dengan menggunakan sistem permutasi dan substitusi (P-Box dan S-Box). Jenis AES terbagi 3, yaitu:

1. AES-128;
2. AES-192;
3. AES-256.

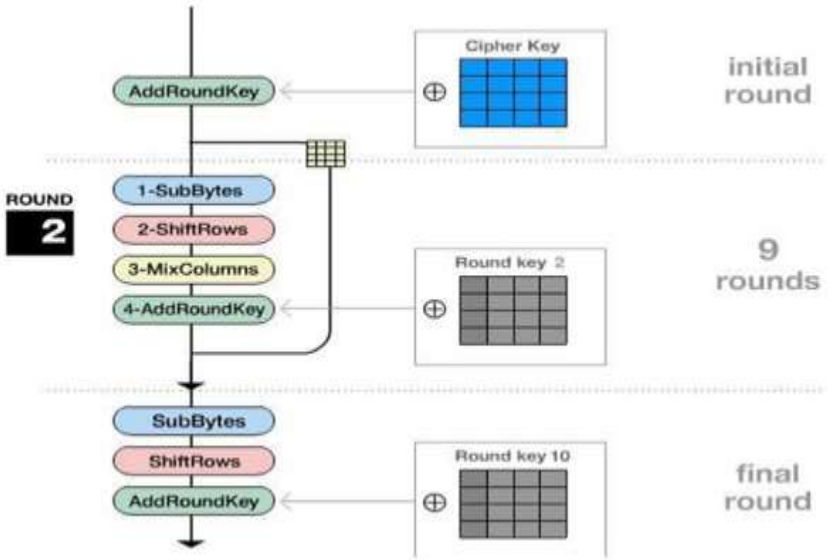
Pengelompokkan jenis AES ini adalah berdasarkan panjang kunci yang digunakan. Angka-angka di belakang kata AES menggambarkan panjang kunci yang digunakan pada tiap-tiap AES. Sedangkan perbedaanya akan ditunjukkan pada table berikut :

Tabel 2.1 Perbedaan AES

	Panjang Kunci	**Ukuran Blok**	**Jumlah putaran**
AES-128	4	4	10
AES-192	6	4	12
AES-256	8	4	14

Sumber : Rinaldi Munir, 2006

Berdasarkan ukuran blok yang tetap, AES bekerja pada matriks berukuran 4x4 di mana tiap-tiap sel matriks terdiri atas 1 *byte* (8 bit). Blok *cipher* tersebut ini akan diasumsikan sebagai sebuah kotak. Setiap *plaintext* akan dikonversikan terlebih dahulu ke dalam blok-blok tersebut dalam bentuk heksadesimal. Barulah kemudian blok itu akan diproses dengan metode yang akan dijelaskan. Secara umum metode yang digunakan dalam pemrosesan enkripsi dalam algoritma ini dapat dilihat melalui Gambar 2.1.



Gambar 2.1: Diagram AES

Sumber : Wikipedia

a. Enkripsi

Pada proses enkripsi dalam algoritma AES terdiri dari beberapa tahapan yaitu *SubBytes, ShiftRows, MixColumns,* dan *AddRoundKey*. Pada proses awal nput yang telah dikopikan kedalam sebuah *State* akan ditransformasikan dengan *AddRoundKey*. Setelah itu dilakukan transformasi *SubBytes, ShiftRows, MiCcolumns,* dan *AddRoundKey* secara berulang-ulang sebanyak Nr (Jumlah putaran). Dalam algoritma AES proses ini dinamakan *round function*. *Round* yang terakhir berbeda dengan *round-round* sebelumnya karena pada *round* yang terakhir *State* tidak mengalami transformasi *MixColumns.*

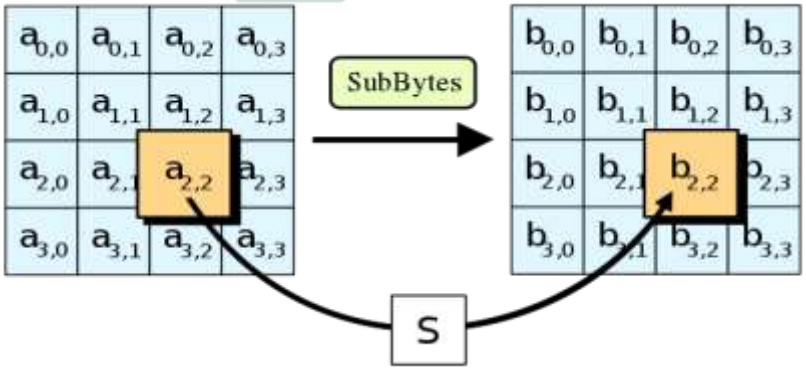
1) *SubBytes*

Pada transformasi *SubBytes* memetakan setiap *byte* dari *array State* dengan menggunakan tabel subtitusi S-Box. Berikut ini adalah tabel S-box :

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Gambar 2.2. Table S-Box.
Sumber : William Stalling, 2003

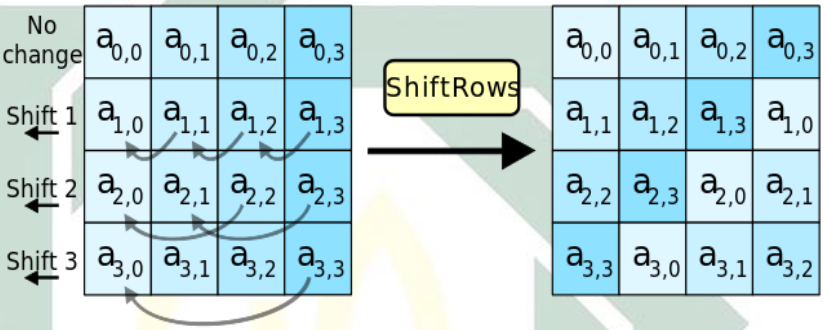
Pensubtitusiannya untuk setiap *byte* pada *array State* misalkan $S[r,c] = xy$ adalah digit heksadesimal dari nilai $S[r,c]$. Maka nilai subtitusi yang dinyatakan dengan $S[r,c]$ adalah elemen dalam S-Box yang merupakan perpotongan baris x dan kolom y. Berikut ini tampilan dari transformasi *SubBytes*.



Gambar 2.3 Transformasi *SubBytes*
Sumber : Rinaldi Munir, 2006

2) *ShiftRows*

Transformasi *ShiftRows* ini pada dasarnya adalah proses pergeseran bit, diana bit yang paling kiri dipindahkan menjadi bit yang paling kanan. Transformasi ini dilakukan pada baris ke 2,3 dan 4. Pada baris kedua bit digeser sebanyak satu kali. Pada baris ke 3 bit digeser dua kali, dan baris ke 4 digeser sebanyak tiga kali.



Gambar 2.4 Transformasi *ShiftRows*
Sumber : Rinaldi Munir, 2006

3) *MixColumns*

Pada tahap *MixColumns* mengoperasikan setiap elemen yang berada dalam satu kolom pada *State*. Elemen pada kolom dikalikan dengan suatu polynomial tetap

$$a(x) = (03)x^3 + (01)x^2 + (01)x^3 + (02) \qquad (2.13)$$

Lebih jelasnya untuk transformasi *MixColumns* dapat dilihat pada perkalian matriks berikut :

$$\begin{bmatrix} S'_{0,c} \\ S'_{1,c} \\ S'_{2,c} \\ S'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} S_{0,c} \\ S_{1,c} \\ S_{2,c} \\ S_{3,c} \end{bmatrix}$$

$$S'_{0,c} = \left((02) \times S_{0,c}\right) - \left((03) \times S_{1,c}\right) - S_{2,c} - S_{3,c}$$

$$S'_{1,c} = S_{0,c} - \left((02) \times S_{1,c}\right) - \left((03) \times S_{2,c}\right) - S_{3,c}$$

$$S'_{2,c} = S_{0,c} - S_{1,c} - \left((02) \times S_{2,c}\right) - \left((03) \times S_{3,c}\right)$$

$$S'_{3,c} = \left((03) \times S_{0,c}\right) - S_{1,c} - S_{2,c} - \left((02) \times S_{3,c}\right)$$

4) *AddRoundKey*

Proses ini *round key* ditambahkan pada *state* menggunakan operasi XOR. Setiap *round key* terdiri dari Nb *word* dimana tiap *word* tersebut kemudian dijumlahkan dengan *word* atau kolom yang bersesuaian dari *State*. Sehingga menjadi :

$$[S'_{0,c}, S'_{1,c}, S'_{2,c}, S'_{3,c}] = [S'_{0,c}, S'_{1,c}, S'_{2,c}, S'_{3,c}] - [w_{round*Nb+c}] \; Untuk \; 0 \leq c < Nb \quad (2.14)$$

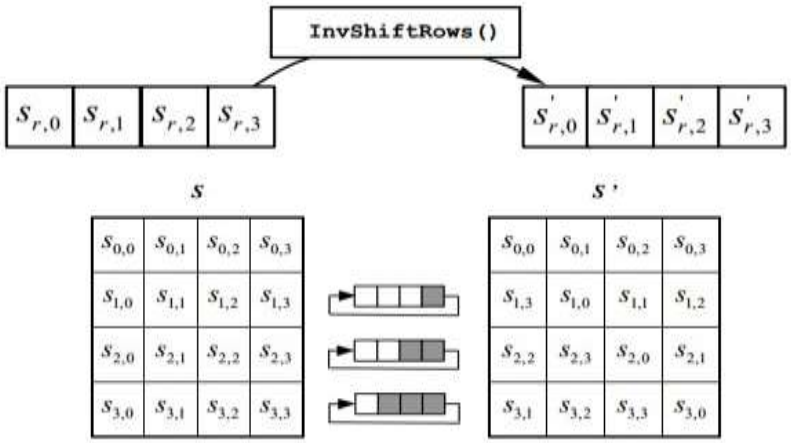
w_i adalah *word* dari *key* yang bersesuaian dimana i = $round * Nb + c$. Transfromasi *AddRoundKey* diimplementasikan pertama kali pada *round* = 0 dimana *key* yang digunakan adalah *initial key* yaitu kunci yang dimasukkan oleh kriptografer dan belum mengalami proses ekspansi kunci.

b. Dekripsi

Transformasi dari sebuah *cipher* bisa dibalikkan dan diimplementasikan pada arah yang berlawanan untuk menghasilkan *invers cipher* yang dapat dipahami untuk algoritma AES. Transformasi yang digunakan pada *invers cipher* adalah *invers* dari semua proses yaitu *InvShiftRows*, *InvSubBytes*, *InvMixColumns*, dan *AddRoundKey*.

1) *InvShiftRows*

InvShiftRows merupakan transformasi dari *byte* yang berkebalikan dengan transformasi *ShiftRows.* Kalau Pada transformasi *ShiftRows* pergeseran bit dilakukan ke kiri, maka pada *InvShiftRows* ini pergeserannya dilakukan ke kanan. Pada baris kedua pergeseran dilakukan sebanyak tiga kali, baris ketiga dilakukan persegeseran dua kali dan baris ke empat digeser satu kali.



Gambar 2.5 *Inverse ShiftRows*

Sumber : Wikipedia

2) *InvSubBytes*

InvSubBytes juga merupakan invers atau kebalikan dari *SubBytes*. Pada proses *InvSubBytes* ini setiap elemen yang ada pada *state* tersebut dipetakan dengan menggunakan table invers S-box. Berikut ini adalah tampilan dari tabel *inverse* S-Box:

		y														
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
	1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
	2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
	3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
	4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
	5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
	6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
	7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
	8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
	9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
	a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
	b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
	c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
	d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
	e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
	f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

Gambar 2.6 Tabel Invers S-Box
Sumber : William Stalling, 2003

Perkalian *invers* yang dilakukan pada transformasi *invSubBytes* ini sama dengan perkalian *invers* yang dilakukan pada transformasi *SubBytes.*

3) *InvMixColumns*

Pada proses *InvMixColumns*, kolom-kolom yang ada pada setiap *State* atau *word* tersebut akan dipandang sebagai polinom atas GF (2^8) dan mengalikan modulo $x^4 + 1$ dengan polinomiaal tetap $a^{-1}(x)$ yang diperoleh dari :

$$a^{-1}(x) = (0b)x^3 + (0d)x^2 + (09)x + (0e) \qquad (2.15)$$

atau dalam matriks :

$$S'(x) = a(x) * S(x) \qquad (2.16)$$

$$\begin{bmatrix} S'_{0,c} \\ S'_{1,c} \\ S'_{2,c} \\ S'_{3,c} \end{bmatrix} = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix} \begin{bmatrix} S_{0,c} \\ S_{1,c} \\ S_{2,c} \\ S_{3,c} \end{bmatrix}$$

$$S'_{0,c} = \left((02) \times S_{0,c}\right) - \left((03) \times S_{1,c}\right) - S_{2,c} - S_{3,c}$$

$$S'_{1,c} = S_{0,c} - \left((02) \times S_{1,c}\right) - \left((03) \times S_{2,c}\right) - S_{3,c}$$

$$S'_{2,c} = S_{0,c} - S_{1,c} - \left((02) \times S_{2,c}\right) - \left((03) \times S_{3,c}\right)$$

$$S'_{3,c} = \left((03) \times S_{0,c}\right) - S_{1,c} - S_{2,c} - \left((02) \times S_{3,c}\right)$$

4) *InvAddRoundKey*

Pada transfromasi *Inverse AddRoundKey* ini tidak memiliki perbedaan dengan transformasi *AddRoundKey* karena pada transformasi ini hanya dilakukan operasi penambahan sederhana dengan menggunakan operasi *bitwise* XOR.

c. Ekspansi kunci

Algoritma AES mengambil kunci *cipher*, k, dan melakukan ekspansi kunci secara rutin atau disebut dengan *key expansion*. Hasil dari ekspansi kunci satu digunakan untuk membentuk kunci selanjutnya dan terus berlanjut hingga selesai. Kunci ekspansi ini digunakan untuk membentuk *key schedule*. Tahap *key schedule* digunakan untuk membangun 10 sub kunci yang digunakan untuk setiap iterasi pada proses enkripsi dan dekripsi. Pada awal proses dimasukkan suatu kunci yang dinamakan *cipher key*. *Cipher key* ini bisa kita definisikan sendiri yang kemudian akan dilakukan pembangkitan atau ekspansi (Hanifah, 2012).

Ekspansi kunci akan menghasilkan total *Nb* (Nr+1) *word*. Pada algoritma ini membutuhkan *set* awal *key* yang terdiri dari *Nb word* dan

setiap *round Nr* membutuhkan data kunci sebanyak *Nb word*. Hasil dari *key schedule* terdiri dari *array byte word linear* yang dinotasikan dengan [W_i] (Hanifah, 2012).

Proses dalam ekspansi kunci ini melalui beberapa tahapan, diantaranya :

1. *RotWord*

   Pada operasi ini dilakukan rotasi pada *word* yaitu operasi perputaran 8 bit dari 32 bit W_{i-1} dengan cara menggeser kolom secara siklis ke atas (Hanifah, 2012).

2. *SubWord*

   SubWord adalah fungsi yang mengambil 4 *byte word input* dan mengaplikasikan S-box ke setiap data 4 *byte* untuk menghasilkan *word output*. Fungi *RotWord* mengambil *word* $[a_0, a_1, a_2, a_3]$ sebagai input kemudian dilakukan pergeseran siklik dengan mengembalikan *word* $[a_1, a_2, a_3, a_0]$ *rcon*[i] terdiri dari nilai-nilai yang diberikan oleh [x^{i-1}, (00), (00),(00)] dengan x^{i-1} adalah pangkat dari x (dinotasikan sebagai (02) dalam *field* GF(2^8). Operasi ini mensubtitusikan 8 bit pada hasil *RotWord* dengan nilai pada tabel s-box. Proses ini sama seperti pada proses *SubByte* pada tahap enkripsi (Hanifah, 2012).

3. XOR

   Hasil dari proses sub*word* kemudian dilakukan XOR dengan *R-con* yang bersesuaian dan setiap *round*nya. *R-con* ini sudah didefinisikan oleh AES sebelumnya (Hanifah, 2012).

D. Citra Digital

Citra merupakan salah satu komponen multimedia yang memiliki peranan penting sebagai bentuk informasi visual (Hanifah, 2012). Citra memiliki karakteristik yang berbeda dengan data teks, dalam citra lebih kaya akan informasi dibanding dengan data teks. Dengan kata lain data citra atau *image* dapat memberikan informasi yang lebih banyak dari pada teks karena informasi itu sendiri disajikan dalam bentuk teks (Hanifah, 2012).

Secara umum, citra merupakan sebuah gambar yang berada pada bidang dua dimensi. Citra merupakan fungsi kontinu dari intentitas cahaya pada bidang dua dimensi yang disimbolkan dengan $f(x, y)$, dalam hal ini (x, y) merupakan kordinat pada bidang dua dimensi dan $f(x, y)$ merupakan intentitas cahaya pada titik (x, y) (Munir, 2006).

Citra digital tersusun dari sejumlah nilai tingkat keabuan yaitu piksel pada posisi tertentu. Piksel adalah elemen terkecil dari sebuah citra digital. Piksel mempunyai dua parameter yaitu titik koordinat dan warna atau intensitas. Koordinat adalah lokasi suatu piksel dari citra digital. Sedangkan warna adalah intensitas cahaya yang dipantulkan oleh suatu obyek.

Citra dibagi menjadi tiga jenis antara lain yaitu citra biner, citra *greyscale*, dan citra warna. Citra biner merupakan citra yang hanya tersusun dari dua warna

dan tiap pikselnya direpresentasikan dalam bit 0 dan 1 yang mana bit 0 berarti hitam dan 1 berarti putih.

Tingkat ketajaman atau resolusi warna pada citra digital tergantung pada jumlah bit yang digunakan oleh computer untuk mempresentasikan setiap piksel tersebut (Hanifah, 2012). Tingkat intensitas *greyscale* adalah 256 dimana 0 menyatakan warna hitam dan 255 menyatakan warna putih.

Citra digital pada umumnya berukuran segi empat yang dinyatakan dengan dimensi ukuran panjang(m) x lebar(n). Ukuran dimensi inilah representasi citra dalam bentuk matriks ukuran m x n piksel. Seperti yang di sajikan dalam fungsi berikut ini :

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,m-1) \\ f(1,0) & f(1,2) & \cdots & f(1,m-1) \\ \vdots & \vdots & \vdots & \vdots \\ f(n-1,0) & f(n-1,1) & \cdots & f(n-1,m-1) \end{bmatrix}$$

BAB III

METODE PENELITIAN

A. Metode Pengumpulan Data

Untuk memenuhi kebutuhan data yang diperlukan dalam penelitian ini, maka pencarian data dan informasi sebagai referensi yang tepat untuk mendukung kebenaran dari uraian materi dan teori serta pembahasan. Sedangkan data yang digunakan sebagai pengujian kali ini adalah file citra lena. Adapun metode pengumpulan data yang digunakan adalah :

1. Studi Pustaka

Studi pustaka merupaka metode pengumpulan data dan informasi dengan mencari dan memperoleh data yang diperlukan melalui berbagai media baik buku, *e-book, website,* serta sumber lainnya yang berkaitan dengan penelitian. Adapun sumber buku-buku, jurnal, dan *website* yang digunakan dalam penulisan proposal skripsi ini dapat dilihat secara lengkap dalam halaman daftar pustaka.

2. Studi Literatur

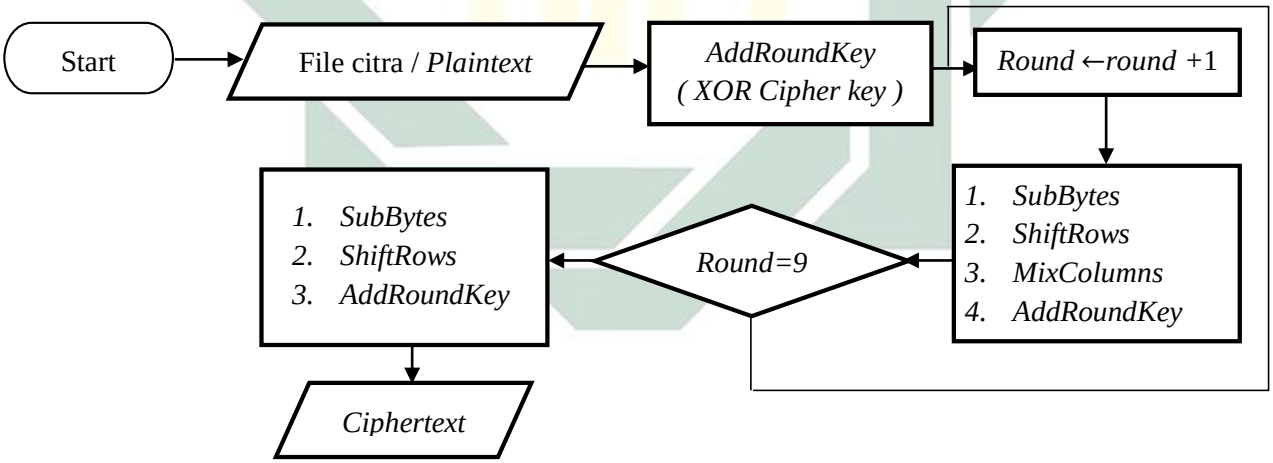
Studi literatur adalah pengembangan instrumen atas penelitian sejenis yang telah dibuat sebelumnya, hal ini dilakukan dengan melihat hasil riset atau penelitian yang sudah ada lalu dikembangkan dari kekurangan-kekurangan yang ada pada hasil riset tersebut.

Studi literatur yang penulis gunakan dalam penelitian ini sehingga berbeda dengan penelitian sebelumnya yaitu pada aplikasi enkripsi dan

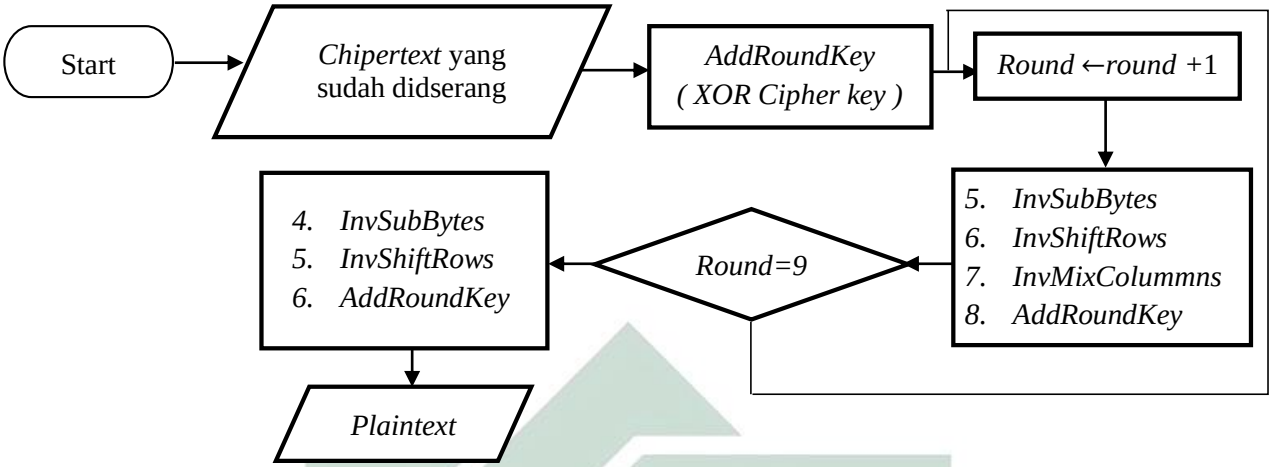
dekripsi data citra atau *image* penulis mengggunakan metode AES dengan pengujian hasil enkripsi terhadap serangan *crop, blur*, dan *enhancement.*

B. Rancangan sistem

Metode pengenkripsian dan pendeskripsian pada data *image* ini menggunakan metode *Advanced Encryption Standart (AES).* Data citra atau *image* akan dienkripsikan terlebih dahulu, kemudian hasil enkripsi tersebut akan dilakukan pengujian terhadap beberapa serangan seperti *crop, blur dan enhancement*. Kemudian dilakukan pendekripsian citra atau *image* yang telah diberi serangan tersebut, apabila citra atau *image* masih bisa dikenali maka metode yang digunakan yaitu AES tahan terhadap serangan. Berikut diagram alir rancangan sistem untuk proses enkripsi dan dekripsi :



Gambar 3.1 Diagram Alir proses enkripsi



Gambar 3.2 Diagram Alir proses dekripsi

BAB IV

HASIL DAN PEMBAHASAN

Dalam penelitian ini akan mengimplementasikan algoritma *Advanced Encryption Standard* (AES) atau Rijndael untuk perancangan aplikasi penyandian menggunakan software *matlab* dan data berupa citra atau *image* sehingga menjadi sebuah aplikasi penyandian citra**.**

Pada bab sebelumnya telah dibahas secara singkat proses pengenkripsian dan pendekripsian dalam penyandian menggunakan metode AES. Pada bab ini akan dijelaskan mengenai perancangan dan setiap tahapannya. Tahap yang pertama dalam perancangan aplikasi tersebut adalah membangkitkan kunci terlebih dahulu kemudian dilakukan tahapan enkripsi dan dekripsi. Untuk lebih jelasnya akan diuraikan dalam setiap tahapan berikut:

A. Key schedule

Key schedule merupakan proses untuk membangkitkan kunci yang akan digunakan dalam proses enkripsi dan dekripsi. Seperti yang dijelaskan pada bab sebelumnya pembentukan kunci ini terdiri dari beberapa tahapan yaitu *RotWord*, *SubWord*, XOR dengan nilai R-con, dan XOR dengan *word* sebelumnya. Pada perancangan system ini cipher key diinputkan sendiri oleh pembuat aplikasi. *Cipher key* yang diinputkan selanjutnya dirubah menjadi bilangan ASCII. Apabila *cipher* yang diinputkan lebih dari 16 *byte*, maka yang digunakan adalah 16 *byte* pertama, namun jika *cipher* yang dimasukkan kurang dari 16 *byte* maka *cipher* akan digenapkan menjadi 16 *byte* dengan

penambahan bilangan 0. Selanjutnya bilangan ASCII dari *cipher key* tersebut dirubah menjadi bilangan heksadesimal dan direpresentasikan dalam matriks yang berukuran 4 x 4.

Input cipher key: "prodi matematika"

Setiap inputan yang dimasukkan akan dirubah menjadi bilangan ASCII. Sehingga bilangan ASCII dari inputan *cipher key* tersebut adalah :

110 172 73 99 62 176 85 60 90 145 103 115 82 168 96 79

Bilangan ASCII tersebut selanjutnya dirubah ke dalam heksadesimal dan direpresentasikan ke dalam matriks berukuran 4 x 4. Sehingga menghasilkan *cipher key* sebagai berikut :

$$Cipher\ key : \begin{bmatrix} 6e & 3e & 5a & 52 \\ ac & b0 & 91 & a8 \\ 49 & 55 & 67 & 60 \\ 63 & 3c & 73 & 4f \end{bmatrix}$$

Tahapan selanjutnya adalah *RotWord*

Tahap ini digunakan untuk menghasilkan kolom pertama pada sub kunci pertama (*Wi*). Yang dilakukan pada tahap ini adalah menggeser setiap *byte* pada kolom terakhir dari *cipher key* secara siklis ke atas satu kali.

$$RotWord : \begin{bmatrix} 52 \\ a8 \\ 60 \\ 4f \end{bmatrix} \rightarrow \begin{bmatrix} a8 \\ 60 \\ 4f \\ 52 \end{bmatrix}$$

Tahap sub*word*

Hasil dari *RotWord* pada tahap ini kemudian di lakukan pensubtitusian dengan tabel S-Box yang sudah ditetapkan.

$$\text{Sub}word : \begin{bmatrix} a8 \\ 60 \\ 4f \\ 52 \end{bmatrix} \rightarrow (S - Box) \rightarrow \begin{bmatrix} c2 \\ d0 \\ 84 \\ 00 \end{bmatrix}$$

Tahap terakhir untuk mendapatkan kunci kolom ke (W_i) yaitu proses XOR yang dilakukan terhadap hasil sub*word* dengan nilai *R-con* yang bersesuaian, lalu XOR lagi dengan kolom (W_{i-4}).

Pada tahap ini bilangan heksadesimal dirubah dulu menjadi bilangan biner untuk dapat dilakukan operasi XOR. Prosesnya adalah sebagai berikut:

$$\begin{bmatrix} c2 \\ d0 \\ 84 \\ 00 \end{bmatrix} \oplus \begin{bmatrix} 01 \\ 00 \\ 00 \\ 00 \end{bmatrix} \oplus \begin{bmatrix} 6e \\ ac \\ 49 \\ 63 \end{bmatrix} =$$

Atau dalam biner

$$\begin{bmatrix} 11000010 \\ 11010000 \\ 10000100 \\ 00000000 \end{bmatrix} \oplus \begin{bmatrix} 00000001 \\ 00000000 \\ 00000000 \\ 00000000 \end{bmatrix} \oplus \begin{bmatrix} 01101110 \\ 10101100 \\ 01001001 \\ 01100011 \end{bmatrix} = \begin{bmatrix} 10101101 \\ 01111100 \\ 11001101 \\ 01100011 \end{bmatrix}$$

Hasil dari operasi biner $\begin{bmatrix} 10101101 \\ 01111100 \\ 11001101 \\ 01100011 \end{bmatrix}$ dalam heksadesimalnya adalah $\begin{bmatrix} ad \\ 7c \\ cd \\ 63 \end{bmatrix}$

Jadi hasil untuk sub kunci pertama kolom pertama adalah = $\begin{bmatrix} ad \\ 7c \\ cd \\ 63 \end{bmatrix}$

Selanjutnya untuk mendapatkan sub kunci pertama kolom kedua hingga keempat dilakukan operasi XOR antara W_i dengan kolom W_{i-3}. Begitu pula untuk mendapatkan kolom ketiga dan keempat pada sub kunci pertama.

Kolom kedua sub kunci pertama (W_{i+1}) = $\begin{bmatrix} ad \\ 7c \\ cd \\ 63 \end{bmatrix} \oplus \begin{bmatrix} 3e \\ b0 \\ 55 \\ 3c \end{bmatrix} = \begin{bmatrix} 93 \\ cc \\ 98 \\ 5f \end{bmatrix}$

Kolom ketiga sub kunci pertama $(W_{i+2}) = \begin{bmatrix} 93 \\ cc \\ 98 \\ 5f \end{bmatrix} \oplus \begin{bmatrix} 5a \\ 91 \\ 67 \\ 73 \end{bmatrix} = \begin{bmatrix} c9 \\ 5d \\ fb \\ 6c \end{bmatrix}$

Kolom keempat sub kunci pertama $(W_{i+3}) = \begin{bmatrix} c9 \\ 5d \\ fb \\ 6c \end{bmatrix} \oplus \begin{bmatrix} 52 \\ a8 \\ 60 \\ 4f \end{bmatrix} = \begin{bmatrix} 9b \\ f5 \\ 9b \\ 23 \end{bmatrix}$

Sehingga didapatkan sub kunci pertama yaitu $\begin{bmatrix} ad & 93 & c9 & 9b \\ 7c & cc & 5d & f5 \\ cd & 98 & fb & 9b \\ 63 & 5f & 6c & 23 \end{bmatrix}$

Proses-proses di atas tersebut diulang sebanyak 10 iterasi hingga menghasilkan 10 sub kunci yang digunakan untuk proses enkripsi dan dekripsi.

B. Enkripsi

Pada bab sebelumnya telah digambarkan alur untuk proses enkripsi ini diawali dengan proses *AddRoundKey* yang selanjutnya diikuti dengan proses *SubBytes, ShiftRows, MixColumns,* dan *AddRoundKey*. Proses tersebut dilakukan iterasi sampai 10 kali. Dan proses terakhir yaitu yang ke-10 tanpa *MixColumns*. Proses iterasi yang dilakukan pada enkripsi ini juga diiringi dengan proses *key schedule* untuk setiap iterasinya.

Input yang dimasukkan berupa citra *greyscale*, lalu direpresentasikan dalam bentuk matriks sesuai dengan ukuran piksel dari citra tersebut. Algoritma AES ini beroperasi menggunakan blok *cipher* sehingga matriks dari citra tersebut akan dipartisi menjadi blok-blok yang berukuran sesuai dengan jenis AES yang digunakan.

Pada perancangan system ini menggunakan AES 128 sehingga matriks nilai piksel dari citra yang diinputkan di partisi menjadi blok 4 x 4 dan direpresentasikan dalam bilangan heksadesimal. Namun jika pada partisi terakhir elemennya tidak mencapai 16 *byte*, maka akan ditambahkan dummy berupa elemen 0 sampai berjumlah 16 *byte*. Berikut proses manual dari algoritma AES:

Yang pertama inputkan citra digital yang akan digunakan berukuran 32 x 32. Citra yang menjadi inputan ini boleh citra apapun asalkan berukuran sama dengan batasan yang ditetapkan.

Gambar 4.1 Citra Lena
Sumber : Wikipedia

Matriks nilai piksel dari citra tersebut sebagai berikut:

103	105	176	162	70	97	94	111	195	120	115	136	97	…	129
102	98	172	161	70	97	99	96	189	110	137	84	59	…	123
102	97	173	163	75	103	106	116	136	126	75	49	47	…	156
98	97	174	163	76	101	99	129	102	105	68	51	53	…	189
93	87	182	160	76	102	99	172	72	69	72	54	49	…	30
100	77	174	169	79	106	126	65	83	73	98	38	39	…	60
78	64	184	170	79	129	81	86	57	62	93	38	135	…	79
80	48	172	164	89	104	88	106	58	54	102	57	157	…	46
77	77	176	173	79	95	77	91	91	57	78	149	50	…	50
62	82	176	168	85	96	60	79	101	95	165	72	49	…	209
49	88	181	172	95	93	52	113	85	158	119	36	55	…	47
.	.	.	.	.	.	.	.	.	.	.	.	.	.	
.	.	.	.	.	.	.	.	.	.	.	.	.	.	
80	192.	72	155	237	234	250	210	200	32	40	57	87	102	45

Matriks diatas merupakan matriks dari citra tersebut yang di kurangi elemennya. *Plaintext* asli berukuran 32 x 32. Matriks tersebut kemudian dibentuk menjadi *array* terlebih dahulu untuk memudahkan partisinya. Dari *array* tersebut dipartisi menjadi blok 16 *byte* atau panjang 128 bit. Dalam pemartisiannya diambil 4 byte pertama menjadi kolom pertama, 4 *byte* kedua menjadi kolom kedua , begitupun seterusnya. Matriks dengan 4 kolom pertama akan menjadi *State* pertama hingga didapatkan *State* yang terakhir yaitu *State* 64. *State-state* tersebut dalam bentuk matriks kemudian direpresentasikan ke dalam bilangan heksadesimal.

State input : $\begin{bmatrix} 103 & 105 & 176 & 162 \\ 102 & 98 & 172 & 161 \\ 102 & 97 & 173 & 163 \\ 98 & 97 & 174 & 163 \end{bmatrix}$

State tersebut kemudian dirubah ke dalam bilangan heksadesimal sehingga menjadi : $\begin{bmatrix} 67 & 69 & b0 & a2 \\ 66 & 62 & ac & a1 \\ 66 & 61 & ad & a3 \\ 62 & 61 & ae & a3 \end{bmatrix}$

Transformasi *AddRoundKey*

State input di operasikan dengan *cipher key* menggunakan XOR. Jadi dirubah menjadi bilangan biner dahulu untuk dapat dilakukan operasi XOR.

Kolom pertama $\begin{bmatrix} 67 \\ 66 \\ 66 \\ 62 \end{bmatrix} \oplus \begin{bmatrix} 6e \\ ac \\ 49 \\ 63 \end{bmatrix} = \begin{bmatrix} 09 \\ ca \\ 2f \\ 01 \end{bmatrix}$

$$\begin{bmatrix} 01100111 \\ 01100110 \\ 01100110 \\ 01100010 \end{bmatrix} \oplus \begin{bmatrix} 01101110 \\ 10101100 \\ 01001001 \\ 01100011 \end{bmatrix} = \begin{bmatrix} 00001001 \\ 11001010 \\ 00101111 \\ 00000001 \end{bmatrix}$$

Kolom kedua $\begin{bmatrix} 69 \\ 62 \\ 61 \\ 61 \end{bmatrix} \oplus \begin{bmatrix} 3e \\ b0 \\ 55 \\ 3c \end{bmatrix} = \begin{bmatrix} 57 \\ d2 \\ 34 \\ 5d \end{bmatrix}$

$$\begin{bmatrix} 01101001 \\ 01100010 \\ 01100001 \\ 01100001 \end{bmatrix} \oplus \begin{bmatrix} 00111110 \\ 10110000 \\ 01010101 \\ 00111100 \end{bmatrix} = \begin{bmatrix} 01010111 \\ 11010010 \\ 00110100 \\ 01011101 \end{bmatrix}$$

Kolom ketiga $\begin{bmatrix} b0 \\ ac \\ ad \\ ae \end{bmatrix} \oplus \begin{bmatrix} 5a \\ 91 \\ 67 \\ 73 \end{bmatrix} = \begin{bmatrix} ea \\ 38 \\ ea \\ dd \end{bmatrix}$

$$\begin{bmatrix} 10110000 \\ 10101100 \\ 10101101 \\ 10101110 \end{bmatrix} \oplus \begin{bmatrix} 010110101 \\ 10010001 \\ 01100111 \\ 01110011 \end{bmatrix} = \begin{bmatrix} 11101010 \\ 00111000 \\ 11101010 \\ 11011101 \end{bmatrix}$$

Kolom keempat $\begin{bmatrix} a2 \\ a1 \\ a3 \\ a3 \end{bmatrix} \oplus \begin{bmatrix} 52 \\ a8 \\ 60 \\ 4f \end{bmatrix} = \begin{bmatrix} f0 \\ 09 \\ c3 \\ ec \end{bmatrix}$

$$\begin{bmatrix} 10100010 \\ 10100001 \\ 10100011 \\ 10100011 \end{bmatrix} \oplus \begin{bmatrix} 01010010 \\ 10101000 \\ 01100000 \\ 01001111 \end{bmatrix} = \begin{bmatrix} 11110000 \\ 00001001 \\ 11000011 \\ 11101100 \end{bmatrix}$$

Jadi hasil dari transformasi *AddRoundKey* adalah $\begin{bmatrix} 09 & 57 & ea & f0 \\ ca & d2 & 38 & 09 \\ 2f & 34 & ea & c3 \\ 01 & 5d & dd & ec \end{bmatrix}$

Berikut adalah *source code* yang dipakai untuk proses *AddRoundKey*.

```
function state_out = add_round_key (state_in, round_key)
%UNTITLED8 Summary of this function goes here
%   Detailed explanation goes here
state_out = bitxor (state_in, round_key);
end
```

Setelah proses tranformasi *AddRoundKey* tahapan selanjutnya memasuki tahapan iterasi dengan 9 kali iterasi pertama yang terdiri dari

SubBytes, *ShiftRows*, *MixColumns*, dan *AddRoundKey* sedangkan iterasi ke-10 tanpa *MixColumns*.

Transformasi *SubBytes* :

$$\begin{bmatrix} 09 & 57 & ea & f0 \\ ca & d2 & 38 & 09 \\ 2f & 34 & ea & c3 \\ 01 & 5d & dd & ec \end{bmatrix} \rightarrow (S-Box) \rightarrow \begin{bmatrix} 01 & 5b & 87 & 8c \\ 74 & b5 & 07 & 01 \\ 79 & 18 & 87 & 2e \\ 7c & 4c & c1 & ce \end{bmatrix}$$

Transformasi *ShiftRows* :

$$\begin{bmatrix} 01 & 5b & 87 & 8c \\ 74 & b5 & 07 & 01 \\ 79 & 18 & 87 & 2e \\ 7c & 4c & c1 & ce \end{bmatrix} \rightarrow \begin{bmatrix} 01 & 5b & 87 & 8c \\ b5 & 07 & 01 & 74 \\ 87 & 2e & 79 & 18 \\ ce & 7c & 4c & c1 \end{bmatrix}$$

Tranformasi *MixColumns* :

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} 01 \\ b5 \\ 87 \\ ce \end{bmatrix} = \begin{bmatrix} b_{1,1} \\ b_{2,1} \\ b_{3,1} \\ b_{4,1} \end{bmatrix} = \begin{bmatrix} 8d \\ 2c \\ a8 \\ bd \end{bmatrix}$$

Pada proses transformasi *MixColumns* ini menggunakan polinomial modulo 8. Dalam bab 2 telah dijelaskan mengenai perkalian modulo polinomial yaitu *Galois Field* 2^8 atau yang disebut dengan $GF(2^8)$. Proses untuk transformasinya adalah sebagai berikut :

$$b_{1,1} = 02 \cdot 01 \oplus 03 \cdot b5 \oplus 01 \cdot 87 \oplus 01 \cdot ce$$

$$= [00000010] \cdot [00000001] \oplus [00000011] \cdot [10110101]$$

$$\oplus [00000001] \cdot [10000111] \oplus [00000001] \cdot [11001110]$$

$$= x(1)\ mod\ (x^8 + x^4 + x^3 + x + 1) +$$

$$(x+1)(x^7 + x^5 + x^4 + x^2 + 1)\ mod\ (x^8 + x^4 + x^3 + x + 1) +$$

$$(x^7 + x^2 + x + 1)\ mod (x^8 + x^4 + x^3 + x + 1) +$$

$$(x^7 + x^6 + x^3 + x^2 + x)\ mod\ (x^8 + x^4 + x^3 + x + 1)$$

$$= x(1)\ mod\ (x^8 + x^4 + x^3 + x + 1) +$$
$$(x^8 + x^6 + x^5 + x^3 + x + x^7 + x^5 + x^4 + x^2 + 1)\ mod$$
$$(x^8 + x^4 + x^3 + x + 1) +$$
$$(x^7 + x^2 + x + 1)\ mod(x^8 + x^4 + x^3 + x + 1) +$$
$$(x^7 + x^6 + x^3 + x^2 + x)\ mod\ (x^8 + x^4 + x^3 + x + 1)$$
$$= (x^8 + x^4 + x^3 + 1) + (x^8 + x^7 + x^6 + x^4 + x^2 + 1) +$$
$$(x^8 + x^7 + x^4 + x^3 + x^2) + (x^8 + x^7 + x^6 + x^4 + x^2 + 1)$$
$$= (x^7 + x^6 + x^2 + x) + (x^8 + x^7 + x^4 + x^3 + x^2) +$$
$$(x^8 + x^7 + x^6 + x^4 + x^2 + 1)$$
$$= (x^8 + x^6 + x^4 + x^3) + (x^8 + x^7 + x^6 + x^4 + x^2 + 1)$$
$$= (x^7 + x^3 + x^2 + 1)$$
$$= 10001101$$
$$= 8d$$

$$b_{2,1} = 01 \cdot 01 \oplus 02 \cdot b5 \oplus 03 \cdot 87 \oplus 01 \cdot ce$$
$$= [00000001] \cdot [00000001] \oplus [00000010] \cdot [10110101] \oplus$$
$$[00000011] \cdot [10000111] \oplus [00000001] \cdot [11001110]$$
$$= (1)\ mod\ (x^8 + x^4 + x^3 + x + 1) +$$
$$(x)(x^7 + x^5 + x^4 + x^2 + 1)\ mod\ (x^8 + x^4 + x^3 + x + 1) +$$
$$(x + 1)(x^7 + x^2 + x + 1)\ mod(x^8 + x^4 + x^3 + x + 1) +$$
$$(1)(x^7 + x^6 + x^3 + x^2 + x)\ mod\ (x^8 + x^4 + x^3 + x + 1)$$
$$= (1)\ mod\ (x^8 + x^4 + x^3 + x + 1) +$$
$$(x^8 + x^6 + x^5 + x^3 + 1)\ mod\ (x^8 + x^4 + x^3 + x + 1) +$$

$$(x^8 + x^3 + x^2 + x + x^7 + x^2 + x + 1)\ mod$$

$$(x^8 + x^4 + x^3 + x + 1) +$$

$$(x^7 + x^6 + x^3 + x^2 + x)\ mod\ (x^8 + x^4 + x^3 + x + 1)$$

$$= (x^8 + x^4 + x^3 + 1) + (x^6 + x^5 + x^4 + 1) + (x^7 + x^4 + x) +$$

$$(x^8 + x^7 + x^6 + x^4 + x^2 + 1)$$

$$= (x^8 + x^6 + x^5 + x^3 + x + 1) + (x^7 + x^4 + x) +$$

$$(x^8 + x^7 + x^6 + x^4 + x^2 + 1)$$

$$= (x^8 + x^7 + x^6 + x^5 + x^4 + x^3 + 1) +$$

$$(x^8 + x^7 + x^6 + x^4 + x^2 + 1)$$

$$= (x^5 + x^3 + x^2)$$

$$= 00101100$$

$$= 2c$$

$$b_{3,1} = 01 \cdot 01 \oplus 01 \cdot b5 \oplus 02 \cdot 87 \oplus 03 \cdot ce$$

$$= [00000001] \cdot [00000001] \oplus [00000001] \cdot [10110101] \oplus$$

$$[00000010] \cdot [10000111] \oplus [00000011] \cdot [11001110]$$

$$= (1)\ mod\ (x^8 + x^4 + x^3 + x + 1) +$$

$$(1)(x^7 + x^5 + x^4 + x^2 + 1)\ mod\ (x^8 + x^4 + x^3 + x + 1) +$$

$$(x)(x^7 + x^2 + x + 1)\ mod(x^8 + x^4 + x^3 + x + 1) +$$

$$(x + 1)(x^7 + x^6 + x^3 + x^2 + x)\ mod\ (x^8 + x^4 + x^3 + x + 1)$$

$$= x(1)\ mod\ (x^8 + x^4 + x^3 + x + 1) +$$

$$(x^7 + x^5 + x^4 + x^2 + 1)\ mod\ (x^8 + x^4 + x^3 + x + 1) +$$

$$(x^8 + x^3 + x^2 + x)\ mod(x^8 + x^4 + x^3 + x + 1) +$$

$$(x^8 + x^7 + x^4 + x^3 + x^2 + x^7 + x^6 + x^3 + x^2 + x)\ mod$$

$$(x^8 + x^4 + x^3 + x + 1)$$

$$= (x^8 + x^4 + x^3 + x) + (x^8 + x^7 + x^5 + x^3 + x^2 + x) +$$

$$(x^4 + x^2 + 1) + (x^3 + 1)$$

$$= (x^7 + x^5 + x^4 + x^2) + (x^4 + x^2 + 1) + (x^3 + 1)$$

$$= (x^7 + x^5 + 1) + (x^3 + 1)$$

$$= (x^7 + x^5 + x^3)$$

$$= 10101000$$

$$= a8$$

$$b_{4,1} = 03 \cdot 01 \oplus 01 \cdot b5 \oplus 01 \cdot 87 \oplus 02 \cdot ce$$

$$= [00000011] \cdot [00000001] \oplus [00000011] \cdot [10110101] \oplus$$

$$[00000001] \cdot [10000111] \oplus [00000010] \cdot [11001110]$$

$$= (x + 1)(1)\ mod\ (x^8 + x^4 + x^3 + x + 1) +$$

$$(x^7 + x^5 + x^4 + x^2 + 1)\ mod\ (x^8 + x^4 + x^3 + x + 1) +$$

$$(x^7 + x^2 + x + 1)\ mod (x^8 + x^4 + x^3 + x + 1) +$$

$$(x)(x^7 + x^6 + x^3 + x^2 + x)\ mod\ (x^8 + x^4 + x^3 + x + 1)$$

$$= (x + 1)\ mod\ (x^8 + x^4 + x^3 + x + 1) +$$

$$(x^7 + x^5 + x^4 + x^2 + 1)\ mod\ (x^8 + x^4 + x^3 + x + 1) +$$

$$(x^7 + x^2 + x + 1)\ mod (x^8 + x^4 + x^3 + x + 1) +$$

$$(x^8 + x^7 + x^4 + x^3 + x^2)\ mod\ (x^8 + x^4 + x^3 + x + 1)$$

$$= (x^8 + x^4 + x^3) + (x^8 + x^7 + x^5 + x^4 + x^3 + x^2 + x) +$$

$$(x^8 + x^7 + x^4 + x^3 + x^2) + (x^7 + x^2 + x + 1)$$

$$= (x^7 + x^5 + x^2 + x) + (x^8 + x^7 + x^4 + x^3 + x^2) + (x^7 + x^2 + x + 1)$$

$$= (x^8 + x^5 + x^4 + x^3) + (x^7 + x^2 + x + 1)$$

$$= (x^8 + x^7 + x^5 + x^4 + x^3 + x^2 + 1$$

$$= 10111101$$

$$= bd$$

Kolom kedua transformasi *MixColumns* $\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} 5b \\ 07 \\ 2e \\ 7c \end{bmatrix} = \begin{bmatrix} 7c \\ 4b \\ d6 \\ 13 \end{bmatrix}$

Kolom ketiga transformasi *MixColumns* $\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} 87 \\ 01 \\ 79 \\ 4c \end{bmatrix} = \begin{bmatrix} 7c \\ 4b \\ d6 \\ 13 \end{bmatrix}$

Kolom keempat transformasi *MixColumns* $\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} 8c \\ 74 \\ 18 \\ c1 \end{bmatrix} = \begin{bmatrix} 7c \\ 4b \\ d6 \\ 13 \end{bmatrix}$

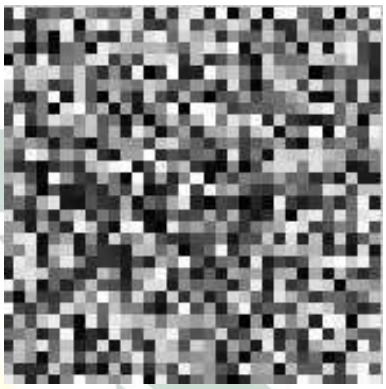
Setelah dilakukan transformasi *MixColumns*, terdapat transformasi *AddRoundKey* dengan langkah yang sama seperti yang dijelaskan sebelumnya, hanya saja pada proses XORnya dengan sub kunci yang bersesuaian untuk setiap iterasinya. Setelah melakukan 10 iterasi dengan iterasi yang terakhir tanpa proses transformasi *MixColumns* akan didapatkan *state output ciphertext* 16 byte yang pertama yaitu :

$$\begin{bmatrix} 51 & 68 & c9 & c0 \\ 92 & 0c & 31 & b9 \\ 6e & 6d & a7 & 63 \\ 7f & af & c0 & 78 \end{bmatrix}$$

Setelah itu dilakukan pula untuk semua *state* lainnya hingga didapatkan keseluruhan *state* untuk citra *ciphertext*, citra inilah yang merupakan citra yang sudah dirahasiakan.

Berikut ini adalah percobaan enkripsi dari citra

Gambar 4.2 Citra sebelum dienkripsi



Gambar 4.3 Citra Lena sudah di enkripsi

Source code yang digunakan dalam matlab untuk proses pengenkripsian ini penulis menggunakan banyak *function* dimana setiap *function* merupakan satu proses yang digunakan untuk satu proses hitung. *Function* lengkap akan disajikan dalam lampiran. Diantara adalah sebagai berikut:

```
function disp_hex (string, hex_array)
%UNTITLED4 Summary of this function goes here
%   Detailed explanation goes here
%cari nomor dari baris dan kolom array yang ditampilkan
[n_hex_array, m_hex_array] = size (hex_array);
%cari panjang title string
n_string = length (string);
%buat string kosong yang panjangnya sam dengan title
empty_string = ones (1, n_string)*' ';
%ulangi pada setiap baris di array
for i = 1 : n_hex_array
    if i == 1
```

```
            line = string;
        else
            line = empty_string;
        end

        %ulangi pada setiap kolom di array
        for j = 1 : m_hex_array
            line = [line, lower(dec2hex(hex_array(i,j),2)),
' '];
        end
        %tampilkan hasil
        disp (line)
    end

    %tampilkan spasi
    disp (' ')

    end

    function b_inv = find_inverse (b_in, mod_pol)
    %UNTITLED7 Summary of this function goes here
    %   Detailed explanation goes here

    for i = 1:255
        prod = poly_mult (b_in, i, mod_pol);
        if prod == 1
            b_inv = i;
            break
        end
    end
    end

    function state_out = mix_columns (state_in, poly_mat)
    %UNTITLED10 Summary of this function goes here
    %   Detailed explanation goes here
    %digunakan di modulo operatin pada poly mat
```

```
mod_pol = bin2dec ('100011011');
%ulanggi setiap kolom di state matrik
for i_col_state = 1:4
    %ulanggi setiap kolom di state matrik
    for i_row_state = 1:4
        temp_state = 0;
        for i_inner = 1:4
            temp_prod = poly_mult (poly_mat(i_row_state,
i_inner), state_in (i_inner, i_col_state), mod_pol);
            temp_state = bitxor (temp_state, temp_prod);
        end
        state_out(i_row_state, i_col_state) =
temp_state;
    end
end
```

sedangkan untuk *script* program AESnya adalah sebagai berikut :

```
% Initialization
[s_box, inv_s_box, w, poly_mat, inv_poly_mat] =
aes_init;

% in the AES-Specification (draft)
 plaintext_hex = { '9d' 'a3' '5d' '69' '5d' 'b6' '4c'
'63' '9a' '96' '67' '7e' '57' 'a0' '66' 'ac' }
% Convert plaintext from hexadecimal (string) to decimal
representation
plaintext = hex2dec (plaintext_hex)
% plaintext1= c(1,1:16)
% This is the real McCoy.
% Convert the plaintext to ciphertext,
% using the expanded key, the S-box, and the polynomial
transformation matrix
ciphertext = cipher (plaintext, w, s_box, poly_mat, 1);
 % Convert the ciphertext back to plaintext
% using the expanded key, the inverse S-box,
% and the inverse polynomial transformation matrix
```

```
re_plaintext = inv_cipher (ciphertext, w, inv_s_box,
inv_poly_mat, 1);
```

C. Dekripsi

Rangkaian pada proses dekripsi ini langkahnya sama seperti proses enkripsi namun pada dekripsi ini merupakan *invers* dari setiap transformasi pada enkripsi. Pada proses dekripsi iterasi yang pertama dilakukan semua tranformasi kecuali *invers MixColumns*.

Ciphertet : $\begin{bmatrix} 51 & 68 & c9 & c0 \\ 92 & 0c & 31 & b9 \\ 6e & 6d & a7 & 63 \\ 7f & af & c0 & 78 \end{bmatrix}$

Transformasi addroundkey dengan sub kunci ke-10 yang telah didapatkan dari proses *key expansion*.

Kolom pertama $\begin{bmatrix} 51 \\ 92 \\ 6e \\ 7f \end{bmatrix} \oplus \begin{bmatrix} f9 \\ e5 \\ 5d \\ da \end{bmatrix} = \begin{bmatrix} a8 \\ 77 \\ 33 \\ a5 \end{bmatrix}$

$$\begin{bmatrix} 01010001 \\ 10010010 \\ 01101110 \\ 01111111 \end{bmatrix} \oplus \begin{bmatrix} 11111001 \\ 11100101 \\ 01011101 \\ 11011010 \end{bmatrix} = \begin{bmatrix} 10101000 \\ 01110111 \\ 00110011 \\ 10100101 \end{bmatrix}$$

Kolom kedua $\begin{bmatrix} 68 \\ 0c \\ 6d \\ af \end{bmatrix} \oplus \begin{bmatrix} e9 \\ 50 \\ de \\ 08 \end{bmatrix} = \begin{bmatrix} 81 \\ 5c \\ b3 \\ a7 \end{bmatrix}$

$$\begin{bmatrix} 01101000 \\ 00000011 \\ 01101101 \\ 10101111 \end{bmatrix} \oplus \begin{bmatrix} 11101001 \\ 01010000 \\ 11011110 \\ 00001000 \end{bmatrix} = \begin{bmatrix} 10000001 \\ 01011100 \\ 10110011 \\ 10100111 \end{bmatrix}$$

Kolom ketiga $\begin{bmatrix} c9 \\ 31 \\ a7 \\ c0 \end{bmatrix} \oplus \begin{bmatrix} 0d \\ ca \\ 94 \\ f7 \end{bmatrix} = \begin{bmatrix} c4 \\ fb \\ 33 \\ 37 \end{bmatrix}$

$$\begin{bmatrix}10110000\\10101100\\10101101\\10101110\end{bmatrix}\oplus\begin{bmatrix}010110101\\10010001\\01100111\\01110011\end{bmatrix}=\begin{bmatrix}11101010\\00111000\\11101010\\11011101\end{bmatrix}$$

Kolom keempat $\begin{bmatrix}c0\\b9\\63\\78\end{bmatrix}\oplus\begin{bmatrix}e3\\73\\dc\\33\end{bmatrix}=\begin{bmatrix}23\\ca\\bf\\4b\end{bmatrix}$

$$\begin{bmatrix}10100010\\10100001\\10100011\\10100011\end{bmatrix}\oplus\begin{bmatrix}01010010\\10101000\\01100000\\01001111\end{bmatrix}=\begin{bmatrix}11110000\\00001001\\11000011\\11101100\end{bmatrix}$$

Jadi hasil dari transformasi *AddRoundKey* adalah $\begin{bmatrix}a8&81&c4&23\\77&5c&fb&ca\\33&b3&33&bf\\a5&a7&37&4b\end{bmatrix}$

Setelah dilakukan transformasi tersebut selanjutnya adalah proses *InvShiftRows*, yaitu menggeser baris secara siklis. Dimana penggeseran baris tersebut kebalikan dari proses S*hiftRows* pada proses enkripsi.

Transformasi *InvShiftRows*

$$\begin{bmatrix}a8&81&c4&23\\77&5c&fb&ca\\33&b3&33&bf\\a5&a7&37&4b\end{bmatrix}\rightarrow\begin{bmatrix}a8&81&c4&23\\ca&77&5c&fb\\33&bf&33&b3\\a7&37&4b&a5\end{bmatrix}$$

Transformasi *InvSubBytes*

$$\begin{bmatrix}a8&81&c4&23\\ca&77&5c&fb\\33&bf&33&b3\\a7&37&4b&a5\end{bmatrix}\rightarrow(InvS-box)\rightarrow\begin{bmatrix}6f&91&88&32\\10&02&a7&63\\66&f4&66&00\\89&b2&cc&29\end{bmatrix}$$

Transformasi *InvMixColumns*

$$\begin{bmatrix}0e&0b&0d&09\\09&0e&0b&0d\\0d&09&0e&0b\\0b&0d&09&0e\end{bmatrix}\begin{bmatrix}6f\\10\\66\\89\end{bmatrix}=\begin{bmatrix}b_{1,1}\\b_{2,1}\\b_{3,1}\\b_{4,1}\end{bmatrix}=\begin{bmatrix}8d\\2c\\a8\\bd\end{bmatrix}$$

Pada proses ini sama dengan proses *MixColumns* yang ada pada enkripsi, hanya saja untuk matriks invers ini sudah ditentukan. Selanjutnya dilakukan perhitungan dari perkalian matriks tersebut. Pertama matriks tersebut dirubah menjadi biner selajutnya direpresentasikan dalam bentuk polinomial modulo 8. Kemudian dijumlahkan. Hasil akhir berupa polinomial lalu dikembalikan dalam bentuk biner dan dirubah menjadi heksadesimal dan didapatkan hasil iterasi terakhir dari proses dekripsi yaitu:

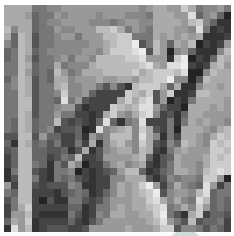
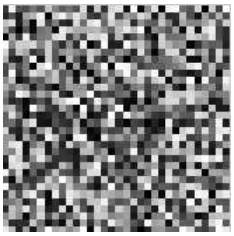
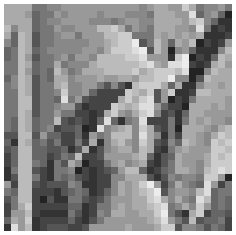
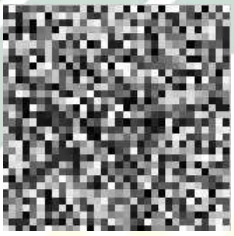
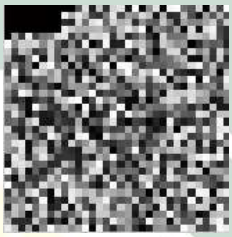
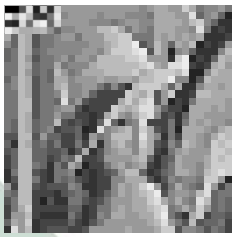
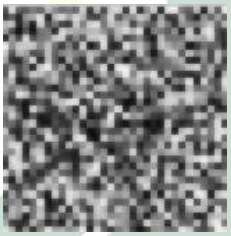
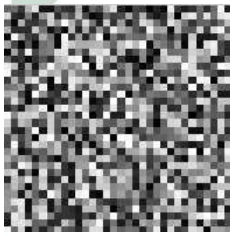
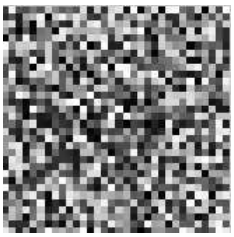
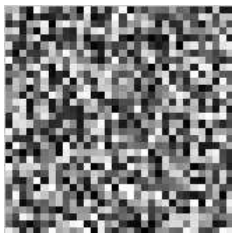
$$\begin{bmatrix} 6e & 3e & 5a & 52 \\ ac & b0 & 91 & a8 \\ 49 & 55 & 67 & 60 \\ 63 & 3c & 73 & 4f \end{bmatrix}$$

Dimana *state* tersebut merupakan state yang menjadi inputan dan nilaiya tidak berubah. Sehingga *state ciphertext* dapat dikembalikan menjadi *plaintext* kembali.

D. Pengujian

Pengujian pada penelitian ini adalah pengujian yang dilakukan terhadap citra hasil enkripsi. Citra yang telah dienkripsikan akan dilakukan serangan berupa *crop, blur,* dan *enhanchment*. Serangan tersebut bisa dilakukan diluar proses baik menggunakan matlab atau aplikasi lain yang mendukung. Pengujian dilakukan pada proses dekripsi dari *ciphertext* yang telah diserang, pengujian ini bertujuan untuk mengetahui ketahanan metode AES terhadap serangan seperti *crop, blur,* dan *enhanchment***.** Berikut tabel pengujian yang dilakukan:

Tabel 4.1 Hasil Pengujian Serangan

Enkripsi & Dekripsi	 Citra asli	 Citra hasil enkripsi	 Citra hasil dekripsi
Cropping	 *Ciphertext* citra enkripsi	 Citra enkripsi yang di *Crop*	 Citra dekripsi Hasil uji *Crop*
Blurring	 *Ciphertext* citra enkripsi	 Citra enkripsi yang di *blur*	 Citra dekripsi Hasil uji *blur*
Enhanchment	 *Ciphertext* citra enkripsi	 Citra enkripsi *enhancement*	 Citra dekripsi hasil uji *enhancement*

Dari hasil pengujian tersebut dilakukan beberapa serangan pada *ciphertext* sehingga dapat diketahui bahwa:

1. Pengujian serangan *Cropping* pada *ciphertext* masih bisa mengenali *plaintext* yang asli. Meskipun pada daerah yang dilakukan *cropping* menjadi berbeda, namun *plaintext* asli masih bisa di deteksi dengan jelas. Hal ini dikarenakan proses yang dilakukan pada tahap dekripsi dilakukan pada setiap blok tidak berpengaruh pada blok lainnya. Pada *cropping* ini dilakukan hanya pada sebagian sehingga yang berubah pada hasil dekripsi hanya pada blok yang terkena *cropping* tersebut. Untuk blok-blok lainnya masih dapat mengembaikan *plaintext* awal secara sempurna. Sehingga pada pengujian ini metode *Advanced Encryption Standard* (AES) tahan terhadap serangan *cropping*.
2. Pengujian serangan *Blurring* yang dilakukan pada *ciphertext* mempengaruhi proses pendekripsian sehingga hasil dari pengujian serangan ini tidak dapat mengembalikan *plaintext* yang asli. Hal tersebut dikarenakan pada proses *blurring* terjadi perubahan nilai piksel yang ada pada matriks citra *ciphertext* tersebut, sehingga sangat mempengaruhi pada proses dekripsi juga. Oleh sebab itu dekripsi dari citra *ciphertext* yang sudah dilakukan *blurring* tidak dapat mengembalikan ke *plaintex*t awal. Jadi metode *Advanced Encryption Standard* (AES) ini tidak tahan terhadap serangan *blur*.
3. Pengujian citra *ciphertext* dari serangan *Enhancement* ini menghasilkan *plaintext* yang berbeda atau *plaintext* tidak dapat kembali ke citra yang

asli karena pada proses *enhancement* tersebut mengubah nilai piksel dari matriks citra, dimana hal tersebut sangat berpengaruh pada proses dekripsi. Sehingga pada proses dekripsi *ciphertext* yang telah dilakukan panajaman atau *enhancement* tersebut tidak dapat mengembalikan *plaintext* semula. Jadi metode *Advanced Encryption Standard* (AES) ini juga tidak tahan terhadap serangan *Enhancement.*

Dari beberapa pengujian serangan yang telah dilakukan terhadap *ciphertext* untuk mengetahui ketahanan metode AES ini dapat dinyatakan bahwa metode AES ini tahan terhadap serangan *cropping*, namun tidak tahan terhadap serangan *blurring* dan *enhancement*.

Dari hasil penelitian ini didapatkan sebuah revolusi baru untuk memecahkan metode AES ini. Meskipun penyerang tidak dapat memecahkan atau menemukan kuncinya namun dengan melakukan serangan pada *ciphertext* tersebut dapat menghalangi penerima untuk dapat membuka *plaintext* yang asli. Serangan yang bisa dilakukan adalah *blur* dan *enhancement*. Dengan adanya hasil dari penelitian ini juga bisa menjadi masukan untuk melakukan perbaikan terhadap metode *Advanced Encryption Standard* (AES).

BAB V

PENUTUP

A. Kesimpulan

Berdasarkan pembahasan dari penelitian yang dilakukan pada perancangan aplikasi kriptografi *image* menggunakan metode *Advanced Encryption Standard* atau yang biasa disebut AES dapat disimpulkan sebagai berikut:

1. Algoritma *Advanced Encryption Standard* berhasil diaplikasikan untuk mengenkripsi sebuah citra atau *image*. Pada proses dekripsi metode ini bisa mengembalikan *plaintext* jelas seperti semula.
2. Pengujian yang dilakukan dengan memberikan serangan pada *ciphertext* berupa *cropping, blurring,* dan *enhancement* menghasilkan bahwa metode ini hanya bisa mengenali *plaintext* dengan jelas untuk serangan *cropping saja*. Namun pada dua serangan lainnya tidak dapat mengenali *plaintext* yang asli. Sehingga dapat disimpulkan bahwa metode *Advanced Encryption Standard* ini dapat dikatakan sebagai metode yang tahan terhadap serangan *cropping* dan tidak tahan terhadap serangan *blur* dan *enhancement*.
3. Metode *Advanced Encryption Standard* masih memiliki kekurangan, yaitu ketika penyerang melakukan serangan seperti *blur* ataupun *enhancement* terhadap citra *ciphertext* mengakibatkan penerima tidak dapat membuka *plaintext* yang asli.

B. Saran

Saran yang bisa dilakukan untuk penelitian lanjutan dari penelitian ini adalah:

1. Menggunakan objek pengaplikasian metode AES terhadap file citra yang berukuran lebih besar atau subjeknya berupa video.
2. Macam serangan yang dilakukan untuk pengujian bisa ditambah untuk mendapatkan hasil yang lebih detail dan kompleks.
3. Dengan pengujian yang dilakukan pada penelitian ini dapat menjadi masukan bahwa perlu adanya penelitian yang lebih untuk pengembangan terhadap metode *Advanced Encryption Standard* (AES) yang lebih baik lagi dan tahan terhadap serangan apapun.

DAFTAR PUSTAKA

Ariyus, Doni. *Keamanan Multimedia*. Yogyakarta: Andi Offset, 2009.

Arya, A. (2016). Effective AES Implementation. *IJECET*.

Bendi, K. J. (2012). Implementasi Algoritma Rijndael Untuk Enkripsi Dan Dekripsi Pada Citra Digital. *Research Gate*, -.

Computer, Wahana. *Memahami Model Enkripsi Dan Security Data.* Andi Offset. Yogyakarta. 2003

Deshmukh, P. (2016). An Image Encryption And Decryption Using AES Algorithm. *IJSER*, 7.

Fachrurrozi, M. F. (2006). Enkripsi Pesan Rahasia Menggunakan Algoritma AES: Rijndael. *Jurnal UIN Syarif Hidayatullah*, Jakarta.

Fricles Ariwisanto Sianturi. Perancangan Aplikasi Pengamanan Data Dengan Kriptografi Advanced Encryption Standard (AES). 2013.Medan. STMIK Budi Darma Medan

Goldreich, Oded. On The Foundation Of Modern Cyptograhy, Department Of Computer Science And Applied Mathematic Weizmann Institute Of Science. Revohot Israel. 2003

Hanifah, F. (2012). Aplikasi Algoritma Rijndael Dalam Pengamanan Citra Digital. *Skripsi*.

Isnaini, Yusran. Hak Cipta Dan Tantangannya Di Era Cyber Space. Jakarta: Ghalia Indonesia, 2009.

Kromodimoeljo, S. (2009). Teori Dan Aplikasi Kriptografi. Jakarta: Spk It Consulting.

kurniawan, R. A. (2017). aplikasi keamanan sms menggunakan algoritma rijndael. *jurnal matrik*.

Munir, R. (2006). Kriptografi. Bandung: Informatika Bandung.

Publications, F. I. (2001). *Announcing the Advanced Encrytion Standard.* New York: National Institute of Standarts and Technologys.

Rahmatullah. Perancangan Aplikasi Kriptografi File Menggunakan Metode Advanced Encryption Standart (AES).2016. Surabaya. Jurnal ITS

Rhee. (2013). *Symmetric Block Ciphers, wireless mobile internet security.* wireless mobile internet security.

Roshni Padate, Aamna Patel. (2015*). Image Encryption And Decryption Using AES Algorithm*. *IJECET*, 23-29.

Sadikin, & Rifki. (2012). Kriptografi untuk Keamanan Jaringan. Yogyakarta: ANDI.

Soleh, M. (2010). Analisis Dan Implementasi Watermarking Dengan Algoritma AES Untuk Pemberian Data Hak Cipta Pada File Audio. UIN Syarif Hidayatullah, Jakarta.

Stalling, W. (2005). *Cryptography and Network Security Principles and Practices, Fourth Edition*. New Jersey: Prentice Hall.

Sukarman, Herry. Teori Bilangan. Pusat Penerbitan Universitas Terbuka. 2001

Tilborgh, H. C. (2005). *Ecnyclopedia of Cryptography and Security.* New York: SpringerScience+Business Med

Vincent, Rijmen Dan Joan Daemen. *AES Proposal: Rijndael*. Federal Information Processing Standart Publication. April 2003

Wagstaff, Samuel S. *Cryptanalysis Of Number Theoretic Ciphers*. Chapman&Hall/CRC

Yoga Palilianto. Rancang Bangun Aplikasi Enkripsi Dan Deskripsi Citra Digital Menggunakan Algoritma Rijndael Berbasis Java SE.2014.