PERBANDINGAN KINERJA ALGORITMA RSA DAN AES DALAM ENKRIPSI DAN DEKRIPSI CITRA DIGITAL

SKRIPSI

OLEH:
RIFQI MUHAMAD SABILAL KARIM
NIM. 19610111

PROGRAM STUDI MATEMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2024

PERBANDINGAN KINERJA ALGORITMA RSA DAN AES DALAM ENKRIPSI DAN DEKRIPSI CITRA DIGITAL

SKRIPSI

Diajukan Kepada
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang
untuk Memenuhi Salah Satu Persyaratan dalam
Memperoleh Gelar Sarjana Matematika (S.Mat)

Oleh
Rifqi Muhamad Sabilal Karim
NIM. 19610111

PROGRAM STUDI MATEMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2024

PERBANDINGAN KINERJA ALGORITMA RSA DAN AES DALAM ENKRIPSI DAN DEKRIPSI CITRA DIGITAL

SKRIPSI

Oleh
Rifqi Muhamad Sabilal Karim
NIM. 19610111

Telah Disetujui Untuk Diuji

Malang, 5 Desember 2024

Dosen Pembimbing I	Dosen Pembimbing II
Muhammad Khudzaifah, M.Si	Dr. Fachrur Rozi, M.Si
NIPPPK. 19900511 202321 1 029	NIP. 19800527 200801 1 012

Mengetahui,
Ketua Program Studi Matematika

Dr. Elly Susanti, M.Sc
NIP. 19741129 200012 2 005

PERBANDINGAN KINERJA ALGORITMA RSA DAN AES DALAM ENKRIPSI DAN DEKRIPSI CITRA DIGITAL

SKRIPSI

Oleh
Rifqi Muhamad Sabilal Karim
NIM. 19610111

Telah Dipertahankan di Depan Penguji Skripsi
dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
untuk Memperoleh Gelar Sarjana Matematika (S.Mat)

Tanggal 12 Desember 2024

Ketua Penguji	: Hisyam Fahmi, M.Kom	.....
Anggota Penguji 1	: Mohammad Nafie Jauhari, M.Si	.....
Anggota Penguji 2	: Muhammad Khudzaifah, M.Si	.....
Anggota Penguji 3	: Dr. Fachrur Rozi, M.Si	.....

Mengetahui,
Ketua Program Studi Matematika

Dr. Elly Susanti, M.Sc
NIP. 19741129 200012 2 00

PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan dibawah ini:

Nama	:	Rifqi Muhamad Sabilal Karim
NIM	:	19610111
Program Studi	:	Matematika
Fakultas	:	Sains dan Teknologi
Judul Skripsi	:	Perbandingan Kinerja Algoritma RSA dan AES dalam Enkripsi dan Dekripsi Citra Digital

Menyatakan dengan sebenarnya bahwa skripsi yang saya tulis ini benar-benar merupakan hasil karya sendiri, bukan merupakan pengambilan data, tulisan atau pikiran orang lain yang saya akui sebagai hasil tulisan dan pikiran saya sendiri, kecuali dengan mencantumkan sumber kutipan atau daftar rujukan. Apabila di kemudian hari terbukti atau dapat dibuktikan skripsi ini hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 12 Desember 2024
Yang membuat pernyataan,

Rifqi Muhamad Sabilal Karim

MOTO DAN PERSEMBAHAN

"*Setiap manusia memiliki garis suksesnya masing-masing. Percayalah bahwa Tuhan tidak pernah membuat rencana yang mengecewakan*"

Saya Persembahkan Skripsi ini Untuk yang Selalu Bertanya:

"Kapan skripsimu selesai?"

Terlambat lulus atau tidak lulus tepat waktu bukanlah suatu kejahatan dan juga bukan suatu hal yang memalukan. Betapa kerdilnya mengukur kecerdasan seseorang hanya dengan melihat siapa yang lulus paling cepat. Bukankah sebaik-baiknya skripsi adalah skripsi yang selesai?

Karena mungkin ada suatu hal dibalik terlambatnya saya lulus, dan percayalah alasan saya disini merupakan alasan yang sepenuhnya baik.

KATA PENGANTAR

Assalamu'alaikum Wr. Wb.

Segala puji bagi Allah SWT yang telah memberikan rahmat, taufik, serta hidayah-Nya, sehingga penulis mampu menyelesaikan skripsi dengan judul Perbandingan Kinerja Algoritma RSA dan AES dalam Enkripsi dan Dekripsi Citra Digital, sebagai salah satu syarat untuk memperoleh gelar sarjana keilmuan matematika di Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang.

Dalam proses penyusunan skripsi ini, penulis banyak mendapatkan arahan, bimbingan, dan motivasi dari berbagai pihak. Maka dari itu, ucapan terimakasih yang sebesar-besarnya penulis sampaikan terutama kepada:

1. Prof. Dr. H. M. Zainuddin, M.A., selaku rektor Universitas Islam Negeri Maulana Malik Ibrahim.
2. Prof. Dr. Hj. Sri Harini, M.Si, selaku dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim.
3. Dr. Elly Susanti, S.Pd., M.Sc, selaku ketua Program Studi Matematika, Universitas Islam Negeri Maulana Malik Ibrahim.
4. Muhammad Khudzaifah, M.Si, selaku dosen pembimbing I yang telah banyak meluangkan waktunya untuk memberikan arahan, motivasi, serta nasihat yang berharga untuk penulis.
5. Dr. Fachrur Rozi, M.Si, selaku dosen pembimbing II yang telah banyak meluangkan waktunya untuk memberikan arahan, motivasi, serta nasihat yang berharga untuk penulis.

6. Hisyam Fahmi, M.Kom, selaku ketua penguji yang telah memberikan arahan dan saran yang membangun kepada penulis.
7. Mohammad Nafie Jauhari, M.Si, selaku dosen penguji yang telah banyak memberikan nasehat dan berbagi ilmu kepada penulis.
8. Seluruh dosen Program Studi Matematika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim.
9. Orang tua penulis yaitu Bapak Moh. Bachrudin dan Ibu Elvi Sayyidah Faiqah yang selalu memberikan dukungan secara lahir dan batin, motivasi yang membangkitkan semangat, nasehat agar penulis tidak salah dalam mengambil langkah selama mengerjakan tugas akhir, serta do'a yang tidak pernah terputus untuk penulis.
10. Seluruh mahasiswa angkatan 2019 Program Studi Matematika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim.
11. Muhammad Ibrahim Albar, Ichsanuddin Noorsy Alfannany, Abu Amar Nashih Bilqisth, Amalia Prastiwi, Syarifudin Dwi Cahyo, selaku sahabat yang selalu menemani penulis selama mengerjakan tugas akhir.
12. Dono Pradana dan Agustian Pratama, selaku *Podcaster* pada Podcast Mendoan yang setia menghibur dengan membawakan banyak topik dan kisah menarik dibalut dengan humor yang sangat lucu, menjadikan otak bekerja dengan lancar ketika mengerjakan tugas akhir.
13. Beberapa nama yang tidak dapat saya sebutkan satu persatu, mereka juga telah banyak membantu dan menemani penulis selama mengerjakan tugas akhir ini.

Penulis berharap semoga Allah SWT selalu melimpahkan rahmat dan karunia-Nya kepada kita semua dan juga semoga skripsi ini dapat segera diselesaikan sehingga bisa segera mendaftarkan diri untuk mengikuti sidang skripsi. Amiinn.

Wassalamu'alaikum Wr. Wb.

Malang, 12 Desember 2024

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR SIMBOL

M	: Lebar citra
N	: Tinggi citra
p dan q	: Bilangan prima untuk proses pembangkitan kunci algoritma RSA
n	: Modulo pembagi untuk algoritma RSA
$\phi(n)$	: Nilai *totient euler* (n)
(e, n)	: Pasangan kunci publik algoritma RSA
(d, n)	: Pasangan kunci privat algoritma RSA
Nk	: Panjang kunci algoritma AES
Nb	: Ukuran blok algoritma AES
Nr	: Jumlah putaran algoritma AES
$S[r, c]$	: Substitusi pada transformasi *SubBytes*
$\cdot$	: perkalian dalam $GF(2^8)$
$\oplus$	: Operasi XOR
MSE	: Nilai *Mean Square Error* pada citra
$A_{i,j}$	: Representasi satu piksel dari *plain image*
$B_{i,j}$	: Representasi satu piksel dari citra hasil dekripsi
$m \times n$	: Dimensi citra

DAFTAR LAMPIRAN

ABSTRAK

Karim, Rifqi Muhamad Sabilal. 2024. **Perbandingan Kinerja Algoritma RSA dan AES dalam Enkripsi dan Dekripsi Citra Digital**. Skripsi. Program Studi Matematika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing (1): Muhammad Khudzaifah, M.Si, Pembimbing (2): Dr. Fachrur Rozi, M.Si

Kata Kunci: Citra Digital, Enkripsi, Dekripsi, Algoritma RSA, Algoritma AES

Dalam era digital, keamanan data menjadi kebutuhan mendesak, terutama untuk melindungi citra digital yang memiliki nilai penting dalam berbagai bidang, seperti komunikasi, keamanan, dan medis. Penelitian ini bertujuan untuk membandingkan kinerja algoritma kriptografi RSA dan AES dalam konteks enkripsi dan dekripsi citra digital. Evaluasi dilakukan berdasarkan tiga parameter utama, yaitu waktu enkripsi dan dekripsi, ukuran file setelah proses, serta kualitas citra hasil enkripsi yang diukur dengan *Mean Squared Error* (MSE). Metode penelitian menggunakan pendekatan kuantitatif eksperimental dengan 50 citra digital beresolusi kecil dan sedang. Algoritma RSA diterapkan melalui proses pembangkitan kunci publik dan privat, sementara AES menggunakan transformasi *SubBytes*, *ShiftRows*, *MixColumns*, dan *AddRoundKey*. Hasil menunjukkan bahwa algoritma AES lebih unggul dalam kecepatan enkripsi dan dekripsi, dengan rata-rata waktu yang lebih cepat dibandingkan RSA. Namun, algoritma RSA menghasilkan kualitas citra yang kurang baik setelah enkripsi, sebagaimana ditunjukkan oleh nilai MSE yang lebih rendah dari algoritma AES. Dari segi ukuran file, RSA cenderung menghasilkan ukuran yang lebih kecil, sedangkan AES menghasilkan ukuran yang lebih besar. Studi ini memberikan kontribusi dalam menentukan algoritma enkripsi yang optimal untuk aplikasi spesifik, dengan implikasi penting dalam keamanan data visual. Hasil ini dapat menjadi acuan bagi mahasiswa maupun praktisi keamanan informasi dalam memilih algoritma yang sesuai untuk melindungi citra digital dari ancaman keamanan.

ABSTRACT

Karim, Rifqi Muhamad Sabilal. 2024. **Comparison of RSA and AES Algorithm Performance in Digital Image Encryption and Decryption**. Thesis. Mathematics Study Program, Faculty of Science and Technology, Universitas Islam Negeri Maulana Malik Ibrahim Malang. Supervisor (1): Muhammad Khudzaifah, M.Si, Supervisor (2): Dr. Fachrur Rozi, M.Si

Keywords: Digital Image, Encryption, Decryption, RSA Algorithm, AES Algorithm

In the digital era, data security has become an urgent necessity, especially to protect digital images that hold significant value in various fields, such as communication, security, and medical. This research aims to compare the performance of the RSA and AES cryptographic algorithms in the context of digital image encryption and decryption. The evaluation was conducted based on three main parameters, namely encryption and decryption time, file size after the process, and the quality of the encrypted image measured by Mean Squared Error (MSE). The research method uses an experimental quantitative approach with 50 digital images of low and medium resolution. The RSA algorithm is applied through the process of generating public and private keys, while AES uses the SubBytes, ShiftRows, MixColumns, and AddRoundKey transformations. The results show that the AES algorithm is superior in encryption and decryption speed, with an average time faster than RSA. However, the RSA algorithm produces lower image quality after encryption, as indicated by the lower MSE value compared to the AES algorithm. In terms of file size, RSA tends to produce smaller sizes, while AES produces larger sizes. This study contributes to determining the optimal encryption algorithm for specific applications, with important implications for visual data security. These results can serve as a reference for students and information security practitioners in selecting the appropriate algorithm to protect digital images from security threats.

مستخلص البحث

الكريم، رفقى محمد سبيل. ٢٠٢٤. **مقارنة أداء خوارزمية *RSA* و *AES* في تشفير وفك تشفير الصور الرقمية. البحث الجامعي**. برنامج دراسات قسم الرياضيات، جامعة مولانا مالك إبراهيم الإسلامية الحكومية، مالانج. المشرف الأول: محمد خذيفة،الماجستير. المشرفة الثانية: د. فخر الرازى، الماجستير

الكلمات المفتاحية: صورة رقمية، تشفير، فك تشفير، خوارزمية*RSA* ، خوارزمية*AES*

في العصر الرقمي، أصبحت أمان البيانات حاجة ملحة، خاصة لحماية الصور الرقمية التي لها قيمة مهمة في مجالات مختلفة، مثل الاتصالات، والأمن، والطب. تهدف هذه الدراسة إلى مقارنة أداء خوارزميات التشفير *RSA* و *AES* في سياق تشفير وفك تشفير الصور الرقمية. تمت التقييم بناءً على ثلاثة معايير رئيسية، وهي وقت التشفير وفك التشفير، وحجم الملف بعد العملية، وجودة الصورة الناتجة عن التشفير التي قياسها باستخدام متوسط الخطأ التربيعي (*MSE*) .يستخدم هذا البحث مدخل الكمي التجريبي مع ٥٠ صورة رقمية ذات دقة منخفضة ومتوسطة. تطبيق خوارزمية *RSA* من خلال عملية توليد المفاتيح العامة والخاصة، وأما *AES* بتحويلات *SubBytes* و *ShiftRows* و *MixColumns* و*AddRoundKey*. ظهرت نتائج البحث أن خوارزمية *AES* تتفوق في سرعة التشفير وفك التشفير، بمتوسط وقت أسرع من *RSA* . ومع ذلك، تصدر خوارزمية *RSA* جودة صورة أقل بعد التشفير، كما يتضح من قيمة *MSE* أقل من بخوارزمية *AES*. من حيث حجم الملف، تميل *RSA* إلى إنتاج أصغر الحجم ، بينما تنتج *AES* أكبر الحجم. تساهم هذا البحث تحديد الخوارزمية المثلى للتشفير للتطبيقات المحددة بتداعيات مهمة في أمان البيانات المرئية. يمكن أن تكون هذه النتائج مرجعًا للطلاب والممارسين في مجال أمن المعلومات عند اختيار الخوارزمية المناسبة لحماية الصور الرقمية من التهديدات الأمنية .

BAB I
PENDAHULUAN

1.1 Latar Belakang

Dalam era digital saat ini, perkembangan teknologi informasi dan komunikasi telah mengubah banyak aspek kehidupan manusia secara signifikan. Salah satu dampak utamanya adalah meningkatnya jumlah data digital yang dihasilkan, ditransmisikan, serta disimpan. Data digital sekarang dianggap sebagai aset yang sangat berharga dan perlu dilindungi. Oleh karena itu, keamanan data menjadi prioritas utama bagi individu, organisasi, dan pemerintah, terutama dengan meningkatnya ancaman keamanan siber seperti *hacking*, pencurian identitas, dan penyadapan data. Citra digital adalah salah satu jenis data yang sering digunakan di berbagai sektor, seperti media sosial, komunikasi, kedokteran, keamanan, dan teknik. Citra digital merupakan representasi numerik dari gambar yang terdiri dari piksel-piksel, di mana setiap piksel memiliki nilai warna tersendiri. Ini memungkinkan gambar untuk diproses dan dimodifikasi secara digital. Teknologi ini telah membawa perubahan besar dalam berbagai bidang, mulai dari fotografi hingga desain grafis, serta dunia medis dan sains. Kelebihan citra digital terletak pada kemudahannya dalam hal penyimpanan, distribusi, pengeditan, serta kemampuannya untuk menganalisis data secara mendalam guna mendapatkan informasi penting (Saravana, dkk., 2021).

Citra digital dihasilkan melalui proses konversi sinyal analog (cahaya) menjadi sinyal digital. Tahap-tahap dalam proses ini mencakup pengambilan sampel pada interval tertentu, dan kuantisasi, yaitu pengubahan nilai analog menjadi angka digital. Hasil akhir dari proses ini adalah matriks numerik yang

merepresentasikan intensitas cahaya di setiap titik gambar. Proses ini memungkinkan komputer untuk memproses dan memanipulasi gambar dengan efisien. Citra digital saat ini telah menjadi elemen penting dalam kehidupan sehari-hari. Dalam bidang fotografi, teknologi ini mempermudah proses pengeditan serta berbagi gambar. Sementara itu, di dunia medis, citra digital digunakan untuk membantu diagnosis penyakit dan memantau kondisi pasien. Industri hiburan juga sangat bergantung pada citra digital untuk menciptakan efek visual yang luar biasa dalam film dan video game. Selain itu, citra satelit dan drone digunakan untuk pemetaan wilayah dan pemantauan perubahan lingkungan (Kuznetsov, 2021).

Konsep dasar dalam citra digital meliputi resolusi, kedalaman bit, dan format file. Resolusi menentukan tingkat ketajaman gambar, sementara kedalaman bit menentukan jumlah warna yang bisa ditampilkan. Format file seperti JPG, PNG, dan BMP memiliki karakteristik kompresi dan kualitas yang bervariasi, sehingga masing-masing cocok untuk aplikasi yang berbeda. Meskipun telah mengalami banyak kemajuan, pengolahan citra digital masih menghadapi beberapa tantangan, seperti gangguan noise, kompresi yang berlebihan, dan segmentasi objek yang kompleks. Namun, dengan kemajuan teknologi kecerdasan buatan (*Artificial Intelligence*) tantangan-tantangan ini mulai dapat diatasi. Algoritma yang semakin canggih memungkinkan pengenalan objek, pelacakan gerakan, dan analisis citra yang lebih presisi (Mittal, dkk., 2021).

Melindungi citra digital menjadi semakin penting, terutama jika digunakan dalam aplikasi yang memproses informasi sensitif atau pribadi. Untuk itu, berbagai teknik keamanan, khususnya enkripsi, telah dikembangkan dan digunakan untuk melindungi citra digital dari akses tidak sah dan manipulasi. Dalam beberapa tahun

terakhir, internet telah berkembang menjadi salah satu alat komunikasi utama. Akibatnya, keamanan internet menjadi isu krusial karena banyak data yang ditransmisikan melalui jaringan, seperti email, dokumen, gambar, dan audio. Melindungi informasi yang dikirimkan serta menjaga privasi dari akses yang tidak sah menjadi sangat penting. Terdapat berbagai metode untuk melindungi data, dan salah satu solusi keamanan informasi yang paling efektif adalah enkripsi, yang mampu melindungi berbagai jenis data (Alsaffar, dkk., 2020).

Enkripsi merupakan proses mengubah data yang mudah dibaca dan dipahami menjadi bentuk yang tidak dapat dimengerti. Sebaliknya, dekripsi adalah proses mengembalikan data tersebut ke bentuk aslinya yang dapat dipahami. Enkripsi data terbagi menjadi dua metode: enkripsi simetris (dengan kunci rahasia) dan enkripsi asimetris (dengan kunci publik). Algoritma enkripsi simetris menggunakan satu kunci rahasia untuk mengenkripsi dan mendekripsi data. Kunci rahasia ini dibagikan antara pengirim dan penerima dan harus tetap dirahasiakan. Banyak standar perangkat lunak dan perangkat keras yang menggunakan enkripsi simetris bergantung pada salah satu algoritma simetris paling umum, seperti *Advanced Encryption Standard* (AES) (Alsaffar, dkk., 2020).

Untuk algoritma enkripsi asimetris, kunci yang berbeda digunakan untuk mengenkripsi dan mendekripsi data: kunci privat yang tetap dirahasiakan, dan kunci publik yang dapat dipublikasikan untuk siapa saja. Banyak standar perangkat lunak dan perangkat keras yang menggunakan enkripsi asimetris bergantung pada salah satu algoritma asimetris paling populer, seperti *Rivest-Shamir-Adleman* (RSA), yang ditemukan oleh Ronald Rivest (Alsaffar, dkk., 2020).

RSA adalah salah satu algoritma enkripsi asimetris yang sangat populer dan banyak digunakan. Pada tahun 1977 algoritma ini diciptakan oleh Ron Rivest, Adi Shamir, dan Leonard Adleman. RSA memanfaatkan prinsip-prinsip matematika, khususnya faktorisasi bilangan prima, untuk menghasilkan sepasang kunci, yaitu kunci publik dan kunci privat. Kunci publik dapat didistribusikan secara terbuka dan digunakan untuk mengenkripsi data, sedangkan kunci privat harus dijaga kerahasiaannya dan digunakan untuk mendekripsi data yang telah dienkripsi dengan kunci publik. RSA dikenal memiliki tingkat keamanan yang tinggi, terutama karena kesulitan dalam memfaktorkan bilangan besar, yang menjadi dasar keamanan algoritma ini. RSA banyak diaplikasikan dalam protokol keamanan seperti SSL (*Secure Sockets Layer*) dan TLS (*Transport Layer Security*) untuk melindungi komunikasi internet, termasuk dalam transaksi keuangan dan pertukaran data sensitif.

AES merupakan algoritma enkripsi simetris yang dikembangkan oleh Joan Daemen dan Vincent Rijmen, dan resmi diadopsi sebagai standar enkripsi oleh *National Institute of Standards and Technology* (NIST) pada tahun 2001. Algoritma ini menggunakan kunci enkripsi dengan ukuran 128-bit, 192-bit, atau 256-bit untuk proses enkripsi dan dekripsi data. Karena AES adalah algoritma simetris, kunci yang sama digunakan dalam kedua proses tersebut. AES dikenal akan kecepatan dan efisiensinya dalam menangani data berukuran besar, sehingga sangat ideal untuk enkripsi citra digital. Algoritma ini digunakan secara luas dalam berbagai aplikasi keamanan, seperti enkripsi disk, file, serta protokol jaringan.

Dalam hal enkripsi citra digital, baik RSA maupun AES masing-masing memiliki kelebihan dan kekurangan. Namun, salah satu tantangan utama dalam

menggunakan algoritma AES adalah pengelolaan kunci enkripsi, karena kunci yang sama harus dilindungi kerahasiaannya dan didistribusikan dengan aman. Sementara itu, tantangan utama dalam menggunakan algoritma RSA terletak pada proses enkripsi data berukuran besar.

Percobaan pengamanan citra digital ini memiliki tujuan untuk melindungi privasi dari pemilik citra digital tersebut. Citra digital dapat dikatakan sebagai sesuatu yang penting bagi setiap orang, karena dari citra digital ini semua orang dapat mengabadikan setiap momen atau peristiwa yang sedang dialami melalui kamera yang menghasilkan berbagai macam gambar. Oleh sebab itu, dapat disimpulkan bahwa citra digital memiliki banyak sekali rekam jejak, baik itu yang bersifat rahasia maupun umum sehingga diperlukan cara untuk mengamankan kerahasiaan itu. Dalam konteks ayat Al-Qur'an, Surat An-Nur ayat 27 memberikan petunjuk tentang pentingnya menjaga privasi setiap individu.

"Wahai orang-orang yang beriman, janganlah memasuki rumah yang bukan rumahmu sebelum meminta izin dan memberi salam kepada penghuninya. Demikian itu lebih baik bagimu agar kamu mengambil pelajaran." (QS. An-Nur:27)

Menurut Tafsir Tahlili di website Qur'an NU, ayat ini mengajarkan orang-orang mukmin tentang tata krama dalam berinteraksi, termasuk cara menjaga dan memupuk cinta serta hubungan baik di antara mereka. Salah satu ajarannya adalah tidak memasuki rumah orang lain tanpa izin dan tanpa memberikan salam terlebih dahulu, untuk menghindari melihat hal-hal yang seharusnya tidak terlihat, seperti aib atau hal-hal pribadi yang dijaga dengan ketat. Jika seseorang meminta izin untuk masuk dan tidak mendapatkan respons, sebaiknya dilakukan hingga tiga kali. Jika

izin diberikan, baru boleh masuk, dan jika tidak, sebaiknya pulang. Dari sini, dapat disimpulkan bahwa untuk mencegah kejahatan digital, penting untuk menjaga keamanan data dan privasi melalui perlindungan data digital.

Pada penelitian sebelumnya yang dilakukan oleh Alsaffar, dkk. (2020) memakai dua algoritma, yakni AES dan RSA untuk memproses enkripsi dan dekripsi pada citra digital. Pada algoritma AES, proses enkripsi dimulai dengan menghasilkan sekumpulan kunci putaran dari kunci cipher yang diberikan. Selanjutnya, algoritma ini menginisialisasi *array* status dengan data blok (*plaintext*) untuk memulai operasi putaran. Setiap putaran melibatkan beberapa langkah, termasuk substitusi, permutasi, dan penambahan kunci, yang bertujuan untuk mengacak data sehingga sulit untuk dikembalikan ke bentuk aslinya tanpa kunci yang tepat. Menggunakan operasi XOR sebagai langkah penting dalam algoritma enkripsi AES, khususnya dalam tahap penambahan kunci (*AddRoundey*). Dari penelitian tersebut terdapat beberapa kekurangan pada hasilnya, yakni lebih menekankan pada kualitas enkripsi dan tidak membahas aspek lain seperti kecepatan enkripsi, efisiensi sumber daya, atau keamanan terhadap serangan tertentu, penggunakan program MATLAB sebagai alat didalam prosesnya menyebabkan hasil yang diperoleh tidak sepenuhnya mencerminkan kinerja algoritmanya, serta keterbatasan dalam variasi gambar citra digitalnya.

Berdasarkan penelitian sebelumnya, maka penelitian ini dibuat bertujuan untuk membandingkan kinerja algoritma RSA dan AES dalam konteks enkripsi dan dekripsi citra digital dengan menggunakan format PNG sebagai objeknya. Perbandingan akan difokuskan pada beberapa parameter seperti kecepatan enkripsi dan dekripsi, penggunaan program komputasi berbasis Python, kualitias citra yang

dihasilkan, serta ukuran *file* yang dihasilkan dari kedua algoritma. Harapannya penelitian ini dapat memberikan kontribusi penting dalam bidang keamanan data, terutama terkait dengan enkripsi citra digital. Dengan semakin tingginya kebutuhan akan perlindungan data visual, hasil penelitian ini diharapkan dapat membantu pengembang dan praktisi keamanan dalam memilih algoritma enkripsi yang paling tepat untuk melindungi citra digital. Selain itu, penelitian ini juga dapat memberikan wawasan lebih lanjut mengenai kinerja berbagai algoritma dalam skenario aplikasi yang berbeda.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan, maka rumusan masalah penelitian dapat dijabarkan dalam beberapa poin berikut:

1. Bagaimana perbandingkan kinerja algoritma AES dan RSA dalam hal waktu yang dibutuhkan untuk enkripsi dan dekripsi?
2. Bagaimana perbandingkan kinerja algoritma AES dan RSA dalam hal ukuran *file* setelah proses enkripsi dan dekripsi?
3. Bagaimana perbandingkan kinerja algoritma AES dan RSA dalam hal kualitas citra yang dihasilkan setelah proses enkripsi?

1.3 Tujuan Penelitian

Tujuan dari penelitian ini adalah:

1. Mengetahui perbandingan kinerja algoritma AES dan RSA dalam hal waktu yang dibutuhkan untuk enkripsi dan dekripsi

2. Mengetahui perbandingan kinerja algoritma AES dan RSA dalam hal ukuran file yang dihasilkan setelah enkripsi dan dekripsi
3. Mengetahui perbandingan kinerja algoritma AES dan RSA dalam hal kualitas citra yang diperoleh setelah proses enkripsi.

1.4 Manfaat Penelitian

Penelitian ini diharapkan akan memunculkan beberapa manfaat sebagai berikut:

1. Menyediakan wawasan yang lebih mendalam mengenai cara kerja dan karakteristik dari dua algoritma enkripsi yang banyak digunakan, yaitu RSA dan AES. Pengetahuan ini dapat membantu akademisi, peneliti, serta praktisi keamanan informasi dalam menentukan algoritma yang paling sesuai untuk berbagai aplikasi tertentu.
2. Menjadi acuan bagi pengembang perangkat lunak, terutama yang berfokus pada keamanan data, dalam memilih algoritma enkripsi yang paling efisien dan efektif untuk melindungi citra digital. Dengan memahami perbedaan performa RSA dan AES dalam enkripsi citra, mereka dapat membuat pilihan yang lebih tepat berdasarkan kebutuhan spesifik aplikasi.
3. Penelitian ini berkontribusi pada pengembangan lebih lanjut dalam bidang kriptografi, terutama terkait dengan aplikasi enkripsi citra digital. Hasil penelitian dapat menjadi referensi bagi studi lanjutan yang bertujuan untuk mengembangkan atau menyempurnakan algoritma enkripsi yang ada.
4. Penelitian ini juga dapat dijadikan acuan akademis bagi mahasiswa dan peneliti lain yang tertarik pada topik kriptografi, khususnya dalam

membandingkan algoritma simetris dan asimetris. Ini akan menjadi tambahan literatur yang berguna dalam bidang keamanan informasi dan kriptografi.

1.5 Batasan Masalah

Beberapa batasan masalah yang dimunculkan pada penelitian ini adalah:

1. Penelitian ini terbatas pada dua algoritma enkripsi, yakni RSA sebagai algoritma asimetris dan AES sebagai algoritma simetris. Algoritma enkripsi lainnya tidak akan dibahas atau dianalisis dalam penelitian ini.
2. Penelitian ini juga dibatasi pada citra digital dengan ukuran tertentu (misalnya, resolusi rendah, sedang, atau tinggi). Selain itu, hanya jenis citra tertentu, seperti citra berwarna (RGB) atau *grayscale*, yang akan dianalisis.
3. Analisis kinerja algoritma enkripsi dalam penelitian ini akan difokuskan pada parameter-parameter tertentu, seperti kecepatan enkripsi dan dekripsi, ukuran file setelah enkripsi, serta kualitas citra yang dihasilkan dari masing-masing algoritma.
4. Penelitian ini akan menekankan pada perbandingan kinerja dan tidak akan mencakup analisis mendalam terkait tingkat keamanan algoritma RSA dan AES, seperti ketahanan terhadap serangan tertentu (misalnya, *brute force* atau serangan *side-channel*).

1.6 Definisi Istilah

1. Algoritma merupakan serangkaian langkah yang disusun secara teratur untuk menyelesaikan suatu masalah dan menghasilkan output tertentu.

2. AES (*Advanced Encryption Standard*) adalah cipher blok simetris yang dapat digunakan untuk mengenkripsi dan mendekripsi data.
3. RSA (*Rivest-Shamir-Adleman*) adalah teknik kriptografi di mana kunci yang digunakan untuk enkripsi berbeda dari kunci yang digunakan untuk dekripsi.
4. Plainteks adalah pesan yang berisi informasi yang dapat dibaca, yang bisa berupa teks, gambar, audio, atau video.
5. Cipherteks adalah pesan yang telah dienkripsi dan tidak dapat dibaca tanpa dekripsi.
6. Enkripsi adalah proses mengubah informasi menjadi format yang tidak dapat dibaca oleh pihak yang tidak berwenang.
7. Dekripsi adalah proses mengubah data yang terenskripsi menjadi bentuk yang dapat dipahami kembali.
8. XOR (*Exclusive OR*) adalah metode operasi logika yang melibatkan dua bit.

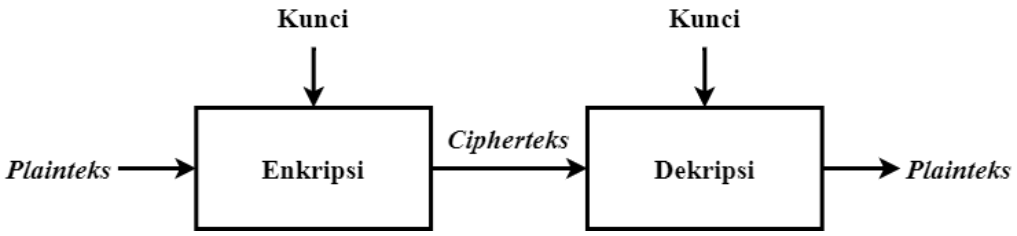
BAB II
KAJIAN TEORI

2.1 Teori Pendukung

2.1.1 Kriptografi

Kriptografi (*Cryptography*) berasal dari bahasa Yunani, di mana "*cryptos*" berarti "rahasia" dan "*graphein*" berarti "menulis". Maka, secara harfiah kriptografi berarti "tulisan rahasia" (Munir, 2019). Ada berbagai definisi kriptografi yang ditemukan dalam literatur. Definisi lama, seperti yang dijelaskan dalam buku-buku sebelum tahun 1980-an, menggambarkan kriptografi sebagai ilmu dan seni untuk menjaga kerahasiaan pesan dengan mengubahnya ke dalam bentuk yang tidak dapat dimengerti (Meyer & Matyas, 1982). Definisi ini relevan pada masa lalu ketika kriptografi digunakan untuk melindungi komunikasi penting dalam konteks militer, diplomatik, dan spionase. Namun, saat ini, kriptografi tidak hanya bertujuan untuk menjaga kerahasiaan, tetapi juga untuk memastikan integritas data, otentikasi, dan non-repudiation. Sebagai alternatif, Menez (1996) mendefinisikan kriptografi sebagai ilmu yang mempelajari teknik-teknik matematika yang berkaitan dengan aspek keamanan informasi, seperti kerahasiaan, integritas data, dan otentikasi.

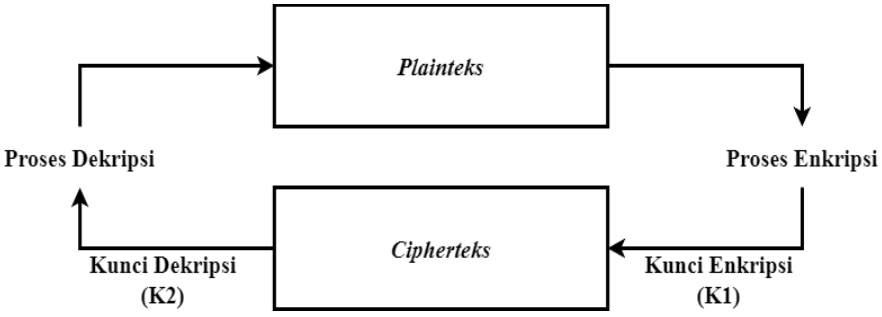
Pada dasarnya, kriptografi digunakan untuk memastikan bahwa pesan yang dikirim kepada penerima tetap aman dari gangguan pihak lain. Proses ini dilakukan dengan mengacak pesan sehingga maknanya tidak dapat diketahui oleh pihak yang tidak berwenang. Setelah itu, penerima pesan menggunakan algoritma yang sama untuk mengembalikan makna asli pesan tersebut.



Gambar 2.1 Proses Enkripsi dan Dekripsi.

Proses mengubah *plainteks* menjadi *cipherteks* disebut "enkripsi" (*encryption*) menurut standar ISO 7498-2. Sebaliknya, proses mengembalikan *cipherteks* menjadi *plainteks* disebut "dekripsi" (*decryption*) menurut standar ISO 7498-2. Diagram untuk enkripsi dan dekripsi dapat dilihat pada Gambar 2.1. Proses enkripsi menerima input berupa plainteks dan kunci, dan menghasilkan cipherteks. Sementara itu, proses dekripsi menerima input berupa cipherteks dan kunci, dan mengembalikan hasilnya menjadi plainteks.

Beberapa faktor kunci dalam enkripsi dan dekripsi data termasuk algoritma yang dirancang untuk menciptakan sistem yang aman serta kunci yang digunakan untuk menjaga kerahasiaan *cipherteks* agar tidak mudah dipecahkan atau bocor. Secara umum, algoritma kriptografi terdiri dari serangkaian langkah-langkah sistematis yang digunakan dalam enkripsi dan dekripsi. Dalam kriptografi, terdapat dua jenis algoritma: algoritma simetris dan algoritma asimetris.



Gambar 2.2 Alur Enkripsi dan Dekripsi Algoritma Simetris.

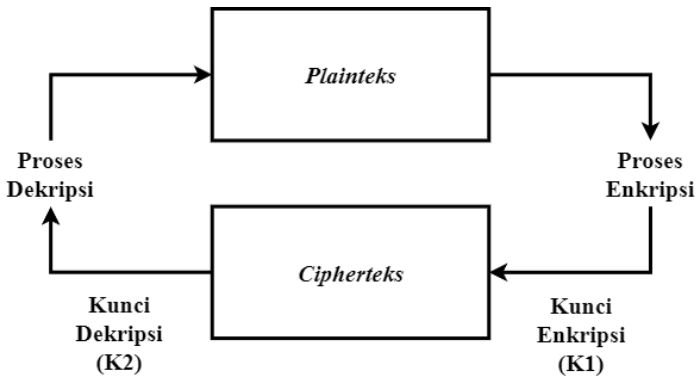
Gambar 2.2 menunjukkan algoritma simetris, yang menggunakan kunci yang sama untuk enkripsi dan dekripsi. Karena itu, algoritma ini sering disebut sebagai algoritma dengan satu kunci (*single-key algorithm*).

Menurut Widiasari (2014), algoritma simetris memiliki sejumlah kelebihan dan kekurangan, seperti yang dijelaskan dalam Tabel 2.1.

Tabel 2.1 Kelebihan dan Kekurangan Algoritma Simetris.

No	Indikator	Keterangan
1	Kelebihan	Kecepatan operasi sistem yang lebih tinggi daripada algoritma asimetris.
		Dapat digunakan pada sistem *real-time* karena kecepatan operasi yang tinggi.
2	Kekurangan	Kesulitan manajemen kunci karena kebutuhan kunci yang berbeda pada tiap pengiriman pesan.

Sebaliknya, algoritma asimetris, seperti yang ditunjukkan pada Gambar 2.3, berbeda dari algoritma simetris. Pada algoritma asimetris, kunci yang digunakan untuk enkripsi dan dekripsi tidaklah sama. Kunci publik yang dapat didistribusikan secara bebas digunakan untuk enkripsi, sedangkan kunci privat, yang harus dirahasiakan oleh pengguna, digunakan untuk dekripsi. Meskipun kunci publik dapat diakses, menemukan kunci privat tetap sangat sulit dalam algoritma asimetris.



Gambar 2.3 Proses Enkripsi dan Dekripsi Algoritma Asimetris.

Menurut Widiasari (2014), juga menuturkan bahwa terdapat beberapa kelebihan dan kekurangan pada algoritma asimetris seperti pada Tabel 2.2.

Tabel 2.2 Kelebihan dan Kekurangan Algoritma Asimetris.

No	Indikator	Keterangan
1	Kelebihan	Keamanan pada distribusi kunci lebih aman dan lebih baik.
		Jumlah kunci lebih sedikit sehingga manajemen kunci dapat lebih baik.
2	Kekurangan	Kecepatan operasi lebih rendah dibandingkan algoritma asimetris.
		Kunci yang digunakan lebih panjang daripada algoritma asimetris pada tingkat keamanan sama.

2.1.2 Citra Digital

Menurut Kamus Webster, citra adalah hasil tiruan atau representasi dari suatu objek. Di sisi lain, dalam KBBI, citra diartikan sebagai gambar atau bentuk rupa (Sulistiyanti, dkk., 2016). Citra terbagi menjadi dua jenis, yaitu citra kontinu yang dihasilkan oleh sinyal analog, seperti yang diambil oleh kamera analog atau mata manusia, dan citra diskrit yaitu hasil digitalisasi dari citra kontinu yang sering disebut sebagai citra digital. Citra digital atau gambar digital memiliki peran penting dalam penyampaian informasi visual karena memiliki karakteristik yang berbeda dari data teks (Sutoyo, dkk., 2009).

Dalam penggunaan citra digital, diperlukan teknik pengolahan gambar agar gambar yang dihasilkan memenuhi kriteria yang dibutuhkan dalam penelitian.

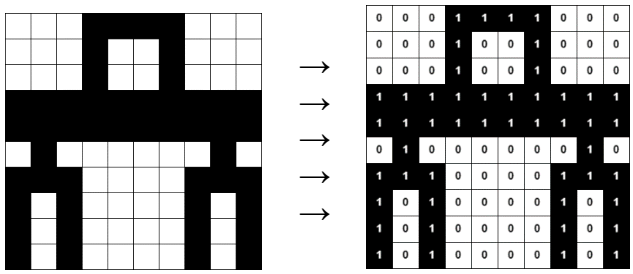
Menurut KBBI, pengolahan adalah proses memodifikasi sesuatu agar menjadi lebih baik atau berubah sesuai keperluan. Oleh karena itu, pengolahan citra dapat diartikan sebagai proses pengolahan gambar untuk menghasilkan citra sesuai dengan yang diinginkan (Sulistiyanti, dkk., 2016). Proses mengubah citra dari fungsi kontinu menjadi nilai-nilai diskrit disebut digitalisasi. Setelah melalui proses ini, citra digital terbentuk dan memiliki dimensi yang dinyatakan dengan lebar (M) $\times$ tinggi (N) (Cahyanti, dkk., 2016).

Citra digital terdiri dari dua komponen utama, yaitu koordinat dan tingkat keabuan (*gray-level*) dan setiap titik koordinat memiliki nilai yang menentukan warna pada setiap piksel. Jumlah baris atau kolom piksel mencerminkan resolusi citra tersebut. Resolusi citra berfungsi untuk menunjukkan tingkat kejelasan sebuah citra. Resolusi citra sebanding dengan detailnya, sehingga dapat disimpulkan bahwa semakin tinggi resolusi suatu citra, semakin detail citra tersebut (Sulistiyanti, dkk., 2016).

Tingkat keabuan pada citra digital berkaitan dengan digit biner atau bit yang terdiri dari angka 0 dan 1, dengan semua warna berasal dari tiga warna utama: merah, hijau, dan biru. Nilai piksel dalam citra adalah bilangan biner dengan panjang k, di mana k menunjukkan kedalaman bit (*bit depth*) citra tersebut. Jenis citra akan mempengaruhi skala bit yang digunakan. Secara umum, citra digital terbagi menjadi tiga jenis: citra biner, citra berwarna, dan citra hitam-putih (Kusumanto & Tompunu, 2011).

Pada Gambar 2.4 terlihat bahwa matriks pada citra biner hanya terdiri dari angka 0 dan 1, sehingga hanya menghasilkan warna hitam dan putih. Citra biner

sering digunakan dalam pemrosesan citra (*image processing*) yang memerlukan deteksi tepi (*edge detection*) untuk mendapatkan bentuk objek yang lebih jelas.



Gambar 2.4 Matriks Citra Digital *Binary Image*.

Gambar 2.5 menjelaskan tentang *Grayscale Image*, di mana gambar disimpan dalam bentuk matriks dengan elemen-elemen yang berada pada rentang nilai 0 (hitam) hingga 255 (putih). Nilai antara 0 dan 255 mewakili berbagai tingkat warna abu-abu. Representasi gambar dalam bentuk *grayscale* sering digunakan dalam pemrosesan citra (Purnama, 2015).

Gambar 2.5 (Kiri) Gambar Direpresentasikan dalam Bentuk *Greyscale Image*, (Kanan) Cuplikan Isi Matriks pada Gambar.

Pada Gambar 2.6 menjelaskan tentang Citra RGB, di mana gambar disimpan dalam matriks dengan elemen-elemen berupa triple (x_1, x_2, x_3), di mana x_1 mewakili intensitas warna merah, x_2 mewakili hijau, dan x_3 mewakili biru. Intensitas masing-masing warna berkisar antara 0-255 (Purnama, 2015).

Pada komponen R (merah), bagian kiri gambar menunjukkan intensitas tinggi (ditunjukkan oleh gambar *grayscale* berwarna putih), yang berarti intensitas warna merah paling tinggi berada di sisi kiri gambar. Dalam pengolahan gambar dengan format RGB, gambar tersebut terlebih dahulu dipisahkan berdasarkan komponennya. Setiap komponen kemudian diproses seperti gambar *grayscale*. Setelah proses selesai, komponen-komponen tersebut digabungkan kembali untuk membentuk gambar RGB yang lengkap.

Gambar 2.6 Gambar RGB Dipisahkan Berdasarkan Komponennya.

Citra digital memiliki beberapa format umum, termasuk Bitmap (BMP), *Joint Photographic Experts Group* (JPEG/JPG), *Graphics Interchange Format* (GIF), dan *Portable Network Graphics* (PNG).

2.1.3 Algoritma RSA

Di antara berbagai algoritma kriptografi kunci-publik yang ada, RSA adalah salah satu yang paling terkenal. Algoritma ini dikembangkan oleh tiga peneliti dari MIT (*Massachusetts Institute of Technology*) pada tahun 1976, yaitu Ron Rivest, Adi Shamir, dan Leonard Adleman, sehingga RSA merupakan singkatan dari Rivest-Shamir-Adleman. Keamanan RSA bergantung pada kesulitan memfaktorkan bilangan besar menjadi faktor-faktor primanya, yang diperlukan

untuk menghasilkan kunci privat. Selama tidak ditemukan metode yang efektif untuk memfaktorkan bilangan bulat besar, keamanan RSA tetap terjaga.

Sebagai algoritma kriptografi kunci-publik, RSA menggunakan dua jenis kunci: kunci publik dan kunci privat. Kunci publik digunakan untuk enkripsi dan dapat diakses oleh siapa saja, sementara kunci privat digunakan untuk dekripsi dan harus disimpan dengan aman, hanya diketahui oleh pengirim dan penerima. Algoritma RSA memiliki besaran-besaran seperti pada Tabel 2.3.

Tabel 2.3 Parameter pada Algoritma RSA.

No	Parameter	Keterangan
1	p dan q adalah bilangan prima	Rahasia
2	$n = p \cdot q$	Tidak Rahasia
3	$\phi(n) = (p-1)(q-1)$	Rahasia
4	e (Kunci Enkripsi)	Tidak Rahasia
5	d (Kunci Dekripsi)	Rahasia
6	m (*Plainteks*)	Rahasia
7	c (*Cipherteks*)	Tidak Rahasia

Algoritma RSA melibatkan tiga tahapan utama: pembuatan kunci, enkripsi, dan dekripsi. Proses enkripsi dan dekripsi dalam RSA didasarkan pada aritmetika modulo dan prinsip bilangan prima (Ginting, dkk., 2015). Dalam algoritma ini, kunci untuk enkripsi dan dekripsi berupa bilangan bulat, di mana kunci enkripsi dikenal sebagai kunci publik, sementara kunci dekripsi disebut kunci privat. Untuk menghasilkan kunci privat, diperlukan pemfaktoran bilangan bulat menjadi faktor-faktor primanya, yang merupakan tugas yang sangat menantang karena belum ada metode efisien untuk pemfaktoran ini. Salah satu teknik yang digunakan untuk pemfaktoran adalah pohon faktor, tetapi RSA memanfaatkan bilangan prima yang

sangat besar, menjadikan pemfaktoran lebih rumit dan memakan waktu. Kondisi ini meningkatkan kekuatan algoritma RSA, sehingga dianggap aman dan mampu menghasilkan proses enkripsi dan dekripsi yang sulit dipecahkan (Belkaid, 2015).

Langkah pertama dalam algoritma RSA adalah pembuatan kunci, yang mencakup pembuatan kunci publik dan kunci privat. Sesuai namanya, kunci publik adalah kunci yang tersedia untuk umum dan digunakan untuk mengenkripsi pesan (Gollagi, dkk., 2021). Pesan yang dikirimkan melalui enkripsi dengan kunci publik kemudian dapat didekripsi menggunakan kunci privat. Proses pembuatan kunci dalam algoritma RSA dilakukan melalui langkah-langkah berikut:

1. Pilih dua bilangan prima yang berbeda, yaitu p dan q.
2. Untuk keamanan, bilangan prima p dan q harus dipilih dengan panjang bit yang sama. Bilangan prima dapat ditemukan secara efisien melalui pengujian primalitas.
3. Kemudian, hitung nilai n yang merupakan hasil perkalian p dan q, yaitu $n = pq$.
4. n adalah modulus yang digunakan untuk kunci publik dan kunci privat. Panjangnya disebut panjang kunci yang biasanya dinyatakan dalam bit.
5. Hitung totient Euler dari n, yaitu

$$\phi(n) = \phi(p)\phi(q) = (p-1)(q-1) \tag{2.1}$$

di sini ϕ adalah fungsi totient Euler. Nilai ini harus disimpan secara rahasia.

6. Pilih bilangan bulat e relatif prina terhadap $\phi(n)$.
7. Tentukan d sebagai

$$d \equiv e - 1\big(mod\ \phi(n)\big) \tag{2.2}$$

yaitu d merupakan invers multiplikatif modular dari

$$e\left(mod\ \phi(n)\right) \tag{2.3}$$

Ini dapat dilakukan dengan menyelesaikan persamaan

$$e \cdot d \equiv 1\ \left(mod\ \phi(n)\right) \tag{2.4}$$

Proses ini menggunakan algoritma Euclidean yang diperpanjang. Input a dan n sesuai dengan e dan $\phi(n)$.

Hasil dari algoritma di atas yaitu, kunci publik dengan pasangan (e, n) yang dapat dikirim kepada siapa saja yang akan berkirim pesan dan kunci privat dengan pasangan (d, n) dimana d harus dijaga kerasahasiannya karena d digunakan untuk mendeskripsikan cipherteks yang diterima.

Pada proses enkripsi dan dekripsi, Munir (2019) menjelaskan dengan sebuah analogi:

1. Enkripsi

   Misalkan Bob akan mengirim pesan m kepada Alice dan akan mengenkripsinya menggunakan algoritma RSA. Nilai m haruslah terletak di dalam $[0, n-1]$. Prosedur yang dilakukan bob adalah mengambil kunci publik Alice, e, dan nilai modulus n. Kemudian menghitung

   $$c = m^e \bmod(n) \tag{2.5}$$

   dan mengirim c kepada Alice.

2. Dekripsi

   Alice mendekripsi *cipherteks* dari Bob dengan menggunakan kunci privatnya, d, dan modulus n. Kemudian menghitung

   $$m = c^d \bmod(n) \tag{2.6}$$

2.1.4 Algoritma AES

Pada bulan November 2001, National Institute of Standards and Technology (NIST) mengumumkan bahwa Rijndael telah dipilih sebagai AES (*Advanced Encryption Standard*), dengan harapan menjadi standar kriptografi utama setidaknya selama 10 tahun ke depan. AES mendukung panjang kunci dari 128 bit hingga 256 bit, dengan kenaikan setiap 32 bit. Panjang kunci dan ukuran blok dapat dipilih secara terpisah. Karena AES menyediakan panjang kunci yang ditetapkan, yaitu 128, 192, dan 256 bit, maka dikenal sebagai AES-128, AES-192, dan AES-256. Tabel 2.4 akan menunjukkan perbedaan antara ketiga versi AES tersebut.

Tabel 2.4 Perbedaan Tiga Versi AES.

Versi	Panjang Kunci (*Nk words*)	Ukuran Blok (*Nb words*)	Jumlah Putaran (*Nr*)
AES-128	4	4	10
AES-192	6	4	12
AES-256	8	4	14

Secara de-fakto, hanya ada dua varian AES, yaitu AES-128 dan AES-256 karena akan sangat jarang seseorang menggunakan kunci yang panjangnya 192 bit. AES memiliki panjang kunci paling sedikit 128 bit yang membuatnya tahan terhadap serangan *brute force* dengan teknologi saat ini. Dengan panjang kunci 128 bit, maka terdapat sebanyak $2^{128} = 3{,}4 \times 10^{38}$ (Munir, 2019).

Dengan 16-*byte*, maka baik blok data dan kunci yang berukuran 128-bit dapat disimpan di dalam ketiga larik tersebut $(128 = 16 \times 8)$. Selama kalkulasi *plainteks* menjadi *cipherteks*, status data sekarang disimpan di dalam matriks dua

dimensi (*State*), bertipe *byte* dan berukuran $Nrows \times Ncols$. Untuk blok data 128-bit, ukuran *state* adalah 4×4. Elemen matriks *state* diacu sebagai $S[r, c]$, dengan $0 \leq r < 4$ dan $0 \leq c < Nb$ (Nb adalah panjang blok dibagi 32. Pada AES-128, $Nb = \frac{128}{32} = 4$) (Munir, 2019). Algortima AES memiliki tiga paramater, seperti pada Tabel 2.5.

Tabel 2.5 Parameter Algoritma AES.

No	Parameter	Keterangan
1	*In*	Larik berukuran 16-*byte* yang berisi data masukan.
2	*Out*	Larik berukuran 16-*byte* yang berisi hasil enkripsi.
3	*Key*	Larik berukuran 16-*byte* yang berisi kunci *cipher key*.

Dalam algoritma AES dimulai dengan tahap *AddRoundKey*, diikuti oleh sembilan putaran yang terdiri dari empat tahap, serta putaran kesepuluh dengan tiga tahap, yang berlaku baik untuk algoritma enkripsi maupun dekripsi. Menurut Munir (2019) tahap-tahap dalam putaran ini diatur oleh:

1. *AddRoundKey*: melakukan XOR antara *state* awal (*plainteks*) dengan *cipher key*. Tahap ini disebut juga putaran awal.
2. Putaran sebanyak $Nr - 1$ kali. Proses yang dilakukan pada setiap putaran adalah:
    a. *SubBytes*: subtitusi *byte* dengan menggunakan tabel substitutsi (*S-box*).
    b. *ShiftRows*: pergeseran baris-baris *array state* secara *wrapping*.
    c. *MixColumns*: mengacak data di masing-masing kolom *array state*.

d. *AddRoundKey*: melakukan XOR antara *state* sekarang *round key*.

3. Proses untuk putaran terakhir:

   a. *SubBytes*.

   b. *ShiftRows*.

   c. *AddRoundKey*.

Hasil putaran akhir adalah *cipherteks* yang dinyatakan didalam *state*.

Menurut Munir (2019) di dalam bukunya yang berjudul Kriptografi, menjelaskan bahwa terdapat empat transformasi pada algoritma AES, yakni Transformasi *SubBytes*, Transformasi *ShiftRows*, Transformasi *MixColumns*, dan Transformasi *AddRoundKey*. Penjelasan terkait keempat transformasi ini akan diulas, yaitu:

1. Transformasi *SubBytes.*

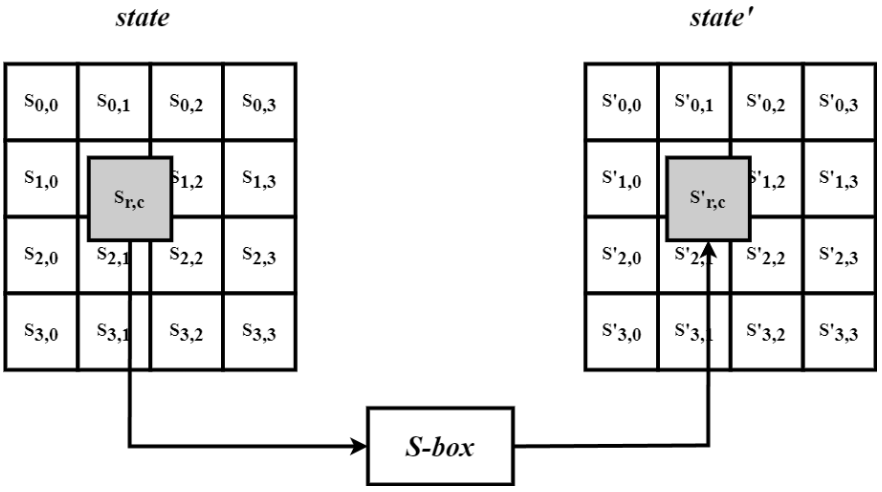
   Merupakan sebuah operasi substitusi di dalam AES. *SubBytes* memetakan setiap *byte* dari larik *state* dengan menggunakan tabel substitusi *S-box*. Tabel *S-box* berupa matriks berukuran 16×16 yang entrinya merupakan permutasi dari 256 kemungkinan susunan karakter heksadesimal.

   Karakter heksadesimal ada 16 kemungkinan $(0,1,2,3, \dots, D, E, F)$. Karena satu *byte* terdiri dari 8 bit dan satu karakter heksadesimal panjangnya 4 bit, maka satu *byte* dinyatakan dalam dua karakter heksadesimal (Arrañaga, 2017). Gambar 2.7 memperlihatkan bentuk tabel *S-box*.

	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	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

Gambar 2.7 Tabel *S-box* dalam AES.

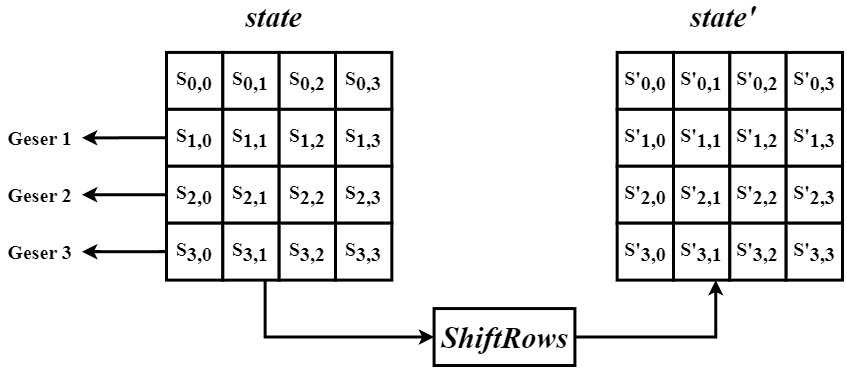
Operasi substitusi menggunakan *S-box* sederhana, caranya adalah untuk setiap *byte* pada matriks *state*, misalkan $S[r, c] = xy$ yang dalam hal ini xy adalah digit heksadesimal dari nilai $S[r, c]$, maka nilai substitusinya adalah $S'[r, c]$ yang merupakan perpotongan baris x dengan kolom y di dalam *S-box*. Gambar 2.8 memperlihatkan transformasi *SubBytes*.



Gambar 2.8 Transformasi *SubBytes*.

2. Transformasi *ShiftRows.*

Transformasi *ShiftRows* melakukan pergeseran secara siklik (*wrapping*) pada tiga baris terahir dari matriks *state*. Jumlah pergeseran bergantung pada nilai baris (*r*). Elemen-elemen pada baris $r = 1$ digeser sejauh 1 *byte* ke kiri, elemen-elemen pada baris $r = 2$ digeser sejauh 2 *byte* ke kiri, dan elemen-elemen pada baris $r = 3$ digeser sejauh 3 *byte*. Baris $r = 0$ tidak digeser.



Gambar 2.9 Transformasi *ShiftRows*.

Gambar 2.9 memperlihatkan transformasi *ShiftRows*. Operasi pergeseran merupakan jenis operasi transposisi yang mengubah urutan data di dalam *plainteks*.

3. Transformasi *MixColumns.*

Transformasi *MixColumns* menghasilkan matriks *state* dengan sebuah matriks tertentu.

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} S_{0,0} & S_{0,1} & S_{0,2} & S_{0,3} \\ S_{1,0} & S_{1,1} & S_{1,2} & S_{1,3} \\ S_{2,0} & S_{2,1} & S_{2,2} & S_{2,3} \\ S_{3,0} & S_{3,1} & S_{3,2} & S_{3,3} \end{bmatrix} = \begin{bmatrix} S'_{0,0} & S'_{0,1} & S'_{0,2} & S'_{0,3} \\ S'_{1,0} & 'S_{1,1} & S'_{1,2} & S'_{1,3} \\ S'_{2,0} & S'_{2,1} & S'_{2,2} & S'_{2,3} \\ S'_{3,0} & S'_{3,1} & S'_{3,2} & S'_{3,3} \end{bmatrix} \quad (2.7)$$

MixColumns memberikan efek difusi pada *chiper*. Setiap kolom diperlakukan sebagai polinom 4-suku pada $GF(2^8)$. Transformasi *MixColumns* pada sebuah kolom matriks *state* dinyatakan:

$$\begin{bmatrix} S'_{0,j} \\ S'_{1,j} \\ S'_{2,j} \\ S'_{3,j} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} S_{0,j} \\ S_{1,j} \\ S_{2,j} \\ S_{3,j} \end{bmatrix} \quad (2.8)$$

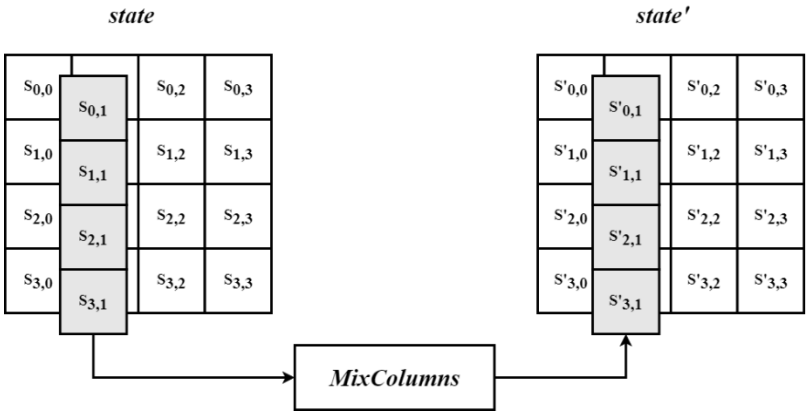
$$S'_{0,j} = (\{02\} \cdot S_{0,j}) \oplus (\{03\} \cdot S_{1,j}) \oplus S_{2,j} \oplus S_{3,j} \quad (2.9)$$

$$S'_{1,j} = S_{0,j} \oplus (\{02\} \cdot S_{1,j}) \oplus (\{03\} \cdot S_{2,j}) \oplus S_{3,j} \quad (2.10)$$

$$S'_{2,j} = S_{0,j} \oplus S_{1,j} \oplus (\{02\} \cdot S_{2,j}) \oplus (\{03\} \cdot S_{3,j}) \quad (2.11)$$

$$S'_{3,j} = (\{03\} \cdot S_{0,j}) \oplus S_{1,j} \oplus S_{2,j} \oplus (\{02\} \cdot S_{3,j}) \quad (2.12)$$

Simbol " · " menyatakan perkalian dalam $GF(2^8)$, sedangkan simbol "⊕" menyatakan operator *bitwise* XOR. Gambar 2.10 memperlihatkan transformasi *MixColumns*.



Gambar 2.10 Transformasi *MixColumns.*

Terdapat cara yang lebih mudah dalam menghitung *MixColumns* dengan menggunakan dua tabel, yaitu tabel E dan tabel L sebagai berikut (Berent, 2016):

E Table

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	01	03	05	0F	11	33	55	FF	1A	2E	72	96	A1	F8	13	35
1	5F	E1	38	48	D8	73	95	A4	F7	02	06	0A	1E	22	66	AA
2	E5	34	5C	E4	37	59	EB	26	6A	BE	D9	70	90	AB	E6	31
3	53	F5	04	0C	14	3C	44	CC	4F	D1	68	B8	D3	6E	B2	CD
4	4C	D4	67	A9	E0	3B	4D	D7	62	A6	F1	08	18	28	78	88
5	83	9E	B9	D0	6B	BD	DC	7F	81	98	B3	CE	49	DB	76	9A
6	B5	C4	57	F9	10	30	50	F0	0B	1D	27	69	BB	D6	61	A3
7	FE	19	2B	7D	87	92	AD	EC	2F	71	93	AE	E9	20	60	A0
8	FB	16	3A	4E	D2	6D	B7	C2	5D	E7	32	56	FA	15	3F	41
9	C3	5E	E2	3D	47	C9	40	C0	5B	ED	2C	74	9C	BF	DA	75
A	9F	BA	D5	64	AC	EF	2A	7E	82	9D	BC	DF	7A	8E	89	80
B	9B	B6	C1	58	E8	23	65	AF	EA	25	6F	B1	C8	43	C5	54
C	FC	1F	21	63	A5	F4	07	09	1B	2D	77	99	B0	CB	46	CA
D	45	CF	4A	DE	79	8B	86	91	A8	E3	3E	42	C6	51	F3	0E
E	12	36	5A	EE	29	7B	8D	8C	8F	8A	85	94	A7	F2	0D	17
F	39	4B	DD	7C	84	97	A2	FD	1C	24	6C	B4	C7	52	F6	01

L Table

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0		00	19	01	32	02	1A	C6	4B	C7	1B	68	33	EE	DF	03
1	64	04	E0	0E	34	8D	81	EF	4C	71	08	C8	F8	69	1C	C1
2	7D	C2	1D	B5	F9	B9	27	6A	4D	E4	A6	72	9A	C9	09	78
3	65	2F	8A	05	21	0F	E1	24	12	F0	82	45	35	93	DA	8E
4	96	8F	DB	BD	36	D0	CE	94	13	5C	D2	F1	40	46	83	38
5	66	DD	FD	30	BF	06	8B	62	B3	25	E2	98	22	88	91	10
6	7E	6E	48	C3	A3	B6	1E	42	3A	6B	28	54	FA	85	3D	BA
7	2B	79	0A	15	9B	9F	5E	CA	4E	D4	AC	E5	F3	73	A7	57
8	AF	58	A8	50	F4	EA	D6	74	4F	AE	E9	D5	E7	E6	AD	E8
9	2C	D7	75	7A	EB	16	0B	F5	59	CB	5F	B0	9C	A9	51	A0
A	7F	0C	F6	6F	17	C4	49	EC	D8	43	1F	2D	A4	76	7B	B7
B	CC	BB	3E	5A	FB	60	B1	86	3B	52	A1	6C	AA	55	29	9D
C	97	B2	87	90	61	BE	DC	FC	BC	95	CF	CD	37	3F	5B	D1
D	53	39	84	3C	41	A2	6D	47	14	2A	9E	5D	56	F2	D3	AB
E	44	11	92	D9	23	20	2E	89	B4	7C	B8	26	77	99	E3	A5
F	67	4A	ED	DE	C5	31	FE	18	0D	63	8C	80	C0	F7	70	07

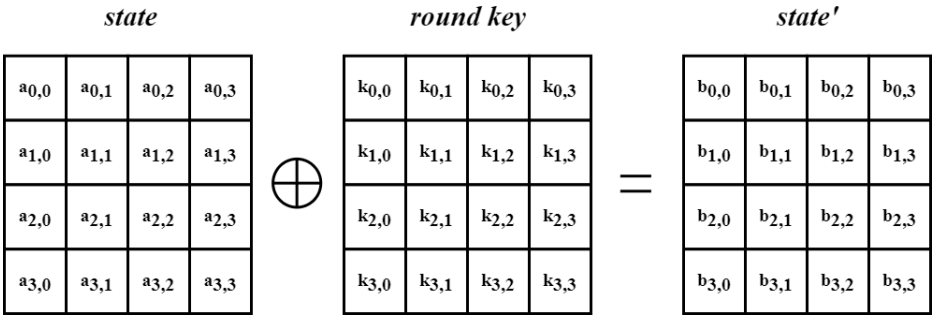
Gambar 2.11 Tabel E dan Tabel L.

Cara menggunakan tabel L dan E adalah misalkan akan menghitung $(02 \cdot 26)$ cari $L(02)$ yaitu perpotongan baris 0 dan kolom 2 pada Tabel L, kemudian cari $L(26)$ yaitu perpotongan baris 2 dan kolom 6 pada Tabel L. Maka diperoleh $L(02) = 19$ dan $L(26) = 27$. Selanjutnya jumlahkan 19

dan 27 dalam basis 16, maka $19 + 27$ menghasilkan nilai 40. Untuk mencari $E(40)$, yaitu perpotongan baris 4 dan kolom 0 pada Tabel E sehingga diperoleh $E(40) = 4C$. Maka didapatlah hasil dari $(02 \cdot 26) = 4C$.

4. Transformasi *AddRoundKey*.

Transformasi ini melakukan operasi *bitwise* XOR antara sebuah *round key* dengan matriks *state*, dan hasilnya disimpan pada matriks *state* yang sama. Gambar 2.12 memperlihatkan transformasi *AddRoundKey*. Elemen-elemen pada matriks *state* di-XOR-kan dengan elemen yang sesuai pada matriks *round key*, yaitu $a_{ij} \oplus k_{ij} = b_{ij}$.



Gambar 2.12 Transformasi *AddRoundKey*.

Dalam algoritma AES, terdapat proses ekspansi kunci, di mana setiap putaran dalam algoritma menggunakan kunci putaran (*round key*). Kunci putaran ini dihasilkan dari kunci utama yang diberikan kepada pengguna, yang dikenal sebagai *cipher key*. Proses pembangkitan semua kunci putaran dilakukan oleh fungsi *KeyExpansion* (Venkatesha, 2019).

Dalam algoritma dekripsi AES, prosesnya hampir serupa dengan algoritma enkripsi, namun terdapat beberapa perubahan pada konstanta dan tabel yang digunakan. Untuk membedakannya dari enkripsi, fungsi-fungsi yang bekerja secara

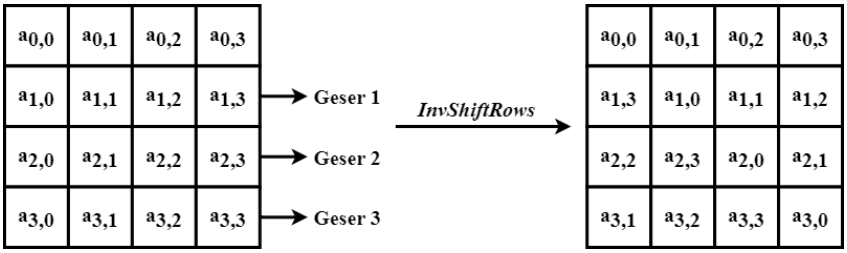
terbalik diberi awalan "*Inv*", yang berarti inversi. Terdapat tiga tahap pada algoritma dekripsi, yakni *InvSubBytes*, *InvShiftRows*, *InvMixColumns*.

Operasi substitusi *byte* dalam *InvSubBytes* mirip dengan *SubBytes*, tetapi menggunakan inversi dari *S-box* yang digunakan dalam *SubBytes* (lihat Gambar 2.8). Tabel inversi *S-box* yang dipakai selama proses dekripsi ditampilkan pada Gambar 2.13.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	52	09	6A	D5	30	36	A5	38	BF	40	A3	9E	81	F3	D7	FB
1	7C	E3	39	82	9B	2F	FF	87	34	83	43	44	C4	DE	E9	CB
2	54	7B	94	32	A6	C2	23	3D	EE	4C	95	0B	42	FA	C3	4E
3	08	2E	A1	66	28	D9	24	B2	76	5B	A2	49	6D	8B	D1	25
4	72	F8	F6	64	86	68	98	16	D4	A4	5C	CC	5D	65	B6	92
5	6C	70	48	50	FD	ED	B9	DA	5E	15	46	57	A7	8D	9D	84
6	90	D8	AB	00	8C	BC	D3	0A	F7	E4	58	05	B8	B3	45	06
7	D0	2C	1E	8F	CA	3F	0F	02	C1	AF	BD	03	01	13	8A	6B
8	3A	91	11	41	4F	67	DC	EA	97	F2	CF	CE	F0	B4	E6	73
9	96	AC	74	22	E7	AD	35	85	E2	F9	37	E8	1C	75	DF	6E
A	47	F1	A1	71	1D	29	C5	89	6F	B7	62	0E	AA	18	BE	1B
B	FC	56	3E	4B	C6	D2	79	20	9A	DB	C0	FE	78	CD	5A	F4
C	1F	DD	A8	33	88	07	C7	31	B1	12	10	59	27	80	EC	5F
D	60	51	7F	A9	19	B5	4A	0D	2D	E5	7A	9F	93	C9	9C	EF
E	A0	E0	3B	4D	AE	2A	F5	B0	C8	EB	BB	3C	83	53	99	61
F	17	28	04	7E	BA	77	D6	26	E1	69	14	63	55	21	0C	7D

Gambar 2.13 Tabel Inversi *S-box*.

Transformasi *InvShiftRows* serupa dengan *ShiftRows*, tetapi pergeseran dilakukan ke arah yang berlawanan (ke kanan) untuk tiga baris terakhir dalam *state*. Proses transformasi *InvShiftRows* dapat dilihat pada Gambar 2.14.



Gambar 2.14 Transformasi *InvShiftRows*.

Terakhir adalah *InvMixColumns*, didefinisikan sebagai perkalian matriks seperti contoh berikut ini:

$$\begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \begin{bmatrix} S_{0,0} & S_{0,1} & S_{0,2} & S_{0,3} \\ S_{1,0} & S_{1,1} & S_{1,2} & S_{1,3} \\ S_{2,0} & S_{2,1} & S_{2,2} & S_{2,3} \\ S_{3,0} & S_{3,1} & S_{3,2} & S_{3,3} \end{bmatrix} = \begin{bmatrix} S'_{0,0} & S'_{0,1} & S'_{0,2} & S'_{0,3} \\ S'_{1,0} & 'S_{1,1} & S'_{1,2} & S'_{1,3} \\ S'_{2,0} & S'_{2,1} & S'_{2,2} & S'_{2,3} \\ S'_{3,0} & S'_{3,1} & S'_{3,2} & S'_{3,3} \end{bmatrix} \quad (2.13)$$

2.1.5 *Mean Square Error* (MSE)

Mean Square Error (MSE) merupakan metode pengukuran yang digunakan untuk menentukan sejauh mana kesalahan terjadi selama proses penyisipan data (Male, dkk., 2012). Dalam bidang kriptografi, MSE digunakan untuk mengevaluasi keakuratan proses enkripsi dan dekripsi. Hal ini dilakukan dengan membandingkan nilai piksel dari citra asli dengan citra yang telah melalui proses dekripsi. Perhitungan MSE dapat dijelaskan secara matematis dengan menggunakan rumus berikut:

$$MSE = \frac{1}{m \times n} \sum_{i=0}^{n-1} \sum_{j=0}^{m-1} \left(A_{i,j} - B_{i,j}\right)^2 \quad (2.14)$$

Untuk menjelaskan persamaan (2.14), maka tabel 2.6 akan menerangkan beberapa indikator dalam rumus.

Tabel 2.6 Indikator untuk Menghitung Nilai MSE.

No	Indikator	Keterangan
1	MSE	Nilai Mean Square Error pada citra.
2	m	Panjang citra hasil dekripsi.
3	n	Lebar citra hasil dekripsi.
4	$A_{i,j}$	Representasi satu piksel dari *plain image*.
5	$B_{i,j}$	Representasi satu piksel dari citra hasil dekripsi.
6	$m \times n$	Dimensi citra.

2.2 Kajian Integrasi Topik dengan Al-Quran/Hadits

Menurut Tafsir Tahlili dari website Qur'an NU, Surat An-Nur ayat 27 mengajarkan kepada umat Islam cara bergaul yang baik untuk menjaga dan memperkuat rasa cinta serta hubungan sosial yang harmonis. Ayat ini menekankan pentingnya meminta izin sebelum memasuki rumah orang lain dan memberi salam terlebih dahulu untuk menghindari melihat hal-hal yang bersifat pribadi atau yang seharusnya tidak terlihat. Jika tidak ada jawaban, permintaan izin sebaiknya dilakukan hingga tiga kali. Jika setelah itu tidak ada izin, lebih baik kembali dan tidak memaksakan diri masuk.

Ibnu Asyur, seorang ahli tafsir terkemuka dari Tunisia, menjelaskan bahwa ayat ini berfungsi sebagai pedoman tentang pentingnya meminta izin sebelum memasuki rumah. Menurutnya, rumah merupakan tempat yang sangat pribadi, bukan sekadar tempat berlindung dari cuaca panas, hujan, atau dingin, melainkan sebuah ruang yang menjaga privasi seseorang. Ibnu Asyur menekankan bahwa ayat ini mencerminkan kepedulian Islam terhadap perlindungan dan penghormatan terhadap privasi individu. Hal ini penting karena ada kemungkinan bahwa di dalam rumah terdapat hal-hal yang tidak ingin ditampilkan kepada orang luar, yang berkaitan dengan kehormatan pemiliknya. Menurut Ibnu Asyur, pada masa jahiliyyah, aturan tentang izin memasuki rumah tidak diterapkan secara ketat, terutama di kalangan orang miskin dan rakyat biasa. Aturan ini umumnya hanya berlaku untuk kalangan yang berkuasa. Oleh karena itu, ayat ini diturunkan untuk menekankan pentingnya menghormati privasi semua orang tanpa memandang status sosial mereka. (Ibnu Asyur, at-Tahrir wat-Tanwir, ad-Dar at-Tunisiyyah, juz 18 hlm. 196)

Menariknya, di masa modern, terdapat sistem keamanan yang berfungsi mencegah akses tidak sah ke jaringan pribadi yang disebut *firewall*. Sistem ini mirip dengan kisah Dzulqarnain dalam surat Al-Kahfi ayat 96, di mana ia diminta oleh suatu kaum untuk membangun tembok besi yang dilapisi api dan tembaga guna melindungi mereka dari serangan Yajuj Majuj yang merusak. Selain itu, banyak hadits Nabi yang menekankan pentingnya melindungi privasi seseorang agar tidak tersebar kepada pihak yang tidak berwenang. Perlindungan terhadap data pribadi sangat ditekankan, seperti yang tercermin dalam hadits yang diriwayatkan oleh Anas yang secara jelas menyampaikan hal ini:

"Dari Anas dia berkata, "Saya pernah didatangi oleh Rasulullah SAW ketika saya sedang bermain dengan teman-teman yang lain. Kemudian beliau mengucapkan salam kepada kami dan menyuruh saya untuk suatu keperluan hingga saya terlambat pulang ke rumah. Sesampainya di rumah, Ibu bertanya kepada saya, 'Mengapa kamu terlambat pulang? Maka saya pun menjawab, 'Tadi saya disuruh oleh Rasulullah untuk suatu keperluan.' Ibu saya terus bertanya, 'Keperluan apa?' Saya menjawab, 'Itu rahasia.' Ibu saya berkata, 'Baiklah, Janganlah kamu ceritakan rahasia Rasulullah SAW kepada siapapun.'" Anas berkata, "Demi Allah, kalau saya boleh menceritakan rahasia itu kepada seseorang, maka saya akan menceritakannya kepadamu, hai Tsabit!" (HR Muslim no 4533).

Melihat konteks dari ayat-ayat dan hadits tersebut, dapat disimpulkan bahwa prinsip-prinsip dalam teks-teks tersebut sangat relevan sebagai dasar untuk melindungi data pribadi di era digital saat ini. Terlihat jelas bahwa dalam maqashidus syariah terdapat prinsip-prinsip seperti menjaga kehormatan (*hifdzul 'irdli*), menjaga jiwa (*hifdzun nafsi*), dan menjaga harta (*hifdzul mali*). Oleh karena itu, penting untuk menerapkan solusi guna mencegah kejahatan digital dengan memastikan keamanan data dan privasi melalui perlindungan data digital.

2.3 Kajian Topik dengan Teori Pendukung

Dalam era teknologi digital yang berkembang pesat, perlindungan informasi menjadi semakin penting. Citra digital memiliki peran krusial dalam berbagai aplikasi seperti media sosial, sektor medis, dan sistem keamanan. Karena citra digital sering mengandung data sensitif, diperlukan metode enkripsi untuk melindunginya. Di dunia kriptografi, berbagai algoritma telah dikembangkan untuk menjaga keamanan informasi, termasuk RSA dan AES. Kedua algoritma ini memiliki karakteristik dan kinerja yang berbeda, terutama saat diterapkan pada enkripsi citra digital. Penelitian ini bertujuan untuk menganalisis dan membandingkan kinerja algoritma RSA dan AES dalam enkripsi citra digital. Evaluasi akan mencakup kecepatan proses enkripsi dan dekripsi, kualitas citra setelah dekripsi, serta tingkat keamanan yang ditawarkan oleh masing-masing algoritma.

Kriptografi adalah ilmu dan teknik untuk melindungi kerahasiaan informasi dengan mengubah data menjadi format yang tidak dapat dipahami oleh pihak yang tidak berwenang. Dalam kriptografi modern, terdapat dua jenis algoritma utama: kriptografi simetris dan asimetris. AES adalah contoh algoritma kriptografi simetris, sedangkan RSA adalah contoh algoritma kriptografi asimetris. Citra digital adalah representasi visual yang disimpan dalam format digital, terdiri dari piksel-piksel yang membentuk gambar. Penting untuk memastikan bahwa struktur dan kualitas citra tetap terjaga saat menerapkan enkripsi pada citra digital, agar informasi dapat diterima dengan baik setelah dekripsi.

RSA adalah algoritma kriptografi asimetris yang menggunakan pasangan kunci publik dan privat untuk proses enkripsi dan dekripsi. RSA dikenal karena

tingkat keamanannya yang tinggi, namun memerlukan sumber daya komputasi yang cukup besar, terutama untuk enkripsi data berukuran besar seperti citra digital. Oleh karena itu, penting untuk menganalisis kinerja RSA dalam konteks ini. Di sisi lain, AES adalah algoritma kriptografi simetris yang menggunakan satu kunci untuk proses enkripsi dan dekripsi. AES terkenal karena kecepatannya dan efisiensinya, sehingga sering digunakan dalam berbagai aplikasi keamanan. Meskipun AES lebih cepat dibandingkan RSA, kajian mendalam diperlukan untuk memahami dampaknya terhadap kualitas citra dan tingkat keamanan data. Penelitian ini bertujuan untuk membandingan kinerja algoritma RSA dan AES dalam enkripsi dan dekripsi citra digital, dengan fokus pada tiga aspek utama: waktu pemrosesan, kualitas citra yang dihasilkan, serta ukuran file hasil enkripsi dan dekripsi oleh masing-masing algoritma. Hasil analisis ini akan memberikan panduan untuk menentukan algoritma yang paling efektif untuk enkripsi citra digital.

BAB III
METODE PENELITIAN

3.1 Jenis Penelitian

Jenis penelitian ini adalah penelitian kuantitatif eksperimental. Penelitian ini difokuskan untuk membandingkan kinerja dua algoritma kriptografi, yaitu RSA dan AES, dalam konteks enkripsi citra digital. Eksperimen ini akan mengevaluasi kinerja kedua algoritma berdasarkan beberapa indikator, termasuk kecepatan enkripsi dan dekripsi, kualitas citra setelah dekripsi yang diukur dengan *Mean Squared Error* (MSE), serta tingkat keamanan yang dihasilkan oleh masing-masing algoritma. Pendekatan kuantitatif dipilih untuk menganalisis data secara objektif dan terukur. Dalam penelitian ini, variabel bebasnya adalah algoritma enkripsi yang digunakan, yaitu RSA dan AES. Sementara itu, variabel terikatnya mencakup kinerja algoritma, yang diukur melalui beberapa parameter, seperti waktu pemrosesan (kecepatan enkripsi dan dekripsi), kualitas citra yang dihasilkan setelah enkripsi (diukur dengan MSE), dan besaran ukuran file hasil enkripsi dan dekripsi dari masing-masing algoritma. Pendekatan ini akan memungkinkan perbandingan yang sistematis dan terukur antara kedua algoritma, serta memberikan hasil yang dapat diandalkan dan dapat dipertanggungjawabkan secara ilmiah.

3.2 Data dan Sumber Data

Data yang akan digunakan dalam penelitian ini adalah citra digital RGB yang memiliki 2 resolusi, yakni kecil (128×128 piksel) dan sedang (256×256 piksel) dengan format data *.png yang terdiri atas 25 gambar ukuran kecil dan 25 gambar ukuran sedang, dengan total objek adalah 50 gambar. Sumber data pada penelitian

ini bisa diperoleh pada situs Pixabay yang memiliki lisensi berdasarkan *Creative Commons Zero* atau disingkat "CC0", yang dapat diakses pada *link* https://pixabay.com/id/service/license-summary.

Citra digital dengan format *.png akan diolah menggunakan komputer yang menjalankan program Python, dengan bantuan *Library Pillow* untuk memproses data tersebut.

3.3 Teknik Analisis Data

Untuk memperoleh hasil yang signifikan dalam penelitian perbandingan kinerja algoritma RSA dan AES dalam enkripsi dan dekripsi citra digital, akan dilakukan proses enkripsi dan dekripsi pada objek citra digital yang berformat PNG. Langkah-langkah yang akan diambil dalam penelitian ini meliputi:

3.4.1 Proses Enkripsi dan Dekripsi dengan Algoritma RSA

Proses ini akan menerapkan algoritma RSA untuk enkripsi dan dekripsi data citra digital, menurut Sahoo, dkk., (2022) langkah-langkahnya sebagai berikut:

1. Pembuatan kunci algoritma RSA
    a. Pilih dua bilangan prima yang berbeda: misalkan p dan q adalah 2 bilangan prima.
    b. Hitung modulus n:

$$n = p \times q \tag{3.1}$$

    Modulus n ini digunakan sebagai bagian dari kunci publik dan kunci privat.
    c. Hitung fungsi euler $\phi(n)$:

$$\phi(n) = (p-1)(q-1) \quad (3.2)$$

d. Pilih eksponen enkripsi e:

Pilih bilangan e yang memenuhi syarat $1 < e < \phi(n)$ dan $GCD(e, \phi(n)) = 1$, artinya e dan $\phi(n)$ relatif prima.

e. Hitung eksponen dekripsi d:

Eksponen dekripsi d adalah invers modular dari e terhadap $\phi(n)$, yang artinya memenuhi:

$$e \times d \equiv 1(\text{mod } \phi(n)) \quad (3.3)$$

Pasangan (n, e) adalah kunci publik dan (n, d) adalah kunci privat.

2. Tahap enkripsi algoritma RSA

Untuk mengenkripsi citra digital, kita perlu terlebih dahulu mengonversi citra tersebut menjadi sekumpulan nilai piksel, dengan setiap piksel diwakili oleh nilai antara 0 hingga 255.

a. Representasi citra:

Membaca file citra PNG menggunakan *library* PIL dan mengonversi setiap komponen warna dari setiap piksel citra menjadi angka. Untuk piksel dengan nilai RGB $(255,0,0)$, maka dapat menggunakan 255 sebagai nilai yang akan dienkripsi.

b. Enkripsi piksel:

Setiap piksel m kemudian dienkripsi menggunakan kunci publik (n, e) dengan formula

$$c = m^e \text{ mod}(n) \quad (3.4)$$

dimana m adalah piksel asli, e adalah eksponen enkripsi dari kunci publik, dan c adalah piksel terenkripsi.

c. Penyimpanan citra terenkripsi:

Setelah seluruh piksel terenkripsi, piksel-piksel tersebut dikombinasikan kembali menjadi citra terenkripsi yang baru yang disimpan dalam format PNG.

3. Tahap dekripsi algoritma RSA

Untuk mendekripsi citra digital yang telah terenkripsi dengan algoritma RSA, akan digunakan kunci privat (n, d).

a. Representasi citra terenkripsi:

Ambil citra terenkripsi yang memiliki piksel c (hasil enkripsi)

b. Dekripsi piksel:

Setiap piksel terenkripsi c akan didekripsi menggunakan kunci privat (n, d) dengan formula

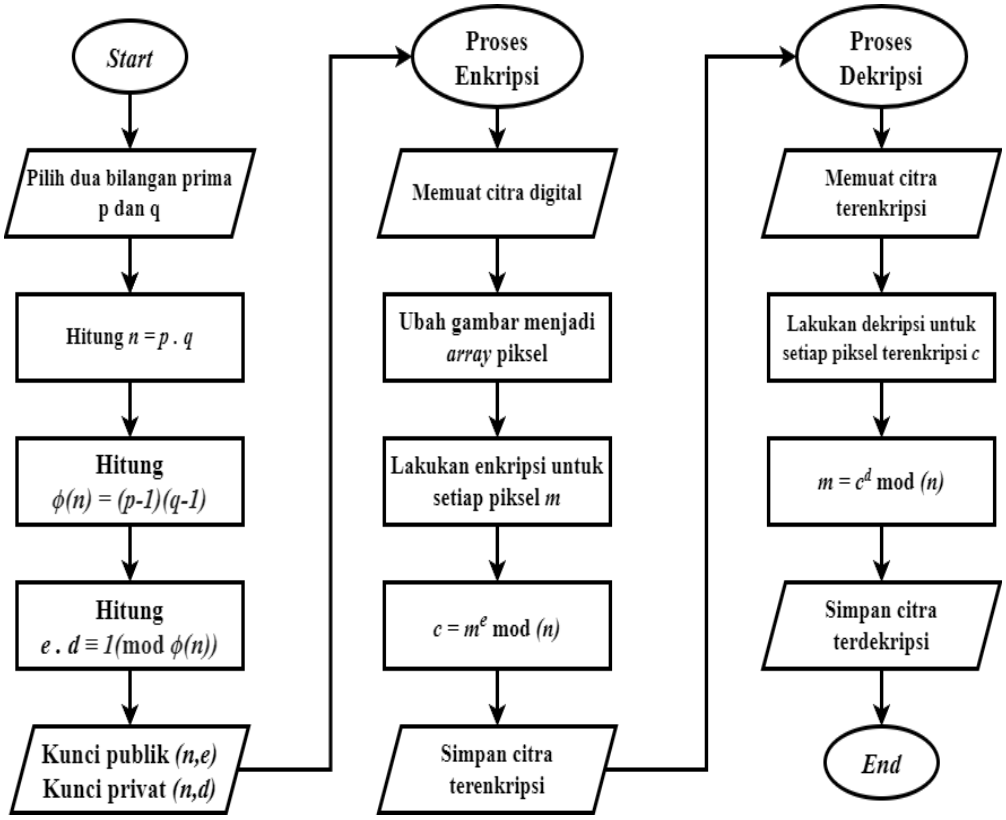
$$m = c^d \bmod(n) \tag{3.5}$$

dimana c adalah piksel terenkripsi, d adalah eksponen dekripsi dari kunci privat, dan m adalah piksel asli yang terdekripsi.

c. Penyimpanan citra terdekripsi:

Setelah semua piksel didekripsi, piksel-piksel tersebut digabungkan kembali untuk membentuk citra asli. Citra terdekripsi ini disimpan kembali dalam format PNG.

Gambar 3.1 akan menggambarkan proses enkripsi dan dekripsi citra digital dengan menggunakan algoritma RSA.



Gambar 3.1 Diagram Alir Enkripsi dan Dekripsi Algoritma RSA.

3.4.2 Proses Enkripsi dan Dekripsi Algoritma AES

Langkah-langkah untuk enkripsi dan dekripsi citra digital menggunakan algoritma AES mengikuti tahapan proses yang telah dijelaskan pada Poin 2.14, yaitu *SubBytes*, *ShiftRows*, *MixColumns*, dan *AddRoundKey*. Tahap enkripsi algoritma AES sebagai berikut:

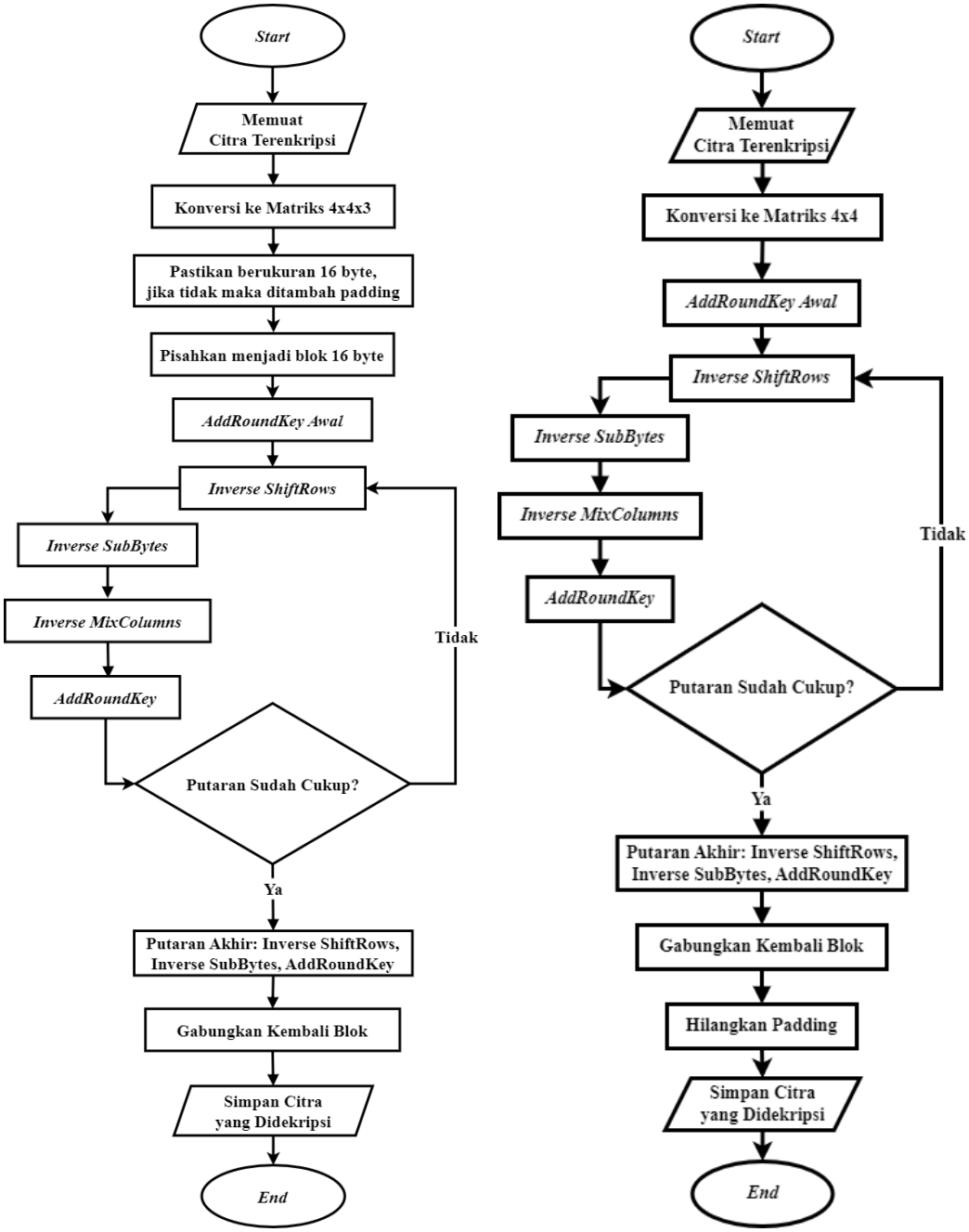
1. Ubah citra digital ke dalam bentuk matriks piksel dalam format RGB, setiap *channel* (R, G, B) akan dienkripsi secara terpisah.
2. Pastikan ukuran citra adalah kelipatan dari 16 *byte*, jika tidak maka tambahkan *padding* agar sesuai.
3. Pisahkah citra menjadi blok 16 *byte* (AES-128 bit) dan susun blok-blok ini ke dalam bentuk matriks 4×4.

4. Menggunakan langkah awal *AddRoundKey*:
    a. Setiap blok 4×4 digabungkan dengan kunci putaran pertama melalui operasi XOR, sebab ini adalah langkah awal dari proses enkripsi.
    b. Jika menggunakan AES-128, maka kunci awal adalah 128-bit (16 *byte*).
5. Menggunakan *SubBytes*:
    a. Setiap *byte* pada matriks diganti dengan *byte* yang sesuai dari *S-box* (lihat Gambar 2.7)
    b. *S-box* adalah tabel substitusi non-linier .
6. Mengaplikasikan *ShiftRows*, di mana baris pertama tetap pada posisinya, baris kedua digeser satu *byte* ke kiri, baris ketiga digeser dua *byte* ke kiri, dan baris keempat digeser tiga *byte* ke kiri.
7. Menggunakan *MixColumns*, yakni setiap kolom dari matriks data digabungkan secara linier menggunakan operasi matriks dengan tujuan untuk menyebarkan efek dari setiap *byte* ke seluruh kolom.
8. Menggunakan *AddRoundKey*, dimana blok data digabungkan kembali dengan kunci putaran berikutnya melalui operasi XOR.
9. Untuk AES-128 diharuskan untuk mengulangi langkah *SubBytes*, *ShiftRows*, *MixColumns*, dan *AddRoundKey* sebanyak sembilan kali lagi
10. Pada putaran terakhir, langkah *MixColumns* tidak dilakukan dan hanya menyisakan langkah *SubBytes*, *ShiftRows*, dan *AddRoundKey*.
11. Menggabungkan kembali setiap blok setelah semua blok dienkripsi agar menjadi citra terenkripsi.
12. Simpan citra yang telah dienkripsi dalam format *.png.

Tahap dekripsi algoritma AES:

1. Memasukkan Citra yang terenkripsi dan konversi ke bentuk blok 4×4.
2. Menggunakan langkah awal *AddRoundKey*, dimana XOR blok dengan kunci terakhir (ini adalah kebalikan dari langkah terakhir enkripsi).
3. Menggunakan *Inverse ShiftRows*, yang menggeser baris kedua satu *byte* ke kanan, baris ketiga dua *byte* ke kanan, dan baris keempat tiga *byte* ke kanan.
4. Menggunakan *Inverse SubBytes*, yakni mengganti setiap *byte* yang sesuai dengan inversi *S-box* (lihat Gambar 2.13).
5. Menggunakan *Inverse MixColumns*, yakni menerapkan operasi matriks *invers* untuk mengembalikan data ke bentuk semula.
6. Menggunakan *AddRoundKey* untuk menggabungkan kembali blok dengan kunci putaran sebelumnya melalui XOR.
7. Mengulangi semua tahap ini sampai putaran pertama tercapai.
8. Pada putaran terakhir, hanya melakukan *InvShiftRows, InvSubBytes,* dan *AddRoundKey*.
9. Gabungkan kembali blok-blok yang telah didekripsi.
10. Menghilangkan *padding* yang ditambahkan selama enkripsi (jika ada penambahan *padding*).
11. Tampilkan kembali citra digital yang telah berhasil didekripsi dalam format *.png.

Untuk memperjelas pemahaman tentang enkripsi dan dekripsi dengan algoritma AES, Gambar 3.2 akan menunjukkan diagram alir yang menggambarkan proses enkripsi serta dekripsi algoritma AES.



Gambar 3.2 *Flowchart* Proses Enkripsi (a) dan Dekripsi (b) Algoritma AES.

3.4.3 Evaluasi Data

1. Dalam penelitian ini, evaluasi data akan dilakukan dengan menggunakan *Mean* Squared *Error* (MSE) untuk mengukur perbedaan antara citra asli dan

citra hasil dekripsi setelah proses enkripsi menggunakan algoritma RSA dan AES. MSE adalah metrik umum yang digunakan untuk menilai kualitas citra, dengan nilai MSE yang lebih rendah menunjukkan bahwa citra hasil dekripsi lebih mendekati citra asli. Langkah-langkah evaluasinya adalah:

a. Lakukan enkripsi pada citra digital menggunakan algoritma RSA dan AES.
b. Setelah proses enkripsi, lakukan dekripsi citra untuk mendapatkan kembali citra dalam bentuk yang dapat dibandingkan dengan citra asli.
c. Ubah citra asli dan citra hasil dekripsi menjadi array numerik yang merepresentasikan setiap piksel dalam citra menggunakan library seperti numpy untuk konversi ini.
d. Hitung perbedaan antara setiap piksel dari citra asli dan citra hasil dekripsi. Hal ini dilakukan dengan mengurangi nilai piksel yang bersesuaian antara kedua citra.
e. MSE dihitung dengan menjumlahkan kuadrat dari perbedaan piksel, kemudian membagi hasilnya dengan jumlah total piksel dalam citra. Formula MSE dapat dilihat pada persamaan (2.14).
f. Buat perbandingan antara nilai MSE dari kedua algoritma untuk menentukan algoritma mana yang menghasilkan kualitas enkripsi yang lebih baik.
g. Bandingkan nilai rata-rata MSE untuk RSA dan AES untuk menilai kinerja algoritma dalam hal keakuratan rekonstruksi citra.

2. Evaluasi statistik adalah menggunakan teknik analisis statistik yang digunakan untuk mengevaluasi dan memahami data hasil pengukuran

(Verma, 2020), dalam penelitian ini data yang terkait adalah kecepatan dan ukuran file hasil enkripsi dan dekripsi menggunakan algoritma RSA dan AES. Tahap analisis statistik untuk kecepatan dan ukuran file hasil enkripsi dan dekripsi:

a. Pengumpulan data

- Ukur waktu yang diperlukan oleh setiap algoritma (RSA dan AES) untuk melakukan enkripsi dan dekripsi pada data citra digital.
- Ukur ukuran file setelah proeses enkripsi dan dekripsi, pastikan juga untuk mencatat ukuran file asli sebagai acuan.

Hasil pengukuran ini akan menjadi data kuantitatif yang bisa dianalisis menggunakan metode statistik.

b. Statistik deskriptif

- Menghitung rata-rata (*mean*) waktu enkripsi dan dekripsi, serta ukuran file untuk kedua algoritma menggunakan rumus

$$Mean = \frac{\sum X}{n} \tag{3.6}$$

- Menghitung nilai tengah (*median*) dari data yang telah diurutkan untuk menghindari nilai ekstrem yang mungkin ada dalam data.
- Menghitung selisih (*range*) antara nilai maksimum dan minimum untuk melihat seberapa jauh variabilitas data menggunakan rumus

$$Range = Nilai\ Max - Nilai\ Min \tag{3.7}$$

BAB IV
HASIL DAN PEMBAHASAN

4.1 Proses Enkripsi dan Dekripsi Citra Digital

4.1.1 Proses Enkripsi Citra Digital dengan Algoritma RSA

Pada proses enkripsi citra digital memakai algoritma RSA (*Rivest-Shamir-Adleman*) memiliki dua prosedur yang akan dijalankan, yakni membangkitkan kunci dan enkripsi. Berikut adalah prosedur membangkitkan kunci untuk algoritma RSA:

1. Memilih dua bilangan prima

   Misalkan $p = 61$ dan $q = 53$

2. Menghitung modulus n

$$n = p \times q = 61 \times 53 = 3233$$

3. Menghitung fungsi euler $\phi(n)$

$$\begin{aligned}\phi(n) &= (p-1) \times (q-1)\\ &= (61-1) \times (53-1)\\ &= 60 \times 52\\ &= 3120\end{aligned}$$

4. Memilih eksponen enkripsi e

   Pilih e yang memenuhi $1 < e < \phi(n)$ dan GCD $\big(e, \phi(n)\big) = 1$

   Misalkan $e = 17$

   Karena GCD $(17,3120) = 1$ dan $e = 17$ maka memenuhi syarat.

   Menggunakan algoritma Euclidian untuk menemukan GCD antara 17 dan 3120 untuk memastikan GCD $(17,3120) = 1$ yang berarti 17 memiliki invers modulo 3120.

$$3120 = 183 \times 17 + 9$$

$$17 = 1 \times 9 + 8$$

$$9 = 1 \times 8 + 1$$

$$8 = 8 \times 1 + 0$$

karena sisanya adalah 0, maka benar bahwa GCD $(17{,}3120) = 1$ sesuai dengan syarat bahwa e dan $\phi(n)$ relatif prima.

5. Menghitung eksponen dekripsi d

   Cari d yang memenuhi

$$e \times d \equiv 1\big(\mathrm{mod}\ \phi(n)\big)$$

$$17 \times d \equiv 1(\mathrm{mod}(3120))$$

   Menggunakan algoritma Euclidian yang diperpanjang untuk menemukan invers modular, akan dikerjakan secara mundur untuk membuktikan bahwa 1 sebagai kombinasi linier dari 17 dan 3120.

   Mulai dari persamaan $1 = 9 - 1 \times 8$, substitusikan nilai 8 dari proses sebelumnya:

$$1 = 9 - 1 \times (17 - 1 \times 9)$$

$$1 = 2 \times 9 - 1 \times 17$$

   Substitusi 9 dari baris sebelumnya,

$$1 = 2 \times (3120 - 183 \times 17) - 1 \times 17$$

$$1 = 2 \times 3120 - 367 \times 17$$

   Dari sini, dapat dilihat bahwa invers modular dari 17 modulo 3120 adalah $d = -367$. Namun, karena bekerja dalam modulo 3120, harus mengonversi -367 ke nilai positif dengan menambahkan 3120.

$$d = -367 + 3120$$

$$= 2753$$

Karena $d = 2753$ maka

$$17 \times 2753 \equiv 1(\mathrm{mod}(3120))$$

Diperoleh hasil kunci RSA adalah

Kunci publik $(n, e) = (3233,17)$

Kunci privat $(n, d) = (3233,2753)$

Kemudian memasuki prosedur enkripsi citra digital untuk algoritma RSA menggunakan Persamaan (3.4):

1. Representasi citra

   Misalkan akan mengenkripsi nilai piksel citra, contohnya nilai piksel $m = 65$ (nilai antara 0 dan 255).

2. Enkripsi piksel

   Untuk enkripsi piksel m menggunakan kunci publik $(n = 3233, e = 17)$ dengan formula:

$$c = m^e \; mod \; (n)$$

$$= 65^{17} \bmod 3233$$

perhitungan ini cukup besar, jadi akan dihitung langkah demi langkah dengan algoritma eksponensial modular.

$$65^2 \bmod 3233 = 4225 \bmod 3233 = 992$$

$$65^4 \bmod 3233 = (65^2)^2 \bmod 3233$$

$$= 992^2 \bmod 3233$$

$$= 984064 \bmod 3233$$

$$= 1232$$

$$65^8 \bmod 3233 = (65^4)^2 \bmod 3233$$

$$= 1232^2 \bmod 3233$$

$$= 1517824 \bmod 3233$$

$$= 1547$$

$$65^{16} \bmod 3233 = (65^8)^2 \bmod 3233$$

$$= 3063^2 \bmod 3233$$

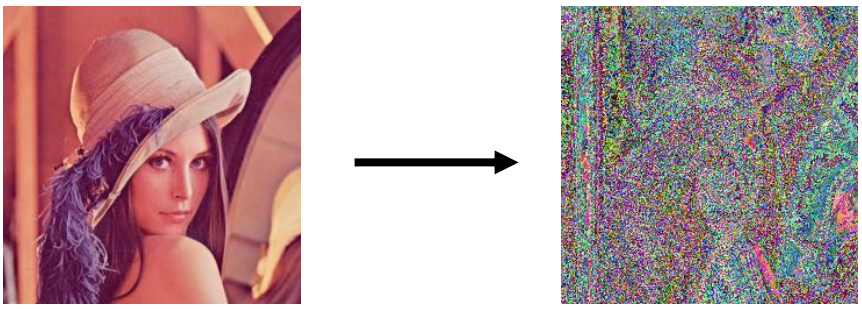
$$= 9381969 \bmod 3233$$

$$= 789$$

$$65^{17} \bmod 3233 = 2790$$

maka nilai piksel terenkripsi $c = 2790$.

Dengan demikian c dapat dinyatakan sebagai hasil enkripsi menggunakan algoritma RSA. Setelah semua piksel terenkripsi, maka piksel-piksel tersebut digabungkan kembali menjadi citra terenkripsi yang baru dan disimpan dalam format *.png.



Gambar 4.1 Hasil Enkripsi Citra Digital Menggunakan Algoritma RSA.

4.1.2 Proses Dekripsi Citra Digital dengan Algoritma RSA

Berikut ini adalah penerapan dekripsi pada citra digital memakai algoritma RSA dengan tipe data RGB.

1. Representasi citra terenkripsi

   Misalnya menyiapkan citra terenkripsi yang memiliki piksel $c = 2790$ (hasil enkripsi).

2. Dekripsi piksel

   Untuk mendekripsikan nilai pikel $c = 2790$, akan digunakan kunci privat $(n = 3233, d = 2753)$ menggunakan formula:

$$m = c^d \bmod (n) = 2790^{2753} \bmod (3233)$$

   Perhitungan ini cukup besar, tetapi bisa menggunakan algoritma eksponensial modular yang sama:

$$2790^2 \bmod 3233 = 7784100 \bmod 3233 = 2269$$

$$2790^4 \bmod 3233 = (2790^2)^2 \bmod 3233$$

$$= 1425$$

$$2790^8 \bmod 3233 = (2790^4)^2 \bmod 3233$$

$$= 301$$

$$2790^{16} \bmod 3233 = (2790^8)^2 \bmod 3233$$

$$= 77$$

$$2790^{32} \bmod 3233 = (2790^{16})^2 \bmod 3233$$

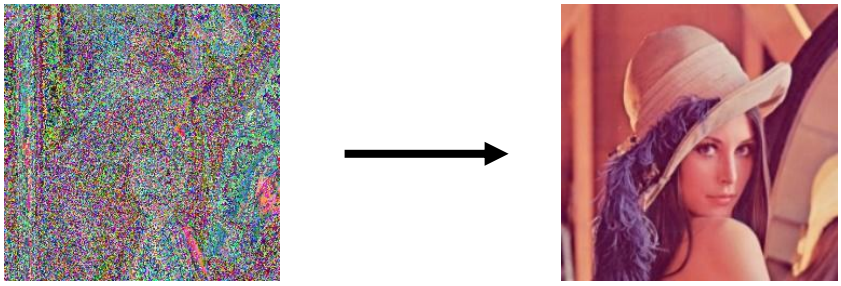
$$\vdots \qquad = 2696$$

$$2790^{2753} \bmod 3233 = 65$$

   Setelah prosedur dekripsi menggunakan kunci privat $(n = 3233, d = 2753)$, nilai $c = 2790$ berhasil dikembalikan menjadi nilai piksel asli $m = 65$.

3. Penyimpanan citra

   Setelah semua piksel didekripsi, piksel-piksel tersebut dapat digabungkan kembali untuk membentuk citra digital yang asli. Citra hasil dekripsi ini disimpan kembali dalam format *.png.



Gambar 4.2 Hasil Dekripsi Citra Digital Menggunakan Algoritma RSA.

4.1.3 Proses Enkripsi Citra Digital dengan Algoritma AES

1. Mengubah citra digital ke dalam bentuk matriks piksel dalam format RGB

   Citra digital terdiri dari piksel yang masing-masing terdiri dari tiga komponen warna: merah (R), hijau (G), dan biru (B). Setiap komponen diwakili oleh nilai heksadesimal. Misalkan sebuah matriks RGB berukuran $4 \times 4 \times 3$ sebagai berikut:

$$\begin{bmatrix} [12,34,56] & [78,90,50] & [45,67,89] & [34,89,23] \\ [56,45,67] & [89,E3,F4] & [7C,CD,CC] & [A3,F6,43] \\ [C8,69,B7] & [47,8A,C3] & [34,46,D1] & [EA,87,14] \\ [81,1C,C6] & [73,DB,B8] & [B5,48,2E] & [6B,0E,90] \end{bmatrix}$$

2. Pastikan ukuran citra adalah kelipatan dari 16 *byte*, jika tidak maka tambahkan *padding* agar sesuai.

   Jika ukuran citra tidak kelipatan 16 byte, maka AES-128 tidak dapat mengenkripsi blok tersebut karena algoritma ini dirancang untuk memproses blok penuh. Penambahan byte *padding* dilakukan untuk melengkapi blok

terakhir agar ukurannya menjadi kelipatan 16 byte. Dalam citra RGB, nilai *padding* diwakili oleh byte heksadesimal yang menunjukkan jumlah byte yang ditambahkan untuk memenuhi panjang blok yang diperlukan. Sebagai contoh, jika 7 byte *padding* dibutuhkan maka setiap byte *padding* akan memiliki nilai 07, yang berarti jumlah byte *padding* yang ditambahkan adalah 7 byte, sesuai dengan panjang yang kurang pada blok terakhir.

3. Pisahkan citra menjadi blok 16 *byte* karena menggunakan AES-128

   Dibagi menjadi blok-blok 16 *byte*.

   Blok 1: [12,34,56,78,90,50,45,67,89,34,89,23,56,45,67,89]

   Blok 2: [$E3, F4,7C, CD, CC, A3, F6,43, CB, 69, B7,47,8A, C3$,34,46]

   Blok 3: [$D1, EA,$ 87,14,81,$1C, C6,73, DB, B8, B5,48,2E, 6B, 0E,$ 90]

   Setiap blok akan menjadi matriks 4×4 yang akan diproses. Misalkan diambil blok pertama dari matriks RGB:

$$Blok\ 1 = \begin{bmatrix} 12 & 34 & 56 & 78 \\ 90 & 50 & 45 & 67 \\ 89 & 34 & 89 & 60 \\ 56 & 45 & 67 & 89 \end{bmatrix}$$

   Dimensi blok adalah 4×4.

4. Langkah awal *AddRoundKey*

   Setiap blok 4×4 digabungkan dengan kunci putaran menggunakan operasi XOR. Misalkan kunci putaran adalah:

$$Kunci = \begin{bmatrix} 4F & 5A & 7B & 10 \\ 8C & CD & D1 & 23 \\ 67 & 2A & FF & 45 \\ 28 & 0D & 93 & 2C \end{bmatrix}$$

   Dengan melakukan operasi XOR antara setiap elemen matriks blok dan matriks kunci, maka hasilnya sebagai berikut:

$$Hasil\ XOR = \begin{bmatrix} 12 \oplus 4F & 34 \oplus 5A & 56 \oplus 7B & 78 \oplus 10 \\ 90 \oplus 8C & 50 \oplus CD & 45 \oplus D1 & 67 \oplus 23 \\ 89 \oplus 67 & 34 \oplus 2A & 89 \oplus FF & 60 \oplus 45 \\ 56 \oplus 28 & 45 \oplus 0D & 67 \oplus 93 & 89 \oplus 2C \end{bmatrix}$$

Menghitung nilai XOR satu persatu:

Baris 1:

$$12 \oplus 4F = 00010010 \oplus 01001111 = 01011101 = 5D$$

$$34 \oplus 5A = 00110100 \oplus 01011010 = 01101110 = 6E$$

$$56 \oplus 7B = 01010110 \oplus 01111011 = 00101101 = 2D$$

$$78 \oplus 10 = 01111000 \oplus 00010000 = 01101000 = 68$$

Baris 2:

$$90 \oplus 8C = 10010000 \oplus 10001100 = 00011100 = 1C$$

$$50 \oplus CD = 01010000 \oplus 11001101 = 10011101 = 9D$$

$$45 \oplus D1 = 01000101 \oplus 11010001 = 10010100 = 94$$

$$67 \oplus 23 = 01100111 \oplus 00100011 = 01000100 = 44$$

Baris 3:

$$89 \oplus 67 = 10001001 \oplus 01100111 = 11101110 = EE$$

$$34 \oplus 2A = 00110100 \oplus 00101010 = 00011110 = 1E$$

$$89 \oplus FF = 10001001 \oplus 11111111 = 01110110 = 76$$

$$60 \oplus 45 = 01100000 \oplus 01000101 = 00100101 = 25$$

Baris 4:

$$56 \oplus 28 = 01010110 \oplus 00101000 = 01111110 = 7E$$

$$45 \oplus 0D = 01000101 \oplus 00001101 = 01001000 = 48$$

$$67 \oplus 93 = 01100111 \oplus 10010011 = 11110100 = F4$$

$$89 \oplus 2C = 10001001 \oplus 00101100 = 10100101 = A5$$

Setelah melakukan operasi XOR, didapatkan matriks hasil XOR sebagai berikut:

$$\begin{bmatrix} 5D & 6E & 2D & 68 \\ 1C & 9D & 94 & 44 \\ EE & 1E & 76 & 25 \\ 7E & 48 & F4 & A5 \end{bmatrix}$$

5. Menggunakan *SubBytes*

Untuk setiap byte pada matriks, misalkan $S[r,c] = xy$, dalam hal ini xy adalah digit hexadesimal dari nilai $S[r,c]$, maka nilai substitusinya adalah $S'[r,c]$ yang merupakan potongan baris x dengan kolom y di dalam *S-box* (Gambar 2.7). Menggunakan matriks hasil *AddRoundKey* awal:

$$\begin{bmatrix} 5D & 6E & 2D & 68 \\ 1C & 9D & 94 & 44 \\ EE & 1E & 76 & 25 \\ 7E & 48 & F4 & A5 \end{bmatrix}$$

elemen pertama, 5D, disubstitusikan dengan elemen pada perpotongan baris 5 dan kolom D pada *S-box*, yaitu 4C. Jadi, operasi substitusi pada $S[r,c] = 5D$ menghasilkan $S'[r,c] = 4C$. Jika operasi ini diterapkan untuk semua matriks lainnya, maka hasilnya adalah:

$$\begin{bmatrix} 5D & 6E & 2D & 68 \\ 1C & 9D & 94 & 44 \\ EE & 1E & 76 & 25 \\ 7E & 48 & F4 & A5 \end{bmatrix} \Rightarrow \begin{bmatrix} 4C & 9F & D8 & 45 \\ 9C & 5E & 22 & 1B \\ 28 & 72 & 38 & 3F \\ F3 & 52 & BF & 06 \end{bmatrix}$$

6. Mengaplikasikan *ShiftRows*

Menggunakan matriks hasil *SubBytes*,

Baris 0: $[4C, 9F, D8,45]$, tidak digeser ke kiri.

Baris 1: $[9C, 5E, 22,1B]$, digeser satu byte ke kiri menjadi $[5E, 22,1B, 9C]$.

Baris 2: $[28,72,38,3F]$, geser dua byte ke kiri menjadi $[38,33,28,72]$.

Baris 3: $[F3,52, BF, 06]$, geser tiga byte ke kiri menjadi $[06, F3,52, BF]$.

Sehingga hasilnya adalah:

$$\begin{bmatrix} 4C & 9F & D8 & 45 \\ 9C & 5E & 22 & 1B \\ 28 & 72 & 38 & 33 \\ F3 & 52 & BF & 06 \end{bmatrix} \Rightarrow \begin{bmatrix} 4C & 9F & D8 & 45 \\ 5E & 22 & 1B & 9C \\ 38 & 3F & 28 & 72 \\ 06 & F3 & 52 & BF \end{bmatrix}$$

7. Menggunakan *MixColumns*

$$\begin{bmatrix} 4C & 9F & D8 & 45 \\ 5E & 22 & 1B & 9C \\ 38 & 3F & 28 & 72 \\ 06 & F3 & 52 & BF \end{bmatrix}$$

Dengan memakai matriks hasil *ShiftRows* di atas, kemudian dilakukan *MixColumns* menggunakan dua tabel, yaitu Tabel E dan Tabel L (Gambar 2.11) maka menghasilkan:

a. Operasi *MixColumns* terhadap kolom pertama

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} 4C \\ 5E \\ 38 \\ 06 \end{bmatrix} = \begin{bmatrix} 65 \\ 5F \\ 8E \\ BD \end{bmatrix}$$

$(02 \cdot 4C) \oplus (03 \cdot 5E) \oplus (01 \cdot 38) \oplus (01 \cdot 06)$

$= E\big(L(02) + L(4C)\big) \oplus E\big(L(03) + L(5E)\big) \oplus E(L(01)$

$+ L(38)) \oplus E(L(01) + L(06))$

$= E(59) \oplus E(90) \oplus E(12) \oplus E(1A)$

$= 98 \oplus C3 \oplus 38 \oplus 06$

$= 10011000 \oplus 11000011 \oplus 00111000 \oplus 00000110$

$= 01100101$

$= 65$

$(01 \cdot 4C) \oplus (02 \cdot 5E) \oplus (03 \cdot 38) \oplus (01 \cdot 06)$

$= E\big(L(01) + L(4C)\big) \oplus E\big(L(02) + L(5E)\big) \oplus E(L(03)$

$$+L(38))\oplus E(L(01)+L(06))$$

$$= E(40)\oplus E(88)\oplus E(13)\oplus E(1A)$$

$$= 4C\oplus 5D\oplus 48\oplus 06$$

$$= 01001100\oplus 01011101\oplus 01001000\oplus 00000110$$

$$= 01011111$$

$$= 5F$$

$$(01\cdot 4C)\oplus(01\cdot 5E)\oplus(02\cdot 38)\oplus(03\cdot 06)$$

$$= E\big(L(01)+L(4C)\big)\oplus E\big(L(01)+L(5E)\big)\oplus E(L(02)$$

$$+L(38))\oplus E(L(03)+L(06))$$

$$= E(40)\oplus E(91)\oplus E(0B)\oplus E(1B)$$

$$= 4C\oplus 5E\oplus 96\oplus 0A$$

$$= 01001100\oplus 01011110\oplus 00001011\oplus 00011011$$

$$= 10001110$$

$$= 8E$$

$$(03\cdot 4C)\oplus(01\cdot 5E)\oplus(01\cdot 38)\oplus(02\cdot 06)$$

$$= E\big(L(03)+L(4C)\big)\oplus E\big(L(01)+L(5E)\big)\oplus E(L(01)$$

$$+L(38))\oplus E(L(02)+L(06))$$

$$= E(41)\oplus E(91)\oplus E(12)\oplus E(03)$$

$$= D4\oplus 5E\oplus 38\oplus 0F$$

$$= 11010100\oplus 01011110\oplus 00111000\oplus 00001111$$

$$= 10111101$$

$$= BD$$

b. Operasi *MixColumns* terhadap kolom kedua

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} 9F \\ 22 \\ 3F \\ F3 \end{bmatrix} = \begin{bmatrix} F7 \\ 3C \\ 73 \\ AE \end{bmatrix}$$

$(02 \cdot 9F) \oplus (03 \cdot 22) \oplus (01 \cdot 3F) \oplus (01 \cdot F3)$

$= E\big(L(02) + L(9F)\big) \oplus E\big(L(03) + L(22)\big) \oplus E(L(01)$

$+ L(33)) \oplus E(L(01) + L(F3))$

$= E(B9) \oplus E(1C) \oplus E(8E) \oplus E(DE)$

$= 25 \oplus 1E \oplus 3F \oplus F3$

$= 00100101 \oplus 00011110 \oplus 00111111 \oplus 11110011$

$= 11110111$

$= F7$

$(01 \cdot 9F) \oplus (02 \cdot 22) \oplus (03 \cdot 3F) \oplus (01 \cdot F3)$

$= E\big(L(01) + L(9F)\big) \oplus E\big(L(02) + L(22)\big) \oplus E(L(03)$

$+ L(3F)) \oplus E(L(01) + L(F3))$

$= E(A0) \oplus E(04) \oplus E(8F) \oplus E(DE)$

$= 9F \oplus 11 \oplus 41 \oplus F3$

$= 10011111 \oplus 00010001 \oplus 01000001 \oplus 11110011$

$= 00111100$

$= 3C$

$(01 \cdot 9F) \oplus (01 \cdot 22) \oplus (02 \cdot 3F) \oplus (03 \cdot F3)$

$= E\big(L(01) + L(9F)\big) \oplus E\big(L(01) + L(22)\big) \oplus E(L(02)$

$$+L(3F))\oplus E(L(03)+L(F3))$$

$$= E(A0)\oplus E(1D)\oplus E(97)\oplus E(DF)$$

$$= 9F\oplus 22\oplus C0\oplus 0E$$

$$= 10011111\oplus 00100010\oplus 10010111\oplus 00001110$$

$$= 01110011$$

$$= 73$$

$$(03\cdot 9F)\oplus(01\cdot 22)\oplus(01\cdot 3F)\oplus(02\cdot F3)$$

$$= E\big(L(03)+L(9F)\big)\oplus E\big(L(01)+L(22)\big)\oplus E(L(01)$$

$$+L(3F))\oplus E(L(02)+L(F3))$$

$$= E(A1)\oplus E(1D)\oplus E(8E)\oplus E(C7)$$

$$= BA\oplus 22\oplus 3F\oplus 09$$

$$= 10111010\oplus 00100010\oplus 00111111\oplus 00001001$$

$$= 10101110$$

$$= AE$$

c. Operasi *MixColumns* terhadap kolom ketiga

$$\begin{bmatrix}02 & 03 & 01 & 01\\ 01 & 02 & 03 & 01\\ 01 & 01 & 02 & 03\\ 03 & 01 & 01 & 02\end{bmatrix}\begin{bmatrix}D8\\ 1B\\ 28\\ 52\end{bmatrix}=\begin{bmatrix}AF\\ 5D\\ 6F\\ 69\end{bmatrix}$$

$$(02\cdot D8)\oplus(03\cdot 1B)\oplus(01\cdot 28)\oplus(01\cdot 52)$$

$$= E\big(L(02)+L(D8)\big)\oplus E\big(L(03)+L(1B)\big)\oplus E(L(01)$$

$$+L(28))\oplus E(L(01)+L(52))$$

$$= E(0D)\oplus E(C9)\oplus E(4D)\oplus E(FD)$$

$$= F8\oplus 2D\oplus 28\oplus 52$$

$= 11111000 \oplus 00101101 \oplus 00101000 \oplus 01010010$

$= 10101111$

$= AF$

$(01 \cdot D8) \oplus (02 \cdot 1B) \oplus (03 \cdot 28) \oplus (01 \cdot 52)$

$= E\big(L(01) + L(D8)\big) \oplus E\big(L(02) + L(1B)\big) \oplus E(L(03)$

$+ L(28)) \oplus E(L(01) + L(52))$

$= E(14) \oplus E(D1) \oplus E(4C) \oplus E(FD)$

$= D8 \oplus CF \oplus 18 \oplus 52$

$= 11011000 \oplus 11001111 \oplus 00011000 \oplus 01010010$

$= 01011101$

$= 5D$

$(01 \cdot D8) \oplus (01 \cdot 1B) \oplus (02 \cdot 28) \oplus (03 \cdot 52)$

$= E\big(L(01) + L(D8)\big) \oplus E\big(L(01) + L(1B)\big) \oplus E(L(02)$

$+ L(28)) \oplus E(L(03) + L(52))$

$= E(14) \oplus E(C8) \oplus E(54) \oplus E(FC)$

$= D8 \oplus 1B \oplus 6B \oplus C7$

$= 11011000 \oplus 00011011 \oplus 01101011 \oplus 11000111$

$= 01101111$

$= 6F$

$(03 \cdot D8) \oplus (01 \cdot 1B) \oplus (01 \cdot 28) \oplus (02 \cdot 52)$

$= E\big(L(03) + L(D8)\big) \oplus E\big(L(01) + L(1B)\big) \oplus E(L(01)$

$$+L(28))\oplus E(L(02)+L(52))$$

$$= E(15)\oplus E(CB)\oplus E(4D)\oplus E(E4)$$

$$= 73\oplus 1B\oplus 28\oplus 29$$

$$= 01110011\oplus 00011011\oplus 00101000\oplus 00101001$$

$$= 01101001$$

$$= 69$$

d. Operasi *MixColumns* terhadap kolom keempat

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} 45 \\ 9C \\ 72 \\ BF \end{bmatrix} = \begin{bmatrix} 5F \\ 01 \\ 0D \\ A5 \end{bmatrix}$$

$$(02 \cdot 45)\oplus(03 \cdot 9C)\oplus(01 \cdot 72)\oplus(01 \cdot BF)$$

$$= E\big(L(02)+L(45)\big)\oplus E\big(L(03)+L(9C)\big)\oplus E(L(01)$$

$$+L(72))\oplus E(L(01)+L(BF))$$

$$= E(C9)\oplus E(9D)\oplus E(0A)\oplus E(9D)$$

$$= 2D\oplus BF\oplus 72\oplus BF$$

$$= 00101101\oplus 10111111\oplus 01110010\oplus 10111111$$

$$= 01011111$$

$$= 5F$$

$$(01 \cdot 45)\oplus(02 \cdot 9C)\oplus(03 \cdot 72)\oplus(01 \cdot BF)$$

$$= E\big(L(01)+L(45)\big)\oplus E\big(L(02)+L(9C)\big)\oplus E(L(03)$$

$$+L(72))\oplus E(L(01)+L(BF))$$

$$= E(D0)\oplus E(85)\oplus E(0B)\oplus E(9D)$$

$= 45 \oplus 6D \oplus 96 \oplus BF$

$= 01000101 \oplus 01101101 \oplus 10010110 \oplus 10111111$

$= 00000001$

$= 01$

$(01 \cdot 45) \oplus (01 \cdot 9C) \oplus (02 \cdot 72) \oplus (03 \cdot BF)$

$= E\big(L(01) + L(45)\big) \oplus E\big(L(01) + L(9C)\big) \oplus E(L(02)$

$+ L(72)) \oplus E(L(03) + L(BF))$

$= E(D0) \oplus E(9C) \oplus E(13) \oplus E(9C)$

$= 45 \oplus 9C \oplus 48 \oplus 9C$

$= 01000101 \oplus 10011100 \oplus 01001000 \oplus 10011100$

$= 00001101$

$= 0D$

$(03 \cdot 45) \oplus (01 \cdot 9C) \oplus (01 \cdot 72) \oplus (02 \cdot BF)$

$= E\big(L(01) + L(45)\big) \oplus E\big(L(01) + L(9C)\big) \oplus E(L(02)$

$+ L(72)) \oplus E(L(03) + L(BF))$

$= E(D1) \oplus E(9C) \oplus E(0A) \oplus E(84)$

$= CF \oplus 9C \oplus 72 \oplus 84$

$= 11001111 \oplus 10011100 \oplus 01110010 \oplus 10000100$

$= 10100101$

$= A5$

Hasil transformasi *MixColumns* secara keseluruhan adalah:

$$\begin{bmatrix} 4C & 9F & D8 & 45 \\ 5E & 22 & 1B & 9C \\ 38 & 3F & 28 & 72 \\ 06 & F3 & 52 & BF \end{bmatrix} \Rightarrow \begin{bmatrix} 65 & F7 & AF & 5F \\ 5F & 3C & 5D & 01 \\ 8E & 73 & 6F & 0D \\ BD & AE & 69 & A5 \end{bmatrix}$$

8. Menggunakan *AddRoundKey*

   Menggunakan sebuah *round key* adalah:

$$\begin{bmatrix} 4F & 5A & 7B & 10 \\ 8C & CD & D1 & 23 \\ 67 & 2A & FF & 45 \\ 28 & 0D & 93 & 2C \end{bmatrix}$$

   dan hasil *MixColumns* sebelumnya adalah:

$$\begin{bmatrix} 65 & F7 & AF & 5F \\ 5F & 3C & 5D & 01 \\ 8E & 73 & 6F & 0D \\ BD & AE & 69 & A5 \end{bmatrix}$$

   Maka *AddRoundKey* menghasilkan:

$$\begin{bmatrix} 65 & F7 & AF & 5F \\ 5F & 3C & 5D & 01 \\ 8E & 73 & 6F & 0D \\ BD & AE & 69 & A5 \end{bmatrix} \oplus \begin{bmatrix} 4F & 5A & 7B & 10 \\ 8C & CD & D1 & 23 \\ 67 & 2A & FF & 45 \\ 28 & 0D & 93 & 2C \end{bmatrix} = \begin{bmatrix} 2A & AD & D4 & 4F \\ D3 & F1 & 8C & 22 \\ E9 & 59 & 90 & 48 \\ 95 & A3 & FA & 89 \end{bmatrix}$$

9. Mengulangi proses *SubBytes, ShiftRows, MixColumns,* dan *AddRoundKey* sebanyak 9 kali untuk AES-128.

   Putaran pertama: $\begin{bmatrix} 2A & AD & D4 & 4F \\ D3 & F1 & 8C & 22 \\ E9 & 59 & 90 & 48 \\ 95 & A3 & FA & 89 \end{bmatrix}$

   Putaran kedua: $\begin{bmatrix} C3 & 26 & C6 & E7 \\ 06 & A1 & 6D & 2C \\ FF & 12 & E0 & 2B \\ 39 & 28 & 5A & 06 \end{bmatrix}$

   Putaran ketiga: $\begin{bmatrix} 53 & 91 & BE & EB \\ 15 & 03 & B6 & 85 \\ A3 & C9 & DD & 32 \\ 5B & D4 & 62 & 69 \end{bmatrix}$

Putaran keempat: $\begin{bmatrix} C1 & 95 & 42 & A0 \\ 2E & A0 & 4C & A7 \\ 5F & 30 & BD & 32 \\ 5F & FB & E2 & 36 \end{bmatrix}$

Putaran kelima: $\begin{bmatrix} D7 & 59 & B0 & C1 \\ 16 & C7 & 42 & 16 \\ 18 & C6 & 3F & D9 \\ B5 & C1 & EC & 65 \end{bmatrix}$

Putaran keenam: $\begin{bmatrix} 87 & BD & 17 & 2F \\ 3F & D4 & 3B & D0 \\ 5F & CA & 2A & 3F \\ AD & 3F & 87 & DD \end{bmatrix}$

Putaran ketujuh: $\begin{bmatrix} 10 & C2 & FB & 70 \\ 21 & 5A & 6E & 22 \\ 43 & CA & 45 & 26 \\ B6 & 02 & 66 & 5B \end{bmatrix}$

Putaran kedelapan: $\begin{bmatrix} 3B & E0 & E7 & BC \\ 5C & A5 & 58 & AA \\ 30 & 93 & 86 & 02 \\ 65 & F9 & 6C & B3 \end{bmatrix}$

Putaran kesembilan: $\begin{bmatrix} AB & D1 & 43 & 3F \\ CE & D0 & 79 & B5 \\ 4E & 10 & 62 & C9 \\ A1 & CB & 32 & 53 \end{bmatrix}$

10. Pada putaran terakhir tidak melakukan MixColumns, hanya melakukan *SubBytes, ShiftRows, dan AddRoundKey*.

    Putaran terakhir menghasilkan: $\begin{bmatrix} 84 & 68 & 53 & F2 \\ CE & 83 & 13 & 60 \\ 3D & 58 & E6 & 9F \\ FD & 40 & D3 & 5B \end{bmatrix}$

    Sehingga putaran terakhir ini dapat disebut sebagai blok terenkripsi.

    Blok 1 $= [84,68,53, F2, CE, 83,13,60, 3D, 58, E6, 9F, FD, 40, D3, 5B]$

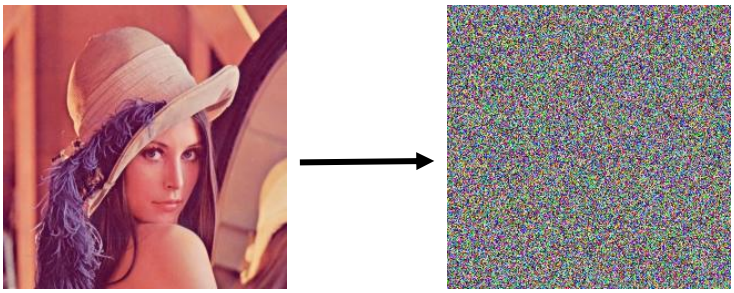
    Blok 2 $= [24, 8E, 96, 27, 6F, D0, 56, 9F, 90, 65, FD, 26, 3D, CE, BF, 38]$

    Blok 3 $= [B9, 97, E1, FB, 6D, 03, 01, 67, C3, 77, 0F, B4, C7, 32, DE, FE]$

11. Setelah semua blok terenkripsi, kemudian menggabungkan kembali semua matriks-matriks menjadi citra utuh

$$\begin{bmatrix} [84,68,53] & [F2,CE,83] & [13,60,3D] & [58,E6,9F] \\ [FD,40,D3] & [5B,24,8E] & [96,27,6F] & [D0,56,9F] \\ [90,65,FD] & [26,3D,CE] & [BF,38,B9] & [97,E1,FB] \\ [6D,03,01] & [67,C3,77] & [0F,B4,C7] & [32,DE,FE] \end{bmatrix}$$

Kemudian disimpan dalam format *.png.



Gambar 4.3 Hasil Enkripsi Citra Digital Menggunakan Algoritma AES.

4.1.4 Proses Dekripsi Citra Digital dengan Algoritma AES

1. Memasukkan citra yang terenkripsi dan ubah ke bentuk blok 4×4

   Citra yang terenkripsi akan diubah ke dalam blok berbentuk mariks 4×4. Citra terenkripsi disebut sebagai C. Setelah dikonversi, maka blok pertama dapat diwakili oleh:

$$C = \begin{bmatrix} 84 & 68 & 53 & F2 \\ CE & 83 & 13 & 60 \\ 3D & 58 & E6 & 9F \\ FD & 40 & D3 & 5B \end{bmatrix}$$

2. Menggunakan langkah awal *AddRoundKey*

   Lakukan *AddRoundKey* dengan melakukan operasi XOR antara blok C dengan kunci terakhir, kunci terakhir adalah:

$$Kunci = \begin{bmatrix} 4F & 5A & 7B & 10 \\ 8C & CD & D1 & 23 \\ 67 & 2A & FF & 45 \\ 28 & 0D & 93 & 2C \end{bmatrix}$$

Dengan melakukan operasi XOR antara setiap elemen matriks blok dan matriks kunci, maka hasilnya sebagai berikut:

$$XOR = \begin{bmatrix} 84 \oplus 4F & 68 \oplus 5A & 53 \oplus 7B & F2 \oplus 10 \\ CE \oplus 8C & 83 \oplus CD & 13 \oplus D1 & 60 \oplus 23 \\ 3D \oplus 67 & 58 \oplus 2A & E6 \oplus FF & 9F \oplus 45 \\ FD \oplus 28 & 40 \oplus 0D & D3 \oplus 93 & 5B \oplus 2C \end{bmatrix}$$

Menghitung nilai XOR satu persatu:

Baris 1:

$$84 \oplus 4F = 10000100 \oplus 01001111 = 11001011 = CB$$

$$68 \oplus 5A = 01101000 \oplus 01011010 = 00110010 = 32$$

$$53 \oplus 7B = 01010011 \oplus 01111011 = 00101000 = 28$$

$$F2 \oplus 10 = 11110010 \oplus 00010000 = 11100010 = E2$$

Baris 2:

$$CE \oplus 8C = 11001110 \oplus 10001100 = 01000010 = 42$$

$$83 \oplus CD = 10000011 \oplus 11001101 = 01001110 = 4E$$

$$13 \oplus D1 = 00010011 \oplus 11010001 = 11000010 = C2$$

$$60 \oplus 23 = 01100000 \oplus 00100011 = 01000011 = 43$$

Baris 3:

$$3D \oplus 67 = 00111101 \oplus 01100111 = 01011010 = 5A$$

$$58 \oplus 2A = 01011000 \oplus 00101010 = 01110010 = 72$$

$$E6 \oplus FF = 11100110 \oplus 11111111 = 00011001 = 19$$

$$9F \oplus 45 = 10011111 \oplus 01000101 = 11011010 = DA$$

Baris 4:

$$FD \oplus 28 = 11111101 \oplus 00101000 = 11010101 = D5$$

$$40 \oplus 0D = 01000000 \oplus 00001101 = 01001101 = 4D$$

$$D3 \oplus 93 = 11010011 \oplus 10010011 = 01000000 = 40$$

$$5B \oplus 2C = 01011011 \oplus 00101100 = 01110111 = 77$$

Setelah melakukan operasi XOR, didapatkan matriks hasil XOR sebagai berikut:

$$Hasil\ XOR = \begin{bmatrix} CB & 32 & 28 & E2 \\ 42 & 4E & C2 & 43 \\ 5A & 72 & 19 & DA \\ D5 & 4D & 40 & 77 \end{bmatrix}$$

3. Menggunakan *InvShiftRows*

Setelah langkah *AddRoundKey*, lakukan inversi *ShiftRows*.

$$\begin{bmatrix} CB & 32 & 28 & E2 \\ 42 & 4E & C2 & 43 \\ 5A & 72 & 19 & DA \\ D5 & 4D & 40 & 77 \end{bmatrix}$$

Baris 0: $[CB, 32, 28, E2]$, tidak digeser.

Baris 1: $[42, 4E, C2, 43]$, digeser satu byte ke kanan $[43, 42, 4E, C2]$.

Baris 2: $[5A, 72, 19, DA]$, digeser dua byte ke kanan $[19, DA, 5A, 72]$.

Baris 3: $[D5, 4D, 40, 77]$, digeser tiga byte ke kanan $[4D, 40, 77. D5]$.

Hasil setelah langkah inversi *ShiftRows*:

$$\begin{bmatrix} CB & 32 & 28 & E2 \\ 42 & 4E & C2 & 43 \\ 5A & 72 & 19 & DA \\ D5 & 4D & 40 & 77 \end{bmatrix} \Rightarrow \begin{bmatrix} CB & 32 & 28 & E2 \\ 43 & 42 & 4E & C2 \\ 19 & DA & 5A & 72 \\ 4D & 40 & 77 & D5 \end{bmatrix}$$

4. Menggunakan *InvSubBytes*

Untuk mengganti setiap *byte* dengan tabel inversi *S-box* (Gambar 2.13), maka menghasilkan:

$$\begin{bmatrix} CB & 32 & 28 & E2 \\ 43 & 42 & 4E & C2 \\ 19 & DA & 5A & 72 \\ 4D & 40 & 77 & D5 \end{bmatrix} \Rightarrow \begin{bmatrix} 59 & A1 & EE & 3B \\ 64 & F6 & B6 & A8 \\ 83 & 76 & 46 & 1E \\ 65 & 72 & 02 & B5 \end{bmatrix}$$

5. Menggunakan *AddRoundKey*

Melakukan proses XOR dengan kunci putaran sebelumnya untuk setiap elemennya.

$$XOR = \begin{bmatrix} 00 \oplus 4F & CE \oplus 5A & 26 \oplus 7B & 14 \oplus 10 \\ 7D \oplus 8C & C0 \oplus CD & 81 \oplus D1 & 7D \oplus 23 \\ 38 \oplus 67 & CA \oplus 2A & AD \oplus FF & 0C \oplus 45 \\ DD \oplus 28 & 29 \oplus 0D & E5 \oplus 93 & F9 \oplus 2C \end{bmatrix}$$

Hasil XOR akhir untuk semua elemen adalah:

Baris 1:

$$00 \oplus 4F = 00000000 \oplus 01001111 = 01001111 = 4F$$

$$CE \oplus 5A = 11001110 \oplus 01011010 = 10010100 = 94$$

$$26 \oplus 7B = 00100110 \oplus 01111011 = 01011101 = 5D$$

$$14 \oplus 10 = 00010100 \oplus 00010000 = 00000100 = 04$$

Baris 2:

$$7D \oplus 8C = 01111101 \oplus 10001100 = 11110001 = F1$$

$$C0 \oplus CD = 11000000 \oplus 11001101 = 00001101 = 0D$$

$$81 \oplus D1 = 10000001 \oplus 11010001 = 01010000 = 50$$

$$7D \oplus 23 = 01111101 \oplus 00100011 = 01011110 = 5E$$

Baris 3:

$$38 \oplus 67 = 00111000 \oplus 01100111 = 01011111 = 5F$$

$$CA \oplus 2A = 11001010 \oplus 00101010 = 11100000 = E0$$

$$AD \oplus FF = 10101101 \oplus 11111111 = 01010010 = 52$$

$$0C \oplus 45 = 00001100 \oplus 01000101 = 01001001 = 49$$

Baris 4:

$$DD \oplus 28 = 11011101 \oplus 00101000 = 11110101 = F5$$

$$29 \oplus 0D = 00101001 \oplus 00001101 = 00100100 = 24$$

$$E5 \oplus 93 = 11100101 \oplus 10010011 = 01110110 = 76$$

$$F9 \oplus 2C = 11111001 \oplus 00101100 = 110010101 = D5$$

Setelah melakukan semua operasi XOR, maka didapatkan matriks hasil berikut ini:

$$\begin{bmatrix} 4F & 94 & 5D & 04 \\ F1 & 0D & 50 & 5E \\ 5F & E0 & 52 & 49 \\ F5 & 24 & 76 & D5 \end{bmatrix}$$

6. Menggunakan *Inverse MixColumns*

Dengan memakai matriks hasil *InvSubBytes* diatas, kemudian diproses dengan menggunakan persamaan (2.13), maka menghasilkan:

a. Kolom pertama

$$\begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \begin{bmatrix} 59 \\ 64 \\ 83 \\ 65 \end{bmatrix} = \begin{bmatrix} 00 \\ 7D \\ 3B \\ DD \end{bmatrix}$$

$$(0E \cdot 59) \oplus (0B \cdot 64) \oplus (0D \cdot 83) \oplus (09 \cdot 65)$$

$$= E\big(L(0E) + L(59)\big) \oplus E\big(L(0B) + L(64)\big) \oplus E(L(0D) + L(83)) \oplus E(L(09) + L(65))$$

$$= E(FA) \oplus E(CB) \oplus E(BE) \oplus E(71)$$

$$= 6C \oplus 99 \oplus C5 \oplus 30$$

$$= 01101100 \oplus 10011001 \oplus 11000101 \oplus 00110000$$

$$= 00000000$$

$$= 00$$

$$(09 \cdot 59) \oplus (0E \cdot 64) \oplus (0B \cdot 83) \oplus (0D \cdot 65)$$

$$= E\big(L(09) + L(59)\big) \oplus E\big(L(0E) + L(64)\big) \oplus E(L(0B) + L(83)) \oplus E(L(0D) + L(65))$$

$$= E(E2) \oplus E(7C) \oplus E(38) \oplus E(58)$$

$$= 5A \oplus E9 \oplus 4F \oplus 81$$

$$= 01011010 \oplus 11101001 \oplus 01001111 \oplus 0010000001$$

$$= 01111101$$

$$= 7D$$

$$(0D \cdot 59) \oplus (09 \cdot 64) \oplus (0E \cdot 83) \oplus (0B \cdot 65)$$

$$= E\big(L(0D) + L(59)\big) \oplus E\big(L(09) + L(64)\big) \oplus E(L(0E) + L(83)) \oplus E(L(0B) + L(65))$$

$$= E(CB) \oplus E(64) \oplus E(8F) \oplus E(DE)$$

$$= 99 \oplus 10 \oplus 41 \oplus F3$$

$$= 10011001 \oplus 00010000 \oplus 01000001 \oplus 11110011$$

$$= 00111011$$

$$= 3B$$

$$(0B \cdot 59) \oplus (0D \cdot 64) \oplus (09 \cdot 83) \oplus (0E \cdot 65)$$

$$= E\big(L(0B) + L(59)\big) \oplus E\big(L(0D) + L(64)\big) \oplus E(L(09) + L(83)) \oplus E(L(0E) + L(65))$$

$$= E(4D) \oplus E(4D) \oplus E(97) \oplus E(69)$$

$$= 28 \oplus 28 \oplus C0 \oplus 1D$$

$$= 00101000 \oplus 00101000 \oplus 11000000 \oplus 00011101$$

$= 11011101$

$= DD$

b. Kolom kedua

$$\begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \begin{bmatrix} A1 \\ F6 \\ 76 \\ 72 \end{bmatrix} = \begin{bmatrix} CE \\ C0 \\ CA \\ 29 \end{bmatrix}$$

$(0E \cdot A1) \oplus (0B \cdot F6) \oplus (0D \cdot 76) \oplus (09 \cdot 72)$

$= E\big(L(0E) + L(A1)\big) \oplus E\big(L(0B) + L(F6)\big) \oplus E(L(0D)$

$+ L(76)) \oplus E(L(09) + L(72))$

$= E(D3) \oplus E(96) \oplus E(B0) \oplus E(CD)$

$= DE \oplus 40 \oplus 9B \oplus CB$

$= 11011110 \oplus 01000000 \oplus 10011011 \oplus 11001011$

$= 11001110$

$= CE$

$(09 \cdot A1) \oplus (0E \cdot F6) \oplus (0B \cdot 76) \oplus (0D \cdot 72)$

$= E\big(L(09) + L(A1)\big) \oplus E\big(L(0E) + L(F6)\big) \oplus E(L(0B)$

$+ L(76)) \oplus E(L(0D) + L(72))$

$= E(CB) \oplus E(21) \oplus E(36) \oplus E(E4)$

$= 99 \oplus 34 \oplus 44 \oplus 29$

$= 10011001 \oplus 00110100 \oplus 01000100 \oplus 00101001$

$= 11000000$

$= C0$

$$(0D \cdot A1) \oplus (09 \cdot F6) \oplus (0E \cdot 76) \oplus (0B \cdot 72)$$

$$= E\big(L(0D) + L(A1)\big) \oplus E\big(L(09) + L(F6)\big) \oplus E(L(0E) + L(76)) \oplus E(L(0B) + L(72))$$

$$= E(E2) \oplus E(39) \oplus E(81) \oplus E(62)$$

$$= 5A \oplus D1 \oplus 16 \oplus 57$$

$$= 01011010 \oplus 11010001 \oplus 00010110 \oplus 01010111$$

$$= 11001010$$

$$= CA$$

$$(0B \cdot A1) \oplus (0D \cdot F6) \oplus (09 \cdot 76) \oplus (0E \cdot 72)$$

$$= E\big(L(0B) + L(A1)\big) \oplus E\big(L(0D) + L(F6)\big) \oplus E(L(09) + L(76)) \oplus E(L(0E) + L(72))$$

$$= E(64) \oplus E(10) \oplus E(99) \oplus E(D5)$$

$$= 10 \oplus 5F \oplus ED \oplus 8B$$

$$= 00010000 \oplus 01011111 \oplus 11101101 \oplus 010001011$$

$$= 00101001$$

$$= 29$$

c. Kolom ketiga

$$\begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \begin{bmatrix} EE \\ B6 \\ 46 \\ 02 \end{bmatrix} = \begin{bmatrix} 26 \\ 81 \\ AD \\ E5 \end{bmatrix}$$

$$(0E \cdot EE) \oplus (0B \cdot B6) \oplus (0D \cdot 46) \oplus (09 \cdot 02)$$

$$= E\big(L(0E) + L(EE)\big) \oplus E\big(L(0B) + L(B6)\big) \oplus E(L(0D)$$

$$+L(46))\oplus E(L(09)+L(02))$$

$$= E(3C)\oplus E(D9)\oplus E(20)\oplus E(DE)$$

$$= D3\oplus E3\oplus E5\oplus F3$$

$$= 11010011\oplus 11100011\oplus 11100101\oplus 11110011$$

$$= 00100110$$

$$= 26$$

$$(09\cdot EE)\oplus(0E\cdot B6)\oplus(0B\cdot 46)\oplus(0D\cdot 02)$$

$$= E\big(L(09)+L(EE)\big)\oplus E\big(L(0E)+L(B6)\big)\oplus E(L(0B)$$

$$+L(46))\oplus E(L(0D)+L(02))$$

$$= E(24)\oplus E(6E)\oplus E(A6)\oplus E(F7)$$

$$= 37\oplus 61\oplus 2A\oplus FD$$

$$= 00110111\oplus 01100001\oplus 00101010\oplus 11111101$$

$$= 10000001$$

$$= 81$$

$$(0D\cdot EE)\oplus(09\cdot B6)\oplus(0E\cdot 46)\oplus(0B\cdot 02)$$

$$= E\big(L(0D)+L(EE)\big)\oplus E\big(L(09)+L(B6)\big)\oplus E(L(0E)$$

$$+L(46))\oplus E(L(0B)+L(02))$$

$$= E(0D)\oplus E(76)\oplus E(11)\oplus E(71)$$

$$= F8\oplus AD\oplus E1\oplus 19$$

$$= 11111000\oplus 10101101\oplus 11100001\oplus 00011001$$

$$= 10101101$$

$$= AD$$

$$(0B \cdot EE) \oplus (0D \cdot B6) \oplus (09 \cdot 46) \oplus (0E \cdot 02)$$

$$= E\big(L(0B) + L(EE)\big) \oplus E\big(L(0D) + L(B6)\big) \oplus E(L(09)$$

$$+ L(46)) \oplus E(L(0E) + L(02))$$

$$= E(8B) \oplus E(5F) \oplus E(09) \oplus E(C6)$$

$$= 56 \oplus 9A \oplus 2E \oplus 07$$

$$= 01010110 \oplus 10011010 \oplus 00101110 \oplus 00000111$$

$$= 11100101$$

$$= E5$$

d. Kolom keempat

$$\begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \begin{bmatrix} 3B \\ A8 \\ 1E \\ B5 \end{bmatrix} = \begin{bmatrix} 14 \\ 7D \\ 0C \\ F9 \end{bmatrix}$$

$$(0E \cdot 3B) \oplus (0B \cdot A8) \oplus (0D \cdot 1E) \oplus (09 \cdot B5)$$

$$= E\big(L(0E) + L(3B)\big) \oplus E\big(L(0B) + L(A8)\big) \oplus E(L(0D)$$

$$+ L(1E)) \oplus E(L(09) + L(B5))$$

$$= E(9A) \oplus E(B0) \oplus E(F2) \oplus E(A7)$$

$$= 2C \oplus 9B \oplus DD \oplus 7E$$

$$= 00101100 \oplus 10011011 \oplus 11011101 \oplus 01111110$$

$$= 00010100$$

$$= 14$$

$$(09 \cdot 3B) \oplus (0E \cdot A8) \oplus (0B \cdot 1E) \oplus (0D \cdot B5)$$

$$= E\big(L(0E) + L(3B)\big) \oplus E\big(L(0B) + L(A8)\big) \oplus E(L(0D) + L(1E)) \oplus E(L(09) + L(B5))$$

$$= E(82) \oplus E(07) \oplus E(74) \oplus E(8E)$$

$$= 3A \oplus FF \oplus 87 \oplus 3F$$

$$= 00111010 \oplus 11111111 \oplus 10000111 \oplus 00111111$$

$$= 01111101$$

$$= 7D$$

$$(0D \cdot 3B) \oplus (09 \cdot A8) \oplus (0E \cdot 1E) \oplus (0B \cdot B5)$$

$$= E\big(L(0D) + L(3B)\big) \oplus E\big(L(09) + L(A8)\big) \oplus E(L(0E) + L(1E)) \oplus E(L(0B) + L(B5))$$

$$= E(AB) \oplus E(1F) \oplus E(C3) \oplus E(08)$$

$$= DF \oplus AA \oplus 63 \oplus 1A$$

$$= 11011111 \oplus 10101010 \oplus 01100011 \oplus 00011010$$

$$= 00001100$$

$$= 0C$$

$$(0B \cdot 3B) \oplus (0D \cdot A8) \oplus (09 \cdot 1E) \oplus (0E \cdot B5)$$

$$= E\big(L(0B) + L(3B)\big) \oplus E\big(L(0D) + L(A8)\big) \oplus E(L(09) + L(1E)) \oplus E(L(0E) + L(B5))$$

$$= E(2D) \oplus E(36) \oplus E(DB) \oplus E(BF)$$

$$= AB \oplus 44 \oplus 42 \oplus 54$$

$$= 10101011 \oplus 01000100 \oplus 01000010 \oplus 01010100$$

$= 11111001$

$= F9$

Hasil transformasi *InvMixColumns* secara keseluruhan adalah:

$$\begin{bmatrix} 59 & A1 & EE & 3B \\ 64 & F6 & B6 & A8 \\ 83 & 76 & 46 & 1E \\ 65 & 72 & 02 & B5 \end{bmatrix} \Rightarrow \begin{bmatrix} 00 & CE & 26 & 14 \\ 7D & C0 & 81 & 7D \\ 38 & CA & AD & 0C \\ DD & 29 & E5 & F9 \end{bmatrix}$$

7. Mengulangi semua tahap sampai putaran tercapai

   Langkah ini akan diulang sesuai jumlah putaran yang ditentukan oleh kunci AES (10 putaran untuk AES-128).

Putaran pertama: $\begin{bmatrix} 4F & 94 & 5D & 04 \\ F1 & 0D & 50 & 5E \\ 5F & E0 & 52 & 49 \\ F5 & 24 & 76 & D5 \end{bmatrix}$

Putaran kedua: $\begin{bmatrix} DE & 36 & 03 & CD \\ D0 & 5E & 92 & 4D \\ F0 & AD & D5 & 32 \\ 47 & DC & C2 & 4E \end{bmatrix}$

Putaran ketiga: $\begin{bmatrix} 1A & 37 & 5A & E4 \\ 52 & 20 & 49 & 64 \\ 45 & F5 & BC & E0 \\ C0 & FD & 5C & 60 \end{bmatrix}$

Putaran keempat: $\begin{bmatrix} 69 & 68 & 96 & F0 \\ E0 & 5E & 70 & 49 \\ 93 & CD & 7B & 15 \\ 1B & D6 & 2C & 72 \end{bmatrix}$

Putaran kelima: $\begin{bmatrix} 1F & EF & 64 & DB \\ 42 & 0A & C9 & 7B \\ E8 & 1F & A3 & 2D \\ 7B & 9E & 1F & 32 \end{bmatrix}$

Putaran keenam: $\begin{bmatrix} 93 & FB & 75 & DA \\ 2B & 8F & 23 & 70 \\ AF & CC & 70 & 6C \\ FB & EB & 66 & 5A \end{bmatrix}$

Putaran ketujuh: $\begin{bmatrix} E6 & 13 & 3B & B4 \\ DC & 05 & 70 & 80 \\ 66 & 5F & 68 & F2 \\ 54 & 5C & 35 & 6D \end{bmatrix}$

Putaran kedelapan: $\begin{bmatrix} B2 & F5 & 2D & EC \\ 7F & E8 & CD & A6 \\ CC & 46 & D6 & 0F \\ 3E & BE & DE & F4 \end{bmatrix}$

Putaran kesembilan: $\begin{bmatrix} 2C & A8 & 40 & 8A \\ 1C & 1C & E6 & B3 \\ 47 & 83 & 5B & 66 \\ 4C & C5 & 79 & C7 \end{bmatrix}$

8. Pada putaran terakhir, hanya melakukan *InvShiftRows, InvSubBytes,* dan *AddRoundKey* tanpa *MixColumns*.

   Putaran terakhir: $\begin{bmatrix} 12 & 34 & 56 & 78 \\ 90 & 50 & 45 & 67 \\ 89 & 34 & 89 & 60 \\ 56 & 45 & 67 & 89 \end{bmatrix}$

   Sehingga putaran terakhir ini dapat disebut sebagai blok terenkripsi.

   Blok 1: [12,34,56,78,90,50,45,67,89,34,89,23,56,45,67,89]

   Blok 2: [$E3, F4,7C, CD, CC, A3, F6,43, CB,$ 69, $B7,47,8A, C3,$34,46]

   Blok 3: [$D1, EA,$ 87,14,81,$1C, C6,73, DB, B8, B5,48,2E, 6B, 0E,$ 90]

9. Gabungkang kembali blok-blok yang didekripsi

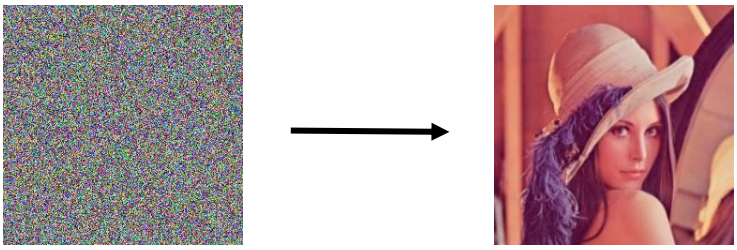
   Setelah semua putaran selesai, maka setiap blok berbentuk matriks 4×4 akan digabungkan menjadi citra yang terdekripsi.

$$\begin{bmatrix} [12,34,56] & [78,90,50] & [45,67,89] & [34,89,23] \\ [56,45,67] & [89, E3, F4] & [7C, CD, CC] & [A3, F6,43] \\ [C8,69, B7] & [47,8A, C3] & [34,46, D1] & [EA, 87,14] \\ [81,1C, C6] & [73, DB, B8] & [B5,48,2E] & [6B, 0E, 90] \end{bmatrix}$$

10. Jika terdapat penambahan *padding*, maka *padding* yang ditambahkan selama enkripsi harus dihilangkan.

    Dengan mengetahui jumlah byte padding, kita dapat menghapus byte-byte tersebut dari akhir data. Misalkan jika padding yang ditambah adalah 7 byte, maka kita harus menghapus 7 byte terakhir dari array byte citra.

11. Menampilkan kembali citra digital yang telah didekripsi dalam format *.png, menandakan bahwa proses dekripsi telah selesai.



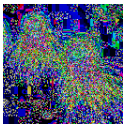
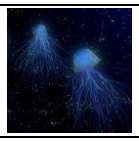
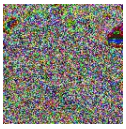
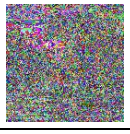
Gambar 4.4 Hasil Dekripsi Citra Digital Menggunakan Algoritma AES.

4.2 Pembahasan Hasil Enkripsi dan Dekripsi Citra Digital

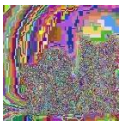
4.2.1 Hasil Enkripsi dan Dekripsi Algoritma RSA

Dari hasil pengolahan data yang telah dilakukan setelah pengujian enkripsi dan dekripsi pada citra digital menggunakan algoritma RSA untuk citra digital yang dibagi menjadi dua resolusi, yakni ukuran 128×128 piksel dan 256×256 piksel. Berikut adalah Tabel 4.1 dan Tabel 4.2 yang berisi data hasil enkripsi dan dekripsi.

Tabel 4.1 Hasil Enkripsi dan Dekripsi Citra Digital Beresolusi $\mathbf{128 \times 128}$ Piksel Menggunakan Algoritma RSA.

No	Nama Citra Digital	Hasil enkripsi algoritma RSA	Waktu proses enkripsi (detik)	Ukuran file enkripsi (kb)	Hasil dekripsi algoritma RSA	Waktu proses dekripsi (detik)	Ukuran file dekripsi (kb)
1	Gambar_1		0,1443	35,23		0,1426	25,85
2	Gambar_2		0,1671	46,20		0,1822	37,59
3	Gambar_3		0,1557	46,14		0,1485	29,86

Tabel 4.2 Hasil Enkripsi dan Dekripsi Citra Digital Beresolusi $\mathbf{256 \times 256}$ Piksel Menggunakan Algoritma RSA.

No	Nama Citra Digital	Hasil enkripsi algoritma RSA	Waktu proses enkripsi (detik)	Ukuran file enkripsi (kb)	Hasil dekripsi algoritma RSA	Waktu proses dekripsi (detik)	Ukuran file dekripsi (kb)
1	Gambar_1		0,5107	100,39		0,6355	58,27
2	Gambar_2		0,5242	188,31		0,6114	134,63
3	Gambar_3		0,5625	100,26		0,6330	60,80

Berdasarkan hasil pengujian, enkripsi dan dekripsi citra digital menggunakan algoritma RSA menunjukkan adanya pengaruh resolusi citra terhadap waktu pemrosesan dan ukuran file yang dihasilkan. Semakin tinggi resolusi citra, semakin besar pula data yang harus diproses, yang berdampak pada waktu eksekusi dan ukuran file. Pengolahan data ini dilakukan dengan menggunakan bahasa pemrograman Python melalui Google Colab.

Pada citra 128×128 piksel, waktu rata-rata proses enkripsi tercatat sebesar 0,1448 detik, dengan median 0,1417 detik dan rentang 0,0480 detik, menunjukkan konsistensi yang baik karena nilai mean dan median berdekatan serta rentang waktu yang kecil. Ukuran rata-rata file hasil enkripsi mencapai 44,33 kb, dengan median 46,12 kb dan rentang 12,43 kb, menunjukkan adanya sedikit variasi ukuran file. Proses dekripsi pada resolusi ini membutuhkan waktu rata-rata 0,1620 detik dengan median 0,1634 detik dan rentang 0,0396 detik, sedikit lebih lama dibandingkan proses enkripsi, yang merupakan karakteristik umum dari algoritma RSA. Ukuran file hasil dekripsi tercatat rata-rata sebesar 30,75 kb, dengan median 31,02 kb dan

rentang 15,47 kb, menunjukkan sedikit penurunan ukuran dibandingkan dengan file enkripsi.

Pada citra beresolusi 256×256 piksel, waktu rata-rata proses enkripsi adalah 0,5691 detik, dengan median 0,5473 detik dan rentang 0,3086 detik. Terjadi peningkatan signifikan dalam waktu pemrosesan dibandingkan dengan citra 128×128 piksel, yang dapat dijelaskan oleh peningkatan kompleksitas data. Ukuran rata-rata file enkripsi pada resolusi ini mencapai 147,30 kb, dengan median 152,53 kb dan rentang 98,18 kb, menunjukkan adanya variasi ukuran yang lebih besar. Proses dekripsi membutuhkan waktu rata-rata 0,6806 detik, dengan median 0,6709 detik dan rentang 0,3265 detik, sedikit lebih lama dibandingkan proses enkripsi. Ukuran rata-rata file dekripsi adalah 91,93 kb, dengan median 89,19 kb dan rentang 100,83 kb, menunjukkan adanya perubahan ukuran setelah melalui proses enkripsi-dekripsi.

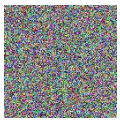
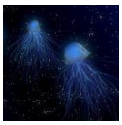
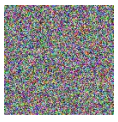
Secara umum, hasil analisis menunjukkan bahwa peningkatan resolusi citra mempengaruhi performa algoritma RSA dalam hal waktu pemrosesan dan ukuran file. Citra beresolusi 256×256 membutuhkan waktu pemrosesan lebih lama dan menghasilkan file dengan ukuran lebih besar dibandingkan citra 128×128 piksel, sesuai dengan peningkatan jumlah data. Baik pada resolusi 128×128 maupun 256×256, waktu dekripsi selalu sedikit lebih lama dibandingkan dengan enkripsi, karena proses dekripsi RSA melibatkan perhitungan tambahan dengan kunci privat. Variasi ukuran file pada kedua resolusi menunjukkan bahwa hasil enkripsi dan dekripsi tidak selalu menghasilkan ukuran yang sama, meskipun tetap dalam batas yang wajar. Secara keseluruhan, algoritma RSA bekerja dengan baik dan konsisten

pada kedua resolusi, namun menunjukkan peningkatan beban komputasi dan ukuran file seiring dengan bertambahnya resolusi citra.

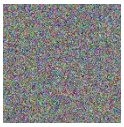
4.2.2 Hasil Enkripsi dan Dekripsi Algoritma AES

Pada bagian ini, hasil pengolahan data yang diperoleh setelah pengujian enkripsi dan dekripsi pada citra digital menggunakan algoritma AES (*Advanced Encryption Standard*) untuk citra digital dibagi menjadi dua resolusi, yaitu 128×128 piksel dan 256×256 piksel. Pengolahan data ini dilakukan dengan menggunakan pemrograman Python dan memanfaatkan berbagai library. Berikut ini adalah Tabel 4.3 dan Tabel 4.4 berisi data yang diperoleh dari pengujian enkripsi dan dekripsi menggunakan algoritma AES terhadap 50 citra digital beresolusi 128×128 piksel dan 256×256 piksel yang diambil dari situs Pixabay.

Tabel 4.3 Hasil Enkripsi dan Dekripsi Citra Digital Beresolusi $\mathbf{128 \times 128}$ Piksel Menggunakan Algoritma AES.

No	Nama Citra Digital	Hasil enkripsi algoritma RSA	Waktu proses enkripsi (detik)	Ukuran file enkripsi (kb)	Hasil dekripsi algoritma RSA	Waktu proses dekripsi (detik)	Ukuran file dekripsi (kb)
1	Gambar_1		0,0040	48,20		0,0048	25,85
2	Gambar_2		0,0032	48,20		0,0032	37,59
3	Gambar_3		0,0037	48,20		0,0042	29,86

Tabel 4.4 Hasil Enkripsi dan Dekripsi Citra Digital Beresolusi $\mathbf{256 \times 256}$ Piksel Menggunakan Algoritma AES.

No	Nama Citra Digital	Hasil enkripsi algoritma RSA	Waktu proses enkripsi (detik)	Ukuran file enkripsi (kb)	Hasil dekripsi algoritma RSA	Waktu proses dekripsi (detik)	Ukuran file dekripsi (kb)
1	Gambar_1		0,0163	192,55		0,0120	58,27
2	Gambar_2		0,0148	192,55		0,0163	134,63
3	Gambar_3		0,0133	192,55		0,0122	60,80

Berdasarkan hasil pengolahan data, proses enkripsi dan dekripsi citra digital menggunakan algoritma AES menunjukkan bahwa resolusi citra mempengaruhi performa waktu pemrosesan, ukuran file, serta kualitas citra yang dihasilkan. Pengujian ini dilakukan menggunakan Python di platform Google Colab.

Pada citra dengan resolusi 128×128 piksel, waktu rata-rata enkripsi adalah 0,0053 detik, dengan median 0,0052 detik dan rentang 0,0053 detik. Waktu pemrosesan enkripsi yang sangat cepat dan rentang yang kecil menunjukkan bahwa algoritma AES memiliki konsistensi yang baik dalam memproses citra beresolusi rendah. Ukuran rata-rata dan median file hasil enkripsi adalah 48,20 kb dengan rentang 0,00 kb, yang mengindikasikan bahwa ukuran file hasil enkripsi konstan untuk setiap kali proses. Untuk proses dekripsi, waktu rata-rata yang dibutuhkan adalah 0,0044 detik dengan median 0,0040 detik dan rentang 0,0038 detik, yang sedikit lebih cepat dibandingkan waktu enkripsi. Ukuran rata-rata file hasil dekripsi adalah 30,76 kb, dengan median 31,02 kb dan rentang 15,47 kb, menunjukkan bahwa ukuran file setelah dekripsi lebih kecil dibandingkan dengan hasil enkripsi.

Pada citra beresolusi 256×256 piksel, waktu rata-rata enkripsi adalah 0,0151 detik, dengan median 0,0148 detik dan rentang 0,0067 detik. Waktu pemrosesan ini lebih lama dibandingkan dengan citra 128×128 piksel, karena peningkatan resolusi meningkatkan kompleksitas data yang diproses. Ukuran file hasil enkripsi rata-rata dan median sebesar 192,55 kb, dengan rentang 0,00 kb, menunjukkan bahwa ukuran file enkripsi pada resolusi ini konsisten di setiap proses. Untuk proses dekripsi, waktu rata-rata yang dibutuhkan adalah 0,0160 detik, dengan median 0,0174 detik dan rentang 0,0163 detik. Waktu dekripsi sedikit lebih lama dibandingkan enkripsi, namun tetap dalam batas yang efisien. Ukuran file dekripsi pada resolusi ini tercatat rata-rata 91,93 kb, dengan median 89,19 kb dan rentang 100,75 kb, menunjukkan bahwa ukuran file setelah dekripsi mengalami perubahan signifikan dibandingkan ukuran file enkripsi.

Secara keseluruhan, analisis menunjukkan bahwa algoritma AES bekerja sangat efisien dalam memproses citra dengan berbagai resolusi. Dibandingkan dengan resolusi 128×128, proses enkripsi dan dekripsi untuk citra 256×256 membutuhkan waktu yang sedikit lebih lama, yang mencerminkan pengaruh dari peningkatan jumlah data yang harus diproses. Ukuran file hasil enkripsi konsisten pada kedua resolusi, dengan rentang 0,00 kb, menunjukkan bahwa algoritma AES mempertahankan struktur file enkripsi dengan baik. Namun, ukuran file hasil dekripsi cenderung lebih kecil daripada file enkripsi, yang dapat disebabkan oleh perbedaan dalam pengolahan data setelah dekripsi. Secara umum, algoritma AES menunjukkan kinerja yang cepat dan konsisten dalam hal waktu pemrosesan, baik untuk enkripsi maupun dekripsi, terutama untuk citra digital berukuran sedang maupun kecil.

4.2.3 Hasil MSE Algoritma RSA dan AES

Berikut ini adalah Tabel 4.5 berisi hasil MSE yang diperoleh dari pengujian algoritma RSA dan AES.

Tabel 4.5 Hasil MSE Citra Digital Algoritma RSA dan AES.

No	MSE citra enkripsi algoritma RSA		MSE citra enkripsi algoritma AES	
	128×128 Piksel	256×256 Piksel	128×128 Piksel	256×256 Piksel
1	10669,9	13832,4	17134,3	17065,1
2	8532,6	8813,9	8791,0	8919,0
3	7581,0	8393,8	8550,9	9833,3

Dari hasil pengolahan data yang telah dilakukan setelah pengujian MSE pada citra digital menggunakan algoritma RSA dan AES untuk citra digital yang dibagi menjadi dua resolusi, yaitu ukuran 128×128 piksel dan 256×256 piksel.

Berdasarkan hasil pengolahan *Mean Squared Error* (MSE) pada citra digital menggunakan algoritma RSA, terlihat adanya perbedaan antara MSE hasil enkripsi untuk citra dengan resolusi 128×128 piksel dan 256×256 piksel. Pada proses enkripsi, MSE untuk citra beresolusi 128×128 piksel memiliki nilai rata-rata (mean) sebesar 9640,8 dengan median 9434,3 dan rentang (range) 5238,0. Sementara itu, untuk citra dengan resolusi 256×256 piksel, MSE enkripsi tercatat lebih tinggi, dengan nilai rata-rata 10826,9, median 9827,6, dan rentang 13581,0. Peningkatan nilai MSE pada citra beresolusi 256×256 piksel mengindikasikan bahwa semakin tinggi resolusi citra, semakin besar perbedaan antara citra asli dan citra hasil enkripsi, yang merupakan dampak dari kompleksitas data selama proses enkripsi.

Pada hasil MSE citra digital menggunakan algoritma AES, terdapat perbedaan antara proses enkripsi untuk citra beresolusi 128×128 piksel dan 256×256 piksel. Pada proses enkripsi, MSE untuk citra dengan resolusi 128×128 piksel memiliki nilai rata-rata (mean) sebesar 10311,0, dengan median 9781,8 dan rentang 9445,7. Sementara itu, pada citra dengan resolusi 256×256 piksel, nilai MSE enkripsi sedikit lebih tinggi, dengan mean sebesar 11180,3, median 10222,5, dan rentang 12661,1. Peningkatan MSE pada citra beresolusi 256×256 piksel menunjukkan bahwa semakin tinggi resolusi, semakin besar perbedaan nilai piksel antara citra asli dan citra terenkripsi, karena jumlah data yang lebih kompleks mengalami transformasi yang lebih signifikan.

4.2.4 Hasil Keseluruhan Perbandingan Algoritma RSA dan AES

Bagian ini menyajikan hasil analisis komprehensif mengenai perbandingan kinerja algoritma RSA dan AES dalam proses enkripsi dan dekripsi citra digital dengan resolusi 128×128 piksel dan 256×256 piksel. Analisis mencakup berbagai parameter penting yang meliputi kecepatan enkripsi dan dekripsi, ukuran file setelah proses enkripsi dan dekripsi, serta tingkat perubahan yang diukur melalui nilai *Mean Squared Error* (MSE) pada citra terenkripsi. Untuk memberikan gambaran yang lebih mendalam, data yang disajikan mencakup statistik deskriptif, yaitu nilai *mean* (M), *median* (Me), dan *range* (R) dari masing-masing parameter. Perbandingan dilakukan dengan mempertimbangkan efisiensi waktu proses, perubahan kualitas citra setelah enkripsi, dan pengaruhnya terhadap ukuran file.

Tabel 4.6 Tabel Statistik Keseluruhan Hasil Kinerja Algoritma RSA dan AES.

Resolusi	Algoritma	Kecepatan Enkripsi	Kecepatan Dekripsi	Ukuran Hasil Enkripsi	Ukuran Hasil Dekripsi	Nilai MSE
128	RSA	M: 0,1448	M: 0,1620	M: 44,33	M: 30,75	M: 9640,8
		Me: 0,1417	Me: 0,1634	Me: 46,12	Me: 31,02	Me: 9434,3
		R: 0,0480	R: 0,0396	R: 12,43	R: 15,47	R: 5238,0
	AES	M: 0,0053	M: 0,0044	M: 48,20	M: 30,76	M: 10311,0
		Me: 0,0052	Me: 0,0040	Me: 48,20	Me: 31,02	Me: 9781,8
		R: 0,0053	R: 0,0038	R: 0,00	R: 15,47	R: 9445,7
256	RSA	M: 0,5691	M: 0,6806	M: 147,30	M: 91,93	M: 10826,9
		Me: 0,5473	Me: 0,6709	Me: 152,53	Me: 89,19	Me: 9827,6
		R: 0,3086	R: 0,3265	R: 98,18	R: 100,83	R: 13581,0
	AES	M: 0,0151	M: 0,0160	M: 192,55	M: 91,93	M: 11180,3
		Me: 0,0148	Me: 0,0174	Me: 192,55	Me: 89,19	Me: 10222,5
		R: 0,0067	R: 0,0163	R: 0,00	R: 100,75	R: 12661,1

Berdasarkan Tabel 4.6, terlihat bahwa AES memiliki keunggulan signifikan dalam efisiensi waktu enkripsi dan dekripsi dibandingkan RSA. Pada resolusi 128×128, rata-rata waktu enkripsi RSA adalah 0,1448 detik, jauh lebih lambat dibandingkan AES yang hanya membutuhkan 0,0053 detik. Hal yang sama berlaku untuk dekripsi, di mana RSA membutuhkan rata-rata waktu 0,1620 detik, sementara AES hanya 0,0044 detik. Performa RSA juga lebih bervariasi dengan rentang waktu yang lebih besar dibandingkan AES. Pada resolusi 256×256, perbedaan semakin mencolok; rata-rata waktu enkripsi RSA adalah 0,5691 detik, sementara AES tetap jauh lebih cepat dengan 0,0151 detik. RSA menunjukkan peningkatan waktu pemrosesan yang signifikan pada resolusi lebih tinggi, sementara AES tetap konsisten dengan variasi yang lebih kecil.

Dalam hal ukuran hasil enkripsi, RSA menghasilkan ukuran yang lebih kecil dibandingkan AES pada kedua resolusi. Namun, AES menunjukkan konsistensi yang lebih baik dengan rentang ukuran hasil enkripsi nol pada semua kasus. Ukuran hasil dekripsi dari kedua algoritma hampir identik pada kedua resolusi, dengan variasi rentang yang serupa. Untuk nilai MSE, pada resolusi 128×128, nilai rata-rata MSE RSA adalah 9640,8 dibandingkan AES dengan nilai 10311,0. Perbedaan ini tetap konsisten pada resolusi 256×256 dengan RSA memiliki rata-rata MSE 10826,9 dibandingkan AES 11180,3. Namun, AES menunjukkan variasi nilai MSE yang lebih kecil, yang menunjukkan stabilitas kualitas enkripsi.

Secara keseluruhan, AES menunjukkan keunggulan dalam efisiensi waktu dan konsistensi, membuatnya lebih cocok untuk aplikasi yang memerlukan kecepatan tinggi dan hasil yang stabil. Di sisi lain, RSA menghasilkan ukuran file terenkripsi yang lebih kecil dan nilai MSE yang sedikit lebih rendah, membuatnya lebih sesuai untuk aplikasi yang memprioritaskan penghematan ruang penyimpanan. Pemilihan algoritma terbaik bergantung pada kebutuhan spesifik aplikasi, apakah mengutamakan kecepatan, konsistensi, atau efisiensi ruang dan akurasi kualitas enkripsi.

4.3 Kajian Keislaman terkait Enkripsi dan Dekripsi

Prinsip menjaga privasi dan keamanan yang diajarkan dalam Surat An-Nur ayat 27 memiliki keselarasan dengan praktik enkripsi dan dekripsi dalam citra digital. Enkripsi adalah proses pengamanan data dengan cara mengubah informasi asli menjadi kode yang tidak dapat dibaca tanpa kunci tertentu. Dekripsi adalah kebalikannya, yaitu mengembalikan data terenkripsi ke bentuk aslinya sehingga

hanya orang yang berhak dapat mengakses informasi tersebut. Ini mencerminkan adab privasi dalam Islam, yang melindungi informasi dari pihak yang tidak berwenang, sebagaimana ajaran Islam menjaga hak privasi dan menghormati kehormatan individu.

Enkripsi melindungi data pribadi dengan cara yang mirip dengan izin masuk dalam konteks fisik. Hanya orang yang memiliki "kunci" atau izin untuk membuka enkripsi yang dapat mengakses informasi tersebut. Dalam Islam, menjaga privasi adalah bentuk kehormatan, sehingga secara digital, enkripsi berfungsi sebagai proteksi dari akses yang tidak sah dan melindungi kehormatan digital seseorang, sejalan dengan konsep *hifdzul 'ird* (menjaga kehormatan) dalam maqashid syariah.

Mengingat hadits yang menyatakan pentingnya menjaga rahasia dan amanah (seperti yang dilakukan Anas bin Malik dalam menjaga tugas yang diberi Rasulullah SAW), enkripsi dalam digital juga menjadi amanah bagi setiap individu yang memiliki data penting. Enkripsi menjamin bahwa data rahasia hanya dapat diakses oleh orang yang memang memiliki otoritas atau amanah untuk mengetahuinya, mirip dengan Anas yang tidak mengungkapkan tugasnya kepada siapapun yang tidak berkepentingan.

Sebagaimana Islam menekankan kewajiban moral dan agama untuk menjaga privasi dan kehormatan, di era digital, setiap orang bertanggung jawab untuk mengamankan data pribadinya. Ini termasuk menggunakan enkripsi untuk melindungi dari potensi penyalahgunaan atau kejahatan siber. Pengamanan data pribadi juga berhubungan dengan *hifdzun nafsi* (menjaga jiwa) karena mencegah potensi ancaman yang bisa menimbulkan bahaya bagi diri dan keluarga, serta

hifdzul mali (menjaga harta), karena melindungi aset-aset digital atau informasi berharga.

Meskipun enkripsi penting untuk melindungi privasi dan keamanan, penggunaannya harus sesuai dengan prinsip Islam dan norma yang berlaku. Enkripsi tidak boleh digunakan untuk menyembunyikan kejahatan, melanggar hak privasi orang lain, atau menghalangi transparansi yang bertujuan baik, seperti penegakan hukum. Kunci enkripsi adalah amanah yang harus dijaga dan tidak boleh disalahgunakan untuk mengakses atau memodifikasi data tanpa izin. Selain itu, enkripsi harus menghindari kerusakan (*mafsadah*), seperti penyebaran konten fitnah atau kebencian, serta mematuhi hukum yang berlaku selama tidak bertentangan dengan syariat. Dengan mengikuti batasan ini, enkripsi dapat menjadi alat yang mendukung kemaslahatan, menjaga keseimbangan antara hak privasi dan tanggung jawab sosial sesuai ajaran Islam.

Secara keseluruhan, praktik enkripsi dan dekripsi dalam menjaga keamanan data digital adalah implementasi modern dari prinsip menjaga privasi, keamanan, dan kehormatan yang diajarkan dalam Islam. Prinsip ini tidak hanya relevan dalam konteks fisik, tetapi juga dalam dunia digital, di mana setiap individu wajib berperan aktif dalam melindungi data serta informasi pribadinya sesuai dengan ajaran Islam dan tuntunan maqashid syariah.

BAB V
PENUTUP

5.1 Kesimpulan

Berdasarkan analisis hasil pengujian enkripsi dan dekripsi citra digital menggunakan algoritma AES dan RSA, kesimpulan yang dapat diambil adalah:

1. AES lebih cepat dibandingkan RSA dalam proses enkripsi dan dekripsi. Pada citra 128×128 piksel, AES hanya memerlukan 0,0053 detik untuk enkripsi dan 0,0044 detik untuk dekripsi, sedangkan RSA membutuhkan 0,1448 detik dan 0,1620 detik. Pada citra 256×256 piksel, perbedaan semakin terlihat, dengan AES memproses lebih cepat (0,0151 detik untuk enkripsi dan 0,0160 detik untuk dekripsi) dibandingkan RSA (0,5691 detik dan 0,6806 detik).
2. AES menghasilkan file terenkripsi lebih besar namun konsisten. Pada citra 256×256 piksel, ukuran file terenkripsi AES tercatat 192,55 kb tanpa variasi (rentang 0,00 kb), sementara RSA menghasilkan file yang lebih kecil, dengan rata-rata 147,30 kb dan rentang 98,18 kb. Namun, hasil dekripsi kedua algoritma mengembalikan ukuran yang lebih kecil. AES mempertahankan konsistensi ukuran, sedangkan RSA menunjukkan sedikit variasi dan perubahan ukuran file dekripsi dibandingkan hasil enkripsi.
3. Berdasarkan perbandingan hasil MSE antara algoritma AES dan RSA, AES menghasilkan nilai MSE enkripsi lebih tinggi dibandingkan RSA. Ini menunjukkan perubahan struktur citra yang lebih signifikan setelah enkripsi. Pada citra 128×128 piksel, MSE enkripsi AES tercatat 10311,0, sedangkan RSA 9640,8. Untuk citra 256×256 piksel, MSE enkripsi AES mencapai 11180,3, sedikit lebih tinggi daripada RSA yang bernilai 10826,9.

5.2 Saran untuk Penelitian Lanjutan

Untuk memperluas wawasan, penelitian selanjutnya dapat membandingkan kinerja algoritma kriptografi lainnya untuk mengidentifikasi keunggulan dan kekurangan masing-masing dalam enkripsi citra digital. Selain itu, penggunaan resolusi citra lebih tinggi akan memberikan pemahaman lebih mendalam terkait dampak resolusi terhadap waktu proses, ukuran file, dan kualitas enkripsi dan dekripsinya. Penelitian juga dapat diperluas dengan menggunakan format video atau yang lainnya untuk mengevaluasi performa algoritma dalam menjaga ukuran dan kualitas citra pada berbagai format.

DAFTAR PUSTAKA

Alsaffar, D. M., Sultan Almutiri, A., Alqahtani, B., Alamri, R. M., Fahhad Alqahtani, H., Alqahtani, N. N., Mohammed alshammari, G., & Ali, Azza. A. (2020). Image Encryption Based on AES and RSA Algorithms. *2020 3rd International Conference on Computer Applications & Information Security (ICCAIS)*, 1–5. https://doi.org/10.1109/ICCAIS48893.2020.9096809

Arrañaga, J. D. R., & Chavarin, J. A. S. (2017). New S-box calculation for Rijndael-AES based on artificial neural networks. *Redalyc*, *1*, 56–57.

Belkaid, B. M. (2015). Encryption of Meteosat images using AES and RSA algorithms. *International Journal of Advanced Computer Science and Applications (IJACSA)*, *6*(6), 23–24.

Berent, A. (2016). *Advanced Encryption Standard by Example*. ABI Software Development.

Cahyanti, M., Salim, R. A., & Wisuda, M. (2016). Implementasi Pengolahan Citra Untuk Pengenalan Citra Bendera Negara Berdasarkan Warna. *Seminar Riset Teknologi Informasi (SRITI)*.

Ginting, A., Isnanto, R. R., & Windasari, I. P. (2015). Implementasi Algoritma Kriptografi RSA untuk Enkripsi dan Dekripsi Email. *Jurnal Teknologi Dan Sistem Komputer*, *3*, 253–258.

Gollagi, S. G., Srividya, R., Santhosh Kumar, G., & Pareek, P. K. (2021). A New Method of Secure Image Encryption by Using Enhanced RSA Algorithm. *2021 International Conference on Forensics, Analytics, Big Data, Security (FABS)*, 1–5. https://doi.org/10.1109/FABS52071.2021.9702550

Hadist riwayat Muslim dari Anas No. 4533.

Ibnu Asyur. (1984). at-Tahrir wat-Tanwir (Vol. 18, p. 196). ad-Dar at-Tunisiyyah.

Kemenag RI. (2024). Qur'an Kemenag. Jakarta Timur : Lajnah Pentashihan Mushaf Al-Qur'an. Diambil kembali dari https://quran.kemenag.co.id

Kusumanto, R. D., & Tompunu, A. N. (2011). Pengolahan Citra Digital Untuk Mendeteksi Objek Menggunakan Pengolahan Warna Model Normalisasi RGB. *Seminar Nasional Teknologi Informasi & Komunikasi Terapan*.

Kuznetsov, Y. V. (2021). Image Sampling, Quantization, and Encoding. In *Principles of Image Printing Technology* (pp. 127–149). Springer International Publishing. https://doi.org/10.1007/978-3-030-60955-9_6

Male, G.M., Wirawan, dan Setijadi, E. (2012). Analisa Kualitas Citra Pada Steganografi Untuk Aplikasi e-Government. Prosiding Seminar Nasional Manajemen Teknologi XV. Institut Teknologi Surabaya, Surabaya.

Menez, A. J., Oorschot, P. C., Vanstone, S. A. (1996). Handbook of Applied Cryptography. CRC Press.

Meyer, C., Matyas, S. M. (1982). Cryptography, A New Dimension in Computer Data Security. John Wiley & Sons.

Mittal, H., Pandey, A. C., Saraswat, M., Kumar, S., Pal, R., & Modwel, G. (2022). A comprehensive survey of image segmentation: clustering methods, performance parameters, and benchmark datasets. *Multimedia Tools and Applications*, *81*(24), 35001–35026. https://doi.org/10.1007/s11042-021-10594-9

Munir, R. (2019). Kriprografi (Edisi Kedua). Bandung: Infomatika.

Purnama, A. (2015). Aplikasi Matriks dalam Pengolahan Gambar . *Jurnal UNSIKA SYNTAX*, *2*(2).

Sahoo, A., Mohanty, P., & Sethi, P. C. (2022). Image Encryption Using RSA Algorithm. *Lecture Notes in Networks and Systems*, *431*, 641–652. https://doi.org/10.1007/978-981-19-0901-6_56

Saravana, V., Singh, A., Li, P., & Singh, K. K. (2021). Real-time intelligent image processing for security applications. *Journal of Real-Time Image Processing*, *18*(5), 1787–1788. https://doi.org/10.1007/s11554-021-01169-w

Sulistiyanti, S. R., Setyawan, A., & Komarudin, M. (2016). *Pengolahan Citra Dasar dan Contoh Penerapannya* (1st ed.). TEKNOSAIN.

Sutoyo, T., Mulyanto, E., Suhartono, V., Nurhayati, O. D., & Wijanarto. (2009). *TEORI PENGOLAHAN CITRA DIGITAL* (1st ed.). ANDI OFFSET.

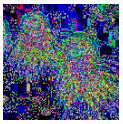
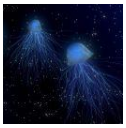
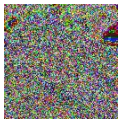
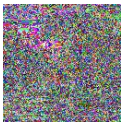
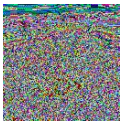
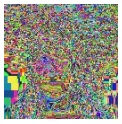
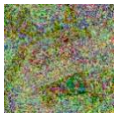
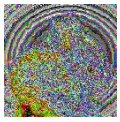
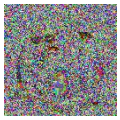
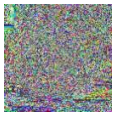
Venkatesha G, Dinesh S, & Manjunath M. (2019). AES-based algorithm for image encryption and decryption. *Pices*, *2*(11).

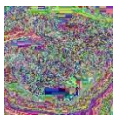
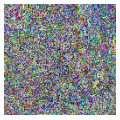
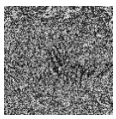
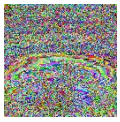
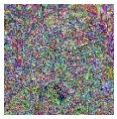
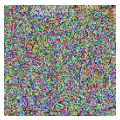
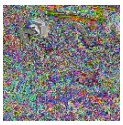
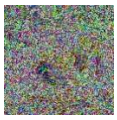
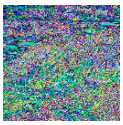
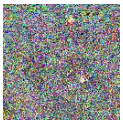
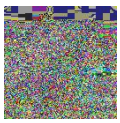
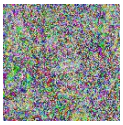
Verma, N. K., Giri, V. K., Patel, R. K., & Singh, V. P. (2020). *Computing Algorithms with Applications in Engineering: Proceedings of ICCAEEE 2019 Algorithms for Intelligent Systems* (Vol. 1). Springer Nature.

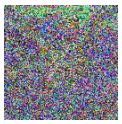
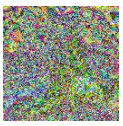
Widiasari, A. (2014). Implementasi Algoritma Kriptografi RSA Pada Aplikasi Smart Card. Skripsi. Malang: Universitas Islam Negeri Maulana Malik Ibrahim.

LAMPIRAN

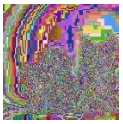
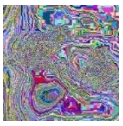
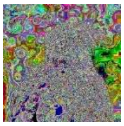
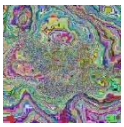
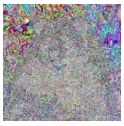
Lampiran 1 Hasil pengolahan citra digital resolusi $\mathbf{128 \times 128}$ piksel menggunakan algoritma RSA.

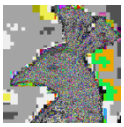
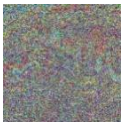
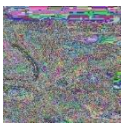
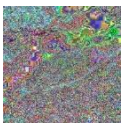
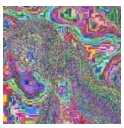
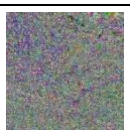
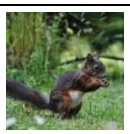
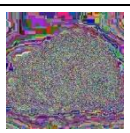
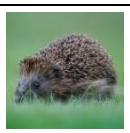
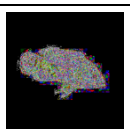
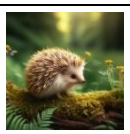
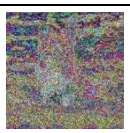
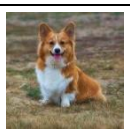
No	Nama Citra Digital	Hasil enkripsi algoritma RSA	Waktu proses enkripsi (detik)	Ukuran file enkripsi (kb)	Hasil dekripsi algoritma RSA	Waktu proses dekripsi (detik)	Ukuran file dekripsi (kb)
1	Gambar_1		0,1443	35,23		0,1426	25,85
2	Gambar_2		0,1671	46,20		0,1822	37,59
3	Gambar_3		0,1557	46,14		0,1485	29,86
4	Gambar_4		0,1382	42,46		0,1699	29,18
5	Gambar_5		0,1402	39,40		0,1500	22,12
6	Gambar_6		0,1418	45,40		0,1437	31,02
7	Gambar_7		0,1421	40,85		0,1555	24,00
8	Gambar_8		0,1424	46,77		0,1547	34,08
9	Gambar_9		0,1391	46,53		0,1656	32,57
10	Gambar_10		0,1355	44,24		0,1677	34,78

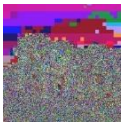
11	Gambar_11		0,1325	40,61		0,1527	26,11
12	Gambar_12		0,1422	46,91		0,1715	32,69
13	Gambar_13		0,1417	38,02		0,1601	29,29
14	Gambar_14		0,1384	44,47		0,1636	26,39
15	Gambar_15		0,1796	46,30		0,1781	31,67
16	Gambar_16		0,1483	47,66		0,1575	34,35
17	Gambar_17		0,1547	46,12		0,1669	29,39
18	Gambar_18		0,1333	46,89		0,1600	33,57
19	Gambar_19		0,1316	44,29		0,1653	29,10
20	Gambar_20		0,1600	46,72		0,1748	29,91
21	Gambar_21		0,1389	41,46		0,1634	30,94
22	Gambar_22		0,1568	47,42		0,1527	33,90
23	Gambar_23		0,1403	44,51		0,1801	33,01

24	Gambar_24		0,1362	47,31		0,1701	34,67
25	Gambar_25		0,1383	46,26		0,1520	32,67

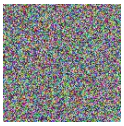
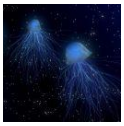
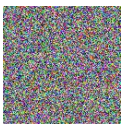
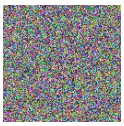
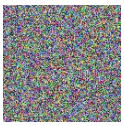
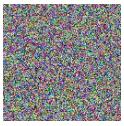
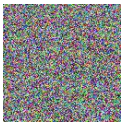
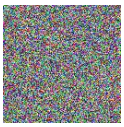
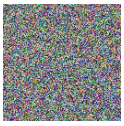
Lampiran 2 Hasil pengolahan citra digital resolusi $\mathbf{256 \times 256}$ piksel menggunakan algoritma RSA.

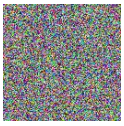
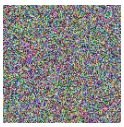
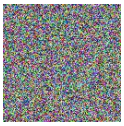
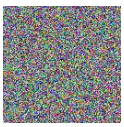
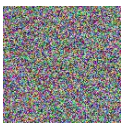
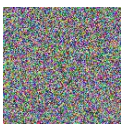
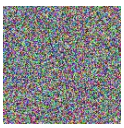
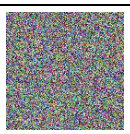
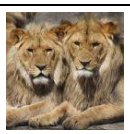
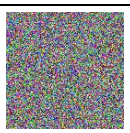
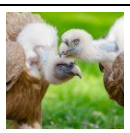
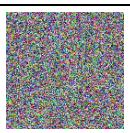
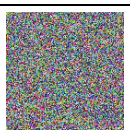
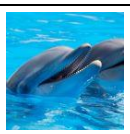
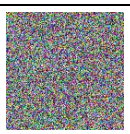
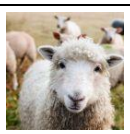
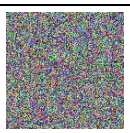
No	Nama Citra Digital	Hasil enkripsi algoritma RSA	Waktu proses enkripsi (detik)	Ukuran file enkripsi (kb)	Hasil dekripsi algoritma RSA	Waktu proses dekripsi (detik)	Ukuran file dekripsi (kb)
1	Gambar_1		0,5107	100,39		0,6355	58,27
2	Gambar_2		0,5242	188,31		0,6114	134,63
3	Gambar_3		0,5625	100,26		0,6330	60,80
4	Gambar_4		0,5590	135,84		0,6311	84,69
5	Gambar_5		0,5410	105,55		0,8073	62,69
6	Gambar_6		0,5612	146,25		0,6775	82,60
7	Gambar_7		0,6409	187,18		0,8501	114,79
8	Gambar_8		0,6238	119,05		0,6684	72,86
9	Gambar_9		0,5878	158,73		0,7126	93,58

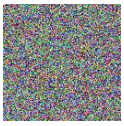
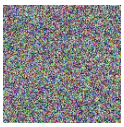
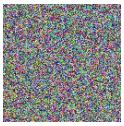
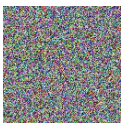
10	Gambar_10		0,6029	182,64		0,7237	127,29
11	Gambar_11		0,5375	90,13		0,5862	64,55
12	Gambar_12		0,5655	184,03		0,9127	127,71
13	Gambar_13		0,5267	157,16		0,6076	99,14
14	Gambar_14		0,5374	168,03		0,6417	107,94
15	Gambar_15		0,5553	185,36		0,6393	126,15
16	Gambar_16		0,7975	114,13		0,6050	60,70
17	Gambar_17		0,5473	188,11		0,6709	121,58
18	Gambar_18		0,7349	127,74		0,6353	88,72
19	Gambar_19		0,5243	137,76		0,6835	33,80
20	Gambar_20		0,5465	172,64		0,6613	109,26
21	Gambar_21		0,5412	169,05		0,6891	103,72
22	Gambar_22		0,5990	170,38		0,6895	117,20

23	Gambar_23		0,4995	152,53		0,6788	89,19
24	Gambar_24		0,4889	117,13		0,6899	70,45
25	Gambar_25		0,5116	124,14		0,6727	85,96

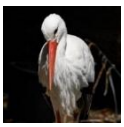
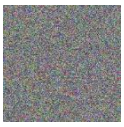
Lampiran 3 Hasil pengolahan citra digital resolusi $\mathbf{128 \times 128}$ piksel menggunakan algoritma AES.

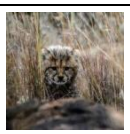
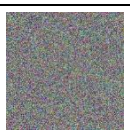
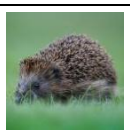
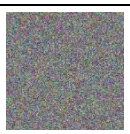
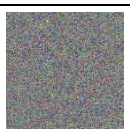
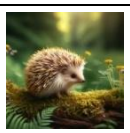
No	Nama Citra Digital	Hasil enkripsi algoritma RSA	Waktu proses enkripsi (detik)	Ukuran file enkripsi (kb)	Hasil dekripsi algoritma RSA	Waktu proses dekripsi (detik)	Ukuran file dekripsi (kb)
1	Gambar_1		0,0040	48,20		0,0048	25,85
2	Gambar_2		0,0032	48,20		0,0032	37,59
3	Gambar_3		0,0037	48,20		0,0042	29,86
4	Gambar_4		0,0052	48,20		0,0040	29,18
5	Gambar_5		0,0038	48,20		0,0055	22,12
6	Gambar_6		0,0064	48,20		0,0040	31,02
7	Gambar_7		0,0052	48,20		0,0041	24,00
8	Gambar_8		0,0050	48,20		0,0038	34,08

9	Gambar_9		0,0044	48,20		0,0035	32,57
10	Gambar_10		0,0085	48,20		0,0050	34,78
11	Gambar_11		0,0061	48,20		0,0070	26,11
12	Gambar_12		0,0041	48,20		0,0050	32,69
13	Gambar_13		0,0038	48,20		0,0054	29,49
14	Gambar_14		0,0061	48,20		0,0049	26,39
15	Gambar_15		0,0058	48,20		0,0039	31,67
16	Gambar_16		0,0062	48,20		0,0038	34,35
17	Gambar_17		0,0050	48,20		0,0040	29,39
18	Gambar_18		0,0052	48,20		0,0062	33,57
19	Gambar_19		0,0069	48,20		0,0042	29,10
20	Gambar_20		0,0049	48,20		0,0047	29,91
21	Gambar_21		0,0066	48,20		0,0034	30,94

22	Gambar_22		0,0033	48,20		0,0038	33,90
23	Gambar_23		0,0056	48,20		0,0037	33,01
24	Gambar_24		0,0069	48,20		0,0036	34,67
25	Gambar_25		0,0057	48,20		0,0040	32,67

Lampiran 4 Hasil pengolahan citra digital resolusi $\mathbf{256 \times 256}$ piksel menggunakan algoritma AES.

No	Nama Citra Digital	Hasil enkripsi algoritma RSA	Waktu proses enkripsi (detik)	Ukuran file enkripsi (kb)	Hasil dekripsi algoritma RSA	Waktu proses dekripsi (detik)	Ukuran file dekripsi (kb)
1	Gambar_1		0,0163	192,55		0,0120	58,27
2	Gambar_2		0,0148	192,55		0,0163	134,63
3	Gambar_3		0,0133	192,55		0,0122	60,80
4	Gambar_4		0,0147	192,55		0,0160	84,69
5	Gambar_5		0,0129	192,55		0,0094	64,55
6	Gambar_6		0,0148	192,55		0,0187	82,60
7	Gambar_7		0,0147	192,55		0,0204	114,79

8	Gambar_8		0,0142	192,55		0,0163	72,86
9	Gambar_9		0,0144	192,55		0,0179	93,58
10	Gambar_10		0,0159	192,55		0,0187	127,29
11	Gambar_11		0,0124	192,55		0,0130	62,69
12	Gambar_12		0,0173	192,55		0,0190	127,71
13	Gambar_13		0,0167	192,55		0,0190	99,14
14	Gambar_14		0,0144	192,55		0,0189	107,94
15	Gambar_15		0,0157	192,55		0,0174	126,15
16	Gambar_16		0,0169	192,55		0,0154	60,70
17	Gambar_17		0,0160	192,55		0,0188	121,58
18	Gambar_18		0,0191	192,55		0,0142	88,72
19	Gambar_19		0,0130	192,55		0,0049	33,88
20	Gambar_20		0,0150	192,55		0,0185	109,26

21	Gambar_21		0,0148	192,55		0,0191	103,72
22	Gambar_22		0,0161	192,55		0,0212	117,20
23	Gambar_23		0,0149	192,55		0,0175	89,19
24	Gambar_24		0,0143	192,55		0,0135	70,45
25	Gambar_25		0,0143	192,55		0,0128	85,96

Lampiran 5 Hasil MSE citra terenkripsi algoritma RSA dan AES

No	MSE citra enkripsi algoritma RSA		MSE citra enkripsi algoritma AES	
	128×128 Piksel	256×256 Piksel	128×128 Piksel	256×256 Piksel
1	10669,9	13832,4	17134,3	17065,1
2	8532,6	8813,9	8791,0	8919,0
3	7581,0	8393,8	8550,9	9833,3
4	10398,2	14881,1	10447,7	14171,8
5	12819,0	20284,7	11778,0	17019,6
6	8901,4	14794,6	12656,6	15831,8
7	8365,5	6703,7	8486,8	7523,7
8	9434,3	8930,0	9781,8	8529,6
9	8141,4	16442,0	8283,2	12391,5
10	11347,4	10432,4	11883,4	10222,5
11	8826,2	11287,3	8443,5	8710,4
12	9332,7	8021,7	9643,2	8104,7

13	8859,7	9523,4	10062,1	9635,4
14	7660,7	13202,4	8220,0	11717,7
15	8593,7	8163,2	8981,1	8138,9
16	7875,4	15836,7	7688,6	14255,3
17	9752,2	8735,8	9753,0	8829,7
18	10849,4	7165,4	12567,4	6922,1
19	10639,6	8178,6	12139,9	19583,3
20	9439,1	9885,9	9934,2	10770,6
21	10953,0	11681,4	12728,2	11491,4
22	10775,5	7092,6	10225,1	6931,7
23	8146,1	10707,7	8447,0	13044,5
24	10756,8	7854,6	9489,5	10781,5
25	12368,2	9827,6	11657,3	9081,9

Lampiran 6 *Script* enkripsi dan dekripsi algoritma RSA

```
import numpy as np
from PIL import Image
import random
import time  # Untuk mengukur waktu eksekusi

# Fungsi untuk menghitung gcd
def gcd(a, b):
    while b != 0:
        a, b = b, a % b
    return a

# Fungsi untuk menghitung mod inverse
def modinv(a, m):
    m0, x0, x1 = m, 0, 1
    if m == 1:
        return 0
    while a > 1:
        q = a // m
        m, a = a % m, m
        x0, x1 = x1 - q * x0, x0
```

```
    if x1 < 0:
        x1 += m0
    return x1

# Fungsi untuk generate kunci RSA
def generate_keypair(keysize):
    # Memilih dua bilangan prima p dan q (dalam skenario
nyata ini harus besar)
    p = 349  # bilangan prima
    q = 337  # bilangan prima
    n = p * q
    phi = (p - 1) * (q - 1)

    # Memilih e yang relatif prima terhadap phi(n)
    e = random.randrange(1, phi)
    while gcd(e, phi) != 1:
        e = random.randrange(1, phi)

    # Menghitung d
    d = modinv(e, phi)
    return ((e, n), (d, n))

# Fungsi untuk enkripsi piksel menggunakan kunci publik
def encrypt_rsa(public_key, plaintext):
    e, n = public_key
    return pow(int(plaintext), e, n)  # konversi ke int

# Fungsi untuk dekripsi piksel menggunakan kunci privat
def decrypt_rsa(private_key, ciphertext):
    d, n = private_key
    return pow(int(ciphertext), d, n)  # konversi ke int

# Fungsi untuk enkripsi citra
def encrypt_image_rsa(image_path, public_key):
    img = Image.open(image_path)
    img = img.resize((256, 256))  # Menggunakan resolusi
128x128 atau 256x256
    img_data = np.array(img)

    encrypted_data = np.zeros(img_data.shape,
dtype=np.uint32)
    for i in range(img_data.shape[0]):
        for j in range(img_data.shape[1]):
            r, g, b = img_data[i, j]
            encrypted_data[i, j, 0] =
encrypt_rsa(public_key, r)
            encrypted_data[i, j, 1] =
encrypt_rsa(public_key, g)
            encrypted_data[i, j, 2] =
encrypt_rsa(public_key, b)

    return img_data, encrypted_data  # Mengembalikan data
asli dan terenkripsi
```

```
# Fungsi untuk dekripsi citra
def decrypt_image_rsa(encrypted_data, private_key):
    decrypted_data = np.zeros(encrypted_data.shape,
dtype=np.uint8)
    for i in range(encrypted_data.shape[0]):
        for j in range(encrypted_data.shape[1]):
            r_enc, g_enc, b_enc = encrypted_data[i, j]
            decrypted_data[i, j, 0] =
decrypt_rsa(private_key, r_enc) % 256
            decrypted_data[i, j, 1] =
decrypt_rsa(private_key, g_enc) % 256
            decrypted_data[i, j, 2] =
decrypt_rsa(private_key, b_enc) % 256

    return decrypted_data  # Mengembalikan data
terdekripsi

# Fungsi untuk menghitung MSE
def calculate_mse(original_image_data,
decrypted_image_data):
    mse = np.mean((original_image_data -
decrypted_image_data) ** 2)
    return mse

# Input gambar JPG yang akan dienkripsi
image_path = '256/gambar_25.png'

# Generate kunci RSA
public_key, private_key = generate_keypair(8)

# Mulai waktu enkripsi
start_encrypt = time.time()

# Enkripsi citra
original_image_data, encrypted_image_data =
encrypt_image_rsa(image_path, public_key)
encrypted_image =
Image.fromarray(encrypted_image_data.astype(np.uint8))  #
Konversi ke uint8 untuk menyimpan gambar
encrypted_image.save('gambar_enkripsi_25.png')

# Akhir waktu enkripsi
end_encrypt = time.time()
print("Gambar berhasil dienkripsi.")
print(f"Waktu enkripsi: {end_encrypt - start_encrypt:.4f}
detik")

# Mulai waktu dekripsi
start_decrypt = time.time()

# Dekripsi citra
```

```
decrypted_image_data =
decrypt_image_rsa(encrypted_image_data, private_key)
decrypted_image =
Image.fromarray(decrypted_image_data.astype(np.uint8))  #
Konversi ke uint8 untuk menyimpan gambar
decrypted_image.save('gambar_dekripsi_25.png')

# Akhir waktu dekripsi
end_decrypt = time.time()
print("Gambar berhasil didekripsi.")
print(f"Waktu dekripsi: {end_decrypt - start_decrypt:.4f}
detik")

# Hitung MSE
mse = calculate_mse(original_image_data,
decrypted_image_data)
print(f"Nilai MSE antara citra asli dan citra terdekripsi:
{mse:.4f}")

# Menampilkan citra hasil dekripsi
decrypted_image.show()  # Untuk menampilkan citra hasil
dekripsi
```

Lampiran 7 *Script* enkripsi dan dekripsi algoritma AES

```
pip install pycryptodome pillow numpy

import os
import time
import numpy as np
from PIL import Image
from Crypto.Cipher import AES
from Crypto.Util.Padding import pad, unpad
from Crypto.Random import get_random_bytes

# Fungsi untuk membaca dan mengonversi gambar menjadi byte
array
def image_to_byte_array(image_path):
    image = Image.open(image_path)
    image = image.convert("RGB")  # Pastikan citra dalam
format RGB
    return image.tobytes(), image.size

# Fungsi untuk menyimpan byte array sebagai gambar
def byte_array_to_image(byte_array, image_size,
output_path):
    image = Image.frombytes('RGB', image_size, byte_array)
    image.save(output_path)

# Fungsi untuk enkripsi gambar dan simpan hasil enkripsi
sebagai citra visual
def encrypt_image_to_visual(image_path, key, output_path):
    # Baca gambar dan ubah menjadi byte array
```

```
    image_bytes, image_size =
image_to_byte_array(image_path)

    # AES mode CBC membutuhkan IV (Initialization Vector)
    iv = get_random_bytes(16)

    # Buat cipher AES dengan kunci dan IV
    cipher = AES.new(key, AES.MODE_CBC, iv)

    # Enkripsi citra
    encrypted_bytes = cipher.encrypt(pad(image_bytes,
AES.block_size))

    # Konversi hasil enkripsi menjadi array yang bisa
divisualisasikan sebagai gambar
    encrypted_array =
np.frombuffer(encrypted_bytes[:image_size[0] *
image_size[1] * 3], dtype=np.uint8)
    encrypted_array =
encrypted_array.reshape((image_size[1], image_size[0],
3))  # Sesuaikan ukuran citra

    # Buat gambar dari array terenkripsi dan simpan
    encrypted_image = Image.fromarray(encrypted_array)
    encrypted_image.save(output_path)

    print("Gambar telah berhasil dienkripsi.")

    # Simpan IV bersamaan dengan gambar
    return iv, encrypted_bytes

# Fungsi untuk dekripsi gambar dari hasil enkripsi visual
def decrypt_image_from_visual(encrypted_image_path, iv,
encrypted_bytes, key, output_path, image_size):
    # Buat cipher AES untuk dekripsi
    cipher = AES.new(key, AES.MODE_CBC, iv)

    # Dekripsi citra
    decrypted_bytes =
unpad(cipher.decrypt(encrypted_bytes), AES.block_size)

    # Simpan hasil dekripsi sebagai gambar
    byte_array_to_image(decrypted_bytes, image_size,
output_path)

    print("Gambar telah berhasil didekripsi.")

# Fungsi untuk menghitung MSE antara dua gambar
def calculate_mse(image_path_1, image_path_2):
    img1 =
np.array(Image.open(image_path_1).convert("RGB"),
dtype=np.float32)
```

```
    img2 =
np.array(Image.open(image_path_2).convert("RGB"),
dtype=np.float32)

    # Hitung MSE
    mse = np.mean((img1 - img2) ** 2)
    return mse

# Fungsi utama untuk enkripsi dan dekripsi citra
def process_image(image_path, encrypted_image_path,
decrypted_image_path, key):
    # Mulai hitung waktu enkripsi
    start_time = time.time()
    iv, encrypted_bytes =
encrypt_image_to_visual(image_path, key,
encrypted_image_path)
    encryption_time = time.time() - start_time

    # Mulai hitung waktu dekripsi
    image = Image.open(image_path)
    image_size = image.size
    start_time = time.time()
    decrypt_image_from_visual(encrypted_image_path, iv,
encrypted_bytes, key, decrypted_image_path, image_size)
    decryption_time = time.time() - start_time

    return encryption_time, decryption_time

# Main function
if __name__ == '__main__':
    # Tentukan path gambar
    image_path = 'gambar_1.png'  # masukkan citra digital
yang asli
    encrypted_image_path = 'gambar_enkripsi_1.png'
    decrypted_image_path = 'gambar_dekripsi_1.png'

    # Kunci 128-bit untuk AES
    key = get_random_bytes(32)

    # Proses enkripsi dan dekripsi
    encryption_time, decryption_time =
process_image(image_path, encrypted_image_path,
decrypted_image_path, key)

    # Cetak waktu enkripsi dan dekripsi
    print(f"Waktu enkripsi: {encryption_time:.4f} detik")
    print(f"Waktu dekripsi: {decryption_time:.4f} detik")

    # Hitung MSE antara gambar asli dan hasil dekripsi
    mse = calculate_mse(image_path, decrypted_image_path)
    print(f"Nilai MSE antara gambar asli dan gambar
dekripsi: {mse:.4f}")
```

Lampiran 8 *Script* MSE citra terenkripsi

```
import numpy as np
from PIL import Image

# Fungsi untuk menghitung MSE antara dua gambar
def calculate_mse(image_path_1, image_path_2):
    # Buka dan konversi gambar ke format RGB dan array
float32
    img1 =
np.array(Image.open(image_path_1).convert("RGB"),
dtype=np.float32)
    img2 =
np.array(Image.open(image_path_2).convert("RGB"),
dtype=np.float32)

    # Hitung MSE dengan mengurangi kedua gambar dan
mengkuadratkan perbedaannya
    mse = np.mean((img1 - img2) ** 2)
    return mse

# Main function untuk menjalankan perhitungan MSE Enkripsi
if __name__ == '__main__':
    # Tentukan path gambar asli dan gambar yang akan
dibandingkan
    image_path_1 = 'gambar_1.png'  # Ganti dengan path
gambar asli
    image_path_2 = 'gambar_enkripsi_1.png’  # Ganti dengan
path gambar hasil enkripsi

    # Hitung MSE
    mse_value = calculate_mse(image_path_1, image_path_2)

    # Cetak hasil MSE
    print(f”Nilai MSE: {mse_value:.4f}”)
```

RIWAYAT HIDUP

Rifqi Muhamad Sabilal Karim lahir di Kabupaten Kediri pada 6 Juni 2001. Memiliki nama panggilan Rifqi. Tempat tinggal di Jalan Brawijaya 140 RT 01 RW 11, Desa Tulungrejo, Kecamatan Pare, Kabupaten Kediri, Jawa Timur. Merupakan anak pertama dari tiga bersaudara dari pasangan Bapak Moh. Bachrudin dan Ibu Elvi Sayyidah Faiqah. Pendidikan yang pernah ditempuh yaitu TK Tauladan Pare. Kemudian melanjutkan sekolah di SDI NU Pare dan lulus pada tahun 2013. Menempuh pendidikan di Madrasah Tsanawiyah Negeri Jombang Kauman dan lulus pada tahun 2016. Melanjutkan pendidikan di Sekolah Menengah Atas Trensains Tebuireng Jombang dan lulus pada tahun 2019. Pada tahun 2019 melanjukan studi ke jenjang pendidikan strata 1 di Universitas Islam Negeri Maulana Malik Ibrahim Malang dan mengambil program studi Matematika di Fakultas Sains dan Teknologi. Kegiatan yang pernah diikuti adalah Praktek Kerja Lapangan (PKL) di Badan Pendapatan Daerah bagian BPHTB pada tahun 2023.

KEMENTERIAN AGAMA RI
UNIVERSITAS ISLAM NEGERI
MAULANA MALIK IBRAHIM MALANG
FAKULTAS SAINS DAN TEKNOLOGI
Jl. Gajayana No.50 Dinoyo Malang Telp. / Fax. (0341)558933

BUKTI KONSULTASI SKRIPSI

Nama : Rifqi Muhamad Sabilal Karim
NIM : 19610111
Fakultas/Jurusan : Sains dan Teknologi/Matematika
Judul Skripsi : Perbandingan Kinerja Algoritma RSA dan AES dalam Enkripsi dan Dekripsi Citra Digital
Pembimbing I : Muhammad Khudzaifah, M.Si
Pembimbing II : Dr. Fachrur Rozi, M.Si

No	Tanggal	Hal	Tanda Tangan
1.	6 September 2024	Konsultasi Bab I, II, dan III	1.
2.	9 September 2024	Konsultasi Revisi Bab I, II, dan III	2.
3.	11 September 2024	Konsultasi Kajian Agama	3.
4.	14 September 2024	Konsultasi Revisi Kajian Agama	4.
5.	16 September 2024	ACC Kajian Agama Bab I dan II	5.
6.	17 September 2024	ACC Bab I, II, dan III	6.
7.	25 September 2024	ACC Seminar Proposal	7.
8.	8 Oktober 2024	Konsultasi Revisi Seminar Proposal	8.
9.	14 Oktober 2024	Konsultasi Bab IV	9.
10.	15 Oktober 2024	Konsultasi Bab IV dan V	10.
11.	18 Oktober 2024	Konsultasi Kajian Agama Bab IV	11.
12.	20 Oktober 2024	ACC Kajian Agama Bab IV	12.
13.	24 Oktober 2024	ACC Bab IV dan V	13.
14.	25 Oktober 2024	ACC Seminar Hasil	14.
15.	19 November 2024	Konsultasi Revisi Seminar Hasil	15.

KEMENTERIAN AGAMA RI
UNIVERSITAS ISLAM NEGERI
MAULANA MALIK IBRAHIM MALANG
FAKULTAS SAINS DAN TEKNOLOGI
Jl. Gajayana No.50 Dinoyo Malang Telp. / Fax. (0341)558933

No	Tanggal	Hal	Tanda Tangan
16.	21 November 2024	ACC Matriks Revisi Seminar Hasil	16.
17.	5 Desember 2024	ACC Sidang Skripsi	17.
18.	12 Desember 2024	ACC Keseluruhan	18.

Malang, 12 Desember 2024
Mengetahui,
Ketua Program Studi Matematika

Dr. Elly Susanti, M.Sc.
NIP. 19741129 200012 2 005