ANALISIS PERFORMA PROSES ENKRIPSI DAN DEKRIPSI MENGGUNAKAN ALGORITME AES-128 PADA BERBAGAI FORMAT *FILE*

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Rohbi Visdya Harris Chandra
115060807111121

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

PENGESAHAN

ANALISA PERFORMA PROSES ENKRIPSI DAN DEKRIPSI MENGGUNAKAN ALGORITME AES 128 PADA BERBAGAI FORMAT *FILE*

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Rohbi Visdya Harris Chandra
115060807111121

Skripsi ini telah diuji dan dinyatakan lulus pada
3 Agustus 2018
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Ari Kusyanti, S.T, M.Sc	Mahendra Data, S.Kom., M.Kom
NIP: 20110283122820001	NIK: 2015038611171001

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP: 19710518 200312 1 001

ii

IDENTITAS TIM PENGUJI

Penguji 1:

Eko Sakti Pramukantoro, S.Kom, M.Kom

NIK: 201102 860805 1 001

Penguji 2:

Dany Primanita Kartikasari, S.T., M.Kom

NIP: 19771116 200501 2 003

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 18 Februari 2018

Rohbi Visdya Harris Chandra
115060807111121

DAFTAR RIWAYAT HIDUP

Nama : Rohbi Visdya Harris Chandra

Tempat, Tanggal Lahir: Surabaya, 8 Januari 1993

Riwayat Sekolah : SD dr. Soetomo V Surabaya

SMPN 5 Malang

SMAN 8 Malang

UCAPAN TERIMA KASIH

Puji syukur kehadirat Tuhan Yang Maha Esa yang telah melimpahkan rahmat dan anugerah-Nya sehingga penulis dapat menyelesaikan laporan Tugas Akhir yang berjudul **"ANALISA PERFORMA PROSES ENKRIPSI DAN DEKRIPSI MENGGUNAKAN ALGORITME AES-128 PADA BERBAGAI FORMAT *FILE*"**

Dalam penyusunan laporan Tugas Akhir ini penulis banyak mendapatkan dukungan, bimbingan, bantuan dan doa dari berbagai pihak. Oleh karena itu melalui kesempatan ini penulis ingin menyampaikan ucapan terima kasih kepada:

1. Allah SWT yang telah memberikan rahmat dan anugerah-Nya sehingga Tugas Akhir ini dapat selesai.
2. Mama Hj. Rr. Dyah Sulistyorini, Papa Dr. H. Is Prijadi, SpOG, dan adik-adik saya Rohbi Visdya Adrian Hadinata, Rohbi Visdya Novrian Ardiansyah, dan Rohbita Visdya Hestika Anggraini, beserta keluarga besar saya yang selalu mendukung, membimbing, mendoakan dan memberi semangat tanpa putus demi kelancaran Tugas Akhir ini.
3. Dekan Fakultas Ilmu Komputer Universitas Brawijaya Bapak Wayan Firdaus Mahmudy, S.Si, M.T, Ph.D, Ketua Jurusan Teknik Informatika Universitas Brawijaya Tri Astoto Kurniawan, S.T, M.T, Ph.D dan Ketua Prodi Teknik Informatika Bapak Agus Wahyu Widodo, S.T, M.Sc.
4. Dosen Pembimbing 1 IbuAri Kusyanti, S.T, M.Sc, yang telah membimbing dan memberikan arahan kepada penulis demi terselesaikannya Tugas Akhir ini.
5. Dosen Pembimbing 2 Bapak Mahendra Data, S.Kom., M.Kom, yang telah memberikan saran dan masukan untuk menyelesaikan penulisan laporan Tugas Akhir ini.
6. Teman-teman angkatan 2011 di kelas kritis 2018.
7. Para dosen dan karyawan FILKOM terutama teruntuk pada bu Wiwin, pak Pras, pak Aswin, pak Nurudin beserta jajaran dosen pengajar dan karyawan FILKOM yang sudah banyak membantu.
8. Dan terima kasih juga kepada Rizal Senggek, Fikar, Tadho, Sena, Aso, Pepeng, Lazu, Stefanus Bayu, Acong, Gendon, Julita, Sari, Alip, Yoga, Lambang, Abyan dan Yoan Anindhita yang selalu semangat mengerjakan, saling membantu dan support satu sama lain.
9. Teman-teman kelas I Informatika 2011, teman-teman DFD, Happy Fams, Keluarga Jancoy, Cuntel dan semua pihak (termasuk para mantan) yang tidak dapat penulis sebutkan satu per satu yang telah memberi support secara langsung maupun tidak langsung.

Dalam penyusunan Tugas Akhir ini, penulis menyadari bahwa masih terdapat kekurangan karena keterbatasan waktu dan ilmu pengetahuan. Oleh karena itu penulis menerima kritik dan saran yang bersifat membangun dari pembaca guna memperbaiki penyusunan karya tulis yang akan datang. Penulis berharap Tugas Akhir ini dapat memberikan manfaat baik bagi diri penulis sendiri maupun masyarakat luas.

Malang, 18 Februari 2018

rohbiharris@gmail.com

ABSTRAK

Setiap perusahaan memiliki data yang harus disimpan dalam bentuk digital. Seiring dengan perkembangan perusahaan maka data yang perlu disimpan akan semakin banyak, sehingga ruang penyimpanan yang dibutuhkan juga akan semakin besar. Selain itu faktor keamanan data juga perlu diperhatikan. Salah satu metode pengamanan data adalah dengan melakukan enkripsi data. Pada penelitian yang dilakukan adalah analisis algoritme enkripsi, dimana menurut hasil analisisnya menunjukkan bahwa algoritme AES menunjukkan algoritme yang direkomendasikan sebagai algoritme untuk mengamankan penggunaan *file.* Penelitian ini mengajukan analisis metode enkripsi AES. Ada beberapa parameter pengujian metode enkripsi, yaitu *cost* enkripsi, waktu komputasi dan menguji beberapa format *file* yang dienkripsi dan didekripsi dengan acuan ukuran *file* untuk format *file* dokumen dan gambar, untuk format *file* audio dan video mengacu menggunakan durasi waktu, kemudian dari tiap format *file* dipilih dua ekstensi *file* yang digunakan untuk pengujian. Berdasarkan hasil pengujian menunjukkan metode enkripsi AES memberikan hasil yang bagus terhadap berbeberapa format *file*, AES-128 mampu mengenkripsi *file* dengan baik walaupun membutuhkan waktu komputasi yang sedikit lama di beberapa ekstensi *file* yang diujikan. Waktu yang dibutuhkan untuk proses enkripsi dan dekripsi dengan menggunakan algoritme AES dipengaruhi tiap ukuran *file* yang berbeda, semakin besar ukuran *file* yang digunakan, waktu komputasi yang dibutuhkan semakin lama.

Kata kunci: *AES, Enkripsi, Dekripsi, Pengamanan, File*

ABSTRACT

Every company has data that must be stored in digital form. Along with the development of the company, the data that needs to be stored will be more and more, so that the storage space needed will also be greater. In addition, data security factors also need to be considered. One method of securing data is by encrypting data. In the research conducted is an encryption algorithm analysis, which according to the results of the analysis shows that the AES algorithm shows an algorithm recommended as an algorithm to secure file usage. This study proposed an analysis of the AES encryption method. There are several parameters for testing the encryption method, namely encryption costs, computation time and testing several file formats encrypted and decrypted with file size references for document and image file formats, for audio and video file formats referring to the time duration, then from each file format selected two file extensions used for testing. Based on the results of the test shows that the AES encryption method provides good results for a number of file formats, AES-128 is able to encrypt files well even though it requires a long computing time in some of the file extensions tested. The time needed for the encryption and decryption process using the AES algorithm is affected by each different file size, the larger the file size used, the longer computation time needed.

Keywords: *AES, Encryption, Decryption, Storage, File*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Pesatnya perkembangan teknologi menjadi semakin mudah dan cepatnya berbagi informasi maupun data (Syaikhu, 2010). Penyimpanan data pada suatu instansi atau perusahaan sangat diperlukan untuk proses berjalannya kegiatan instansi atau perusahaan tersebut. Dengan seiring berjalannya waktu penyimpanan data yang besar sangat diperlukan. Akan tetapi perlu suatu pengamanan data yang bagus sehingga data yang disimpan terjamin keamanannya. Terlebih lagi data tersebut adalah suatu data rahasia dari perusahaan yang sangat riskan jika data tersebut berada pada orang yang tidak berhak. Sehingga pengamanan data menjadi hal yang sangat penting untuk menjadi perhatian. Berdasarkan hal tersebut keamanan menjadi hal pennting yang perlu diperhatikan dan didalami.

Ada berbagai metode untuk mengatasi keamanan pada penggunaan data. Enkripsi salah satu metode yang andal untuk digunakan untuk mengamanankan data. Enkripsi merupakan metode pengamanan data dengan melakukan pengkodean. Sehingga data tetap aman dan terjaga kerahasiannya. Ada berbagai macam algoritme enkripsi yang dapat digunakan untuk menjaga keamanan data yang dikirimkan (Bhardwaj, 2016). Berbagai macam parameter yang digunakan untuk mengukur performa dari algoritme yang di gunakan, sehingga diperlukan analisis hasil untuk menguji seberapa bagus performa sebuah algoritme dalam mengolah data *file*.

Penelitian Jeva *et al* (2012) melakukan analisis algoritme enkripsi. Pada penelitian tersebut dilakukan analisis perbandingan algoritme enkripsi AES, DES, Triple DES, BLOWFISH, RC4. Ada beberapa faktor yang digunakan untuk melakukan analisis seperti rasio enkripsi, kecepatan komputasi, panjang *key*, *tunability*, dan *security against attacks*, tetapi belum ada pengujian waktu eknrispi dan waktu dekripsi dan *cost* enkripsi. Hasil analisis menunjukkan bahwa algoritme AES menunjukkan algoritme yang direkomendasikan sebagai algoritme untuk mengamankan penggunaan *file*. Pada penelitian lain hasil analisis algoritme untuk mengamankan data menunjukkan algoritme *Advanced Encryption Standard* (AES) memiliki hasil yang lebih baik dibandingkan dengan DES 36 dan 3DES 268. Penelitian tersebut menunjukkan bahwa AES merupakan algoritme yang bagus untuk enkripsi. Berdasarkan penelitian tersebut AES merupakan metode yang handal untuk mengamankan data.

Fokus penelitian ini, pertama adalah mengkaji tentang algortime AES sebagai algoritme pengamanan *file* yang aman dan kecepatan enkripsi dan dekripsi ketika algoritme AES melakukan pengamanan pada tiap *file*. Kenapa menggunakan *file*? karena *file* adalah hal yang penting dan selalu digunakan oleh orang-orang, perusahaan dan perkantoran di seluruh dunia sebagai sebuah hal penting yang memiliki kegunaan sangat banyak, mulai dari kerahasiaan yang sangat dijaga keamanannya sampe hal-hal sepele yang di simpan oleh banyak orang. Kedua

mengimplementasikan algoritme AES untuk enkripsi dan dekripsi terhadap berbagai format *file*, yang bertujuan untuk mengetahui seberapa aman dari penilaian pengujian cost enkripsi dan seberapa cepat waktu komputasi algoritme AES saat enkripsi dan dekripsi *file* dengan diambilkan dua contoh ekstensi *file* dengan besar ukuran *file* yang berbeda-beda dari tiap format *file* yang diujikan. Karena dari tiap ekstensi *file* memiliki karakter data yang berbeda-beda, contohnya dari format *file* video, penulis memilih ekstensi *file* .mp4 dan .mkv untuk diuji sesuai parameter yang digunakan untuk pengujian dan analisis dengan menentukan lima ukuran *file* yang berbeda besarnya dan lima macam durasi waktu dengan ukuran *file* sudah di sesuaikan dengan *file* yang lain. Ketiga melakukan analisis perbandingan hasil terhadap keamanan proses enkripsi dekripsi dan kecepatan waktu komputasi algoritme AES saat proses enrkipsi dan dekripsi *file* dengan menggunakan parameter saat pengujian dan analisis.

1.2 Rumusan masalah

Berdasarkan latar belakang yang telah diuraikan, penulis merumuskan permasalahan yang akan diselesaikan sebagai berikut:

1. Bagaimana penerapan algoritme AES untuk proses enkripsi dan dekripsi *file*?
2. Bagaimana performa algoritme AES dalam melakukan proses enkripsi dan dekripsi dalam berbagai format *file*?
3. Bagaimana pengaruh perbedaan ukuran dan format *file* terhadap hasil enkripsi dan dekripsi dengan menggunakan algoritme AES?

1.3 Tujuan

Berdasarkan rumusan masalah yang telah dijelasakan, maka tujuan penelitian yang ingin dicapai adalah sebagai berikut:

1. Menerapkan algoritme AES untuk enkripsi dan dekripsi *file*.
2. Menganalisis hasil enkripsi dan dekripsi algoritme AES pada ukuran *file*.
3. Membandingkan performa proses enkripsi dekripsi saat menenkripsi dan mendekripsi *file* yang sudah di siapkan.

1.4 Manfaat

Berdasarkan beberapa urai yang telah dijelaskan, manfaat yang diharapkan penulis dari hasi penelitian ini adalah untuk mendapatkan hasil perbandingan dari proses enkripsi dan dekripsi menggunakan algoritme AES untuk mengamankan data *file* yang memiliki berbagai macam ukuran, sehingga dapat menjadi rekomendasi bagi pembangun sistem penyimpanan data secara *virtual* sebagai algoritme pengamanan data.

1.5 Batasan masalah

Adapun batasan masalah dalam penelitian ini, antara lain:

1. Proses simulasi enkripsi menggunakan AES-128 yang dilakukan pada pemrograman Java.
2. Yang digunakan untuk batasan pengujiannya adalah menilai hasil kecepatan waktu enkripsi dan dekripsi untuk menguji kecepatan waktu komputasi, panjang *ciphertext*, dan cost enkrips untuk menguji keamanan proses enkripsi dan dekripsi *file*.
3. Dari setiap format *file* akan dipilih 2 tipe data dari setiap format *file*, yaitu untuk format *file* dokumen menggunakan .txt dan .docx, untuk format *file* audio menggunakan .mp3 dan .wav, untuk format *file* video menggunakan .mp4 dan .mkv, dan yang terakhir unutk format *file* gambar menggunakan .jpeg dan .png.

1.6 Sistematika pembahasan

Penelitian ini, sistematika pembahasan menjelaskan tiap bab secara garis besar sebagai berikut:

BAB 1: Pendahuluan

Bagian ini membahas latar belakang penyusunan penelitian, rumusan masalah, tujuan yang ingin dicapai dalam penelitian, manfaat yang diharapkan dalam penelitian, batasan masalah, serta sistematika pembahasan.

BAB 2: Landasan Kepustakaan

Bagian ini menjelaskan tentang kajian pustaka, enkripsi, AES, serta teori-teori yang berkaitan dengan penelitian tentang optimasi komputasi pada berberapa format *file*.

BAB 3: Metodologi

Bagian ini menjelasakan tahapan atau langkah-langkah dalam melakukan penelitian, penerapan algoritme AES sebagai algoritme pengamanan data, serta analisi algoritme AES.

BAB 4: Perancangan

Bagian ini menjelasakan perancangan dan proses dalam sistem yang digunakan untuk penerapan algoritme AES sebagai algoritme pengamanan data, serta analisi algoritme AES.

BAB 5: Implementasi dan analisis

Bagian ini menunjukna hasil dari implementasi penerapan algoritme AES sebagai algoritme pengamanan data, serta menunjukkan analisis algoritme AES pada berbagai tipe *file*.

BAB 6: Kesimpulan

Bagian ini menunjukkan kesimpulan penelitian dengan menjawab rumusan masalah serta memberika saran untuk penelitian kedepannya.

BAB 2 LANDASAN KEPUSTAKAAN

Bagian ini akan dijelaskan kajian pustaka dan dasar teori yang digunakan untuk menunjang penelitian ini. Kajian pustaka berisi penelitian sebelumnya yang terkait dengan penelitian ini, yang dijadikan acuan. Sedangkan dasar teori berisi tentang bahasan teori-teori yang digunakan sebagai aspek pendukung penelitian ini. Dasar teori yang akan dibahas adalah enkripsi, algoritme AES.

2.1 Kajian Pustaka

Pada bagian ini akan dibahas penelitian sebelumnya tentang analisis performa algoritme untuk pengamanan data. Pada penelitian Jeeva *et al* (2012) dilakukan analisis beberapa metode pengamanan data. Metode simetrik yang dilakukan analisis adalah AES, DES, Triple DES, Blowfish, RCA. Dimana algoritme AES merupakan algoritme terbaik yang direkomendasikan, dengan beberapa parameter pengujian seperti ratio enkripsi, kecepatan, *tunability*, *security against attacks*. Pada penelitian lain juga dilakukan analisis perbandingan algoritme pengamanan, diantara algoritme AES, DES 36, 3DES 268 algoritme AES memberikan hasil analisis terbaik dan direkomendasikan sebagai algoritme untuk pengamanan data (Bhardwaj *et al*, 2016). Penelitian ini akan melakukan analisis algoritme untuk pengamanan data. Algoritme yang dilakukan analisis adalah algoritme. Ada beberapa parameter yang akan digunakan untuk melakukan analisis.

2.2 Kriptografi

Kriptografi merupakan salah satu teknik untuk mengamankan data dengan menggunakan enkripsi, dimana data diacak menggunakan kunci enkripsi menjadi suatu data yang sulit dibaca oleh pihak lain dan tidak memiliki kunci dekripsinya (Kromodimoeljo, 2009). Penggunaan kriptografi sangat dibutuhkan untuk mengamankan informasi dalam sebuah data penting, seperti komunikasi e-mail, transaksi bank, rekenik bank, PIN, password dan transaksi kartu kredit, tanda tangan elektronik. Selain itu kriptografi juga digunakan sebagai pengamanan data.

Kriptografi berasal dari bahasa Yunani, yaitu *crypto* dan *graphia* yang keduanya memiliki arti masing-masing adalah penulisan dan rahasia. Kriptografi merupakan suatu ilmu yang mempelajari penulisan secara rahasia sehingga informasi yang dikandung dalam data tersebut tidak diketahui oleh pihak yang tidak sah. Kriptografi merupakan cabang ilmu dari matematika yang biasa disebut *cryotology*, yaitu bagaimana menjaga keamanan data dengan mengkodekannya. Kriptografi memiliki beberapa tujuan, antara lain:

1. *Confidentiality* adalah suatu jaminan bahwa data yang telah dilakukan enkripsi memiliki kerahasian yang tidak dapat dibuka oleh siapapun kecuali oleh pihak yang memiliki wewenang untuk membukanya.

2. *Integrity* yaitu adalah jaminan bahwa data yang telah dilakukan enkripsi masih sah keasliannya. Sehingga sistem perlu kemampuan untuk mendekteksi manipulasi yang dilakukan oleh pihak lain.
3. *Authentication* adalah suatu jaminan mampu mengenali atau identifikasi suatu data. Mulai dari pengenalan keaslian, isi, waktu penggunaan dan lain sebagainya. Selain itu pihak yang komunikasi adalah benar-benar pihak yang bersangkutan.
4. *Non-repudation* adalah suatu usaha yang dapat mencegah terjadinya penyangkalan terhadap penggunaan informasi oleh *user*. Suatu jaminan sistem mampu membuktikan korespodensi antara pihak yang mengirimkan data sehingga dapat memastikan bahwa indentitas *user* dan tidak terjadinya penyangkalan oleh pihak tersebut.

Ada dua jenis algoritme enkripsi pertama algoritme simetrik dan algoritme asimetrik. Algoritme simetrik adalah algoritme yang hanya menggunakan satu untuk untuk proses enkripsi dan dekripsi sehingga keamanan algoritme ini terletak pada kuncinya. Sedangkan asimetrik adalah algoritme yang ada dua kunci yang berbeda. Kunci publik dan kunci privat. Kunci ini berisi *cipher* yang berguna mengubah *text* menjadi *ciphertext* sehingga dapat melakukan proses enrkipsi ataupun dekripsi. Pada algoritme enkripsi model simetrik terdapat dua jenis pengelompokan. Kelompok pertama adalah *block cipher*, yaitu algoritme enkripsi yang cara kerjanya membagikan *plaintext* ke dalam ukuran dan panjang tertentu, untuk mempersulit serangan ada berbagai polanya. Contoh dari *block chiper* adalah DES, 3DES, AES, Blowfish, IDEA. Sedangkan kelompok kedua adalah *stream chiper*, yaitu suatu algoritme enkripsi yang mengenkripsikan data persatuan data, seperti *bit*, *byte*, dan lain sebagainya. Contoh algoritme ini adalah RC4, SEAL, WAKE, Cellular Automaton.

2.3 *File*

File merupakan sekumpulan dokumen atau data yang berisi informasi tertentu dan dapat dibuka dengan menggunakan program komputer tertentu. Sebuah *file* mempunyai atribut nama *file*, ukuran, waktu penyimpanan *file* dan tipe *file*. *File* mempunyai beberapa format. Beberapa format *file* antara lain video, audio, dokumen dan gambar.

2.3.1 *File* Dokumen

File dokumen merupakan sebuah format *file* yang berisi kumpulan teks atau kalimat. Dalam *file* dengan format dokumen memiliki beberapa ekstensi. Ekstensi merupakan tipe *file* yang menunjukkan bahwa *file* dapat dibuka dengan program tertentu sesuai dengan ekstensinya. Ekstensi dalam *file* dokumen antara lain .doc dan .txt. *File* dokumen dengan ekstensi .doc merupakan sebuah bentuk *file* yang dihasilkan dan dapat dibuka dengan menggunakan program microsoft word. *File* dokumen dengan ekstensi .txt merupakan dokumen yang dihasilkan dan dapat dibuka dengan program notepad.

2.3.2 *File* Gambar

File gambar merupakan sebuah format *file* yang berisi sebuah gambar dari suatu obyek dalam pandangan 2D atau 3D. *File* gambar memiliki beberapa ekstensi antara lain .jpeg dan .png. *File* dengan ekstensi .jpeg merupakan singkatan dari *Joint Photographic Experts Group*. Format ini banyak digunakan untuk menyimpan *file* gambar dengan ukuran *file* lebih kecil. Pada umumnya digunakan untuk menyimpan *file* foto .png (*Portable Network Graphics)* merupakan suatu ekstensi *file* yang digunakan untuk menyimpan gambar dengan metode pemadatan yang tidak menghilangkan bagian dari citra tersebut (*lossless compression*).

2.3.3 *File* Audio

File audio merupakan sebuah *file* yang berisi suara digital yang tersimpan dalam sebuah komputer. Beberapa ekstensi dalam *file* audio diantaranya adalah .mp3 dan .wav. Mp3 merupakan sebuah kompresi *file* audio dengan metode pengodean *Pulse Code Modulation* yang memungkinkan sebuah *file* audio menjadi lebih kecil dengan menghilangkan bit dari komponen suara yang tidak terdengar manusia. Wav (*waveform audio format*) merupakan sebuah format *file* audio yang dikembangkan oleh microsoft dan IBM. *File* .wav bisa jadi berisi *file* audio yang telah terkompressi. Namun pada umumnya *file* .wav merupakan sebuah bentuk *file* audio yang tidak dikompressi dalam format *linear pulse code modulation*.

2.3.4 *File* Video

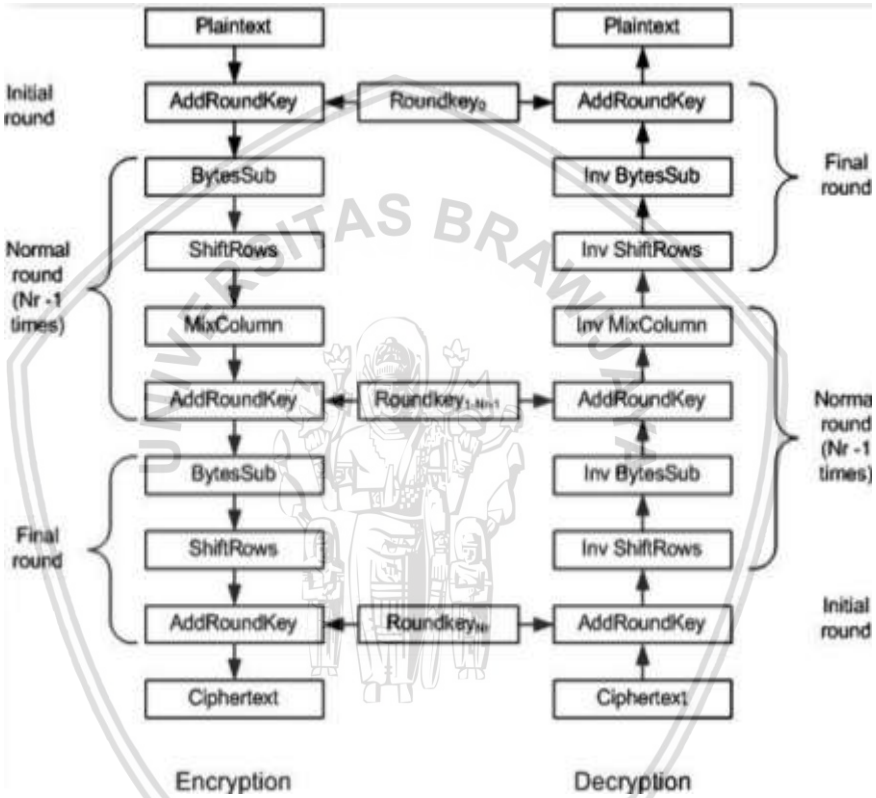
File video merupakan sebuah *file* untuk menyimpan video digital atau gambar gerak dengan suara. Pada umumnya *file* video akan dikompresi untuk mengurangi ukuran *file* yang terlalu besar. Beberapa ekstensi *file* video antara lain .mp4 dan .mkv. Mp4 merupakan sebuah kependekan dari Mpeg-4. Mp4 merupakan sebuah format berkas suara dan gambar/video digital yang dikeluarkan oleh sebuah organisasi MPEG. *File* ini merupakan pengembangan dari format *QuickTime* dari Apple. Mkv (*Matroska video*) merupakan standard format Multimedia yang bersifat terbuka. Format *file* ini dapat menyimpan banyak jumlah video, audio, gambar, track *subtitle* hanya dalam sebuah *file*. Matroska merupakan format dengan spesifikasi yang terbuka sepenuhnya (open source).

2.4 Algoritme AES

Advanced Encryption Standard (AES) adalah suatu algoritme enkripsi tipe simetrik *block cipher* yang dijadikan standard FIPS oleh NIST tahun 2001. Pada abad 21 di amerika secara perlahan algoritme DES digantikan oleh AES.

Algoritme AES merupakan algoritme terpopuler pada tipe simetrik yang digunakan saat ini (Kromodimoeljo, 2009).

AES merupakan algoritme *block cipher* dengan sistem permutasi dan substitusi. Ada tiga jenis algoritme AES, yaitu AES-128, AES-192, dan AES-256. Pengelompokan ini berdasarkan panjang kunci yang digunakan pada algoritme AES. Selain itu ada beberapa hal lain yang membedakan antar jenis algoritme AES, yaitu *Round* yang digunakan. AES-128 menggunakan 10 *Round*, AES-192 menggunakan 12 *Round*, dan AES-256 menggunakan 14 *Round* (Kromodimoeljo, 2009). Gambar 2.1 menunjukkan secara garis beras proses enkripsi yang dilakukan oleh algoritme AES.



Gambar 2.1 Diagram alir algoritme AES

2.4.1 Perhitungan enkripsi algoritme AES

Pada bagian ini akan dicontohkan proses enkripsi AES sesuai dengan prosesnya yang telah dijelaskan pada sub-bab 2.4. Dimisalkan terdapat suatu *key* dan *plaintext*, sebagai berikut:

Key = “Hello Sponge bob”

Plaintext =“One Eight NineDI”

Pertama dalam algoritme AES memasukan *key* dan *plaintext*, lalu mengubah *key* dan *plaintext* dalam bentuk ASCII Hex karakter, sebagai berikut:

key

H	e	l	l	o		S	p	o	n	g	e		b	o	b
48	65	6c	6c	6f	20	53	70	6f	6e	67	65	20	62	6f	62

Plaintext

O	n	e		E	i	g	h	t		N	i	n	e	D	l
4f	6e	65	20	45	69	67	68	74	20	4e	69	6e	65	44	49

Setelah dilakukan perubahan *key* dan *plaintext* menjadi ASCII Hex, selanjutnya dilakukan proses *byteSub*, yaitu proses memecah *key* dan *plaintext* menjadi beberapa bagian. Pada kasus ini dilakukan pemecahan menjadi 4 bagian, sebagai berikut:

w[0] = (48 65 6c 6c)

w[1] = (6f 20 53 70)

w[2] = (6f 6e 67 65)

w[3] = (20 62 6f 62)

Setelah dilakukan pemecahan dilakukan perubahan nilai ASCII Hex sesuai dengan tabel SBOX pada Tabel 2.1.

Tabel 2.1 Tabel SBOX

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
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	3	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	BB	BB	16

Sehingga diperoleh hasilnya sebagai berikut.

w[0] = (52, 4d, 50, 50)

w[1] = (a8, b7, ed, 51)

w[2] = (a8, 9f, 85, 4d)

w[3] = (b7, aa, a8, aa)

Langkah selanjutnya adalah proses *ShiftRow*, yaitu mengeser satu *row* ke belakang, dimisalkan pada *w*[0] dimensi pertama adalah 52 dipindah ke dimensi terahkir dan dimensi kedua menjadi pertama dan seterusnya bergeser ke kiri.

w[0] = (4d, 50, 50, 52)

w[1] = (b7, ed, 51, a8)

w[2] = (9f, 85, 4d, a8)

w[3] = (aa, a8, aa, b7)

Setelah dilakukan proses *ShiftRow*, langkah selanjutnya adalah *mixColumn*, yaitu proses kombinasi dari beberapa bagian *key*.

Setelah selesai dilakukan proses *mixColumn* maka diperoleh nilai pertama dari *Round* pertama: 6b 7b 61 e8 f4 77 dd 13 5e 13 76 a1 06 4e b0 09. Pada penelitian ini akan digunakan AES-128 sehingga akan dilakukan *Round* mulai proses *byteSub*, *ShiftRow*, dan *mixColumn* sebanyak 10 kali. Dengan cara yang sama maka akan diperoleh nilai *Round* sebagai berikut:

- *Round* 0: 07 0b 09 4c 2a 49 34 18 1b 4e 29 0c 4e 07 2b 2b
- *Round* 1: 6b 7b 61 e8 f4 77 dd 13 5e 13 76 a1 06 4e b0 09
- *Round* 2: da 54 ae bb 0b 92 63 33 25 ac 05 42 fb 5b 2c 09
- *Round* 3: 94 73 71 ff 71 13 03 2c 70 3c c5 ef 7e ec 55 40
- *Round* 4: 0b 37 1a 81 e2 08 e6 95 28 b3 68 c2 15 d7 2a 33
- *Round* 5: 78 07 96 0f fe 55 1e e9 68 3a 0c d0 f0 6c 66 ad
- *Round* 6: 81 2e f7 3c ae 51 1d 93 73 47 c1 5d 8e 5a 3d ae
- *Round* 7: 08 47 d6 07 1d 71 3a 0e 16 d7 45 e2 2c 3a 74 4d
- *Round* 8: 5a 1b c2 2f a6 bb 64 e6 86 c6 a8 f3 6a e3 dc f2
- *Round* 9: 2f 84 fa e8 8a 99 ac 78 48 fd fc f2 93 3ff f3 97
- *Round* 10: e2 45 84 5b 1f b5 2d 44 6b ee 67 6e d6 74 14 3c

Sehingga didapatkan hasil *ciphertext*= e2 45 84 5b 1f b5 2d 44 6b ee 67 6e d6 74 14 3c.

Berikut adalah penjabaran perhitungan secara manualnya.

Untuk AES *AddRoundkey*, *Round* 0

Matriks dan *Roundkey* No.0 Matriks :

$$\begin{pmatrix} 4f & 45 & 74 & 6e \\ 6e & 69 & 20 & 65 \\ 65 & 67 & 4e & 44 \\ 20 & 68 & 69 & 49 \end{pmatrix} \begin{pmatrix} 48 & 6f & 6f & 20 \\ 65 & 20 & 6e & 62 \\ 6c & 53 & 67 & 6f \\ 6c & 70 & 65 & 62 \end{pmatrix}$$

Matriks barunya adalah :

$$\begin{pmatrix} 07 & 2a & 1b & 4e \\ 0b & 49 & 4e & 07 \\ 09 & 34 & 29 & 2b \\ 4c & 18 & 0c & 2b \end{pmatrix}$$

AES - *Round* 1

Dari hasil *Round* 0, hasil *AddRoundKey* nya diproses ke tahap *SubBytes*

$$\begin{pmatrix} c5 & e5 & af & 2f \\ 2b & 3b & 2f & c5 \\ 01 & 18 & a5 & f1 \\ 29 & ad & fe & f1 \end{pmatrix}$$

Subtitutesetiap entri (*byte*) dari Matriks saat ini dengan entri yang sesuai di AES S-Box

Kemudian *ShiftRows*

$$\begin{pmatrix} c5 & e5 & af & 2f \\ 3b & 2f & c5 & 2b \\ a5 & f1 & 01 & 18 \\ f1 & 29 & ad & fe \end{pmatrix}$$

Kemudian lanjut ke proses *MixColumns* dimana *MixColumns* mengalikan Matriks tetap dengan Matriks hasil *ShiftRow*

$$\begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \begin{pmatrix} c5 & e5 & af & 2f \\ 3b & 2f & c5 & 2b \\ a5 & f1 & 01 & 18 \\ f1 & 29 & ad & fe \end{pmatrix} = \begin{pmatrix} 88 & 78 & bd & c5 \\ b6 & 9a & 90 & af \\ a7 & 48 & 84 & 2d \\ 33 & b8 & 6f & a5 \end{pmatrix}$$

Kemudian hasil *MixColumns* di *XOR* kan dengan *RoundKey*

$$\begin{pmatrix} 88 & 78 & bd & c5 \\ b6 & 9a & 90 & af \\ a7 & 48 & 84 & 2d \\ 33 & b8 & 6f & a5 \end{pmatrix} \oplus \begin{pmatrix} a3 & 8c & e3 & c3 \\ cd & ed & 83 & e1 \\ c6 & 95 & f2 & 9d \\ db & ab & ce & ac \end{pmatrix}$$

Kemudian mendapatkan hasil *AddRoundKey* sebagai berikut

$$\begin{pmatrix} 6b & f4 & 5e & 06 \\ 7b & 77 & 13 & 4e \\ 61 & dd & 76 & b0 \\ e8 & 13 & a1 & 09 \end{pmatrix}$$

AES - *Round* 2

Dari hasil *Round* 1, hasil *AddRoundKey* nya diproses ke tahap *SubBytes*

$$\begin{pmatrix} 7f & bf & 58 & 6f \\ 21 & f5 & 7d & 2f \\ ef & c1 & 38 & e7 \\ 9b & 7d & 32 & 01 \end{pmatrix}$$

Kemudian *ShiftRows*

$$\begin{pmatrix} 7f & bf & 58 & 6f \\ f5 & 7d & 2f & 21 \\ 38 & e7 & ef & c1 \\ 01 & 9b & 7d & 32 \end{pmatrix}$$

Kemudian lanjut ke proses *MixColumns*

$$\begin{pmatrix} c3 & 9e & 53 & 4e \\ c7 & ec & 51 & 47 \\ f9 & a1 & 35 & 81 \\ 4e & 6d & d2 & 35 \end{pmatrix}$$

Kemudian hasil *MixColumns* di *XOR* kan dengan *RoundKey*

$$\begin{pmatrix} c3 & 9e & 53 & 4e \\ c7 & ec & 51 & 47 \\ f9 & a1 & 35 & 81 \\ 4e & 6d & d2 & 35 \end{pmatrix} \oplus \begin{pmatrix} 19 & 95 & 76 & b5 \\ 93 & 7e & fd & 1c \\ 57 & c2 & 30 & ad \\ f5 & 5e & 90 & 3c \end{pmatrix}$$

Kemudian mendapatkan hasil *AddRoundKey* sebagai berikut

$$\begin{pmatrix} da & 0b & 25 & fb \\ 54 & 92 & ac & 5b \\ ae & 63 & 05 & 2c \\ bb & 33 & 42 & 09 \end{pmatrix}$$

AES - *Round* 3

Dari hasil *Round* 2, hasil *AddRoundKey* nya diproses ke tahap *SubBytes*

$$\begin{pmatrix} 57 & 2b & 3f & 0f \\ 20 & 4f & 91 & 39 \\ e4 & fb & 6b & 71 \\ ea & c3 & 2c & 01 \end{pmatrix}$$

Kemudian *ShiftRows*

$$\begin{pmatrix} 57 & 2b & 3f & 0f \\ 4f & 91 & 39 & 20 \\ 6b & 71 & e4 & fb \\ 01 & ea & c3 & 2c \end{pmatrix}$$

Kemudian lanjut ke proses *MixColumns*

$$\begin{pmatrix} 15 & 65 & 12 & a9 \\ 75 & 6b & b9 & 75 \\ cd & 7d & 8b & b6 \\ df & 52 & 01 & 92 \end{pmatrix}$$

Kemudian hasil *MixColumns* di *XOR* kan dengan *RoundKey*

$$\begin{pmatrix} 15 & 65 & 12 & a9 \\ 75 & 6b & b9 & 75 \\ cd & 7d & 8b & b6 \\ df & 52 & 01 & 92 \end{pmatrix} \oplus \begin{pmatrix} 81 & 14 & 62 & d7 \\ 06 & 78 & 85 & 99 \\ bc & 7e & 4e & e3 \\ 20 & 7e & ee & d2 \end{pmatrix}$$

Kemudian mendapatkan hasil *AddRoundKey* sebagai berikut

$$\begin{pmatrix} 94 & 71 & 70 & 7e \\ 73 & 13 & 3c & ec \\ 71 & 03 & c5 & 55 \\ ff & 2c & ef & 40 \end{pmatrix}$$

AES - *Round* 4

Dari hasil *Round* 3, hasil *AddRoundKey* nya diproses ke tahap *SubBytes*

$$\begin{pmatrix} 22 & a3 & 51 & f3 \\ 8f & 7d & eb & ce \\ a3 & 7b & a6 & fc \\ 16 & 71 & df & 09 \end{pmatrix}$$

Kemudian *ShiftRows*

$$\begin{pmatrix} 22 & a3 & 51 & f3 \\ 7d & eb & ce & 8f \\ a6 & fc & a3 & 7b \\ 09 & 16 & 71 & df \end{pmatrix}$$

Kemudian lanjut ke proses *MixColumns*

$$\begin{pmatrix} 6c & 91 & 39 & d3 \\ 20 & 67 & 59 & a4 \\ 13 & 91 & 51 & f0 \\ af & c5 & 7c & 5f \end{pmatrix}$$

Kemudian hasil *MixColumns* di *XOR* kan dengan *RoundKey*

$$\begin{pmatrix} 6c & 91 & 39 & d3 \\ 20 & 67 & 59 & a4 \\ 13 & 91 & 51 & f0 \\ af & c5 & 7c & 5f \end{pmatrix} \oplus \begin{pmatrix} 67 & 73 & 11 & c6 \\ 17 & 6f & ea & 73 \\ 09 & 77 & 39 & da \\ 2e & 50 & be & 6c \end{pmatrix}$$

Kemudian mendapatkan hasil *AddRoundKey* sebagai berikut

$$\begin{pmatrix} 0b & e2 & 28 & 15 \\ 37 & 08 & b3 & d7 \\ 1a & e6 & 68 & 2a \\ 81 & 95 & c2 & 33 \end{pmatrix}$$

AES - *Round* 5

Dari hasil *Round* 4, hasil *AddRoundKey* nya diproses ke tahap *SubBytes*

$$\begin{pmatrix} 2b & 98 & 34 & 59 \\ 9a & 30 & 6d & 0e \\ a2 & 8e & 45 & e5 \\ 0c & 2a & 25 & c3 \end{pmatrix}$$

Kemudian *ShiftRows*

$$\begin{pmatrix} 2b & 98 & 34 & 59 \\ 30 & 6d & 0e & 9a \\ 45 & e5 & a2 & 8e \\ c3 & 0c & 2a & 25 \end{pmatrix}$$

Kemudian lanjut ke proses *MixColumns*

$$\begin{pmatrix} 80 & 75 & f2 & ac \\ 47 & 7a & ff & da \\ cf & 30 & 1b & ab \\ 95 & 23 & a4 & b5 \end{pmatrix}$$

Kemudian hasil *MixColumns* di *XOR* kan dengan *RoundKey*

$$\begin{pmatrix} 80 & 75 & f2 & ac \\ 47 & 7a & ff & da \\ cf & 30 & 1b & ab \\ 95 & 23 & a4 & b5 \end{pmatrix} \oplus \begin{pmatrix} f8 & 8b & 9a & 5c \\ 40 & 2f & c5 & b6 \\ 59 & 2e & 17 & cd \\ 9a & ca & 74 & 18 \end{pmatrix}$$

Kemudian mendapatkan hasil *AddRoundKey* sebagai berikut

$$\begin{pmatrix} 78 & fe & 68 & f0 \\ 07 & 55 & 3a & 6c \\ 96 & 1e & 0c & 66 \\ 0f & e9 & d0 & ad \end{pmatrix}$$

AES - *Round* 6

Dari hasil *Round* 5, hasil *AddRoundKey* nya diproses ke tahap *SubBytes*

$$\begin{pmatrix} bc & bb & 45 & 8c \\ c5 & fc & 80 & 50 \\ 90 & 72 & fe & 33 \\ 76 & 1e & 70 & 95 \end{pmatrix}$$

Kemudian *ShiftRows*

$$\begin{pmatrix} bc & bb & 45 & 8c \\ fc & 80 & 50 & c5 \\ fe & 33 & 90 & 72 \\ 95 & 76 & 1e & 70 \end{pmatrix}$$

Kemudian lanjut ke proses *MixColumns*

$$\begin{pmatrix} 17 & be & f4 & 55 \\ d3 & 83 & 50 & fb \\ 03 & c7 & 0c & 3d \\ ec & 89 & 33 & d8 \end{pmatrix}$$

Kemudian hasil *MixColumns* di *XOR* kan dengan *RoundKey*

$$\begin{pmatrix} 17 & be & f4 & 55 \\ d3 & 83 & 50 & fb \\ 03 & c7 & 0c & 3d \\ ec & 89 & 33 & d8 \end{pmatrix} \oplus \begin{pmatrix} 96 & 1d & 87 & db \\ fd & d2 & 17 & a1 \\ f4 & da & cd & 00 \\ d0 & 1a & 6e & 76 \end{pmatrix}$$

Kemudian mendapatkan hasil *AddRoundKey* sebagai berikut

$$\begin{pmatrix} 81 & ae & 73 & 8e \\ 2e & 51 & 47 & 5a \\ f7 & 1d & c1 & 3d \\ 3c & 93 & 5d & ae \end{pmatrix}$$

AES - *Round* 7

Dari hasil *Round* 6, hasil *AddRoundKey* nya diproses ke tahap *SubBytes*

$$\begin{pmatrix} 0c & e4 & 8f & 19 \\ 31 & d1 & a0 & be \\ 68 & a4 & 78 & 27 \\ eb & dc & 4c & e4 \end{pmatrix}$$

Kemudian *ShiftRows*

$$\begin{pmatrix} 0c & e4 & 8f & 19 \\ d1 & a0 & be & 31 \\ 78 & 27 & 68 & a4 \\ e4 & eb & dc & 4c \end{pmatrix}$$

Kemudian lanjut ke proses *MixColumns*

$$\begin{pmatrix} ec & e4 & 68 & 89 \\ d9 & 3d & 8c & c0 \\ 1a & 2c & 9e & af \\ 6e & 7d & ff & 26 \end{pmatrix}$$

Kemudian hasil *MixColumns* di *XOR* kan dengan *RoundKey*

$$\begin{pmatrix} ec & e4 & 68 & 89 \\ d9 & 3d & 8c & c0 \\ 1a & 2c & 9e & af \\ 6e & 7d & ff & 26 \end{pmatrix} \oplus \begin{pmatrix} e4 & f9 & 7e & a5 \\ 9e & 44 & 5b & fa \\ cc & 16 & db & db \\ 69 & 73 & 1d & 6b \end{pmatrix}$$

Kemudian mendapatkan hasil *AddRoundKey* sebagai berikut

$$\begin{pmatrix} 08 & 1d & 16 & 2c \\ 47 & 71 & d7 & 3a \\ d6 & 3a & 45 & 74 \\ 07 & 0e & e2 & 4d \end{pmatrix}$$

AES - *Round* 8

Dari hasil *Round* 7, hasil *AddRoundKey* nya diproses ke tahap *SubBytes*

$$\begin{pmatrix} 30 & a4 & 47 & 71 \\ a0 & a3 & 0e & 80 \\ f6 & 80 & 6e & 92 \\ c5 & ab & 98 & e3 \end{pmatrix}$$

Kemudian *ShiftRows*

$$\begin{pmatrix} 30 & a4 & 47 & 71 \\ a3 & 0e & 80 & a0 \\ 6e & 92 & f6 & 80 \\ e3 & c5 & ab & 98 \end{pmatrix}$$

Kemudian lanjut ke proses *MixColumns*

$$\begin{pmatrix} 13 & 16 & 48 & 01 \\ 3c & d0 & f6 & 29 \\ 71 & c1 & d6 & 79 \\ 40 & fa & f2 & 98 \end{pmatrix}$$

Kemudian hasil *MixColumns* di *XOR* kan dengan *RoundKey*

$$\begin{pmatrix} 13 & 16 & 48 & 01 \\ 3c & d0 & f6 & 29 \\ 71 & c1 & d6 & 79 \\ 40 & fa & f2 & 98 \end{pmatrix} \oplus \begin{pmatrix} 49 & b0 & ce & 6b \\ 27 & 6b & 30 & ca \\ b3 & a5 & 7e & a5 \\ 6f & 1c & 01 & 6a \end{pmatrix}$$

Kemudian mendapatkan hasil *AddRoundKey* sebagai berikut

$$\begin{pmatrix} 5a & a6 & 86 & 6a \\ 1b & bb & c6 & e3 \\ c2 & 64 & a8 & dc \\ af & e6 & f3 & f2 \end{pmatrix}$$

AES – *Round* 9

Dari hasil *Round* 8, hasil *AddRoundKey* nya di proses ke tahap *SubBytes*

$$\begin{pmatrix} be & 24 & 44 & 02 \\ af & ea & b4 & 11 \\ 25 & 43 & c2 & 86 \\ 15 & 8e & 0d & 89 \end{pmatrix}$$

Kemudian *ShiftRows*

$$\begin{pmatrix} be & 24 & 44 & 02 \\ ea & b4 & 11 & af \\ c2 & 86 & 25 & 43 \\ 89 & 15 & 8e & 0d \end{pmatrix}$$

Kemudian lanjut ke proses *MixColumns*

$$\begin{pmatrix} 09 & 1c & 10 & a0 \\ a5 & d3 & 87 & 8f \\ 4b & b8 & 98 & 3c \\ f8 & 74 & ff & f0 \end{pmatrix}$$

Kemudian hasil *MixColumns* di *XOR* kan dengan *RoundKey*

$$\begin{pmatrix} 09 & 1c & 10 & a0 \\ a5 & d3 & 87 & 8f \\ 4b & b8 & 98 & 3c \\ f8 & 74 & ff & f0 \end{pmatrix} \oplus \begin{pmatrix} 26 & 96 & 58 & 33 \\ 21 & 4a & 7a & b0 \\ b1 & 14 & 6a & cf \\ 10 & 0c & 0d & 67 \end{pmatrix}$$

Kemudian mendapatkan hasil *AddRoundKey* sebagai berikut

$$\begin{pmatrix} 2f & 8a & 48 & 93 \\ 84 & 99 & fd & 3f \\ fa & ac & fc & f3 \\ e8 & 78 & f2 & 98 \end{pmatrix}$$

AES - *Round* 10

Dari hasil *Round* 9, hasil *AddRoundKey* nya di proses ke tahap *SubBytes*

$$\begin{pmatrix} 15 & 7e & 52 & dc \\ 5f & ee & 54 & 75 \\ 2d & 91 & b0 & 0d \\ 9b & bc & 89 & 88 \end{pmatrix}$$

Kemudian *ShiftRows*

$$\begin{pmatrix} 15 & 7e & 52 & dc \\ ee & 54 & 75 & 5f \\ b0 & 0d & 2d & 91 \\ 88 & 9b & bc & 89 \end{pmatrix}$$

Kemudian hasil *ShiftRows* di *XOR* kan dengan *RoundKey*

$$\begin{pmatrix} 15 & 7e & 52 & dc \\ ee & 54 & 75 & 5f \\ b0 & 0d & 2d & 91 \\ 88 & 9b & bc & 89 \end{pmatrix} \oplus \begin{pmatrix} f7 & 61 & 39 & 0a \\ ab & e1 & 9b & 2b \\ 34 & 20 & 4a & 85 \\ d3 & df & d2 & b5 \end{pmatrix}$$

Kemudian mendapatkan hasil *AddRoundKey* sebagai berikut

$$\begin{pmatrix} e2 & 1f & 6b & d6 \\ 45 & b5 & ee & 74 \\ 84 & 2d & 67 & 14 \\ 5b & 44 & 6e & 3c \end{pmatrix}$$

Kemudian didapat *ciphertext*: e2 45 84 5b 1f b5 2d 44 6b ee 67 6e d6 74 14 3c

2.4.2 Perhitungan dekripsi algoritme AES

Kemudian untuk proses dekripsinya, transformasi *cipher* dapat dibalikkan dan diimplementasikan dalam arah yang berlawanan untuk menghasilkan *inverse cipher* yang mudah dipahami untuk algoritme AES. Transformasi *byte* yang digunakan pada invers *cipher* adalah *InvShiftRows*, *InvSubBytes*, *InvMixColumns*, dan *AddRoundKey*.

AES*InvAddRoundKey*

$$\begin{pmatrix} e2 & 1f & 6b & d6 \\ 45 & b5 & ee & 74 \\ 84 & 2d & 67 & 14 \\ 5b & 44 & 6e & 3c \end{pmatrix} \oplus \begin{pmatrix} f7 & 61 & 39 & 0a \\ ab & e1 & 9b & 2b \\ 34 & 20 & 4a & 85 \\ d3 & df & d2 & b5 \end{pmatrix}$$

XOR antara chiperteks hasil enkripsi dengan *key Round* 10

e2⊕f7=15	45⊕ab=ee	84⊕34=b0	5b⊕d3=88
1f⊕61=7e	b5⊕e1=54	2d⊕20=0d	44⊕df=9b
6b⊕39=52	ee⊕9b=75	67⊕4a=2d	6e⊕d2=bc
d6⊕0a=dc	74⊕2b=5f	14⊕85=91	3c⊕b5=89

Kemudian hasil *InvAddRoundKey* adalah

$$\begin{pmatrix} 15 & 7e & 52 & dc \\ ee & 54 & 75 & 5f \\ b0 & 0d & 2d & 91 \\ 88 & 9b & bc & 89 \end{pmatrix}$$

AES – *Round* 9

Untuk *InvShiftRow* digeser ke kiri. Jadi di baris pertama tidak digeser, baris ke 2 digeser 1x, baris ke 3 digeser 2x, kemudian baris terakhir digeser 3x.

$$\begin{pmatrix} 15 & 7e & 52 & dc \\ ee & 54 & 75 & 5f \\ b0 & 0d & 2d & 91 \\ 88 & 9b & bc & 89 \end{pmatrix} \rightarrow \begin{pmatrix} 15 & 7e & 52 & dc \\ 5f & ee & 54 & 75 \\ 2d & 91 & b0 & 0d \\ 9b & bc & 89 & 88 \end{pmatrix}$$

Setelah di *InvShiftRow* kemudian ke tahap *InvSubBytes*, *InvSubBytes* dilakukan pemecahan dilakukan perubahan nilai ASCII Hex sesuai dengan tabel invSBOX pada Tabel 2.2.

$$\begin{pmatrix} 2f & 8a & 48 & 93 \\ 84 & 99 & fd & 3f \\ fa & ac & fc & f3 \\ e8 & 78 & f2 & 97 \end{pmatrix}$$

Tabel 2.2 Tabel InvSBOX

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	52	9	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	FFA	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	0	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	F0	78	CD	5A	F4
C	1F	DD	A8	33	88	7	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	4	7E	BA	77	D6	26	E1	69	14	63	55	21	0C	7D

Kemudian hasil *InvSubBytes* di *XOR* dengan *InvRoundKey*

$$\begin{pmatrix} 2f & 8a & 48 & 93 \\ 84 & 99 & fd & 3f \\ fa & ac & fc & f3 \\ e8 & 78 & f2 & 97 \end{pmatrix} \oplus \begin{pmatrix} 26 & 96 & 58 & 33 \\ 21 & 4a & 7a & b0 \\ b1 & 14 & 6a & cf \\ 10 & 0c & 0d & 67 \end{pmatrix}$$

2f⊕26=09 84⊕21=a5 fa⊕b1=4b e8⊕10=f8

8a⊕96=1c 99⊕4a=d3 ac⊕14=b8 78⊕0c=74

48⊕fd=10 fd⊕7a=87 fc⊕6a=96 f2⊕0d=ff

93⊕33=a0 3f⊕b0=8f f3⊕cf=3c 97⊕67=f0

Kemudian didapat hasil *InvAddRoundKey* sebagai berikut

$$\begin{pmatrix} 09 & 1c & 10 & a0 \\ a5 & d3 & 87 & 8f \\ 4b & b8 & 96 & 3c \\ f8 & 74 & ff & f0 \end{pmatrix}$$

Kemudian lanjut proses *InvMixColumns* dan mendapatkan hasil sebagai berikut

$$\begin{pmatrix} be & 24 & 44 & 02 \\ ea & b4 & 11 & af \\ c2 & 86 & 25 & 43 \\ 89 & 15 & 8e & 0d \end{pmatrix}$$

AES – *Round* 8

Dari hasil *InvMixColumns* dari *Round* 9, dilakukan *InvShiftRows*

$$\begin{pmatrix} be & 24 & 44 & 02 \\ af & ea & b4 & 11 \\ 25 & 43 & c2 & 86 \\ 15 & 8e & 0d & 89 \end{pmatrix}$$

Kemudian *InvSubBytes*

$$\begin{pmatrix} 5a & a6 & 86 & 6a \\ 1b & bb & c6 & e3 \\ c2 & 64 & a8 & dc \\ 2f & e6 & f3 & f2 \end{pmatrix}$$

Kemudian hasil *InvSubBytes* di *XOR* kan dengan *InvRoundKey*

$$\begin{pmatrix} 5a & a6 & 86 & 6a \\ 1b & bb & c6 & e3 \\ c2 & 64 & a8 & dc \\ 2f & e6 & f3 & f2 \end{pmatrix} \oplus \begin{pmatrix} 49 & b0 & ce & 6b \\ 27 & 6b & 30 & ca \\ b3 & a5 & 7e & a5 \\ 6f & 1c & 01 & 6a \end{pmatrix}$$

Kemudian mendapatkan hasil *InvAddRoundKey* sebagai berikut

$$\begin{pmatrix} 13 & 16 & 48 & 01 \\ 3c & d0 & f6 & 29 \\ 71 & c1 & d6 & 79 \\ 40 & fa & f2 & 98 \end{pmatrix}$$

Kemudian lanjut ke proses *InvMixColumns* dan mendapatkan hasil sebagai berikut

$$\begin{pmatrix} 30 & a4 & 47 & 71 \\ a3 & 0e & 80 & a0 \\ 6e & 92 & f6 & 80 \\ e3 & c5 & ab & 98 \end{pmatrix}$$

AES – *Round* 7

Dari hasil *InvMixColumns* dari *Round* 8, dilakukan *InvShiftRows*

$$\begin{pmatrix} 30 & a4 & 47 & 71 \\ a0 & a3 & 0e & 80 \\ f6 & 80 & 6e & 92 \\ c5 & ab & 98 & e3 \end{pmatrix}$$

Kemudian *InvSubBytes*

$$\begin{pmatrix} 08 & 1d & 16 & 2c \\ 47 & 71 & d7 & 3a \\ d6 & 3a & 45 & 74 \\ 07 & 0e & e2 & 4d \end{pmatrix}$$

Kemudian hasil *InvSubBytes* di *XOR* kan dengan *InvRoundKey*

$$\begin{pmatrix} 08 & 1d & 16 & 2c \\ 47 & 71 & d7 & 3a \\ d6 & 3a & 45 & 74 \\ 07 & 0e & e2 & 4d \end{pmatrix} \oplus \begin{pmatrix} e4 & f9 & 7e & a5 \\ 9e & 4c & 5b & fa \\ cc & 16 & db & db \\ 69 & 73 & 1d & 6b \end{pmatrix}$$

Kemudian mendapatkan hasil *InvAddRoundKey* sebagai berikut

$$\begin{pmatrix} ec & e4 & 68 & 89 \\ d9 & 3d & 8c & c0 \\ 1a & 2c & 9e & af \\ 6e & 7d & ff & 26 \end{pmatrix}$$

Kemudian lanjut ke proses *InvMixColumns* dan mendapatkan hasil sebagai berikut

$$\begin{pmatrix} 0c & e4 & 8f & 19 \\ d1 & a0 & be & 31 \\ 78 & 27 & 68 & a4 \\ e4 & eb & dc & 4c \end{pmatrix}$$

AES – *Round* 6

Dari hasil *InvMixColumns* dari *Round* 7, dilakukan *InvShiftRows*

$$\begin{pmatrix} 0c & e4 & 8f & 19 \\ 31 & d1 & a0 & be \\ 68 & a4 & 78 & 27 \\ eb & dc & 4c & e4 \end{pmatrix}$$

Kemudian *InvSubBytes*

$$\begin{pmatrix} 81 & ae & 73 & 8e \\ 2e & 51 & 47 & 5a \\ f7 & 1d & c1 & 3d \\ 3c & 93 & 5d & ae \end{pmatrix}$$

Kemudian hasil *InvSubBytes* di *XOR* kan dengan *InvRoundKey*

$$\begin{pmatrix} 81 & ae & 73 & 8e \\ 2e & 51 & 47 & 5a \\ f7 & 1d & c1 & 3d \\ 3c & 93 & 5d & ae \end{pmatrix} \oplus \begin{pmatrix} 96 & 1d & 87 & db \\ fd & d2 & 17 & a1 \\ f4 & da & cd & 00 \\ d0 & 1a & 6e & 76 \end{pmatrix}$$

Kemudian mendapatkan hasil *InvAddRoundKey* sebagai berikut

$$\begin{pmatrix} 17 & b3 & f4 & 55 \\ d3 & 83 & 50 & fb \\ 03 & c7 & 0c & 3d \\ ec & 89 & 33 & d8 \end{pmatrix}$$

Kemudian lanjut ke proses *InvMixColumns* dan mendapatkan hasil sebagai berikut

$$\begin{pmatrix} bc & bb & 45 & 8c \\ fc & 80 & 50 & c5 \\ fe & 33 & 90 & 72 \\ 95 & 76 & 1e & 70 \end{pmatrix}$$

AES – *Round* 5

Dari hasil *InvMixColumns* dari *Round* 6, dilakukan *InvShiftRows*

$$\begin{pmatrix} bc & bb & 45 & 8c \\ c5 & fc & 80 & 50 \\ 90 & 72 & fe & 33 \\ 76 & 1e & 70 & 95 \end{pmatrix}$$

Kemudian *InvSubBytes*

$$\begin{pmatrix} 78 & fe & 68 & f0 \\ 07 & 55 & 3a & 6c \\ 96 & 1e & 0c & 66 \\ 0f & e9 & d0 & ad \end{pmatrix}$$

Kemudian hasil *InvSubBytes* di *XOR* kan dengan *InvRoundKey*

$$\begin{pmatrix} 78 & fe & 68 & f0 \\ 07 & 55 & 3a & 6c \\ 96 & 1e & 0c & 66 \\ 0f & e9 & d0 & ad \end{pmatrix} \oplus \begin{pmatrix} f8 & 8b & 9a & 5c \\ 40 & 2f & c5 & b6 \\ 59 & 2e & 17 & cd \\ 9a & ca & 74 & 18 \end{pmatrix}$$

Kemudian mendapatkan hasil *InvAddRoundKey* sebagai berikut

$$\begin{pmatrix} 80 & 75 & f2 & ac \\ 47 & 7a & ff & da \\ cf & 30 & 1b & ab \\ 95 & 23 & a4 & b5 \end{pmatrix}$$

Kemudian lanjut ke proses *InvMixColumns* dan mendapatkan hasil sebagai berikut

$$\begin{pmatrix} 2b & 98 & 34 & 59 \\ 30 & 6d & 0e & 9a \\ 45 & e5 & a2 & 8e \\ c3 & 0c & 2a & 25 \end{pmatrix}$$

AES – *Round* 4

Dari hasil *InvMixColumns* dari *Round* 5, dilakukan *InvShiftRows*

$$\begin{pmatrix} 2b & 98 & 34 & 59 \\ 9a & 30 & 6d & 0e \\ a2 & 8e & 45 & e5 \\ 0c & 2a & 25 & c3 \end{pmatrix}$$

Kemudian *InvSubBytes*

$$\begin{pmatrix} 0b & e2 & 28 & 15 \\ 37 & 08 & b3 & d7 \\ 1a & e6 & 68 & 2a \\ 81 & 95 & c2 & 33 \end{pmatrix}$$

Kemudian hasil *InvSubBytes* di *XOR* kan dengan *InvRoundKey*

$$\begin{pmatrix} 0b & e2 & 28 & 15 \\ 37 & 08 & b3 & d7 \\ 1a & e6 & 68 & 2a \\ 81 & 95 & c2 & 33 \end{pmatrix} \oplus \begin{pmatrix} 67 & 73 & 11 & c6 \\ 17 & 6f & ea & 73 \\ 09 & 77 & 39 & da \\ 2e & 50 & be & 6c \end{pmatrix}$$

Kemudian mendapatkan hasil *InvAddRoundKey* sebagai berikut

$$\begin{pmatrix} 6c & 91 & 39 & d3 \\ 20 & 67 & 59 & a4 \\ 13 & 91 & 51 & f0 \\ af & c5 & 7c & 5f \end{pmatrix}$$

Kemudian lanjut ke proses *InvMixColumns* dan mendapatkan hasil sebagai berikut

$$\begin{pmatrix} 22 & a3 & 51 & f3 \\ 7d & eb & ce & 8f \\ a6 & fc & a3 & 7b \\ 09 & 16 & 71 & df \end{pmatrix}$$

AES – *Round* 3

Dari hasil *InvMixColumns* dari *Round* 4, dilakukan *InvShiftRows*

$$\begin{pmatrix} 22 & a3 & 51 & f3 \\ 8f & 7d & eb & ce \\ a3 & 7b & a6 & fc \\ 16 & 71 & df & 09 \end{pmatrix}$$

Kemudian *InvSubBytes*

$$\begin{pmatrix} 94 & 71 & 70 & 7e \\ 73 & 13 & 3c & ec \\ 71 & 03 & c5 & 55 \\ ff & 2c & ef & 40 \end{pmatrix}$$

Kemudian hasil *InvSubBytes* di *XOR* kan dengan *InvRoundKey*

$$\begin{pmatrix} 94 & 71 & 70 & 7e \\ 73 & 13 & 3c & ec \\ 71 & 03 & c5 & 55 \\ ff & 2c & ef & 40 \end{pmatrix} \oplus \begin{pmatrix} 81 & 14 & 62 & d7 \\ 06 & 78 & 85 & 99 \\ bc & 7e & 4e & e3 \\ 20 & 7e & ee & d2 \end{pmatrix}$$

Kemudian mendapatkan hasil *InvAddRoundKey* sebagai berikut

$$\begin{pmatrix} 15 & 65 & 12 & a9 \\ 75 & 6b & b9 & 75 \\ cd & 7d & 8b & b6 \\ df & 52 & 01 & 92 \end{pmatrix}$$

Kemudian lanjut ke proses *InvMixColumns* dan mendapatkan hasil sebagai berikut

$$\begin{pmatrix} 57 & 2b & 3f & 0f \\ 4f & 91 & 39 & 20 \\ 6b & 71 & e4 & fb \\ 01 & ea & c3 & 2c \end{pmatrix}$$

AES – *Round* 2

Dari hasil *InvMixColumns* dari *Round* 3, dilakukan *InvShiftRows*

$$\begin{pmatrix} 57 & 2b & 3f & 0f \\ 20 & 4f & 91 & 39 \\ e4 & fb & 6b & 71 \\ ea & c3 & 2c & 01 \end{pmatrix}$$

Kemudian *InvSubBytes*

$$\begin{pmatrix} da & 0b & 25 & fb \\ 54 & 92 & ac & 5b \\ ae & 63 & 05 & 2c \\ bb & 33 & 42 & 09 \end{pmatrix}$$

Kemudian hasil *InvSubBytes* di *XOR* kan dengan *InvRoundKey*

$$\begin{pmatrix} da & 0b & 25 & fb \\ 54 & 92 & ac & 5b \\ ae & 63 & 05 & 2c \\ bb & 33 & 42 & 09 \end{pmatrix} \oplus \begin{pmatrix} 19 & 95 & 76 & b5 \\ 93 & 7e & fd & 1c \\ 57 & c2 & 30 & ad \\ f5 & 5e & 90 & 3c \end{pmatrix}$$

Kemudian mendapatkan hasil *InvAddRoundKey* sebagai berikut

$$\begin{pmatrix} c3 & 9e & 53 & 4e \\ c7 & ec & 51 & 47 \\ f9 & a1 & 35 & 81 \\ 4e & 6d & d2 & 35 \end{pmatrix}$$

Kemudian lanjut ke proses *InvMixColumns* dan mendapatkan hasil sebagai berikut

$$\begin{pmatrix} 7f & bf & 58 & 6f \\ f5 & 7d & 2f & 21 \\ 38 & e7 & ef & c1 \\ 01 & 9b & 7d & 32 \end{pmatrix}$$

AES – *Round* 1

Dari hasil *InvMixColumns* dari *Round* 2, dilakukan *InvShiftRows*

$$\begin{pmatrix} 7f & bf & 58 & 6f \\ 21 & f5 & 7d & 2f \\ ef & c1 & 38 & e7 \\ 9b & 7d & 32 & 01 \end{pmatrix}$$

Kemudian *InvSubBytes*

$$\begin{pmatrix} 6b & f4 & 5e & 06 \\ 7b & 77 & 13 & 4e \\ 61 & dd & 76 & b0 \\ e8 & 13 & a1 & 09 \end{pmatrix}$$

Kemudian hasil *InvSubBytes* di *XOR* kan dengan *InvRoundKey*

$$\begin{pmatrix} 6b & f4 & 5e & 06 \\ 7b & 77 & 13 & 4e \\ 61 & dd & 76 & b0 \\ e8 & 13 & a1 & 09 \end{pmatrix} \oplus \begin{pmatrix} e3 & 8c & e3 & c3 \\ cd & ed & 83 & e1 \\ c6 & 95 & f2 & 9d \\ db & ab & ce & ac \end{pmatrix}$$

Kemudian mendapatkan hasil *InvAddRoundKey* sebagai berikut

$$\begin{pmatrix} 88 & 78 & bd & c5 \\ b6 & 9a & 90 & af \\ a7 & 48 & 84 & 2d \\ 33 & b8 & 6f & a5 \end{pmatrix}$$

Kemudian lanjut ke proses *InvMixColumns* dan mendapatkan hasil sebagai berikut

$$\begin{pmatrix} c5 & e5 & af & 2f \\ 3b & 2f & c5 & 2b \\ a5 & f1 & 01 & 18 \\ f1 & 29 & ad & fe \end{pmatrix}$$

AES – *Round* 0

Dari hasil *InvMixColumns* dari *Round* 1, dilakukan *InvShiftRows*

$$\begin{pmatrix} c5 & e5 & af & 2f \\ 2b & 3b & 2f & c5 \\ 01 & 18 & a5 & f1 \\ 29 & ad & fe & f1 \end{pmatrix}$$

Kemudian *InvSubBytes*

$$\begin{pmatrix} 07 & 2a & 1b & 4e \\ 0b & 49 & 4e & 07 \\ 09 & 34 & 29 & 2b \\ 4c & 18 & 0c & 2b \end{pmatrix}$$

Kemudian hasil *InvSubBytes* di *XOR* kan dengan *InvRoundKey*

$$\begin{pmatrix} 07 & 2a & 1b & 4e \\ 0b & 49 & 4e & 07 \\ 09 & 34 & 29 & 2b \\ 4c & 18 & 0c & 2b \end{pmatrix} \oplus \begin{pmatrix} 48 & 6f & 6f & 20 \\ 65 & 20 & 6e & 62 \\ 6c & 53 & 67 & 6f \\ 6c & 70 & 65 & 62 \end{pmatrix}$$

Kemudian mendapatkan hasil *InvAddRoundKey* sebagai berikut

$$\begin{pmatrix} 4f & 45 & 74 & 6e \\ 6e & 69 & 20 & 65 \\ 65 & 67 & 4e & 44 \\ 20 & 68 & 69 & 49 \end{pmatrix}$$

Setelah proses dekripsi selesai akan didapat kembali *plaintext* : 4f 6e 65 20 45 69 67 68 74 20 4e 69 6e 65 44 49

2.5 Analisa Statistika

2.5.1 Pengujian *Kruskal Wallis*

Pengujian *Kruskal-Wallis* adalah metode pengujian statistik *non-parametric* yang dapat digunakan untuk menguji apakah sejumlah sampel data berasal dari distribusi yang sama. Metode ini dapat digunakan untuk membandingkan dua atau lebih sampel data yang memiliki ukuran yang sama atau berbeda satu sama lain. Hasil keluaran dari pengujian ini adalah nilai signifikansi. Apabila nilai signifikan sampel yang diuji bernilai sama atau lebih dari 0,05 (level signifikasi) maka dapat diambil kesimpulan bahwa sampel yang diuji terdisitribusi normal.

Karena pengujian ini merupakan pengujian non parametris di mana asumsi normalitas boleh dilanggar, maka tidak perlu lagi ada pengujian normalitas, misal pengujian *Saphiro Wilk* atau *Liliefors. Saphiro Wilk* adalah sebuah metode uji normalitas yang efektif dan valid digunakan untuk sempel berjumlah kecil. *Liliefors* adalah metode yang menggunakan data dasar yang belum diolah dalam tabel distribusi frekuensi.

2.5.1.1 Asumsi dan Hipotesis Uji *Kruskal Wallis*

Asumsi dari uji *Kruskal Wallis* berupa:

1. Data yang dianalisis terdiri lebih dari 2 ampel acak ($k_1, k_2 \dots, k_n$)
2. Skala data yang digunakan minimum adalah ordinal
3. Variabel yang diamati harus *continue*
4. Jenis skala untuk variabel dependen adalah ordinal

Kemudian untuk hipotesis yang digunakan untuk uji *Kruskal WallisI* adalah ada tidaknya perbedaan dari beberapa kelompok populasi yang diamati. Katakanlah satu variabel mewakili satu populasi sehingga terdapat beberapa populasi yang diamati. Maka hopotesisnya terhadap populasi ke-k.

2.5.1.2 Rumus *Kruskal Wallis*

Berikut di bawah ini adalah rumus dari *Kruskal Wallis*:

$$H = \frac{12}{N(N+1)} \sum_{i=1}^{k} \frac{r_i^2}{ni} - 3(N+1) \qquad (2.1)$$

k = banyaknya sampel

n_i = banyaknya kasus pada setiap sampel ke-i

R_i = Total ranking untuk setiap sampel ke-i

N = $\sum ni$ = Jumlah banyaknya seluruh kasus.

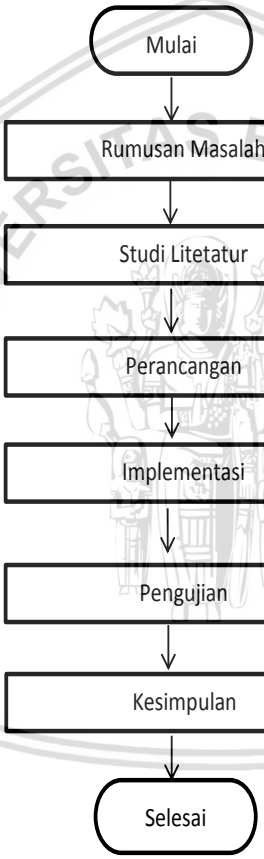
$\sum_{i=1}^{k}$ = menunjukan penjumlahan seluruh k sampel (kolom-kolom) mendekati distribusi Chi square dengan db = $k-1$ untuk ukuran sampel sebesar n yang cukup besar.

BAB 3 METODOLOGI

Penelitian tentang analisis algoritme untuk pengamanan data merupakan penelitian implementatif dalam pendekatan perancangan. Yang nantinya akan memberikan rekomendasi hasil algoritme untuk pengamanan data yang memberikan performa terbaik.

3.1. Tahapan Penelitian

Pada penelitian ini ada beberapa tahapan mulai dari rumusan masalah, studi literature, perancangan, implementasi, pengujian, dan kesimpulan yang ditunjukan pada Gambar 3.1.



Gambar 3.1 Diagram Alir Tahapan Penelitian

3.2. Rumusan Masalah

Permasalahan pada penelitian ini adalah bagaimana mengamankan data. Berdasarkan penjelasan pada sub-bab 1.1 menunjukkan bahwa adanya kemungkinan ancaman yang diberikan saat penyimpanan data. Menurut (Bhardwaj, 2016) algoritme enkripsi merupakan metode pengamanan data dengan melakukan pengkodean. Pada penelitian yang dilakukan Jeva *et al* (2012) dilakukan analisis algoritme enkripsi, dimana menurut hasil analisisnya

menunjukkan bahwa algoritme AES menunjukkan algoritme yang direkomendasikan sebagai algoritme untuk mengamankan penggunaan *file.*

3.3. Studi Literatur

Dalam tahapan penelitian ini pertama yaitu studi literatur yang dilakukan dengan mencari, dasar-dasar teori dan sumber acuan analisis algoritme untuk pengamanan datayang nantinya agar penelitian ini dapat memberikan rekomendasi hasil algoritme enkripsi terbaik. Penulis melakukan pencarian referensi di perpustakaan, ruang baca, jurnal, penjelasan dari dosen pembimbing dalam penyelesaian permasalahan ini.

3.4. Perancangan

Pada bagian ini akan melakukan perancangan algoritme AES untuk pengamanan data, yang nantinya akan dilakukan perancangan secara detail tentang aplikasi yang akan digunakan untuk melakukan analisis metode pengamanan data. Perancangan ini akan menunjang untuk proses implementasi aplikasi yang akan dibangun untuk analisis metode terhadap beberapa format *file*.

3.5. Implementasi

Tahap implementasi sistem ini akan dilakukan beberapa hal. Pertama adalah menerapkan algoritme AES untuk enkripsi *file*. Lalu akan akan dibangun aplikasi untuk analisis hasil enkripsi dan dekripsi.

3.6. Pengujian

Pada tahap pengujian sistem ini penulis menggunakan data yang telah didapatkan. Data tersebut nantinya akan digunakan sebagai masukan dari sistem dan nantinya akan dilakukan proses enkripsi menggunakan algoritme AES. Pengujian perangkat lunak tersebut dilakukan agar dapat membuktikan bahwa hasil berupa perangkat lunak (*software*) tersebut telah mampu bekerja dengan baik sesuai dengan kebutuhan.

3.7. Kesimpulan

Langkah terakhir adalah pengambilan kesimpulan berdasarkan pengujian sistem. Dalam pengujian sistem didapatkan algoritme AES adalah algoritme pengamanan data yang bisa mengamankan berbagai maam format *file* yang akan dijadikan rekomendasi bagi pembangun sistem penyimpanan data secara *virtual*.

BAB 4 PERANCANGAN

Pada bagian ini akan dibahas secara detail perancangan sistem untuk pengujian algoritme untuk pengamanan data menggunakan algoritme AES. Perancangan ini akan melakukan pembahasan proses mekanisme sistem pengamanan mulai dari enkripsi dan dekripsi yang dilakukan algoritme AES.

Perancangan sistem ini dibagi menjadi beberapa bagian, mulai dari analisa kebutuhan, mekanisme proses, dan perancangan *interface*. Analisa kebutuhan berisi kebutuhan yang dibutuhkan oleh pengguna dan sistem. Makanisme proses akan membahas secara mendalam proses enkripsi oleh algoritme AES serta bagaimana sistem dapat berjalan sesuai dengan semestinya. Sedangkan perancangan *interface* akan membahas tentang perancangan antar muka yang akan digunakan untuk melakukan analisi algoritme enkripsi.

4.1 Perancangan Sistem

Perancangan sistem terdapat dua hal yang menjadi focus. Pertamaperancangan dari fungsi yang digunakan oleh pengguna dan perancangan darisistem. Perancangan fungsi menjelaskan secara garis besar tentang fungsi-fungsi dari sistem yang dapat digunakan oleh pengguna dan perancangan sistem menjelaskan secara detail tentang sistem itu harus berjalan.

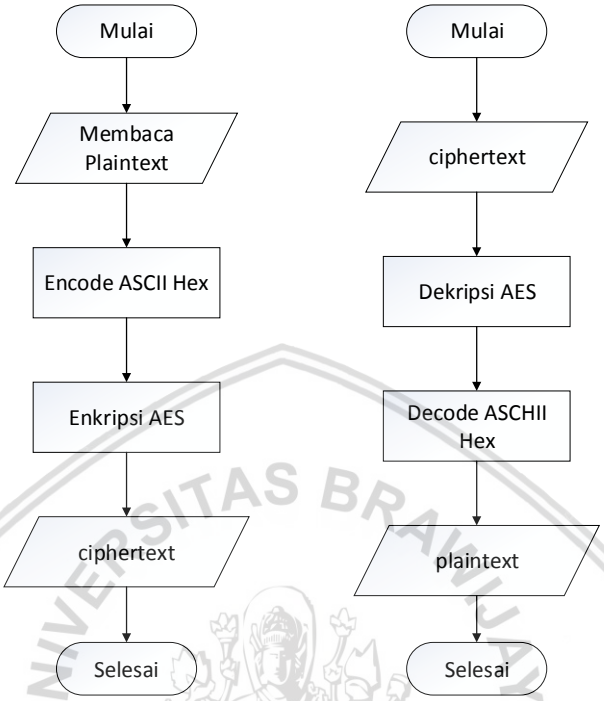
4.1.1 Perancangan Fungsi

Perancangan fungsi sebagai berikut:

1. *User* pertama memilih *file* yang akan di enkripsi, kemudian di *encode* menggunakan alat bantu peng*encode file*.
2. *User* pertama dapat mengirim *file* hingga 100 MB, *file* secara otomatis akan terenkripsi (*ciphertext*).
3. *File* yang diterima *user* kedua dalam bentuk terdekripsi (*plaintext*).
4. *User* ke dua dapat mengunduh *file* yang telah terdekripsi dari *user* kedua.

4.1.2 Perancangan Sistem

Perancangan sistem berfungsi menjelaskan fungsi-fungsi yang harus disediakan oleh sistem sesuai mekanisme proses didalam sistem.



Gambar 4.1 Diagram alir proses enkripsi dan dekripsi

Pada diagram alir diatas menjelaskan bagaimana proses enkripsi dan dekripsi berlangsung. Pertama proses enkripsi dilakukan dengan membaca *file* yang akan dienkripsi lalu dilakukan *encode* menggunakan ASCII Hex seletah itu dilakukan enkripsi menggunakan algorithma AES dan memberikan hasil enkripsinya. Sedangkan proses dekripsi dilakukan dengan membaca hasil enkripsi lalu melakukan *decode* hasil tersebut dan melakukan dekripsi menggunakan AES sehingga mendapatkan hasil dari dekripsi.

4.2 Mekanisme Proses

Pada bagian ini akan dibahas tentang proses enkripsi dekripsi menggunakan metode AES. Selain itu juga akan dibahan parameter yang digunakan untuk analisis algoritme enkripsi.

4.2.1 Analisis Metode

Pada bagian ini dibahas parameter yang digunakan untuk analisis dan pengujian dari algoritme enkripsi AES. Beberapa parameter tersebut antara lain (Bhardwaj *et al*, 2016; Jeeva *et al*, 2012):

1. *Cost* enkripsi

   Cost enkripsi adalah seberapa banyak memori yang digunakan untuk proses enkripsi.

2. Waktu komputasi

   Waktu komputasi sangat berpengaruh besar terhadap penggunaan metode, metode enkripsi dapat melakukan enkripsi dengan hasil yang maksimal tetapi waktu komputasinya sangatlah lama sehingga metode tersebut tidak mungkin akan digunakan.

3. Format *File*

   Melakukan uji menggunakan 4 format *file* yang berbeda, yaitu menggunakan tipe format *file* dan data berupa dokumen (.txt dan .docx), foto (.jpeg dan .png), audio (.mp3 dan .wav), dan video (.mp4 dan .avi).

4. Ukuran Data

   Menyesuaikan ukuran dari tiap format data dengan berbagai macam ukuran tipe format data yang digunakan.

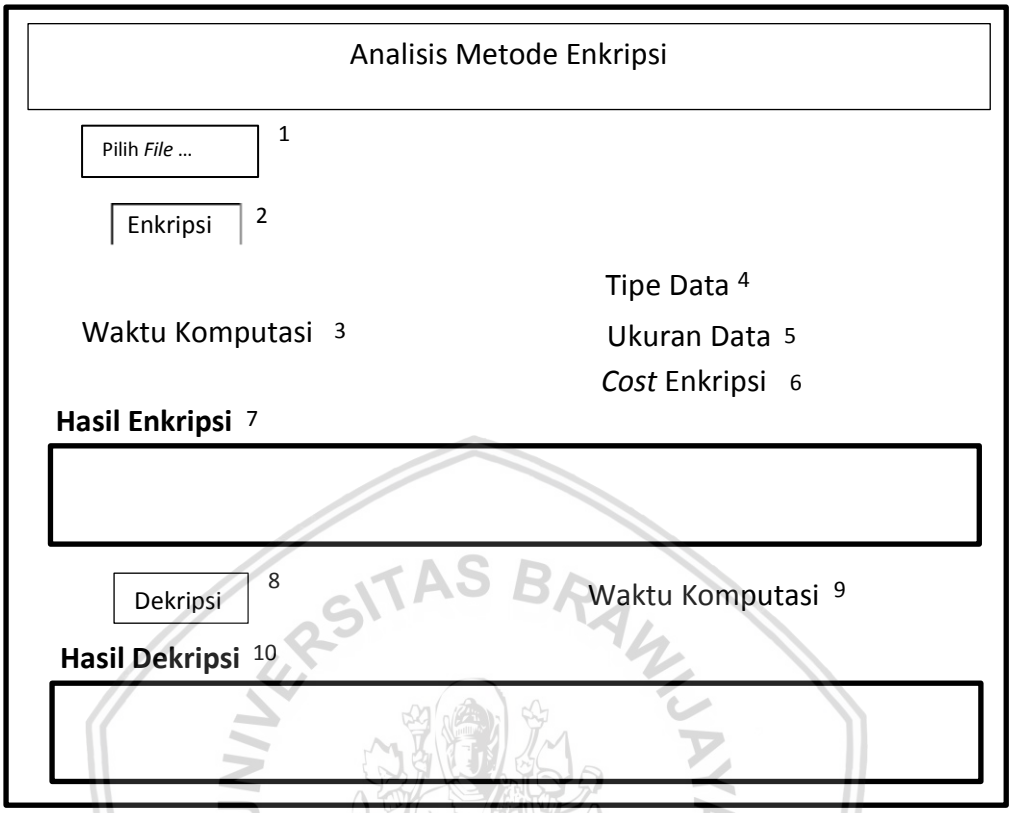
4.3 Perancangan Skenario Pengujian

Berdasarkan rumusan masalah yang telah dijelaskan di bab 1, maka tujuan pengujian yang ingin dicapai adalah membandingkan hasil komputasi metode AES dengan menggunakan format *file* berupa dokumen, gambar, audio, dan video. Kemudian dari setiap format *file* yang diujikan, yaitu format *file* dokumen menggunakan tipe data .txt dan .docx, kemudian format *file* gambar menggunakan tipe data .jpeg dan .png, lalu untuk format *file* audio menggunakan tipe data .mp3 dan .wav, dan yang terakhir format *file* video menggunakan tipe data .mp4 dan .avi.

4.3.1 Pengujian

Dari tiap format *file* yang di gunakan akan dilakukan pengujian pada parameter waktu komputasi untuk enkripsi, dekripsi, panjang *ciphertext*, serta *cost* yang dihasilkan untuk setiap proses. Parameter tersebut akan dujikan untuk setiap tipe data yaitu teks, gambar, audio, dan video. Selain itu untuk audio dan video juga dilihat dari lama waktu audio dan videonya.

4.4 Perancangan Antarmuka



Gambar 4.2 Perancangan antarmuka program

Keterangan:

- 1: *button* untuk memilih *file* yang akan di enkripsi
- 2: *button* untuk menjalankan enrkipsi menggunakan algorithma AES
- 3: teks yang menunjukkan hasil waktu komputasi dari enkripsi
- 4: teks yang menunjukkan tipe data dari *file*yang dienkripsi
- 5: teks yang menunjukkan ukuran data dari *file*yang dienkripsi
- 6: teks yang menunjukkan hasil *cost* dari enkripsi yang telah dilakukan
- 7: hasil dari proses enkripsi
- 8: *button* untuk melakukan proses dekripsi
- 9: teks yang menunjukkan hasil waktu komputasi dari dekripsi
- 10: hasil dari proses dekripsi

BAB 5 IMPLEMENTASI

5.1 Hasil Implementasi Program

Pada bagian ini akan ditunjukan hasil implementasi program untuk menguji algoritme enkripsi dan dekripsi yang dibangun menggunakan Java dengan *library* metode AES. Berdasarkan hal tersebut maka metode AES telah diimplementasikan dalam program Java. Berikut adalah beberapa potongan program; yaitu proses enkripsi, proses dekripsi, dan menampilkan hasil ke antarmuka.

Tabel 5.1 Tabel kode program untuk menampilkan hasil enkripsi

Algoritme 1: Potongan program untuk menampilkan hasil enkripsi

```
1  private void jButtonEnkripsiActionPerformed(java.awt.event.ActionEvent
2  evt) {
3  byte[] keyByte;
4          byte[] plaintextByte;
5          String encrypted = null;
6          keyByte = text.getBytes();
7          plaintextByte = isi_text.getBytes();
8          double start = Sistem.nanoTime();
9              try {
10                 ciphertext = encrypt(keyByte, plaintextByte);
11             } catch (Exception ex) {
12                 Logger.getLogger(GUI.class.getName()).log(Level.SEVERE,
   null, ex);
13             }
14             encrypted = new String(ciphertext);
15         jTextHasilEnkripsi.setText(encrypted);
16         double finish = Sistem.nanoTime();
17         double time = (finish - start) / 1000000000;
18
19 jLabelValuePanjangFile.setText(String.valueOf(encrypted.length()));
20         jLabelValueWaktuKomputasi.setText(String.valueOf(time) + " S");
21         jLabelValueTipeData.setText(jenis_file);
22         jLabelValueUkuranData.setText(ukuran_file);
23         double cost = time * 1000;
24         jLabelValueCostEnkripsi.setText(String.valueOf(cost));
25         Sistem.out.println();
26         Sistem.out.println(String.valueOf(encrypted.length()));
27         Sistem.out.println(String.valueOf(time));
```

```
28          Sistem.out.println(ukuran_file);
29          Sistem.out.println(String.valueOf(cost));
30  }
```

Tabel 5.1 merupakan potongan kode program untuk proses enkripsi saat tombol perintah enkripsi ditekan. Kode ini melakukan pembacaan isi text kemudian dari isi text tersebut di ubah menjadi bentuk *byte*. Setelah didapatkan *key* dan *plaintext* berbentuk *byte*, dilakukan proses enkripsi dengan memanggil metode encrypt() dimana *key* dan *plaintext* berbentuk *byte* tsb menjadi parameternya. Setelah dilakukan proses enkripsi, hasil dari proses tersebut dicetak pada komponen jTextHasilEnkripsi. Penjelasan dari Kode 5.3 adalah sebagai berikut:

1. jButtonEnkripsiActionPerformed merupakan fungsi yang dijalankan ketika tombol 'Enkripsi' ditekan.
2. Baris 3-8 merupakan kode inisialisasi variabel yang diperlukan dalam proses enkripsi.
3. Baris 9-13 memanggil fungsi *encrypt()* kemudian menyimpan nilai kembalian ke dalam variable *ciphertext* dan menampilkan pesan *error* ke dalam *console* apabila terdapat kegagalan proses *encrypt().*
4. Baris 14-15 melakukan konversi nilai variabel *ciphertext* dari tipe data *byte* menjadi tipe data *String yang* kemudian ditampilkan pada *User Interface* (UI).
5. Baris 16-17 menghitung waktu yang dibutuhkan selama proses *encrypt()* berjalan.
6. Baris 18-24 menampilkan tipe *file,* ukuran file, dan *cost* proses enkripsi ke UI.
7. Baris 25-30 menampilkan tipe *file*, ukuran *file*, dan *cost* proses enkripsi ke *console.*Kode ini merupakan proses enkripsi saat tombol perintah enkripsi ditekan. Kode ini pertama-tama melakukan pembacaan isi text kemudian dari isi text tersebut di ubah menjadi bentuk *byte*. Setelah didapatkan *key* dan *plaintext* berbentuk *byte*, dilakukan proses enkripsi dengan memanggil metode encrypt() dimana *key* dan *plaintext* berbentuk *byte* tsb menjadi parameternya. Setelah dilakukan proses enkripsi, hasil dari proses tersebut dicetak pada komponen jTextHasilEnkripsi.

Tabel 5.2 Tabel kode program fungsi enkripsi AES

Algoritme 2: Potongan program untuk proses enkripsi AES	
1	`private byte[] encrypt(byte[] raw, byte[] clear) throws Exception {`
2	`SecretKeySpec skeySpec = new SecretKeySpec(raw, "AES");`
3	`Cipher cipher = Cipher.getInstance("AES");`
4	`cipher.init(Cipher.ENCRYPT_MODE, skeySpec);`

```
5           byte[] encrypted = cipher.doFinal(clear);
6           return encrypted;
7       }
```

Tabel 5.2 merupakan potongan kode program proses enkripsi dimana data mentah diubah menjadi *ciphertext.* Penjelasan dari Kode 5.4 adalah sebagai berikut:

1. *encrypt()* adalah fungsi utama proses enkripsi, didalamnya data masukan dari *user* dikonversi menjadi *ciphertext* menggunakan algoritme AES-128.
2. Baris 2 adalah kode generasi *secretKey* untuk data masukan menggunakan algoritme AES.
3. Baris 3-4 menginisialisasi kelas *Cipher* dan mengatur jenis enkripsi menjadi AES.
4. Baris 5-6 menyimpan hasil konversi masukan menjadi *ciphertext* kedalam variabel *encrypted* kemudian mengembalikan nilai *encrypted* ke pemanggil fungsi *encrypt().*

Tabel 5.3 Tabel kode program untuk menampilkan hasil dekripsi

Algoritme 3: Potongan program untuk menampilkan hasil dekripsi

```
1   private void jButtonDekripsiActionPerformed(java.awt.event.ActionEvent
2   evt) {
3   String plaintextDec = null;
4           byte[] plaintextByte;
5           byte[] keyByte;
6           keyByte = text.getBytes();
7           double start = Sistem.nanoTime();
8           try {
9                   plaintextByte = decrypt(keyByte, ciphertext);
10                  plaintextDec = new String(plaintextByte);
11              } catch (Exception ex) {
12                  Logger.getLogger(GUI.class.getName()).log(Level.SEVERE,
    null, ex);
13              }
14          double finish = Sistem.nanoTime();
15          double time = (finish - start) / 1000000000;
16          jLabel12.setText(String.valueOf(time) + " S");
17          Sistem.out.println();
```

```
18          Sistem.out.println("Decryption");
19          Sistem.out.println(String.valueOf(time));
20          jTextHasilDeskripsi.setText(plaintextDec);
21  }
22
```

Kode 5.5 merupakan potongan kode program untukk proses dekripsi saat tombol perintah dekripsi ditekan. Kode ini membaca *chipertext* hasil enkripsi kemudian dan mengembalikan data menjadi data sebenarnya. Penjelasan Kode 5.5 adalah sebagai berikut:

1. *jButtonDekripsiActionPerformed* adalah fungsi yang dijalankan ketika tombol 'Dekripsi' pada UI ditekan.
2. Baris 3-7 merupakan kode inisialisasi variabel yang dibutuhkan untuk proses dekripsi.
3. Baris 8-14 memanggil fungsi *decrypt()* dimana hasil disimpan dalam variable *plaintextByte*, kemudian dikonversi ke tipe data *String* dan disimpan ke variabel *plaintextDec.*
4. Baris 15-17 menghitung waktu proses dekripsi kemudian menampilkan nilainya ke UI.
5. Baris 18-22 menampilkan teks hasil dekripsi ke UI.

Tabel 5.4 Tabel kode programproses dekripsi

Algoritme 4: Potongan program proses dekripsi	
1	`private byte[] decrypt(byte[] raw, byte[] encrypted) throws Exception {`
2	`SecretKeySpec skeySpec = new SecretKeySpec(raw, "AES");`
3	`Cipher cipher = Cipher.getInstance("AES");`
4	`cipher.init(Cipher.DECRYPT_MODE, skeySpec);`
5	`byte[] decrypted = cipher.doFinal(encrypted);`
6	`return decrypted;`
7	`}`

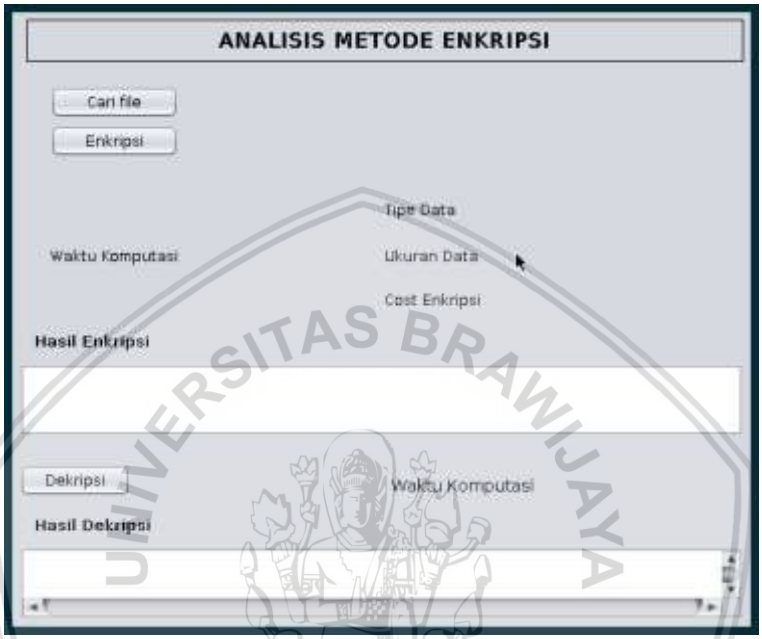
Kode 5.6 adalah potongan kode utama proses dekripsi dimana didalamnya dilakukan konversi dari *ciphertext* ke teks asli *(plaintext)*. Penjelasan Kode 5.6 adalah sebagai berikut:

1. *decrypt()* adalah fungsi utama untuk melakukan konversi *ciphertext* ke *plaintext.*
2. Baris 2 adalah kode generasi secret key untuk proses dekripsi AES.
3. Baris 3-4 melakukan inisialisasi kelas *Cipher* dan mengatur mode dekripsi menggunakan algoritme AES.

4. Baris 5-6 menyimpan hasil dekripsi ke variabel *decrypted* dan mengembalikannya ke pemanggil fungsi *decrypt().*

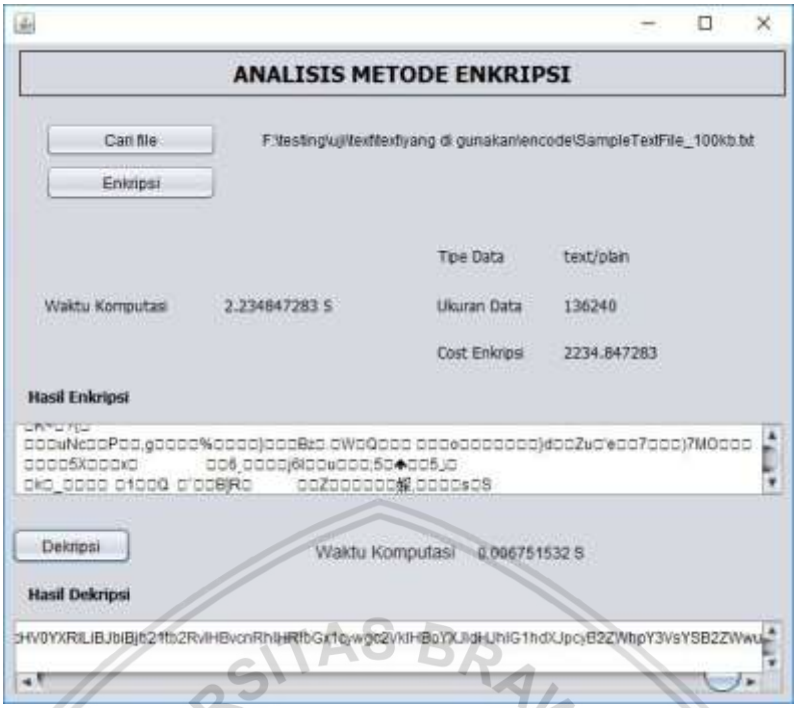
5.2 Implementasi Antarmuka

Antarmuka program yang dibangun sesuai dengan perancangan antarmuka yang telah dibahas pada sub-bab sebelumnya yang ditunjukan pada **Error! eference source not found.**.



Gambar 5.1 Hasil implementasi program

Setelah dilakukan percobaan *running* program untuk metode AES dengan teks Lampiran 1 yang digunakan untuk membangkitkan *plaintext* yang digunakan ditunjukan pada **Gambar 5.2**.



Gambar 5.2 Hasil proses enkripsi metode AES

BAB 6 PENGUJIAN DAN ANALISIS

6.1 Pengujian Validasi

Pengujian validasi enkripsi dan dekripsi digunakan untuk memastikan *file* yang di enkripsi sesuai dengan algoritme yang digunakan. Untuk mendapatkan hasil validasi tersebut dapat dibuktikan dengan cara mencocokkan hasil dekripsi dengan *plaintext* sebelum dilakukan proses enkripsi.

6.1.1 Pengujian enkripsi *file*

Dalam Tabel 6.1, dilakukan pengujian enkripsi *file* dengan prosedur menerima *plaintext* dan *key* lalu melakukan proses enkripsi dan mengembalikan hasil proses enkripsinya. Hasil yang diharapkan pada pengujian enkripsi *file* adalah program berhasil mengembalikan hasil nilai proses enkripsi. Hasil yang didapatkan dari pengujian sesuai dengan hasil yang diharapkan, sehingga pengujian enkripsi *file* dapat dinyatakan valid.

Tabel 6.1 Pengujian enkripsi *file*

Nama Kasus Uji	Melakukan proses enkripsi *file*
Prosedur	Menerima *plaintext* dan *key* lalu melakukan tahapan enkripsi dan memberikan hasil enkripsinya
Hasil yang diharapkan	Program akan memberikan pengembalian hasil enkripsi *file*
Hasil	Program memberikan pengembalian hasil enkripsi *file*
Status	Valid

6.1.2 Pengujian dekripsi *file*

Dalam Tabel 6.2, dilakukan pengujian dekripsi *file* dengan prosedur menerima *chipertext* dan *key* lalu melakukan proses dekripsi dan memberikan hasil dekripsinya. Hasil yang diharapkan pada pengujian dekripsi *file* adalah program bisa memberikan pengembalian hasil nilai proses dekripsi. Kemudian hasil yang didapatkan dari pengujian sesuai dengan hasil yang diharapkan, sehingga pengujian dekripsi *file* dapat dinyatakan valid.

Tabel 6.2 Pengujian dekripsi *file*

Nama Kasus Uji	Melakukan proses dekripsi *file*
Prosedur	Menerima *ciphertext* dan *key* lalu melakukan tahapan dekripsi dan memberikan hasil dekripsinya
Hasil yang diharapkan	Program akan memberikan pengembalian hasil dekripsi *file*
Hasil	Program memberikan pengembalian hasil dekripsi *file*
Status	Valid

6.1.3 Pengujian menampilkan hasil enkripsi

Pada Tabel 6.3 menunjukkan proses menampilkan hasil enkripsi dengan harapan program menjalankan prosedur enkripsi dan menerima hasilnya lalu program akan menampilkan detail hasil enkripsi antarmuka. Yang ditampilkan di antarmuka antara lain: panjang *ciphertext*, waktu komputasi, tipe data, ukuran data *cost* enkripsi, dan hasil dari dekripsi. Setelah hasil yang dibutuhkan sesuai dengan yang diharapkan disitulah dinyatakan valid.

Tabel 6.3 Pengujian menampilkan hasil enkripsi

Nama Kasus Uji	Melakukan proses menampilkan hasil enkripsi
Prosedur	Menjalankan prosedur enkripsi dan menerima hasilnya lalu program akan menampilkan detail hasil enkripsi ke antarmuka
Hasil yang diharapkan	Program akan menampilkan detail enkripsi ke antarmuka antara lain: panjang *ciphertext*, waktu komputasi, tipe data, ukuran data, *cost* enkripsi, dan hasil dari enkripsi.
Hasil	Program akan menampilkan detail enkripsi ke antarmuka antara lain: panjang *ciphertext*, waktu komputasi, tipe data, ukuran data, *cost* enkripsi, dan hasil dari enkripsi.
Status	Valid

6.1.4 Pengujian menampilkan hasil dekripsi

Pada Tabel 6.4 menunjukkan proses menampilkan hasil dekripsi dengan harapan program menjalankan prosedur dekripsi dan menerima hasilnya lalu program akan menampilkan detail hasil enkripsi antarmuka. Yang ditampilkan di

antarmuka adalah program menampilkan detail waktu komputasi hasil dari dekripsi. Setelah hasil yang di butuhkan sesuai dengan yang di harapkan disitulah dinyatakan valid.

Tabel 6.4 Pengujian Menampilkan Hasil Dekripsi

Nama Kasus Uji	Melakukan proses menampilkan hasil dekripsi
Prosedur	Menjalankan prosedur dekripsi dan menerima hasilnya lalu program akan menampilkan detail hasil enkripsi ke antarmuka
Hasil yang diharapkan	Program akan menampilkan detail dekripsi ke antarmuka waktu komputasi hasil dari dekripsi.
Hasil	Program akan menampilkan detail dekripsi ke antarmuka waktu komputasi hasil dari dekripsi.
Status	Valid

6.2 Analisis Hasil Pengujian AES

Pada bagian ini akan dilakukan analisis algoritme berdasarkan hasil implementasi pada sub-bab sebelumnya. Parameter analisis yang digunakan telah dibahas secara detail pada bab sebelumnya antara lain *cost* enkripsi, waktu enkripsi, waktu dekripsi, dan ukuran data.

Berdasarkan hasil tersebut ada beberapa perbedaan nilai pada beberapa parameter, yaitu waktu komputasi. Pada parameter *cost* enkripsi, metode memberikan hasil yaitu ringan, artinya ketika program dijalankan akan membutuhkan memori yang ringan atau sedikit untuk melakukan komputasinya. Pada parameter panjang *ciphertext* metode AES menghasilkan enkripsi yang panjang. Jika dilihat dari selisih panjang *ciphertext* sangat besar, hal ini menunjukkan perbedaan yang cukup terlihat. Metode AES miliki panjang *ciphertext* yang bernilai besar artinya metode AES menghasilkan enkripsi yang komplek. Pada parameter terahkir, yaitu ukuran data, metode AES memiliki nilai besar. Hal ini menunjukkan bahwa metode AES dapat melakukan proses enkripsi untuk data yang berukuran besar.

Selain itu, akan dilakukan perbandingan pengujian ukuran *file* yang akan dilakukan enkripsi. Mulai ukuran data yang menyesuaikan dari tiap *file* yang ditunjukan pada Tabel 6.5 sampai Tabel 6.8. Perbedaan ukuran yang dilakukan pengujian sudah dapat menyimpulkan perbandingan hasil enkripsi dari metode AES.

Tabel 6.5 Hasil Pengujian AES pada Tipe Data Dokumen

Pengujian	tipe: Dokumen			
	.txt		.docx	
	Waktu komputasi Enkripsi	Waktu komputasi Dekripsi	Waktu komputasi Enkripsi	Waktu komputasi Dekripsi
100 KB	3,8007	0,0201	3,5549	0,0209
200 KB	6,5755	0,0149	6,1324	0,0114
500 KB	14,8771	0,0228	11,8714	0,0375
1000 KB	21,8642	0,06007	21,3581	0,0529
5000 KB	63,7808	0,03810	64,9484	0,03688

Tabel menunjukkan hasil proses AES untuk enkripsi dan dokumen dimana semakin besar proses data yang digunakan maka waktu komputasi cenderung semaking lama, hasil waktu komputasi dalam satuan detik. Selain itu proses untuk tipe ".txt" dan ".docs" tidak terdapat perbedaan yang signifikan.

Tabel 6.6 Hasil Pengujian AES pada Tipe Data Gambar

Pengujian	tipe: Gambar			
	.jpeg		.png	
	Waktu komputasi Enkripsi	Waktu komputasi Dekripsi	Waktu komputasi Enkripsi	Waktu komputasi Dekripsi
100 KB	3,8522	0,0205	4,2719	0,0079
200 KB	5,8409	0,0302	8,25007	0,0148
500 KB	11,8999	0,0355	11,7897	0,0341
1000 KB	22,2847	0,0597	23,0577	0,0568
5000 KB	68,6701	0,0611	67,8505	0,0602

Tabel menunjukkan hasil proses AES untuk enkripsi gambar sama dengan hasil pada tipe data dokumen dimana semakin besar proses data yang digunakan maka waktu komputasi cenderung semaking lama, hasil waktu komputasi dalam satuan detik. Selain itu proses untuk tipe ".jpeg" dan ".png" tidak terdapat perbedaan yang signifikan.

Tabel 6.7 Hasil Pengujian AES pada Tipe Data Audio

Pengujian	tipe: Audio					
	.mp3			.wav		
	Ukuran Data	Waktu komputasi Enkripsi	Waktu komputasi Dekripsi	Ukuran Data	Waktu komputasi Enkripsi	Waktu komputasi Dekripsi
1 menit	310868	4,0965	0,0088	77568	2,7271	0,0148
2 menit	629908	7,1420	0,0143	157288	4,6996	0,0190
3 menit	961768	11,057	0,0216	240232	5,8502	0,0237
4 menit	1308676	15,349	0,0283	327012	8,3345	0,0252
5 menit	1597344	17,418	0,0298	399148	8,3626	0,0368

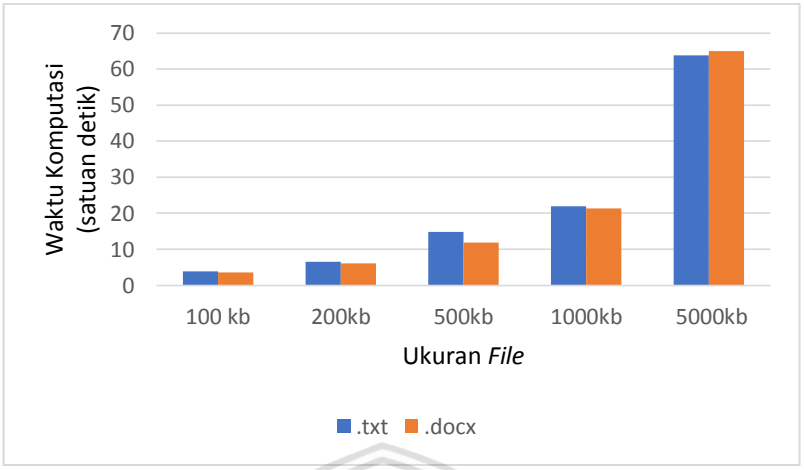
Tabel menunjukna hasil proses AES untuk enkripsi audio sama dengan hasil pada tipe data dokumen dimana semakin besar proses data yang digunakan maka waktu komputasi cenderung semaking lama, hasil waktu komputasi dalam satuan detik. Selain itu proses untuk tipe ".mp3" membutuhkan waktu lebih lama dibandingkan dengan tipe data ".wav".

Tabel 6.8 Hasil Pengujian AES pada Tipe Data Video

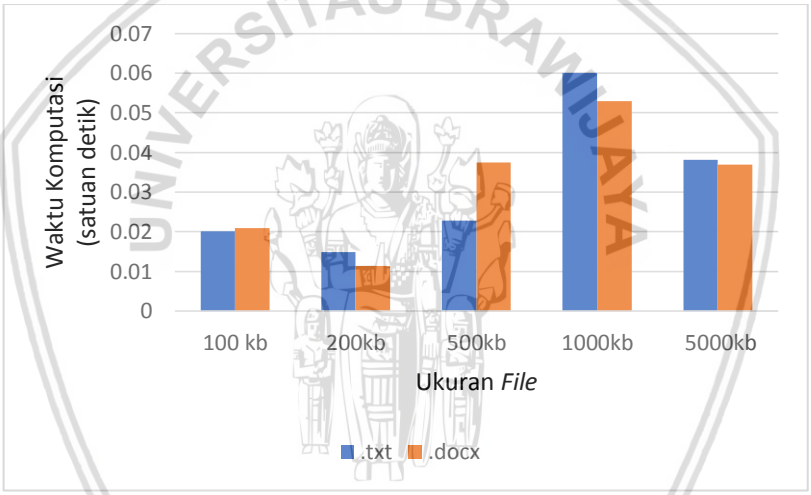
Pengujian	tipe: Video					
	.mp4			.mkv		
	Ukuran Data	Waktu komputasi Enkripsi	Waktu komputasi Dekripsi	Ukuran Data	Waktu komputasi Enkripsi	Waktu komputasi Dekripsi
1 menit	1243082	18,6202	0,0547	252672	5,4728	0,0195
2 menit	2950626	49,4840	1,9993	602312	10,7091	0,0328
3 menit	4424920	75,9905	1,0789	906998	17,7913	0,0481
4 menit	5703640	57,9287	0,0371	11544763	12,1534	0,0251
5 menit	5034272	47,9863	1,0733	885664	9,49857	0,0192

Tabel menunjukkan hasil proses AES untuk enkripsi video dimana tipe data ".mp4" membutuhkan waktu yang lebih lama dibandingkan tipe data ".mkv" hal ini dipengaruhi oleh ukuran data ".mp4" cenderung lebih besar data lama video yang sama, hasil waktu komputasi dalam satuan detik.

Analisis pengujian AES lebih dalam akan ditunjukan pada Gambar 6.1 sampai Gambar 6.8

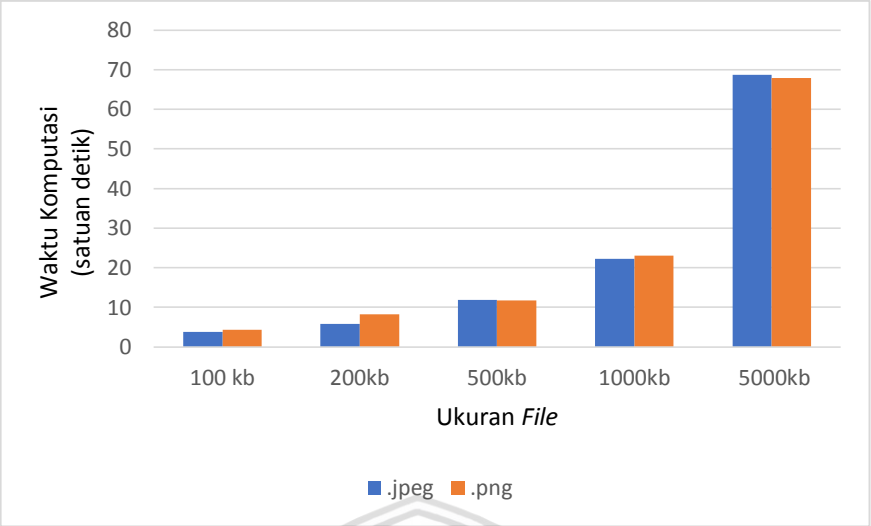


Gambar 6.1 Perbandingan Tipe Data Dokumen Berdasarkan Waktu Enkripsi

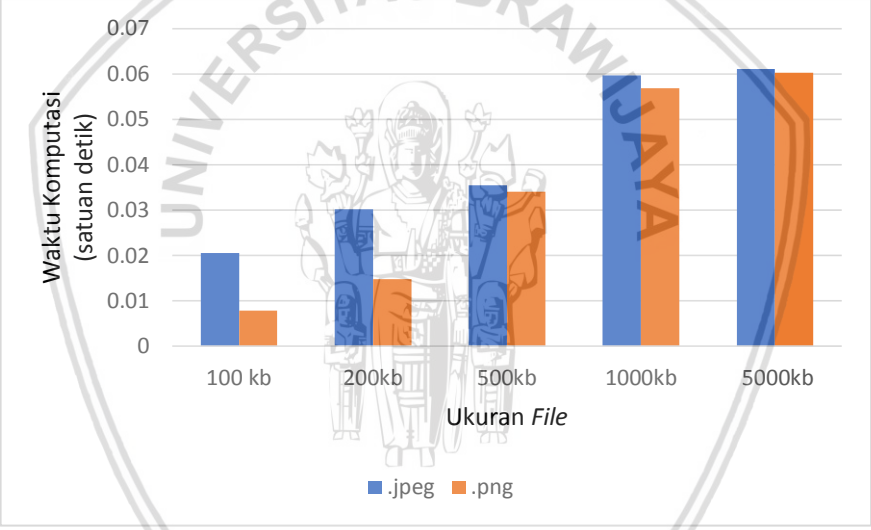


Gambar 6.2 Perbandingan Tipe Data Dokumen Berdasarkan Waktu Dekripsi

Berdasarkan grafik menunjukkan bahwa tipe data ".docx" lebih cepat waktu enkripsinya dibandingkan tipe data ".txt" untuk semua ukuran data kecuali pada ukuran data 5000 KB. Sedangkan pada waktu dekripsi kedua tipe data tidak memiliki perbedaan yang signifikan kecuali pada ukuran data 500 KB tipe data ".txt" memiliki waktu yang lebih cepat.

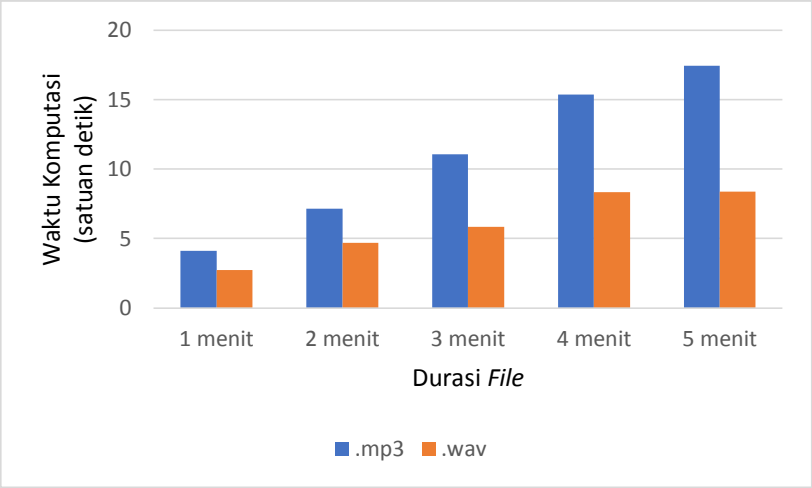


Gambar 6.3 Perbandingan Tipe Data Gambar Berdasarkan Waktu Enkripsi

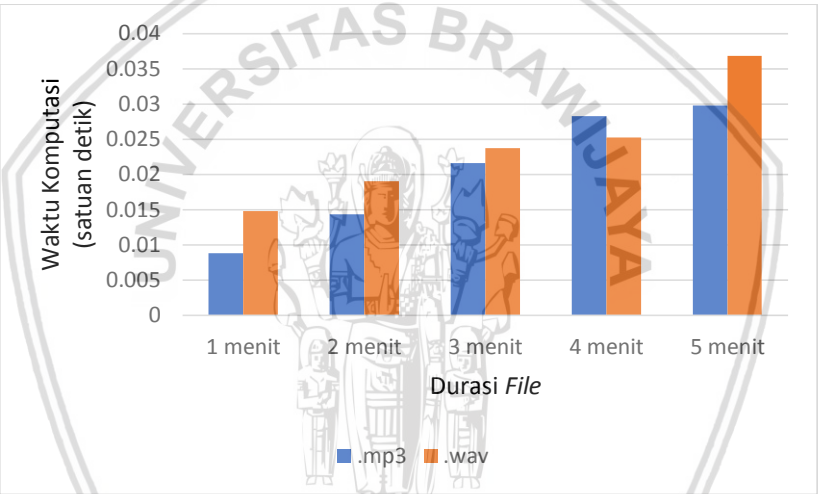


Gambar 6.4 Perbandingan Tipe Data Gambar Berdasarkan Waktu Dekripsi

Berdasarkan grafik menunjukkan perbandingan tipe data ".jpeg" dan ".png" tidak memiliki perbedaan yang cukup signifikan untuk setiap ukuran data yang diuji akan tetapi pada ukuran data 2000 KB kedua tipe data mengalami kenaikan waktu enkripsi sekitar 250% dari pengujian ukuran data 1000 KB. Sedangkan pada pengujian waktu dekripsi tipe data ".png" cenderung lebih cepat dibandingkan tipe data ".jpeg".

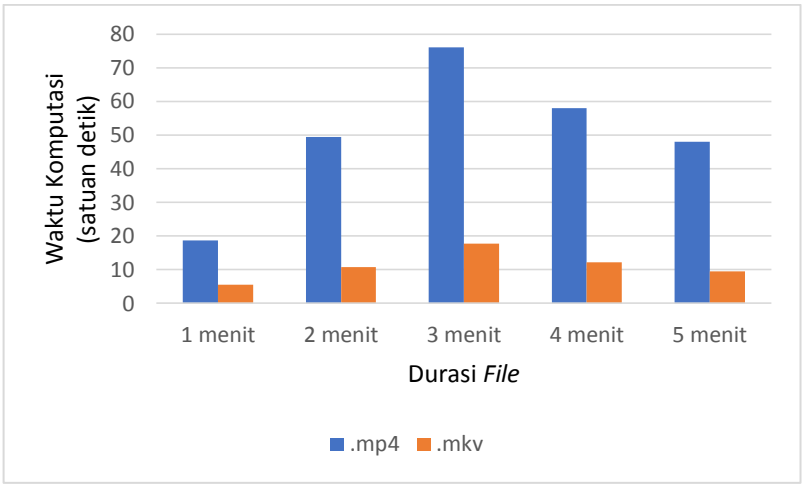


Gambar 6.5 Perbandingan Tipe Data Audio Berdasarkan Waktu Enkripsi

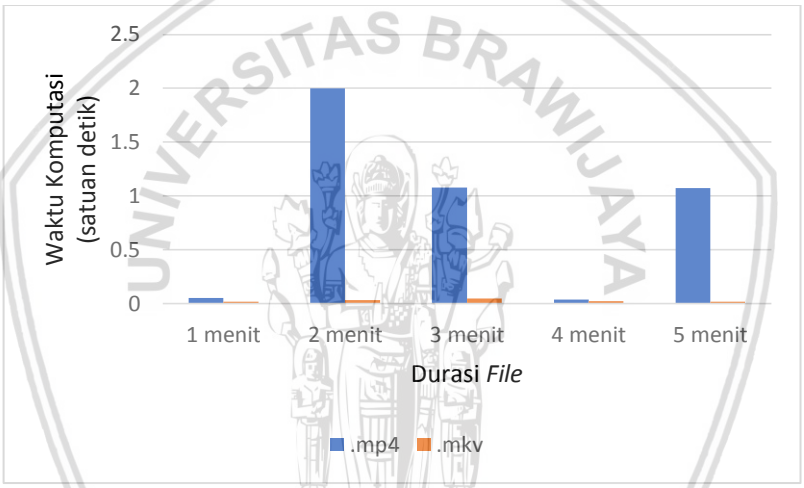


Gambar 6.6 Perbandingan Tipe Data Audio Berdasarkan Waktu Dekripsi

Berdasarkan grafik menunjukkan bahwa tipe data ".wav" memiliki waktu enkripsi yang lebih singkat dari pada tipe data ".mp3", selain itu tipe data ".wav" tidak mengalami kenaikan waktu enkripsi yang signifikan dibandingkan tipe data ".mp3" yang cenderung tidak stabil. Sedangkan pada waktu dekripsi ".wav" cenderung lebih besar dibandikan tipe data ".mp3" akan tetapi keduanya tidak menunjukkan perubahan yang cukup berarti.



Gambar 6.7 Perbandingan Tipe Data Video Berdasarkan Waktu Enkripsi



Gambar 6.8 Perbandingan Tipe Data Video Berdasarkan Waktu Dekripsi

Berdasarkan grafik menunjukkan bahwa tipe data ".mp4" membutuhkan waktu enrkipsi yang cukup besar dibandingkan tipe data ".mkv" begitu juga yang terjadi pada waktu dekripsi. Selain itu, hasil uji *Kruskal Wallis* untuk melihat pengaruh panjang enkripsi, *cost* enkripsi, waktu enkripsi, dan waktu dekripsi terhadap format *file* tidak memiliki pengaruh.

Berdasarkan hasil pengujian AES ada beberapa hal yang dapat menjadi fokus pada penelitian ini. Hasil pengujian menunjukkan metode enkripsi AES dapat diterapkan pada tipe data teks, gambar, audio, dan video. Secara umum waktu komputasi AES dipengaruhi oleh ukuran data dan tipe datanya. Berdasarkan pengujian tersebut, AES dapat direkomendasikan untuk pengamanan data.

6.3 Hasil Pengujian *Kruskal Wallis*

Dari hasil proses enkripsi dan enkripsi dilakukan uji *Kruskal Wallis* untuk melihat pengaruh variabel bebas terhadap variabel terikat. Tabel 6.9 menunjukkan data mentah hasil pengujian AES untuk pengujian *Kruskal Wallis*.

Tabel 6.9 Data Mentah Hasil Pengujian AES

nomor	format *file*	*cost* enkripsi	waktu ekripsi (detik)	waktu dekripsi (detik)
1	1	2234,847283	2,234847283	0,006751532
2	1	3469,860703	3,469860703	0,008233161
3	1	7179,888054	7,179888054	0,01548159
4	1	13723,7615	13,7237615	0,060492305
5	1	63780,82083	63,78082083	0,038105886
6	2	1991,43872	1,99143872	0,005669761
7	2	3308,484328	3,308484328	0,007920146
8	2	6877,526963	6,877526963	0,014646002
9	2	13111,5481	13,1115481	0,026612688
10	2	64948,37897	64,94837897	0,036877237
11	3	2264,334875	0,061107764	0,015234274
12	3	3693,319755	3,693319755	0,008581668
13	3	7699,851157	7,699851157	0,015488386
14	3	14633,59696	14,63359696	0,025858658
15	3	68670,11852	68,67011852	2,264334875
16	4	2331,558701	2,331558701	0,005581408
17	4	3683,481496	3,683481496	0,008382683
18	4	7529,691863	7,529691863	0,015201047
19	4	14761,89608	14,76189608	0,027438081
20	4	67850,52121	67,85052121	0,060219692
21	5	4096,485624	4,096485624	0,00881086
22	5	7141,978511	7,141978511	0,014398686
23	5	11056,9	11,0569	0,021638809
24	5	15348,5038	15,3485038	0,028342765
25	5	17417,77147	17,41777147	0,029773799
26	6	2727,05521	2,72705521	0,014830778
27	6	4699,576623	4,699576622	0,019009259
28	6	5850,261036	5,850261036	0,023660929
29	6	8334,483174	8,334483174	0,025148733
30	6	8362,589219	8,362589219	0,036795862
31	7	11409,64025	11,40964026	0,023935258
32	7	30082,1768	30,0821768	0,029608796
33	7	43805,28168	43,80528168	0,044834008
34	7	57928,69603	57,92869603	0,037112847
35	7	47986,3123	47,9863123	1,07331459
36	8	3497,257627	3,497257627	0007635827
37	8	7035,287241	7,035287241	0,014039229
38	8	9543,371258	9,543371258	0,020388637
39	8	12153,49587	12,15349587	0,025114067
40	8	9498,5701	9,4985701	0,019233238

Dalam kolom format *file*, tiap ekstensi *file* diberi perumpamaan kode nomor, yaitu:

1= .txt	5= .mp3
2= .docx	6= .wav
3= .jpeg	7= .mp4
4= .png	8= .mkv

Pada pengujian krusal willis akan dilakukan untuk *cost* enkripsi, waktu enkripsi, dan waktu dekripsi.

Tabel 6.10 Hasil Dekriptif pada *Cost* Enkripsi

Descriptive Statistics

	N	Mean	Std. Deviation	Minimum	Maximum
cost enkripsi	40	17293,015497	20665,7205484	1991,4387	68670,1185
format file	40	4,50	2,320	1	8

Table diatas menujukan dekriptif masing-masing variabel yaitu *cost* enkripsi dan format *file*.

Tabel 6.11 Hasil Ranking pada *Cost* Enkripsi

Ranks

	format file	N	Mean Rank
cost enkripsi	.txt	5	18,20
	.docs	5	17,20
	.jpeg	5	20,20
	.png	5	20,00
	.mp3	5	22,80
	.wav	5	14,20
	.mp4	5	32,60
	.mkv	5	18,80
	Total	40	

Hasil diatas menjukan masing-masing ranking dari variabel bebas. Selanjutnya akan melihat hasil dari uji statistika untuk melihat pengaruh variabel *cost* enkripsinya terhadap variabel format *file*.

Tabel 6.12 Hasil Uji *Kruskal Wallis* pada *Cost* Enkripsi

Test Statistics[a,b]

	cost enkripsi
Chi-Square	7,712
df	7
Asymp. Sig.	0,359

a. *Kruskal Wallis* Test

b. Grouping Variable: format file

Hasil uji statistika menunjukkan nilai Asymp. Sig. 0,910 dan lebih besar dari batas kritis yaitu 0,05 sehingga variabel *cost* enkripsi tidak berpengaruh terdapat variabel format *file*.

Tabel 6.13 Hasil Dekriptif pada Waktu Enkripsi

Descriptive Statistics

	N	Mean	Std. Deviation	Minimum	Maximum
waktu enkripsi	40	17,237935	20,7096932	0,0611	68,6701
format file	40	4,50	2,320	1	8

Table diatas menujukan dekriptif masing-masing variabel yaitu waktu enkripsi dan format *file*.

Tabel 6.14 Hasil Ranking pada Waktu Enkripsi

Ranks

	format file	N	Mean Rank
waktu enkripsi	.txt	5	18,40
	.docs	5	17,40
	.jpeg	5	19,80
	.png	5	20,00
	.mp3	5	22,80
	.wav	5	14,20
	.mp4	5	32,60
	.mkv	5	18,80
	Total	40	

Hasil diatas menjukan masing-masing ranking dari variabel bebas. Selanjutnya akan melihat hasil dari uji statistika untuk melihat pengaruh variabel waktu enkripsinya terhadap variabel format *file*.

Tabel 6.15 Hasil Uji *Kruskal Wallis* pada Waktu Enkripsi

Test Statistics[a,b]

	waktu enkripsi
Chi-Square	7,648
df	7
Asymp. Sig.	0,365

a. *Kruskal Wallis* Test

b. Grouping Variable: format file

Hasil uji statistika menunjukkan nilai Asymp. Sig. 0,931 dan lebih besar dari batas kritis yaitu 0,05 sehingga variabel waktu enkripsi tidak berpengaruh terdapat variabel format *file*.

Tabel 6.16 Hasil Dekriptif pada Waktu Dekripsi

Descriptive Statistics

	N	Mean	Std. Deviation	Minimum	Maximum
waktu dekripsi	40	0,104618	0,3878694	0,0056	2,2643
format file	40	4,50	2,320	1	8

Table diatas menujukan dekriptif masing-masing variabel yaitu waktu dekripsi dan format *file*.

Tabel 6.17 Hasil Ranking pada Waktu Dekripsi

Ranks

	format file	N	Mean Rank
waktu dekripsi	.txt	5	19,60
	.docs	5	15,80
	.jpeg	5	21,20
	.png	5	17,40
	.mp3	5	20,20
	.wav	5	22,00
	.mp4	5	32,40
	.mkv	5	15,40
	Total	40	

Hasil diatas menjukan masing-masing ranking dari variabel bebas. Selanjutnya akan melihat hasil dari uji statistika untuk melihat pengaruh variabel waktu dekripsinya terhadap variabel format *file*.

Tabel 6.18 Hasil Uji *Kruskal Wallis* pada Waktu Dekripsi

Test Statistics[a,b]

	waktu dekripsi
Chi-Square	7,425
df	7
Asymp. Sig.	0,386

a. *Kruskal Wallis* Test

b. Grouping Variable: format file

Hasil uji statistika menunjukkan nilai Asymp. Sig. 0,906 dan lebih besar dari batas kritis yaitu 0,05 sehingga variabel waktu dekripsi tidak berpengaruh terdapat variabel format *file*.

BAB 7 PENUTUP

7.1 Kesimpulan

Berdasarkan implementasi dan analisis yang telah dilakukan ada beberapa kesimpulan yang didapatkan untuk menjawab rumusan masalah, antara lain:

1. Metode enkripsi AES dapat diterapkan untuk melakukan enkripsi pada data dengan tipe data teks, gambar, audio, dan video yang disimulasikan pada pemrograman Java.
2. Dari pengujian algoritme AES yang dilakukan terhadap berbagai format *file*, didapatkan hasil enkripsi dan dekripsi yang berbeda-beda pada setiap ekstensi *file* yang diujikan. Pada format dokumen menunjukkan bahwa hasil enkripsi *file* terhadap ekstensi .docx lebih baik dibandingkan dengan .txt hanya saat *file* berukuran 5000 KB. Sedangkan hasil dekripsi menunjukkan hasil yang bermacam-macam pada setiap ukuran *file* yang diujikan dengan rata-rata hasil ekstensi .txt lebih kecil dibandingkan ekstensi .docx, dengan nilai masing-masing yaitu 0,031194 dan 0,031916. Hasil enkripsi menunjukkan hasil yang bermacam-macam pada setiap ukuran *file* yang diujikan dengan rata-rata hasil ekstensi .jpeg lebih kecil dibandingkan ekstensi .png, dengan nilai masing-masing yaitu 22,509 dan 23,043. Sedangkan hasil dekripsi *file* yang diperoleh menunjukkan bahwa *file* dengan ekstensi .png lebih bauk dari pada .jpeg.
3. Pada pengujian algoritme AES terhadap format *file* audio, didapatkan hasil enkripsi pada *file* ekstensi .wav lebih baik dibandingkan *file* ekstensi .mp3, sedangkan pada hasil dekripsi, *file* ekstensi .mp3 menghasilkan nilai rata-rata yang lebih kecil yaitu 0,02056 dibandingkan *file* ekstensi .wav dengan nilai rata-rata yaitu 0,0239. Pada format *file* video, hasil enkripsi dan dekripsi menunjukkan bahwa *file* ekstensi .mkv lebih baik dibandingkan *file* ekstensi .mp4.
4. Pada pengujian beberapa tipe data. AES cenderung memberikan waktu komputasi yang cukup lama pada tipe data video yaitu antara 20 detik sampai 60 detik, hal ini karena memerlukan waktu untuk mengubah video dalam bentuk teks yang akan diproses dalam AES selain itu proses perubahan ini akan sangat tergantung dari ukuran data video tersebut. Selain memiliki ukuran data yang banyak juga membutuhkan waktu dalam mengubah dalam bentuk teks. Sedangkan pada tipe data teks, gambar, dan audio AES memberikan performa yang bagus yaitu 5 MB membutuhkan waktu rata-rata enkripsi dan dekripsi masing-masing 60 detik dan 0,02 detik.

7.2 Saran

Hasil penelitian ini menunjukkan bahwa metode enkripsi AES dapat memberikan hasil yang maksimal sehingga metode AES dapat diterapkan pada sistem penyimpanan data *virtual*untuk dapat melindungi data yang dikirimkan. Selain itu untuk mendapatkan metode enkripsi yang lebih baik lagi penelitian selanjutnya dapat melakukan perbandingan metode AES dengan metode enkripsi lainnya seperti *DES* dan *IDEA*.

DAFTAR PUSTAKA

Abdullah, A., 2017. *Advanced Encryption Standart (AES) Algorithm to Encrypt and Decrypt Data*. Eastern Mediteraanean University, Cyprus.

Bhardwaj, A., Subrahmanyam, G. V. B., Avasthi, V. dan Sastry, H., 2016. *Security algorithms for cloud computing*. International Conference on Computational Modeling and Security (CMS), vol. 85, pp. 535-542, 19 Sept, Karnataka, India.

Jeeva, A. L., Palanisamy, V. dan Kanagaram, K., 2012. Comparative analysis of performance efficiency and security measures of some encryption. *International Journal of Engineering Reaserch and Application (IJERA)*, vol. 2, issue 3, pp. 3033-3037.

Kromodimoeljo, S., 2009. Teori dan aplikasi kriptografi. SPK IT Consulting.

Syaikhu, A., 2010. Komputasi awan (Cloud computing) perpustakaan pertanian. *Jurnal Pustakawan Indonesia*, vol. 10, no. 1, pp. 1-12.