KOMBINASI PERLINDUNGAN DATA MATERIAL SAP MENGGUNAKAN ALGORITMA AES 128 BIT DAN REVERSE CIPHER DI PT INDONESIA COMNET PLUS

SKRIPSI

OLEH:
DWIPA OKI DEWANTARA
NIM. 210601110095

PROGRAM STUDI MATEMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2025

KOMBINASI PERLINDUNGAN DATA MATERIAL SAP MENGGUNAKAN ALGORITMA AES 128 BIT DAN REVERSE CIPHER DI PT INDONESIA COMNET PLUS

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
Dwipa Oki Dewantara
NIM. 210601110095

PROGRAM STUDI MATEMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHI MALANG
2025

KOMBINASI PERLINDUNGAN DATA MATERIAL SAP MENGGUNAKAN ALGORITMA AES 128 BIT DAN *REVERSE CIPHER* DI PT INDONESIA COMNET PLUS

SKRIPSI

Oleh
Dwipa Oki Dewantara
NIM. 210601110095

Telah Disetujui Untuk Diuji

Malang, 26 Mei 2025

Dosen Pembimbing I	Dosen Pembimbing II
Muhammad Khudzaifah, M.Si.	Ach. Nashichuddin, M.A.
NIPPPK. 19900511 202321 1029	NIP 19730705 200003 1 002

Mengetahui,
Ketua Program Studi Matematika

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

KOMBINASI PERLINDUNGAN DATA MATERIAL SAP MENGGUNAKAN ALGORITMA AES 128 BIT DAN *REVERSE CIPHER* DI PT INDONESIA COMNET PLUS

SKRIPSI

Oleh
Dwipa Oki Dewantara
NIM. 210601110095

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

Tanggal 16 Juni 2025

Ketua Penguji	: Hisyam Fahmi, M.Kom	.....
Anggota Penguji 1	: Mohammad Nafie Jauhari, M.Si.	.....
Anggota Penguji 2	: Muhammad Khudzaifah, M.Si.	.....
Anggota Penguji 3	: Ach. Nashichuddin, M.A.	.....

Mengetahui,
Ketua Program Studi Matematika

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

PERNYATAN KEASLIAN TULISAN

Saya yang bertanda tangan dibawah ini:

Nama	: Dwipa Oki Dewantara
NIM	: 210601110095
Program Studi	: Matematika
Fakultas	: Sains dan Teknologi
Judul Skripsi	: Kombinasi Perlindungan Data Material SAP Mengguna kan Algoritma AES 128 Bit dan Reverse Chiper di PT Indonesia Comnet Plus

Menyatakan dengan sebenarnya bahwa skripsi yang saya tulis ini benarbenar 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 cuplikan pada daftar pustaka. Apabila di kemudian hari terbukti atau dapat dibuktikan skripsi ini hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 16 Juni 2025
Yang Membuat Pernyataan,

Dwipa Oki Dewantara
NIM. 210601110095

MOTTO

“Tetaplah optimis, dengan ketidak optimisan itu sendiri”

KATA PENGANTAR

Alhamdulillahirabbil'alamin,

Segala puji dan syukur penulis panjatkan kehadirat Allah SWT atas segala limpahan rahmat, taufik, dan hidayah-Nya, sehingga penulis dapat menyelesaikan proposal penelitian ini dengan judul "Kombinasi Perlindungan Data Sensitif Menggunakan Algoritma AES 128 bit dan *Reverse Cipher* di PT.Indonesia Comnet Plus" sebagai bagian dari persiapan dalam rangkaian kegiatan akademik untuk memperoleh gelar Sarjana pada Program Studi Matematika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim.

Dalam proses penyusunan proposal ini, penulis banyak menerima dukungan, bimbingan, serta bantuan dari berbagai pihak. Oleh karena itu, pada kesempatan ini, penulis ingin menyampaikan rasa terima kasih yang sebesar-besarnya kepada:

1. Prof. Dr. H. M. Zainuddin, M.A., selaku Rektor Universitas Islam Negeri Maulana Malik Ibrahim, yang telah memberikan kesempatan untuk menimba ilmu di universitas ini.
2. Prof. Dr. Hj. Sri Harini, M.Si, selaku Dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim, atas segala arahan dan kebijakan yang telah memfasilitasi kegiatan akademik ini.
3. Dr. Elly Susanti, M.Sc, selaku Ketua Program Studi Matematika, Universitas Islam Negeri Maulana Malik Ibrahim, atas bimbingan, dukungan, dan motivasinya selama proses studi.
4. Muhammad Khudzaifah, M.Si, selaku dosen pembimbing I, atas bimbingan, saran, dan arahannya dalam penyusunan proposal ini.
5. Ach. Nasichuddin, M.A, selaku dosen pembimbing II, yang telah memberikan masukan berharga dalam proses penyusunan proposal ini.
6. Mohammad Nafie Jauhari, M.Si., selaku Penguji Utama dalam Ujian Skripsi.
7. Hisyam Fahmi, M.Kom, selaku Ketua Penguji dalam Ujian Skripsi.

8. Seluruh dosen Program Studi Matematika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim, atas ilmu dan bimbingan yang telah diberikan selama masa studi.
9. Orang tua dan seluruh keluarga, atas doa, dukungan, serta dorongan yang tak pernah putus.
10. Seluruh mahasiswa angkatan 2021, atas kebersamaan, dukungan, dan semangatnya selama masa studi.
11. Dhea Adrika Zahro Asyhari selaku teman perjuangan saya untuk menyelesaikan skripsi saya.

Akhir kata, penulis mengucapkan terima kasih yang sebesar-besarnya atas segala perhatian, bantuan, dan dukungan dari berbagai pihak yang telah berkontribusi dalam penyusunan karya ini. Penulis menyadari bahwa tanpa bantuan dan dukungan tersebut, karya ini tidak akan dapat terselesaikan dengan baik. Semoga tulisan ini dapat memberikan manfaat bagi pembaca dan menjadi kontribusi positif dalam pengembangan ilmu pengetahuan.

Malang, 16 Juni 2025

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

ABSTRAK

Dewantara, Dwipa Oki. 2025. Kombinasi algoritma Advanced Encryption Standard (AES) 128-bit dan *Reverse Cipher* untuk perlindungan data material SAP di PT Indonesia Comnet Plus. Skripsi. Program Studi Matematika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing: (1) Muhammad Khudzaifah, M.Si. (2) Ach. Nashichuddin, M.A.

Kata Kunci: AES 128 bit, *Reverse Cipher*, enkripsi, material SAP, keamanan data.

Penelitian ini mengeksplorasi kombinasi Advanced Encryption Standard (AES) 128-bit dan *Reverse Cipher* untuk mengamankan data material SAP (termasuk GroupDesc, MaterialNumber, Catalog Data, dan Material Description) di PT Indonesia Comnet Plus. Kami menganalisis efisiensi enkripsi dan *Avalanche Effect* dari algoritma ini. Hasil menunjukkan proses enkripsi dan dekripsi yang efisien, dengan rata-rata waktu 1.07 detik untuk enkripsi dan 1.00 detik untuk dekripsi file hingga 600 KB. Namun, *Avalanche Effect* yang dihasilkan untuk *padding Fixed Length* menunjukkan rata-rata 33.31% (berkisar antara 24.36% hingga 37.00%). Sedangkan untuk *padding* PKCS#7, *Avalanche Effect* menunjukkan rata-rata 49.93% (berkisar antara 45.96% hingga 53.39%). Meskipun PKCS#7 mendekati standar ideal 50%, nilai *Avalanche Effect* keseluruhan masih dianggap kurang optimal dalam difusi acak. Hal ini disebabkan oleh sifat deterministik *Reverse Cipher* dan penggunaan mode Electronic Codebook (ECB) pada AES yang membatasi penyebaran bit. Oleh karena itu, kami merekomendasikan penggunaan mode AES yang lebih kuat seperti Cipher Block Chaining (CBC) atau Galois/Counter Mode (GCM) untuk meningkatkan keamanan. Studi ini memberikan kontribusi penting dalam bidang kriptografi di sektor telekomunikasi dan industri lainnya.

ABSTRACT

Dewantara, Dwipa Oki. 2025. Combination of Advanced Encryption Standard (AES) 128-bit and *Reverse Cipher* Algorithms for SAP Material Data Protection at PT Indonesia Comnet Plus. Thesis. Mathematics Study Program, Faculty of Science and Technology, Universitas Islam Negeri Maulana Malik Ibrahim Malang. Supervisors: (1) Muhammad Khudzaifah, M.Si. (2) Ach. Nashichuddin, M.A.

Keywords: AES 128 bit, *Reverse Cipher*, encryption, SAP material, data security.

This research explores the combination of 128-bit Advanced Encryption Standard (AES) and *Reverse Cipher* to secure SAP material data (including GroupDesc, MaterialNumber, Catalog Data, and Material Description) at PT Indonesia Comnet Plus. We analyzed the encryption efficiency and *Avalanche Effect* of these algorithms. The results show an efficient encryption and decryption process, with an average time of 1.07 seconds for encryption and 1.00 seconds for decryption of files up to 600 KB. However, the resulting *Avalanche Effect* for *Fixed Length padding* shows an average of 33.31% (ranging from 24.36% to 37.00%). As for PKCS#7 *padding*, the *Avalanche Effect* shows an average of 49.93% (ranging from 45.96% to 53.39%). Although PKCS#7 is close to the ideal standard of 50%, the overall *Avalanche Effect* value is still considered suboptimal in random diffusion. This is due to the deterministic nature of ReverseCipher and the use of Electronic Codebook (ECB) mode in AES which limitsbit spread. Therefore, we recommend the use of stronger AES modes such as Cipher Block Chaining (CBC) or Galois/Counter Mode (GCM)to improve security. This study makes an important contribution to the field of cryptography in the telecommunications sector and other industries.

مستخلص البحث

ديوانتارا، دوبيا أوكي .٢٠٢٥ .دمج خوارزمية التشفير المتقدم_(إيه.إي.إس) ١٢٨-بت وخوارزمية التشفير العكسي لحماية بيانات المواد في نظام إس.إيه.بي بشركة بي.تي إندونيسيا كومنت بلس .أطروحة .برنامج :الرياضيات، كلية العلوم والتكنولوجيا، الجامعة الإسلامية الحكومية مولانا مالك إبراهيم مالانج .المشرفون .)١ (محمد حُذيفة، ماجستير في علوم الحاسوب،)٢ (الأستاذ ناشيش الدين، ماجستير

الكلمات المفتاحية :خوارزمية التشفير المتقدم ١٢٨-بت، التشفير العكسي، التشفير، بيانات مواد .إس.إيه.بي، أمن البيانات

تهدف هذه الدراسة إلى استكشاف دمج خوارزمية التشفير المتقدم)إيه.إي.إس (١٢٨-بت وخوارزمية التشفير العكسي لتأمين بيانات المواد في نظام إس.إيه.بي)بما في ذلك وصف المجموعة، ورقم المادة، وبيانات الكتالوج، ووصف المادة (في شركة بي.تي إندونيسيا كومنت بلس .تم تحليل كفاءة التشفير وتأثير الانهيار الثلجي لهذه الخوارزميات .أظهرت النتائج كفاءة في عملية التشفير وفك التشفير، حيث بلغ متوسط الوقت ١,٠٧ ثانية للتشفير و١,٠٠ ثانية لفك التشفير للملفات التي تصل إلى ٦٠٠ كيلوبايت .ومع ذلك، كان ٪متوسط تأثير الانهيار الثلجي لطريقة الحشو ذات الطول الثابت ٣٣,٣١٪)تتراوح بين ٢٤,٣٦ ٪و٣٧,٠٠٪(، بينما بلغ متوسط التأثير لطريقة بي.كي.سي.إس#٧ نسبة ٤٩,٩٣٪)تتراوح بين ٤٥,٩٦ و٥٣,٣٩٪ .(على الرغم من أن طريقة بي.كي.سي.إس#٧ تقترب من المعيار المثالي البالغ ٥٠٪، إلا أن قيمة تأثير الانهيار الثلجي لا تزال دون المستوى الأمثل في الانتشار العشوائي .يُعزى ذلك إلى الطبيعة الحتمية ،لخوارزمية التشفير العكسي واستخدام وضع دفتر الشيفرة الإلكتروني)إي.سي.بي (في خوارزمية إيه.إي.إس الذي يحد من انتشار البتات .لذلك، نوصي باستخدام أوضاع أقوى مثل وضع سلسلة الكتل المشفّرة)سي.بي.سي (أو وضع جالوا/عداد)جي.سي.إم (لتعزيز الأمان .تساهم هذه الدراسة بشكل مهم في مجال .التشفير في قطاع الاتصالات والصناعات الأخرى

BAB I

PENDAHULUAN

1.1 Latar Belakang

Di era digital saat ini, kemajuan teknologi informasi dan komunikasi telah memberikan dampak signifikan terhadap cara manusia berinteraksi, menjalankan bisnis, dan mengelola informasi. Dengan meningkatnya volume data yang dihasilkan dan dipertukarkan secara daring, perlindungan terhadap data material dalam sistem SAP ICON+ menjadi isu yang sangat krusial. Data material mencakup informasi strategis seperti klasifikasi barang, harga satuan, ketersediaan stok, lokasi penyimpanan, hingga riwayat transaksi logistik dan pengadaan. Jika data ini jatuh ke tangan yang tidak berwenang, dapat menimbulkan kerugian operasional, finansial, maupun reputasional bagi perusahaan. Ancaman terhadap keamanan data SAP juga semakin meningkat seiring dengan kemajuan teknologi digital. Serangan siber seperti peretasan, malware, dan phishing menjadi semakin kompleks dan sulit dideteksi, sehingga meningkatkan risiko kebocoran dan manipulasi data penting.

PT Indonesia Comnet Plus (ICON+) sebagai perusahaan yang bergerak di bidang teknologi informasi dan komunikasi bertanggung jawab atas pengelolaan serta keamanan informasi bisnis, termasuk data material dalam sistem SAP. Kebocoran informasi sensitif seperti identitas barang, estimasi nilai transaksi, hingga logistik internal dapat berdampak besar terhadap stabilitas operasional perusahaan (Apriansyah, Saputra, & Khoir, 2022). Oleh karena itu, perlindungan terhadap data material harus menjadi prioritas utama dalam tata kelola keamanan informasi perusahaan. Penelitian ini menggunakan data material SAP ICON+

sebagai objek pengujian, yang memiliki nilai strategis dalam sistem *Enterprise Resource Planning* (ERP). Tujuan penelitian ini adalah untuk menganalisis efektivitas algoritma enkripsi *Advanced Encryption Standard* (AES) 128-bit dan *Reverse Cipher* dalam melindungi data pada lingkungan sistem informasi perusahaan, serta memberikan gambaran mengenai kinerja kedua algoritma dalam menjaga kerahasiaan dan integritas data material pada sistem operasional ICON+.

Algoritma enkripsi AES 128 bit adalah salah satu metode yang paling banyak digunakan untuk melindungi data sensitif. AES bekerja dengan cara mengubah data asli (*plaintext*) menjadi format yang tidak dapat dibaca (*ciphertext*) menggunakan kunci enkripsi. Keunggulan AES terletak pada kecepatan dan efisiensinya dalam memproses data, serta tingkat keamanannya yang tinggi, sehingga sulit untuk dipecahkan oleh pihak yang tidak berwenang. Dengan menggunakan AES 128 bit, perusahaan dapat memastikan bahwa data sensitif mereka terlindungi dengan baik (Setiawan & Mufarrihah, 2024).

Reverse Cipher adalah metode enkripsi tambahan yang dapat digunakan bersamaan dengan AES untuk meningkatkan keamanan data. Metode ini bekerja dengan cara membalikkan urutan data sebelum atau setelah proses enkripsi, sehingga menambah lapisan perlindungan ekstra. Dengan menggabungkan AES dan *Reverse Cipher*, data sensitif akan lebih sulit untuk diakses oleh pihak yang tidak berwenang, karena mereka harus melewati dua lapisan enkripsi yang berbeda (Harris & Sari, 2024).

Material SAP ICON+ merupakan komponen vital dalam sistem pengelolaan sumber daya perusahaan (*Enterprise Resource Planning*/ERP) yang diterapkan oleh PT. Indonesia Comnet Plus (ICON+), anak perusahaan PLN yang bergerak di

bidang penyediaan layanan telekomunikasi. Sistem ini menggunakan SAP (*Systems, Applications, and Products in Data Processing*), yaitu perangkat lunak ERP terintegrasi yang banyak digunakan oleh perusahaan besar untuk mengelola proses bisnis secara otomatis dan efisien. Data material dalam sistem SAP mencakup informasi penting mengenai inventaris, klasifikasi barang, harga, ketersediaan stok, lokasi penyimpanan, serta riwayat transaksi yang berkaitan langsung dengan proses operasional dan pengambilan keputusan strategis perusahaan. Keakuratan dan kerahasiaan data material ini menjadi sangat krusial, mengingat informasi tersebut dapat memengaruhi efisiensi rantai pasok, perencanaan anggaran, hingga kualitas layanan yang diberikan kepada pelanggan. Dalam konteks keamanan informasi, data material SAP termasuk dalam kategori data sensitif yang rentan terhadap risiko kebocoran, manipulasi, dan akses tidak sah, yang dapat menimbulkan kerugian operasional maupun reputasional bagi perusahaan. Oleh karena itu, penelitian ini difokuskan pada pengamanan data material SAP dengan menerapkan kombinasi algoritma enkripsi *Advanced Encryption Standard* (AES) 128-bit dan *Reverse Cipher*. Kombinasi metode ini diharapkan mampu meningkatkan lapisan keamanan data, sehingga informasi yang tersimpan maupun yang ditransmisikan melalui sistem tetap terjaga kerahasiaannya dan hanya dapat diakses oleh pihak yang berwenang.

Studi sebelumnya menunjukkan keandalan AES 128 Bit dalam berbagai industri. Sebagai contoh, dalam penelitian mereka di PT. Mitsubishi Electric Indonesia, (Sitorus, Nugroho, & Pane, 2021) menunjukkan bahwa menggunakan AES 128 Bit dapat dengan sukses melindungi data transaksi penjualan dari kebocoran informasi. (Prayudha, 2019) juga menemukan bahwa AES 128 Bit

berhasil menjaga data keuangan PT. Capella Medan. Selain itu, penelitian sebelumnya telah melihat bagaimana algoritma *Reverse Cipher* dapat meningkatkan keamanan data (Santoso, Riski, & Kamsyakawuni, 2018), penelitian ini menguji seberapa baik algoritma *Reverse Cipher* melindungi data dari gangguan yang tidak diinginkan. Meskipun tidak sekuat algoritma enkripsi lainnya, seperti AES, mereka menemukan bahwa algoritma ini dapat menyediakan lapisan perlindungan tambahan.

Dalam penelitian ini, variabel yang akan dianalisis adalah efektivitas kombinasi algoritma AES 128 Bit dan *Reverse Cipher* dalam melindungi data sensitif di PT. Indonesia Comnet Plus. Algoritma AES 128 Bit dikenal sebagai salah satu metode enkripsi yang paling aman dan banyak digunakan di seluruh dunia, sementara *Reverse Cipher* menawarkan pendekatan tambahan yang dapat meningkatkan kompleksitas enkripsi. Dengan menggabungkan kedua metode ini, diharapkan dapat tercipta sistem perlindungan data yang lebih kuat dan efektif.

Dalam perpektif Islam, menjaga keamanan data adalah bagian dari kewajiban menjaga amanah. Al-Quran menegaskan pentingnya menjaga amanah dalam Surah An-Nisa ayat 58:

اِنَّ اللّٰهَ يَأْمُرُكُمْ اَنْ تُؤَدُّوا الْاَمٰنٰتِ اِلٰٓى اَهْلِهَاۙ وَاِذَا حَكَمْتُمْ بَيْنَ النَّاسِ اَنْ تَحْكُمُوْا ۞ بِالْعَدْلِۗ اِنَّ اللّٰهَ نِعِمَّا يَعِظُكُمْ بِهٖۗ اِنَّ اللّٰهَ كَانَ سَمِيْعًاۢ بَصِيْرًا

"Sesungguhnya Allah menyuruh kamu menyampaikan amanah kepada pemiliknya. Apabila kamu menetapkan hukum di antara manusia, hendaklah kamu tetapkan secara adil. Sesungguhnya Allah memberi pengajaran yang paling baik kepadamu. Sesungguhnya Allah Maha Mendengar lagi Maha Melihat (QS. An-Nisa/4:58)"

Menurut Tafsir Al-Misbah oleh M. Quraish Shihab (Shihab, 2002), Al-Qur'an mengungkapkan keburukan orang Yahudi, seperti mengabaikan janji Allah

untuk mengamalkan kitab suci. Al-Qur'an juga mengingatkan umat Muslim untuk tidak mengikuti jejak mereka. Allah memerintahkan agar amanah, baik kepada-Nya maupun sesama manusia, ditunaikan dengan baik dan tepat waktu. Selain itu, Allah menginstruksikan untuk menetapkan hukum secara adil, tanpa memihak, dan tidak menganiaya orang lain. Allah mengawasi segala tindakan dan ucapan kita. Amanah, yang berlawanan dengan khianat, hanya diberikan kepada orang yang mampu melindunginya. Dalam ajaran agama, amanah adalah dasar iman, seperti yang dinyatakan Nabi Muhammad SAW, "Tidak ada iman bagi orang yang tidak memiliki amanah." Amanah penting dalam interaksi manusia karena menciptakan kepercayaan, yang membawa ketenangan dan keyakinan yang kuat.

Dengan meningkatnya ancaman terhadap data sensitif, penelitian ini sangat relevan bagi PT. Indonesia Comnet Plus, di mana perlindungan data menjadi bagian penting dari strategi bisnis untuk membangun kepercayaan pelanggan. Penelitian ini bertujuan untuk mengevaluasi efektivitas kombinasi algoritma AES 128 bit dan *Reverse Cipher* dalam melindungi data sensitif, khususnya data *Backbone Access Link* (BAKL) dalam infrastruktur jaringan telekomunikasi ICON+. Hasil analisis diharapkan memberikan rekomendasi praktis untuk langkah-langkah keamanan yang lebih baik dan berkontribusi pada pengembangan ilmu pengetahuan di bidang kriptografi serta memberikan referensi bagi industri lain yang menghadapi tantangan serupa.

1.2 Rumusan Masalah

Berdasarkan latar belakang masalah di atas, manfaat dari penelitian ini sebagai berikut:

1. Bagaimana kinerja kombinasi algoritma AES 128 Bit dan *Reverse Cipher* dalam mengenkripsi *plaintext* dari segi waktu pemrosesan kriptografi dan efisiensi algoritma?
2. Sejauh mana kombinasi algoritma ini mempengaruhi *Avalanche Effect* dalam proses enkripsi dan dekripsi?

1.3 Tujuan Penelitian

Berdasarkan rumusan masalah di atas, manfaat dari penelitian ini sebagai berikut:

1. Menganalisis kinerja kombinasi algoritma AES 128 Bit dan *Reverse Cipher* dalam mengenkripsi data Material SAP, khususnya dalam aspek waktu pemrosesan dan efisiensi algoritma.
2. Mengukur dan mengevaluasi dampak kombinasi algoritma AES 128 Bit dan *Reverse Cipher* terhadap *Avalanche Effect* dalam proses enkripsi dan dekripsi.

1.4 Manfaat Penelitian

Berdasarkan tujuan masalah di atas, manfaat dari penelitian ini sebagai berikut:

1. Penelitian ini memperdalam pemahaman tentang efektivitas kombinasi algoritma AES 128 Bit dan *Reverse Cipher* dalam mengamankan data sensitif, serta memberikan perspektif baru dalam penerapan kriptografi pada sektor telekomunikasi.

2. Penelitian ini memberikan wawasan lebih dalam tentang bagaimana algoritma AES 128 Bit dan *Reverse Cipher* dapat diintegrasikan dalam praktik pengamanan data di perusahaan, yang memperkaya literatur tentang penerapan enkripsi dalam dunia industri.
3. Memberikan solusi nyata bagi PT. Indonesia Comnet Plus untuk memperkuat perlindungan data sensitif menggunakan algoritma enkripsi yang terbukti efektif.
4. Hasil penelitian ini dapat diterapkan di perusahaan telekomunikasi lainnya, memberikan dasar praktis bagi mereka untuk meningkatkan sistem keamanan data.

1.5 Batasan Masalah

Untuk mengarahkan agar penelitian ini tidak menyimpang dari tujuan, maka batasan masalah dalam penelitian ini sebagai berikut:

1. Penelitian ini terbatas pada dua algoritma enkripsi, yaitu AES 128 bit dan *Reverse Cipher*. Algoritma enkripsi lainnya tidak akan dibahas atau dianalisis dalam penelitian ini.
2. Penelitian ini hanya akan fokus pada enkripsi *plaintext* berupa *GroupDesc, MaterialNumber, Catalog Data, MaterialDesc,* dan tidak akan mencakup jenis data lain seperti Stock/Non Stock, Material Group, Brand, dan jenis data lainnya
3. Analisis kinerja algoritma dalam penelitian ini akan difokuskan pada parameter tertentu, seperti waktu kriptografi, efek *avalanche*, dan efisiensi

penggunaan algoritma. Aspek lain dari keamanan algoritma, seperti ketahanan terhadap serangan tertentu, tidak akan dianalisis.

4. Penelitian ini akan lebih menekankan pada kombinasi kinerja dan tidak akan mencakup analisis mendalam terkait tingkat keamanan algoritma AES dan *Reverse Cipher*, seperti ketahanan terhadap serangan tertentu (misalnya, *brute force* atau serangan *side-channel*).

1.6 Definsi Istilah

Terdapat beberapa istilah yang digunakan pada penelitian ini sebagai berikut:

1. *Advanced Encryption Standard* (AES) 128 Bit : Algoritma kriptografi simetris yang berfungsi untuk melindungi data dengan mengubahnya menjadi bentuk terenkripsi (*Ciphertext*) menggunakan kunci sepanjang 128 bit. Algoritma ini terkenal dengan kecepatan dan keamanannya, sering digunakan dalam sektor seperti telekomunikasi dan keuangan untuk mengamankan informasi sensitif.
2. *Enkripsi* : Proses mengubah informasi *plaintext* (yang dapat dibaca) menjadi *Ciphertext* (yang tidak dapat dipahami) untuk menjaga kerahasiaannya. Hanya orang yang memiliki kunci enkripsi yang tepat yang dapat mengubah *Ciphertext* menjadi *plaintext*.
3. *Dekripsi* : Proses kebalikan dari enkripsi, yaitu mengembalikan *Ciphertext* menjadi bentuk aslinya (*plaintext*) yang bisa

dibaca. Dekripsi memerlukan kunci yang sesuai dengan kunci yang digunakan saat proses enkripsi.

4. Material SAP : Material SAP ICON+ merupakan komponen vital dalam sistem pengelolaan sumber daya perusahaan (Enterprise Resource Planning/ERP) yang diterapkan oleh PT. Indonesia Comnet Plus (ICON+), anak perusahaan PLN yang bergerak di bidang penyediaan layanan telekomunikasi. Sistem ini menggunakan SAP (*Systems, Applications, and Products in Data Processing*), yaitu perangkat lunak ERP terintegrasi yang banyak digunakan oleh perusahaan besar untuk mengelola proses bisnis secara otomatis dan efisien.
5. Keamanan Data : Serangkaian upaya untuk melindungi data dari risiko seperti pencurian, perubahan, atau akses yang tidak sah. Hal ini mencakup penggunaan metode seperti enkripsi untuk memastikan bahwa data tetap aman, utuh, dan hanya dapat diakses oleh pihak yang berwenang.
6. Transposisi : Kelemahan atau celah dalam sistem yang bisa dimanfaatkan oleh pihak yang tidak berwenang untuk mengakses atau merusak data. Kerentanan ini bisa muncul karena masalah teknis, kesalahan pengaturan, atau ketidaksempurnaan dalam langkah-langkah keamanan.

7. Integritas Data : Mengacu pada jaminan bahwa data tidak diubah atau dirusak oleh pihak yang tidak berwenang selama proses penyimpanan atau pengiriman. Algoritma enkripsi seperti AES membantu menjaga integritas data dengan memastikan bahwa hanya pihak yang memiliki otoritas yang dapat mengakses atau mengubah data tersebut.
8. Efisiensi : Efisiensi sistem mengacu pada bagaimana sistem, seperti algoritma enkripsi, memanfaatkan sumber daya (seperti identitas, waktu, dll) dalam memproses data. Sistem yang efisien adalah sistem yang dapat bekerja dengan baik tanpa membebani atau menghabiskan sumber daya secara berlebihan.
9. *Reverse Cipher* : Algoritma kriptografi klasik seperti *Vigenere cipher*, tetapi kunci yang digunakan ditransposisi kebalikan, atau *permutation reversed*, terlebih dahulu.

BAB II

KAJIAN TEORI

2.1 Teori Pendukung

2.1.1 Kriptografi

Istilah *crypto* yaitu rahasia dan *grapho* yaitu menulis. *Cryptography* merupakan seni dan ilmu penyandian yang mempunyai tujuan yaitu menjaga keamanan dan kerahasiaan pesan yang dikirim. Sejarah menunjukkan bahwa sejak 4000 tahun yang lalu, raja-raja Mesir telah menggunakan kriptografi untuk mengirimkan pesan rahasia kepada panglima perangnya melalui kurir-kurir mereka (Ariyus, 2008).

Kriptografi berbicara tentang bagaimana informasi atau pesan tetap aman saat dikirim dari pengirim ke penerima. Kriptografi merupakan bidang keilmuan yang terdiri dari teknik enkripsi yang digunakan untuk mengacak pesan sehingga tidak dapat dipahami. Secara umum, penyembunyian informasi ditujukan untuk orang yang diizinkan. (Wibowo,, Susanto, & Karel, 2011)

Prinsip-prinsip yang mendasari kriptografi yakni:

1. Kesetiaan (kerahasiaan) yaitu prinsip dasar kriptografi. Ini berarti bahwa pesan yang dikirim tetap rahasia dan tidak dapat diketahui oleh orang yang tidak memiliki izin kecuali pengirim dan penerima. Untuk mencapai tujuan ini, biasanya digunakan algoritma matematis yang memiliki kemampuan untuk mengubah data sehingga sulit dibaca dan dipahami.

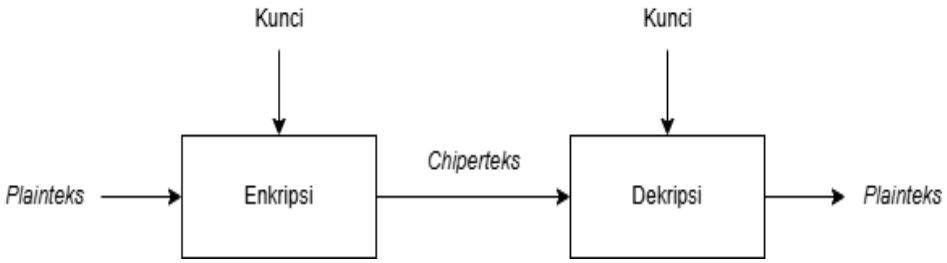
2. Integritas data (keutuhan data) adalah layanan yang dapat mengenali atau mengidentifikasi manipulasi data yang tidak sah oleh pihak lain, seperti penghapusan, pengubahan, atau penambahan.
3. *Authentication* adalah layanan yang berkaitan dengan identifikasi.
4. *Non-repudiation* (anti-penyangkalan) merupakan kemampuan untuk mencegah suatu pihak untuk menyangkal tindakan yang dilakukan sebelumnya, seperti menyangkal bahwa pesan tersebut berasal dari dirinya sendiri.

Kriptografi modern berbeda dengan kriptografi klasik, yang berfokus pada kerahasiaan algoritma yang digunakan; kriptografi modern berfokus pada kerahasiaan kunci yang digunakan oleh pemakai algoritma, yang memungkinkan algoritma menyebar ke seluruh masyarakat. Bidang kriptografi menggunakan istilah berikut:

1. *Plaintext* (M) adalah pesan yang akan dikirimkan dan berisi data asli.
2. *Ciphertext* (C) adalah pesan ter-enkrip atau tersandi yang dihasilkan dari enkripsi.
3. Enkripsi (fungsi E) adalah proses mengubah teks biasa menjadi teks yang dikodekan atau *ciphertext*.
4. Dekripsi (fungsi D) adalah proses yang berlawanan dengan enkripsi, mengubah teks *cipher* menjadi teks biasa, yang merupakan data awal atau asli.
5. Kunci adalah suatu bilangan rahasia yang digunakan dalam enkripsi.

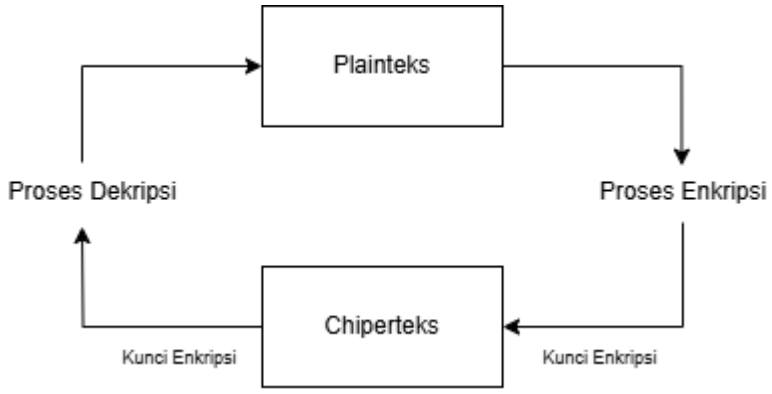
Dua proses utama dalam kriptografi adalah enkripsi dan dekripsi. Menurut (Semarang, 2003), enkripsi yaitu mengubah kode yang dapat dipahami menjadi

kode tidak dapat dipahami. Dekripsi yaitu algoritma yang mengembalikan informasi yang tidak dapat dipahami ke bentuk aslinya. Diagram proses enkripsi dan dekripsi dijelaskan pada Gambar 2.1.



Gambar 2.1 Diagram Enkripsi dan Dekripsi.

Ada banyak komponen penting yang terlibat dalam enkripsi dan dekripsi data, seperti kunci yang menjaga kerahasiaan *ciphertext* dan mencegah pelanggaran data serta algoritma yang membangun sistem yang aman. Algoritma kriptografi biasanya terdiri dari kumpulan langkah teratur yang digunakan untuk enkripsi dan dekripsi. Ada dua jenis algoritma dalam sistem kriptografi: algoritma simetris dan asimetris. Metode simetris yang menggunakan kunci yang sama untuk enkripsi dan dekripsi ditunjukkan pada Gambar 2.2. Sehingga, metode ini terkadang disebut sebagai "algoritma kunci tunggal".



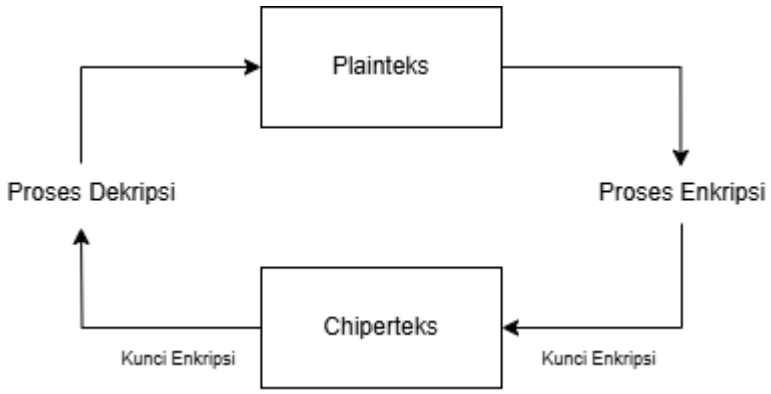
Gambar 2.2 Proses Enkripsi dan Dekripsi Algoritma Simetris

Tabel 2.1 menunjukkan keuntungan dan kelemahan algoritma simetris, menurut (Widiasari, 2014).

Tabel 2.1 Kelebihan dan Kekurangan Algoritma Simetris

No	Indikator	Keterangan
1	Kelebihan	Sistem bekerja lebih cepat daripada algoritma asimetris.
		Karena kecepatan sistem yang tinggi, dapat digunakan dalam sistem *real-time*.
2	Kekurangan	Kesulitan dengan manajemen kunci karena kunci yang dibutuhkan untuk setiap pesan berbeda

Seperti yang ditunjukkan pada Gambar 2.3, algoritma asimetris dan simetris tidak dapat dipertukarkan. Untuk dekripsi algoritma asimetris, kunci privat digunakan, tetapi untuk enkripsi, kunci publik harus dirahasiakan oleh pengguna. Dalam strategi asimetris, menemukan kunci tersembunyi masih sangat sulit, bahkan dengan kunci publik.



Gambar 2.3 Proses Enkripsi dan Dekripsi Algoritma Asimetris

Menurut (Widiasari, 2014), algoritma asimetris memiliki sejumlah kelebihan dan kekurangan, yang ditampilkan dalam Tabel 2.2.

Tabel 2.2 Kelebihan dan Kekurangan Algoritma Asimetris

No	Indikator	Keterangan
1	Kelebihan	Keamanan distribusi kunci lebih baik dan aman.
		Manajemen kunci lebih baik dengan jumlah kunci yang lebih sedikit.
2	Kekurangan	Jika dibandingkan dengan algoritma asimetris, kecepatan operasinya lebih lambat.
		Selain menggunakan kunci yang lebih lama pada tingkat keamanan yang sama, algoritma asimetris memiliki kecepatan operasi yang lebih lambat.

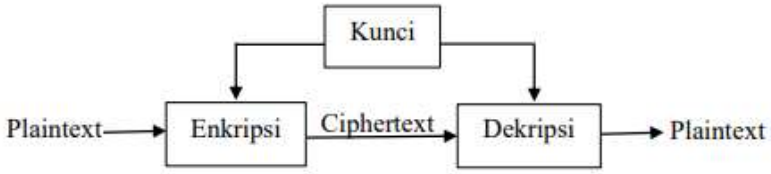
2.1.2 Algoritma Kriptografi

Algortima adalah urutan atau tahap yang disusun secara matematis untuk menyelesaikan masalah. Namun, menurut (Ariyus, 2008) algoritma kriptografi dibagi menjadi tiga bagian berdasarkan kunci yang digunakan untuk menyembunyikan isi pesan rahasia dari orang yang tidak dapat mengaksesnya:

1. Algoritma simetri

   Algoritma simetri juga disebut sebagai "kunci rahasia" yaitu algoritma kriptografi yang menggunakan kunci yang sama untuk melakukan enkripsi dan dekripsi. Maka dari itu, kedua pihak yang mengirim dan menerima pesan harus memastikan bahwa kuncinya tetap rahasia agar tidak dapat diakses orang lain. Karena jika seseorang memiliki kuncinya, mereka dapat melakukan enkripsi dan dekripsi pesan. Menurut (Ariyus, 2008), algoritma kriptografi yang menggunakan algoritma simetri

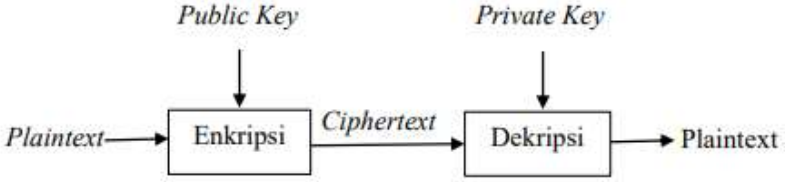
termasuk *Vigenere Cipher, One Time Pad (*OTP*), Standard Data Encryption (*DES*), dan Advance Encryption Standard (*AES*).*



Gambar 2.4 Algoritma Simetri

2. Algoritma Asimetri

Algoritma asimetri menggunakan dua kunci untuk enkripsi dan dekripsi. Kedua jenis kuncinya berinteraksi satu sama lain, yaitu kunci umum atau publik, yang dapat diketahui oleh semua orang, dan kunci rahasia atau private, yang disembunyikan. Algoritma kunci publik lebih aman daripada algoritma simetri karena kriptanalis dapat menggunakan kunci umum untuk mengenkripsi pesan. Karena kunci rahasia hanya dimiliki oleh orang yang menerima pesan, algoritma ini tidak dapat mendekripsinya. Salah satu algoritma yang telah diciptakan oleh para ahli kriptografi yang menggunakan algoritma asimetri adalah algoritma *Digital Signature* (DSA), *Diffle-Hellman* (DH), kriptografi quantum, *Elliptic Curve Cryptography* (ECC), dan lainnya.



Gambar 2.5 Algoritma Asimetri

2.1.3 Algoritma AES

Pada bulan November 2001, *National Institute of Standards and Technology* (NIST) mengumumkan bahwa *Rijndael* akan menjadi Standar Enkripsi Tinggi (AES). AES diharapkan akan menjadi standar kriptografi yang paling banyak digunakan selama setidaknya sepuluh tahun. AES dapat menambah 32-bit kunci dengan panjang kunci hingga 256 bit. Ukuran blok dan panjang kunci dapat dipilih secara mandiri. AES dikenal sebagai AES-128, AES-192, dan AES-256 karena AES menentukan panjang kunci 128, 192, dan 256. Tabel 2.3 menunjukkan perbedaan antara masing-masing versi AES.

Karena jarang menggunakan kunci yang lebih besar dari 192 bit, hanya ada dua versi AES, yaitu AES-128 dan AES-256. Karena teknologi modern hanya memiliki kunci 128 bit, AES lebih tahan terhadap serangan. Banyak yang memiliki panjang kunci 128 bit (Munir, 2019).

Tabel 2.3 Perbedaan Ketiga Buah 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

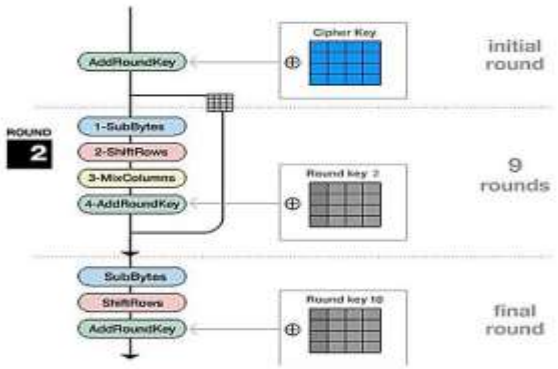
Kunci 128-bit dan blok data dapat dimasukkan ke dalam larik ketiga sebelumnya, yang memiliki 16 *byte*. Status data saat ini direkam dalam dua dimensi (*State*), yang terdiri dari *byte* dan unit, untuk mengubah *plaintext* menjadi *ciphertext*. Panjang blok elemen 32 dilambangkan dengan simbol dan dalam

representasi matematis elemen tersebut. (Munir, 2019) mengklaim bahwa pada AES-128.

Tabel 2.4 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*.

Karena metode enkripsi yang dapat digunakan untuk berbagai *network*, AES dapat dianggap sangat baik.



Gambar 2.6 Diagram Algoritma AES

1. *Key Expansion*

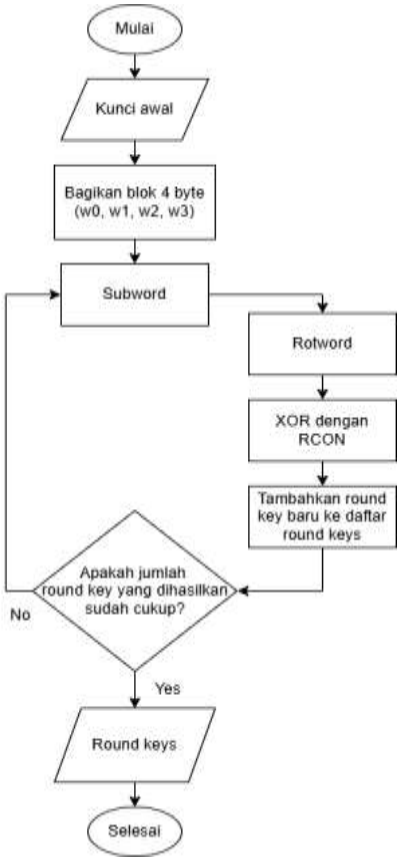
   Key Expansion adalah proses yang digunakan dalam algoritma enkripsi simetris, seperti AES (*Advanced Encryption Standard*), untuk menghasilkan serangkaian kunci dari kunci awal (master *key*) yang akan

digunakan dalam proses enkripsi dan dekripsi. Proses ini sangat penting karena kunci yang dihasilkan akan digunakan pada setiap putaran enkripsi, sehingga meningkatkan keamanan algoritma dengan memastikan bahwa setiap putaran menggunakan kunci yang berbeda.

Dalam algoritma AES, kunci awal yang diberikan akan diperluas menjadi sejumlah kunci yang lebih kecil, yang disebut sebagai "*round keys*." Proses ini dilakukan melalui beberapa langkah sebagai berikut:

a. Input Kunci Awal: Kunci awal yang digunakan dalam AES dapat memiliki panjang 128, 192, atau 256 bit. Panjang kunci ini akan menentukan jumlah putaran (rounds) yang akan dilakukan selama proses enkripsi.
b. Pembentukan Round Keys: Kunci awal akan dipecah menjadi blok-blok yang lebih kecil. Untuk kunci 128 bit, kunci ini akan dibagi menjadi 4 kolom (4 *byte*) yang membentuk matriks 4x4. Proses pembentukan round keys melibatkan beberapa langkah, termasuk:
c. Rotasi: Kolom terakhir dari kunci sebelumnya akan dipindahkan ke posisi pertama.
d. Substitusi: Setiap *byte* dari kolom yang telah dipindahkan akan diganti dengan *byte* yang sesuai dari tabel substitusi (S-Box).
e. XOR dengan Rcon: Hasil dari substitusi akan di-XOR dengan nilai dari Rcon (Round Constant) yang sesuai dengan putaran saat ini.
f. Iterasi: Proses di atas diulang untuk menghasilkan semua round keys yang diperlukan. Jumlah round keys yang dihasilkan tergantung pada panjang kunci awal:

1. Untuk kunci 128 bit, diperlukan 10 round keys.
2. Untuk kunci 192 bit, diperlukan 12 round keys.
3. Untuk kunci 256 bit, diperlukan 14 round keys.



Gambar 2.7 Proses Key Expansion

2. Enkripsi Algoritma *Advanced Encryption Standard* (AES)

Algoritma *Advanced Encryption Standard* (AES) menggunakan transformasi *cipher* dalam beberapa langkah untuk menghasilkan *cipher* yang terenkripsi. *SubBytes*, *ShiftRows*, *MixColumns*, dan *AddRoundKey* adalah transformasi *byte* yang digunakan dalam proses enkripsi:

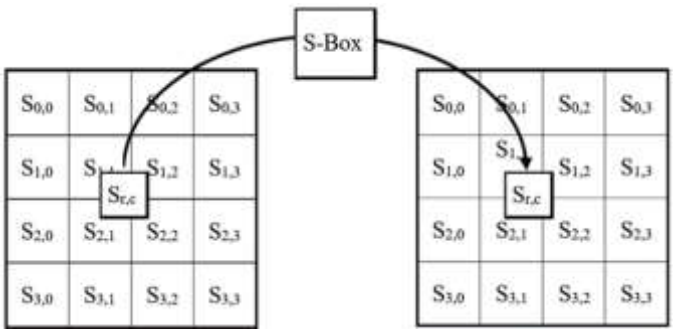
a. Transformasi *SubBytes*

SubBytes adalah transformasi *byte* yaitu setiap elemen pada *state* akan dipetakan melalui tabel substitusi, juga dikenal sebagai *S-Box*, yang ditunjukkan dalam Gambar 2.5.

x \ y	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 2.8 S-Box Rijdael

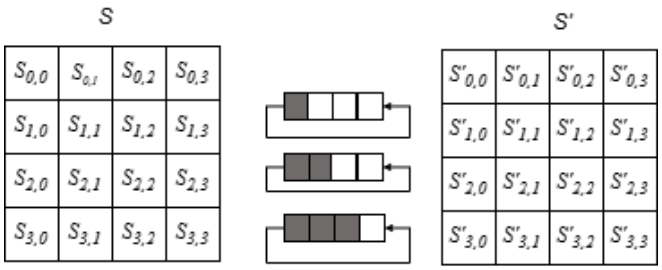
Misalkan untuk setiap *byte* pada *array state* S[r, c] = xy, di mana xy adalah nilai heksadesimal dari S[r, c]. Oleh karena itu, elemen dalam tabel subtitusi, yang terdiri dari perpotongan baris x dengan kolom y, adalah S'[r, c]. Bagaimana pemetaan *byte* mempengaruhi semua *byte* dalam *state* ditunjukkan dalam Gambar 2.6.



Gambar 2.9 Pengaruh Pemetaan pada setiap *Byte* dalam *state*

b. Shiftrows

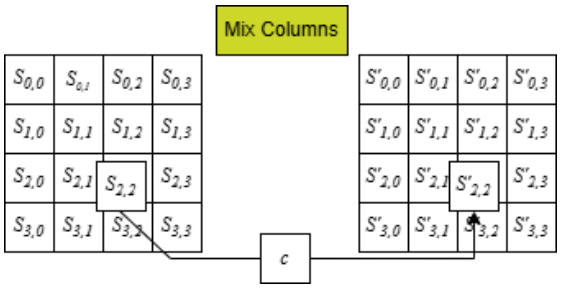
Gambar 2.7 berikut menunjukkan proses transformasi *shiftrows*, yang pada dasarnya adalah pergeseran *byte*, atau rotasi *byte*.



Gambar 2.10 Proses *Shiftrows*

c. MixColumns

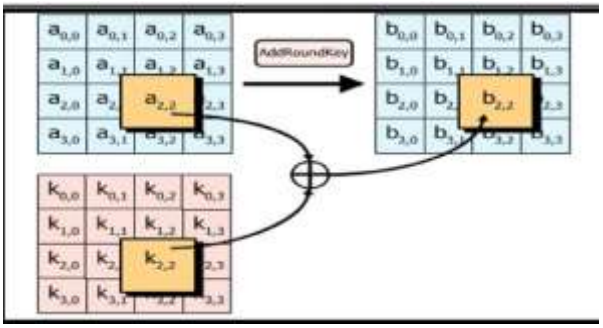
MixColumns menjalankan setiap elemen dalam satu kolom pada *state* tertentu. Perkalian matriks berikut menunjukkan transformasi:



Gambar 2.11 Proses Mix Columns

d. AddRoundKey

Pada *AddRoundKey* dilakukan proses XOR antara *state* sekarang dengan *round key*.



Gambar 2.12 Proses Addround Key

3. Dekripsi Algoritma Advanced Encryption Standard (AES)

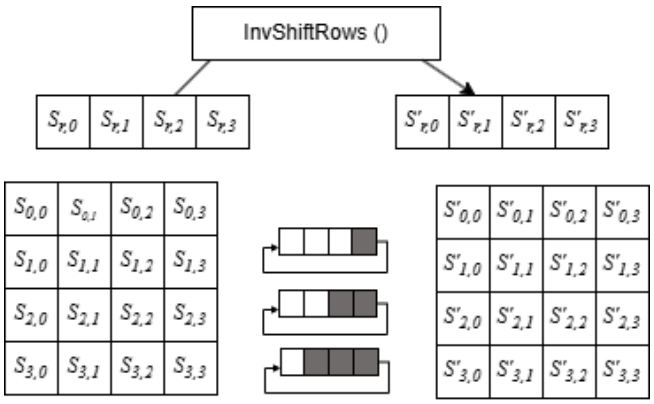
Algoritma dekripsi Advanced Encryption Standard (AES) menggunakan transformasi *cipher* yang dapat dibalikkan dan digunakan dalam arah yang berlawanan untuk menghasilkan *invers cipher* yang mudah dipahami untuk algoritma AES. *InvShiftRows*, *InvSubBytes*, *InvMixColumns*, dan *AddRoundKey* adalah transformasi *byte* yang digunakan pada *invers cipher* AES:

a. AddRoundKey

Matriks *byte-byte* yang terdiri dari *ciphertext* dan *byte-byte roundkey* sebelumnya adalah operasi XOR. *Roundkey* yang digunakan di setiap iterasi berbeda dengan *roundkey* yang digunakan dalam enkripsi. Inversi transformasi ini digunakan untuk dekripsi.

b. Inverse ShiftRows

Transformasi *byte InvShiftRows* adalah kebalikan dari transformasi *ShiftRows*. *InvShiftRows* mengubah bit ke kanan, sedangkan *ShiftRows* mengubah bit ke kiri.



Gambar 2.13 Proses *Invshiftrows*

c. InvSubBytes

Tabel *Inverse S-Box* digunakan untuk memetakan semua elemen pada *state* dalam *InvSubBytes*, Selain itu, itu adalah transformasi *byte* yang berkebalikan dengan transformasi *subbyte.*

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

Gambar 2.14 Proses *InvSubBytes*

d. *Inverse MixColumns*

AES menggunakan matrik perkalian untuk kalikan setiap kolom *state*. Perkalian matrik dapat dituliskan sebagai berikut:

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

Gambar 2.15 Proses *Inverse Mix Colums*

e. *Inverse AddroundKey*

Transformation Reverse Addroundkey tidak jauh berbeda dengan tranformasi *AddroundKey* karena hanya lakukan operasi penambahan dasar menggunakan operasi bitwise XOR.

2.1.4 *Reverse Cipher*

Kriptografi *Reverse Cipher* merupakan kriptografi klasik yang menggunakan transposisi yaitu mengganti satu huruf dengan huruf yang lain. Algoritma ini adalah contoh yang paling sederhana dari kriptografi transposisi yaitu mengubah suatu kalimat dengan menuliskan setiap kata secara terbalik (Hamdah, Harahap, & Usman, 2020).

Contoh kriptografi *Reverse Cipher* sebagai berikut:

Plainteks : SAYA BERMAIN BOLA

*Ciphertek*s : AYAS NIAMREB ALOB

2.1.5 Teknik Transposisi

Teknik transposisi kolom adalah contoh sederhana dari teknik transposisi yang mengubah posisi karakter asli dalam teks ke posisi yang berbeda tanpa mengubah nilai aslinya. Dalam teknik ini, karakter asli ditulis sama dengan panjang karakter, dan untuk mendapatkan kata sandi, teks harus ditulis sesuai dengan kolom yang disepakati.

Kunci : 4 2 1 6 3 5

Teks asli : J U R U S A N M

A T E M A T I K

A X

Dalam tabel, isi sel yang kosong dengan string X. Kemudian, tulis teks sandi sesuai urutan dengan orientasi kolom, sehingga teks sandi yang dihasilkan adalah RAIUMTSEAJNAAMXUTK.

2.1.6 *Avalanche Effect*

Dalam kriptografi, konsep yang dikenal sebagai *Avalanche Effect* mengukur sensitivitas algoritma enkripsi terhadap perubahan kecil pada input, seperti perubahan satu bit pada *plaintext* atau kunci. Prinsip ini sangat penting untuk menjamin keamanan algoritma kriptografi karena diharapkan bahwa perubahan kecil pada input menghasilkan output dengan perubahan besar, sehingga orang yang tidak berwenang akan sulit menebak *ciphertext* (Sugiyanto & Hapsari, 2016)

Tabel 2.5 Indikator untuk Mengukur *Avalanche Effect*

No	Indikator	Deskripsi
1	*Avalanche Effect*	Persentase perubahan bit pada *ciphertext* yang dihasilkan akibat perubahan kecil pada input.
2	Bit Awal	*Ciphertext* yang diperoleh dari *plaintext* atau kunci asli.
3	Bit Ubah	*Ciphertext* yang dihasilkan setelah dilakukan perubahan satu bit pada input.
4	Δ Bit (Perubahan Bit)	Jumlah bit yang berbeda antara *ciphertext* awal dan *ciphertext* setelah input dimodifikasi.
5	n	Total jumlah bit dalam *ciphertext*.

Dalam kriptografi, efek *avalanche* dapat digunakan untuk menilai kemampuan enkripsi menghasilkan *ciphertext* yang sangat sensitif terhadap perubahan pada *plaintext* atau kunci. Ini dapat dilakukan dengan menghitung persentase bit yang berubah antara dua *ciphertext*. Berikut merupakan rumus matematis untuk menghitung efek *Avalanche*:

$$Avalanche\ Effect\ =\ (\Delta\ Bit)/n\ \times 100\%$$

Rumus ini menghitung persentase perubahan bit antara dua *ciphertext*: *ciphertext* asli yang dibuat dari *plaintext* awal dan *ciphertext* yang dibuat setelah sedikit perubahan pada *plaintext* atau kunci. Dengan persentase perubahan ini, algoritma dapat menghasilkan efek *avalanche*, dengan perubahan besar pada output meskipun perubahan kecil pada input. Untuk membantu memahami elemen-elemen yang terlibat dalam pengukuran *Avalanche Effect*, Tabel 2.5 menjelaskan berbagai indikator yang digunakan dalam rumus di atas.

2.1.7 Material SAP

Material SAP merupakan komponen utama dalam modul *Material Management (MM)* pada sistem *Enterprise Resource Planning (ERP)* SAP. Modul ini dirancang untuk mengelola data terkait seluruh proses pengadaan dan pengelolaan stok material dalam perusahaan, termasuk informasi seperti kode material (*Material Number*), deskripsi barang (*Material Description*), grup material, satuan pengukuran, hingga klasifikasi jenis barang. Dalam konteks operasional perusahaan, data material berfungsi sebagai basis untuk mendukung proses perencanaan kebutuhan, manajemen persediaan, serta pengendalian logistik dan keuangan.

Menurut SAP SE, setiap entri data material dalam SAP bersifat *master data*, yang berarti bersifat sentral dan digunakan lintas modul, seperti *Production Planning (PP), Sales and Distribution (SD)*, dan *Warehouse Management (WM)*. Oleh karena itu, keakuratan dan keamanan informasi material sangat penting karena kesalahan atau manipulasi data pada satu modul dapat berdampak sistemik terhadap keseluruhan proses bisnis.

Dalam penelitian ini, data material yang digunakan diperoleh dari sistem SAP milik PT Indonesia Comnet Plus (ICON+), sebuah perusahaan yang mengimplementasikan SAP ERP untuk mengelola proyek jaringan dan infrastruktur telekomunikasi. Data ini mencakup beberapa kolom informasi penting, antara lain:

a. Group Description (GroupDesc): klasifikasi kelompok barang.
b. Material Number: kode unik untuk setiap barang/material.
c. Catalog Data: informasi katalog teknis dari material.

d. Material Description: deskripsi detail mengenai barang.

Data tersebut dinilai sebagai *data sensitif* karena berisi informasi yang apabila bocor atau dimanipulasi, dapat menimbulkan gangguan pada rantai suplai, kesalahan estimasi biaya proyek, dan potensi kebocoran strategi logistik perusahaan. Oleh karena itu, perlindungan data material melalui metode kriptografi menjadi aspek yang sangat krusial, terlebih di era digital yang rawan terhadap ancaman siber seperti *data breach*, *man-in-the-middle attack*, dan *spoofing*.

Penggunaan algoritma enkripsi seperti AES-128 yang dikombinasikan dengan *Reverse Cipher* dalam penelitian ini bertujuan untuk memberikan perlindungan ganda terhadap data material tersebut. Hal ini sejalan dengan penelitian oleh yang menyatakan bahwa penerapan kriptografi simetris pada data master SAP dapat meningkatkan *confidentiality* dan *data integrity* pada sistem ERP.

Secara umum, pentingnya pengamanan *material master data* tidak hanya bersifat teknis, namun juga strategis. Menurut studi oleh (Harsono, Nugroho, & Suryanto), perusahaan yang gagal menjaga integritas data SAP memiliki risiko tinggi terhadap ketidakefisienan operasional, audit yang tidak akurat, dan rendahnya kepercayaan pengguna terhadap sistem informasi yang digunakan.

2.1.8 *Streamlit*

Streamlit adalah salah satu *framework Python* berbasis web yang bersifat *open-source* dan digunakan untuk membangun antarmuka pengguna (*Graphical User Interface*/GUI) secara cepat dan interaktif. Framework ini sangat cocok

untuk aplikasi berbasis analisis data, *machine learning*, dan implementasi sistem kriptografi karena hanya membutuhkan skrip *Python* tanpa memerlukan pengetahuan mendalam tentang HTML, CSS, atau *JavaScript*.

Menurut (Raschka & Mirjalili, 2019), *Python* menjadi bahasa yang sangat populer di bidang sains data dan kriptografi karena mendukung banyak pustaka (*library*) dan *framework*, seperti *Streamlit* dan *PyCryptodome*, yang dapat mempermudah pengembangan sistem keamanan data berbasis algoritma kriptografi.

Beberapa kelebihan implementasi *Streamlit* dalam penelitian ini:

1. Mempermudah validasi hasil enkripsi dan dekripsi.
2. Mendukung integrasi *Avalanche Effect* dan ekspor hasil (PDF, Excel, JSON).
3. Mudah digunakan oleh pengguna teknis maupun non-teknis.
4. Dapat dikembangkan menjadi *dashboard* proteksi data internal untuk PT Indonesia Comnet Plus.

2.1.9 *Time and Space Complexity*

Time Complexity mengacu pada jumlah waktu komputasi yang dibutuhkan oleh sebuah algoritma untuk menyelesaikan tugasnya, sedangkan *Space Complexity* mengacu pada jumlah total ruang memori yang dibutuhkan. Keduanya diukur sebagai fungsi dari ukuran input. Penting untuk dicatat bahwa waktu yang diukur di sini bukanlah waktu absolut dalam detik atau menit, melainkan jumlah operasi dasar yang dilakukan, dan ruang yang diukur adalah alokasi memori, bukan ukuran file. Ini karena waktu absolut dan konsumsi memori aktual dapat

bervariasi tergantung pada spesifikasi perangkat keras, bahasa pemrograman, atau faktor eksternal lainnya. Oleh karena itu, baik *time* maupun *Space Complexity* biasanya dinyatakan dalam notasi asimtotik, yang paling umum adalah Big O *notation* (O).

Notasi Big O memberikan batas atas (*upper bound*) pada pertumbuhan waktu eksekusi atau penggunaan memori algoritma seiring dengan bertambahnya ukuran input. Analisis kompleksitas melibatkan identifikasi operasi-operasi dominan atau alokasi memori utama dalam algoritma dan menghitung berapa kali operasi tersebut dieksekusi atau berapa banyak memori yang dialokasikan sebagai fungsi dari ukuran input. Fokus utama adalah pada skenario kasus terburuk (*worst-case scenario*) karena ini memberikan jaminan mengenai batas atas kinerja algoritma.

Beberapa kelas Big O yang umum untuk *time* dan *Space Complexity* antara lain:

1. $O(1)$ - *Constant Time/Space:* Waktu eksekusi atau penggunaan memori tidak bergantung pada ukuran input. Contoh: mengakses elemen array berdasarkan *indeks (time)*, atau menukar dua variabel tanpa memori tambahan (*space*).
2. $O(\log n)$ - *Logarithmic Time/Space:* Waktu eksekusi atau penggunaan memori meningkat secara logaritmik seiring dengan ukuran input. Biasanya lebih efisien untuk input besar. Contoh: pencarian biner (*binary search*) (*time*).

3. $O(n)$ - *Linear Time/Space:* Waktu eksekusi atau penggunaan memori berbanding lurus dengan ukuran input. Contoh: mencari elemen dalam array tanpa urutan (time), atau membuat salinan *array input (space).*
4. $O(\log n)$ - *Linearithmic Time:* Sering ditemukan pada algoritma pengurutan yang efisien. Contoh: *merge sort, quick sort (time)*.
5. $O(n^2)$ - *Quadratic Time/Space:* Waktu eksekusi atau penggunaan memori berbanding lurus dengan kuadrat ukuran input. Contoh: *nested loop, bubble sort (time)*, atau menggunakan matriks n × n (space).
6. $O(2^n)$ - *Exponential Time:* Waktu eksekusi meningkat secara eksponensial. Biasanya terjadi pada masalah-masalah yang diselesaikan dengan *brute-force* atau rekursi tanpa optimasi.
7. $O(n!)$ - *Factorial Time:* Waktu eksekusi meningkat sangat cepat. Biasanya terbatas pada masalah input yang sangat kecil.

Dalam banyak kasus, terdapat *trade-off* antara *Time Complexity* dan *Space Complexity*. Algoritma yang lebih cepat mungkin memerlukan lebih banyak memori, dan sebaliknya. Pemilihan algoritma yang tepat seringkali melibatkan penimbangan antara kedua faktor ini, tergantung pada batasan dan persyaratan spesifik dari masalah yang dihadapi. Misalnya, dalam sistem dengan memori terbatas, algoritma dengan *Space Complexity* $O(1)$ atau $O(\log n)$ akan lebih diutamakan, meskipun mungkin memiliki *Time Complexity* yang sedikit lebih tinggi.

2.2 Amanah dalam Ajaran Islam

Dalam ajaran Islam, amanah memiliki arti yang luas dan mendalam. Amanah secara terminologi berarti kepercayaan atau tanggung jawab yang diberikan kepada seseorang untuk mempertahankan dan melaksanakan tindakan sesuai dengan keadaan. Konsep ini mencakup tanggung jawab moral, sosial, dan keagamaan selain hal-hal material. Amanah dalam Islam adalah segala sesuatu yang diberikan Allah SWT kepada manusia, baik itu perintah, larangan, atau tanggung jawab yang harus mereka penuhi. Al-Qur'an secara eksplisit menegaskan pentingnya amanah, salah satunya dalam surat An-Nisa ayat 58 yang berbunyi (NU Online, 2024):

اِنَّ اللّٰهَ يَأْمُرُكُمْ اَنْ تُؤَدُّوا الْاَمٰنٰتِ اِلٰٓى اَهْلِهَاۙ وَاِذَا حَكَمْتُمْ بَيْنَ النَّاسِ اَنْ تَحْكُمُوْا بِالْعَدْلِۗ اِنَّ اللّٰهَ نِعِمَّا يَعِظُكُمْ بِهٖۗ اِنَّ اللّٰهَ كَانَ سَمِيْعًاۢ بَصِيْرًا

"Sesungguhnya Allah menyuruh kamu menyampaikan amanah kepada pemiliknya. Apabila kamu menetapkan hukum di antara manusia, hendaklah kamu tetapkan secara adil. Sesungguhnya Allah memberi pengajaran yang paling baik kepadamu. Sesungguhnya Allah Maha Mendengar lagi Maha Melihat (QS. An-Nisa/4:58)"

Dalam Tafsir Al-Misbah karya M. Quraish Shihab (Shihab, 2002), dijelaskan bahwa Al-Qur'an menjelaskan beberapa keburukan yang dilakukan oleh orang Yahudi, seperti mengabaikan janji untuk mengamalkan Al-Qur'an dan tidak menyembunyikannya. Al-Qur'an juga mengingatkan umat Islam agar tidak mengikuti perilaku orang Yahudi. Arahan ini sangat penting karena Allah sendiri menegaskannya dengan nama-Nya yang mulia. "Sesungguhnya Allah Yang Maha Agung, yang memiliki sifat-sifat terpuji dan terhindar dari segala sifat tercela, memerintahkan kalian untuk melaksanakan amanah dengan baik dan tepat waktu

kepada yang berhak menerimanya," demikian firman-Nya. Ini berlaku terlepas dari seberapa banyak amanah yang diberikan kepada kalian.

Amanah adalah kebalikan dari khianat dan hanya diberi kepada mereka yang dianggap mampu untuk menjaga dan melindunginya saat diminta oleh pemiliknya. Amanah, atau kepercayaan, merupakan fondasi iman dalam ajaran agama. Seperti yang disampaikan oleh Nabi Muhammad SAW, "Tidak ada iman bagi orang yang tidak memiliki amanah." Dalam surat Al-Anfal ayat 27, amanah digambarkan sebagai tanggung jawab besar yang Allah tawarkan kepada makhluk-Nya, namun hanya manusia yang bersedia menerimanya (Kemenag, 2024):

يٰٓاَيُّهَا الَّذِيْنَ اٰمَنُوْا لَا تَخُوْنُوا اللّٰهَ وَالرَّسُوْلَ وَتَخُوْنُوْٓا اَمٰنٰتِكُمْ وَاَنْتُمْ تَعْلَمُوْنَ

"*Wahai orang-orang yang beriman, janganlah kamu mengkhianati Allah dan Rasul serta janganlah kamu mengkhianati amanat yang dipercayakan kepadamu, sedangkan kamu mengetahui. (QS. Al-Anfal/8:27)"*

Dalam Tafsir Al-Misbah karya M. Quraish Shihab (Shihab, 2002), dijelaskan bahwa kata (تخونو) takbünu berasal dari kata (الحون) al-khaun, yang berarti "kekurangan", sedangkan kebalikannya adalah (الوفاء) al-wafás, yang artinya "kesempurnaan". "Khianat" merupakan lawan dari "amanah" karena tindakan tersebut menghilangkan tanggung jawab yang seharusnya dipenuhi akibat pengkhianatan terhadap orang lain. Kata "أمانات" amānat berasal dari kata "المنة" amanah, yang artinya "merasa aman" dan "percaya." Barang yang dititipkan sebagai amanah menunjukkan bahwa orang yang menitipkannya memiliki kepercayaan dan merasa yakin bahwa barang tersebut akan dijaga—baik secara

aktif maupun pasif—sehingga pemiliknya dapat mengambilnya kembali ketika diperlukan. Selain itu, barang yang dititipkan sebagai amanah akan lebih terjaga jika dirawat dengan baik melalui pemeliharaan yang aktif.

Seorang muslim harus menjaga amanah sebaik mungkin dan menyerahkannya kepada orang yang berhak menerimanya, menurut Hadis Tirmidzi no. 1853. Karena sikap khianat bertentangan dengan nilai-nilai amanah dan sangat dikecam dalam Islam, harus dihindari sepenuhnya saat melakukannya. Sebagaimana disebutkan dalam sebuah riwayat hadits, Rasulullah SAW pernah menyatakan bahwa orang yang tidak dapat menjaga amanah tidak akan memiliki iman yang sempurna (Ilmu Islam, 2024):

> *"Barangsiapa yang meringankan (menghilangkan) kesulitan seorang muslim kesulitan-kesulitan duniawi, maka Allah akan meringankan (menghilangkan) baginya kesulitan di akhirat kelak. Barangsiapa yang memberikan kemudahan bagi orang yang mengalami kesulitan di dunia, maka Allah akan memudahkan baginya kemudahan (urusan) di dunia dan akhirat. Dan barangsiapa yang menutupi (aib) seorang muslim sewaktu di dunia, maka Allah akan menutup (aibnya) di dunia dan akhirat. Sesungguhnya Allah akan senantiasa menolong seorang hamba selalu ia menolong saudaranya."* [HR. Tirmidsi].

Seorang Muslim juga harus selalu bergantung pada Allah SWT saat menjalankan amanah. Sadar bahwa Allah Maha Mengetahui dan senantiasa mengawasi segala sesuatu mendorong mereka untuk melaksanakan amanah dengan penuh tanggung jawab. Dengan demikian, menjalankan amanah bukan hanya menjadi bukti keimanan, tetapi juga membawa keberkahan bagi kehidupan dunia dan akhirat mereka.

2.3 Kajian Topik dengan Teori Pendukung

Perlindungan informasi menjadi semakin penting seiring dengan pertumbuhan pesat teknologi digital. Pengamanan data sangat penting untuk berbagai aplikasi, seperti sistem keamanan, industri medis, dan media sosial. Data biasanya berisi data sensitif, jadi diperlukan metode enkripsi untuk melindunginya. Dalam kriptografi, ada banyak algoritma yang telah dikembangkan untuk melindungi informasi, seperti AES dan *Reverse Cipher*. Kedua algoritma ini memiliki karakteristik dan kinerja yang berbeda, terutama ketika datang ke pengamanan data. Tujuan dari penelitian ini adalah untuk melakukan perbandingan antara kinerja AES dan *Reverse Cipher* dalam hal keamanan data. Pengukuran kinerja akan dilakukan dengan mempertimbangkan kecepatan enkripsi dan dekripsi.

Seni dan ilmu kriptografi melindungi data pribadi dengan mengubah formatnya sehingga orang lain tidak dapat memahaminya. Dalam kriptografi modern, ada dua jenis algoritma, yaitu simetris dan asimetris. *Advanced Encryption Standard* (AES) merupakan algoritma enkripsi simetris yang umum digunakan untuk menjaga keamanan data, dan *Reverse Cipher* adalah algoritma kriptografi klasik berbasis karakter yang tergolong dalam kategori simetris. Ketika data digital disimpan dalam berbagai format, penting untuk memastikan bahwa kualitas dan struktur data tetap terjaga sehingga data dapat diterima dengan baik setelah proses dekripsi.

Algoritma kriptografi simetris yang dikenal sebagai AES hanya memanfaatkan satu kunci untuk proses enkripsi dan dekripsi. AES dikenal karena kecepatannya dan efisiensi, sehingga sering digunakan dalam berbagai aplikasi

keamanan. Sebaliknya, *Reverse Cipher* adalah algoritma dalam kriptografi konvensional yang hanya memakai satu kunci untuk proses enkripsi dan dekripsi, tetapi berbeda dari AES dalam kompleksitas dan metode enkripsi, meskipun AES lebih aman dan efisien daripada *Reverse Cipher*. Hasil analisis perbandingan ini akan digunakan sebagai dasar untuk mengusulkan algoritma pengamanan data yang lebih baik.

Melalui kajian ini, diharapkan penelitian ini dapat memberikan kontribusi dalam bidang kriptografi dan pengolahan data digital, serta membantu dalam memilih algoritma enkripsi yang paling tepat untuk aplikasi tertentu.

BAB III

METODE PENELITIAN

3.1 Jenis Penelitian

Jenis penelitian ini merupakan penelitian kuantitatif eksperimental. Penelitian ini bertujuan untuk membandingkan kinerja dua algoritma kriptografi, yaitu AES dan *Reverse Cipher*, dalam konteks pengamanan data. Eksperimen ini akan mengevaluasi kinerja kedua algoritma berdasarkan beberapa indikator, termasuk kecepatan enkripsi dan dekripsi, tingkat perubahan data yang diukur dengan *Avalanche Effect*, 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 AES dan *Reverse Cipher*. Sementara itu, variabel terikatnya mencakup kinerja algoritma, yang diukur melalui beberapa parameter, seperti waktu pemrosesan (kecepatan enkripsi dan dekripsi), perubahan data setelah enkripsi yang diukur dengan *Avalanche Effect*, serta tingkat keamanan dari hasil enkripsi tersebut. 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

Penelitian ini menggunakan satu jenis data utama yang diperoleh dari PT Indonesia Comnets Plus (ICON+), anak perusahaan dari PT PLN (Persero) yang

bergerak dalam bidang teknologi informasi dan komunikasi (TIK). Data tersebut berasal dari sistem Enterprise Resource Planning (ERP) SAP (System Application and Product) yang diimplementasikan oleh perusahaan sebagai bagian dari manajemen terintegrasi terhadap proses bisnisnya.

Jenis data yang dimaksud adalah Material SAP, yaitu data proyek jaringan yang disimpan dalam format Microsoft Excel. Data ini mencakup informasi penting berupa teks seperti nama material, kode material, jumlah (kuantitas), serta deskripsi teknis dari komponen jaringan yang digunakan. Data ini memiliki struktur yang sistematis dan konsisten, sehingga sangat mendukung dalam proses analisis berbasis algoritma kriptografi, baik secara manual maupun otomatis.

Dalam penelitian ini, sejumlah 1000 baris data Material SAP digunakan untuk dua tujuan utama, yaitu:

1. Implementasi manual algoritma kriptografi, seperti *Reverse Cipher* dan AES-128, yang diaplikasikan pada beberapa entri data terpilih guna menunjukkan proses enkripsi dan dekripsi secara rinci dan terstruktur.
2. Pengujian performa algoritma secara otomatis, meliputi analisis *Avalanche Effect* dan pengukuran waktu proses enkripsi-dekripsi dalam skala data yang besar, menggunakan pendekatan pemrograman dengan Python.

Pemanfaatan data Material SAP yang berasal langsung dari sistem operasional perusahaan bertujuan untuk memberikan representasi yang realistis dan aplikatif terhadap efektivitas algoritma kriptografi dalam mengamankan data proyek jaringan. Dengan demikian, penelitian ini diharapkan mampu memberikan kontribusi signifikan terhadap pengembangan metode perlindungan data dalam konteks TIK korporasi.

3.3 Teknik Pengumpulan Data

Pengumpulan data dalam penelitian ini dilakukan dengan metode studi dokumentasi. Artinya, kami mengumpulkan data dengan cara mempelajari dan mengambil langsung dokumen internal perusahaan, yaitu data yang sudah ada di sistem SAP milik PT Indonesia Comnet Plus (ICON+). Oleh karena itu, data yang digunakan dalam penelitian ini adalah data sekunder

Data Material SAP (System Application and Product) merupakan data terstruktur dalam format Excel yang diperoleh dari sistem ERP SAP, digunakan oleh perusahaan untuk mencatat seluruh aktivitas pengadaan dan penggunaan material dalam proyek jaringan. Informasi yang tercakup dalam data ini meliputi kode material, nama barang, jumlah (kuantitas), dan deskripsi teknis dari setiap komponen.

Dalam penelitian ini, data tersebut dimanfaatkan pada dua tahap utama, yakni:

1. Pengolahan manual untuk mendemonstrasikan proses enkripsi dan dekripsi dengan algoritma tertentu terhadap sebagian data.
2. Pengolahan otomatis untuk melakukan pengujian *Avalanche Effect* dan evaluasi performa waktu terhadap algoritma kriptografi menggunakan bahasa pemrograman *Python*.

Dengan memanfaatkan satu sumber data yang representatif dan berasal dari lingkungan operasional nyata, penelitian ini memiliki landasan empiris yang kuat untuk mengevaluasi kinerja algoritma kriptografi secara menyeluruh, baik dari sisi keamanan data maupun efisiensi proses komputasi.

3.4 Teknik Analisis Data

Untuk menghasilkan hasil signifikan dalam penelitian ini mengenai perbandingan kinerja algoritma AES dan *Reverse Cipher* dalam pengamanan data, beberapa proses analisis akan dilakukan. Penelitian ini akan mengikuti langkah-langkah berikut:

3.4.1 Proses Enkripsi AES dan *Reverse Cipher*

1. Persiapan Data : Tahap awal dalam proses enkripsi dimulai dengan mempersiapkan data yang akan diamankan. Pada penelitian ini, data yang digunakan adalah data material SAP. hanya bagian *Customer Name* yang digunakan, yaitu: ALFITRAHSURABAYA. Data ini disimpan dalam bentuk teks sebelum diproses lebih lanjut ke tahap enkripsi.
2. Konversi Data ke Hexadecimal : Setelah data disiapkan, langkah berikutnya adalah mengonversi data teks menjadi bilangan hexadecimal. Proses ini dimulai dengan mengubah setiap karakter dalam *plaintext* ke dalam kode ASCII, kemudian dikonversi menjadi bentuk hexadecimal. Misalnya, karakter 'C' dikonversi menjadi ASCII 67, lalu menjadi hexadecimal 43 ; karakter 'I' menjadi 49 , dan seterusnya. Hasil dari konversi ini menghasilkan deretan hexadecimal sebagai berikut:

   41 59 41 42 41 52 55 53 48 41 52 54 49 46 4C 41

   Selanjutnya, data hexadecimal tersebut disusun ke dalam bentuk matriks 4x4 (State Matrix) untuk keperluan proses enkripsi AES:

$$\begin{bmatrix} 41 & 41 & 48 & 49 \\ 59 & 52 & 41 & 46 \\ 41 & 55 & 52 & 4C \\ 42 & 53 & 54 & 41 \end{bmatrix}$$

3. *Key expansion* : Kunci yang digunakan untuk proses enkripsi adalah string "KRIPTOGRAFIAESKU" yang memiliki panjang 16 karakter atau 128 bit. Setiap karakter dari kunci ini dikonversi ke dalam bentuk hexadecimal, misalnya 'S' menjadi 53, 'M' menjadi 4D, dan seterusnya. Hasilnya adalah:

   4B 52 49 50 54 4F 47 52 41 46 49 41 45 53 4B 55

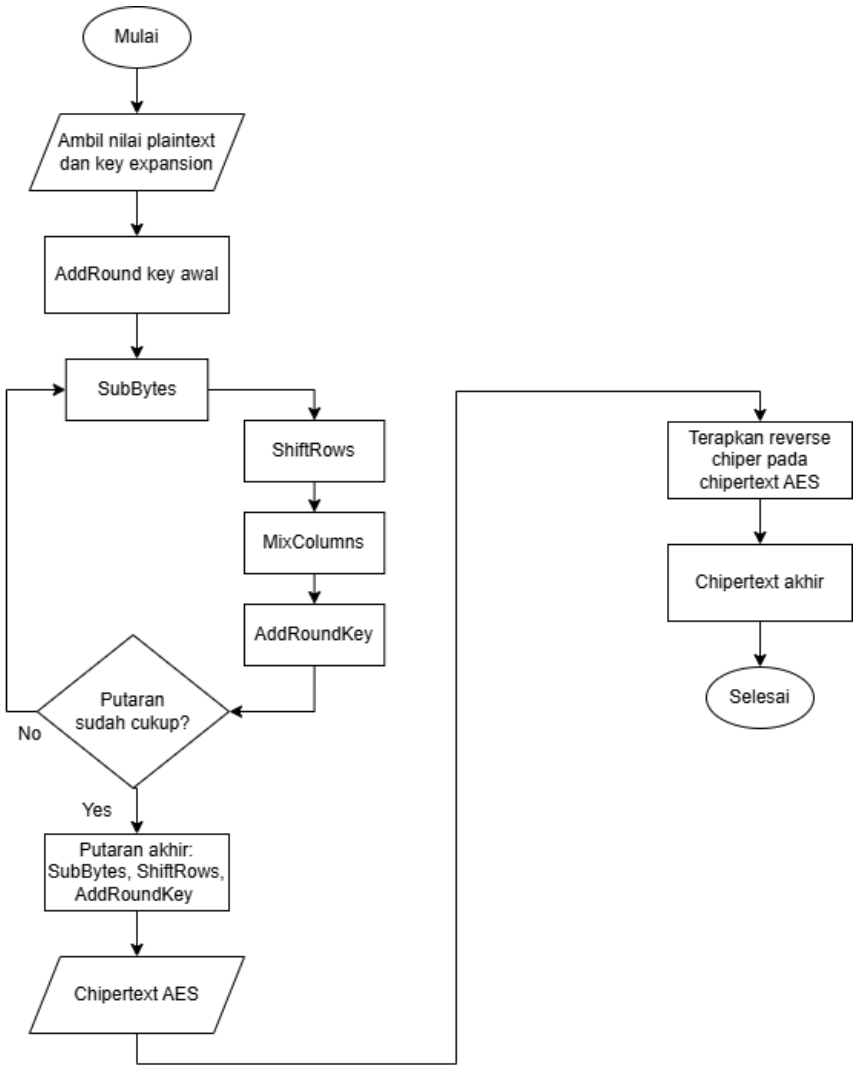
   Kunci ini kemudian disusun dalam bentuk matriks 4x4 untuk membentuk RoundKey ke-0:

$$\begin{bmatrix} 4B & 54 & 41 & 45 \\ 52 & 4F & 46 & 53 \\ 49 & 47 & 49 & 4B \\ 50 & 52 & 41 & 55 \end{bmatrix}$$

4. *AddRoundKey* (Round Pertama) : Proses enkripsi dimulai dengan menambahkan kunci *round* pertama ke data yang telah dipecah menjadi blok 128-bit. Kunci ini dihasilkan dari *key expansion*, yang mengubah kunci utama menjadi beberapa sub-kunci untuk digunakan dalam setiap *round*.
5. *SubBytes* : Setiap *byte* dalam blok data dienkripsi menggunakan tabel substitusi yang disebut *S-Box*, yang menggantikan *byte* yang ada dengan *byte* yang terenkripsi. Proses ini bertujuan untuk mengacak *byte* dalam data dan meningkatkan keamanan *cipher*.
6. *ShiftRows* : Dalam langkah ini, baris-baris dalam blok data akan digeser. Setiap baris digeser sesuai dengan jumlah tertentu: Baris pertama tetap, baris kedua digeser satu *byte*, baris ketiga dua *byte*, dan baris keempat tiga *byte*. Transformasi ini membantu meningkatkan difusi dalam data.
7. *MixColumns* : Setiap kolom dalam blok data akan diperlakukan sebagai polinomial dan dikalikan dengan matriks tertentu untuk memecah pola

dalam data. Langkah ini membantu meningkatkan kerumitan *cipher*, sehingga sulit bagi pihak yang tidak berwenang untuk mengeksploitasi pola dalam data.

8. *AddRoundKey* (Setiap *Round*) : Setiap *round* diulang dengan langkah *AddRoundKey*, yang menambahkan sub-kunci tertentu yang dihasilkan dari *key expansion*. Setiap langkah ini memastikan data terenkripsi tetap aman selama proses enkripsi berlangsung.
9. *Round* Terakhir (Tanpa *MixColumns*) : Pada *round* terakhir, transformasi *SubBytes*, *ShiftRows*, *dan AddRoundKey* diterapkan. Namun, langkah *MixColumns* tidak dilakukan pada *round* terakhir untuk menghasilkan *ciphertext* final yang siap untuk dikirim atau disimpan.
10. Hasil Enkripsi Akhir : *Ciphertext* yang telah terbalik urutannya setelah diterapkan *Reverse Cipher* adalah *ciphertext* final yang siap untuk dikirim atau disimpan. Dengan dua lapisan enkripsi (AES dan *Reverse Cipher*), keamanan data lebih terjamin dan lebih sulit untuk diserang atau dieksploitasi.
11. Terapkan *Reverse Cipher* pada *Ciphertext* AES : Setelah proses enkripsi AES selesai, langkah selanjutnya adalah menerapkan *Reverse Cipher* pada *ciphertext* yang dihasilkan. *Reverse Cipher* akan membalikkan urutan *byte* atau karakter dalam *ciphertext*. Sebagai contoh, jika *ciphertext* AES yang dihasilkan adalah A1B2C3D4, setelah diterapkan *Reverse Cipher*, hasilnya akan menjadi D4C3B2A1. Langkah ini memberikan lapisan keamanan tambahan dengan mengacak urutan data.



Gambar 3.1 Proses Enkripisi AES dan *Reverse Cipher*

3.4.2 Proses Dekripsi *Reverse Cipher* dan AES

Proses dekripsi dalam *Reverse Cipher* adalah kebalikan langsung dari proses enkripsi, yaitu dengan membalikkan urutan karakter pada blok data yang telah dienkripsi. Langkah-langkah dekripsi adalah sebagai berikut:

1. Persiapan Data *Ciphertext* : Tahap awal dalam proses dekripsi dimulai dengan mempersiapkan ciphertext yang akan dikembalikan ke bentuk aslinya. Ciphertext ini merupakan hasil dari dua tahap enkripsi, yaitu

enkripsi dengan AES 128-bit diikuti oleh *Reverse Cipher*. Contoh ciphertext yang digunakan adalah ciphertext yang telah dibalik urutannya,

4159414241525553484152544964C41

2. Proses *Reverse Cipher* (Pembalikan Ciphertext) : Langkah pertama dari dekripsi adalah mengembalikan urutan ciphertext ke bentuk semula sebelum dibalik. *Reverse Cipher* akan mengurutkan kembali karakter atau *byte* ke urutan asal seperti saat keluar dari enkripsi AES. Jika input ciphertext adalah:

41 59 41 42 41 52 55 53 48 41 52 54 49 46 4C 41

Maka hasil pembalikannya menjadi:

41 4C 46 49 54 52 41 48 53 55 52 41 42 41 59 41

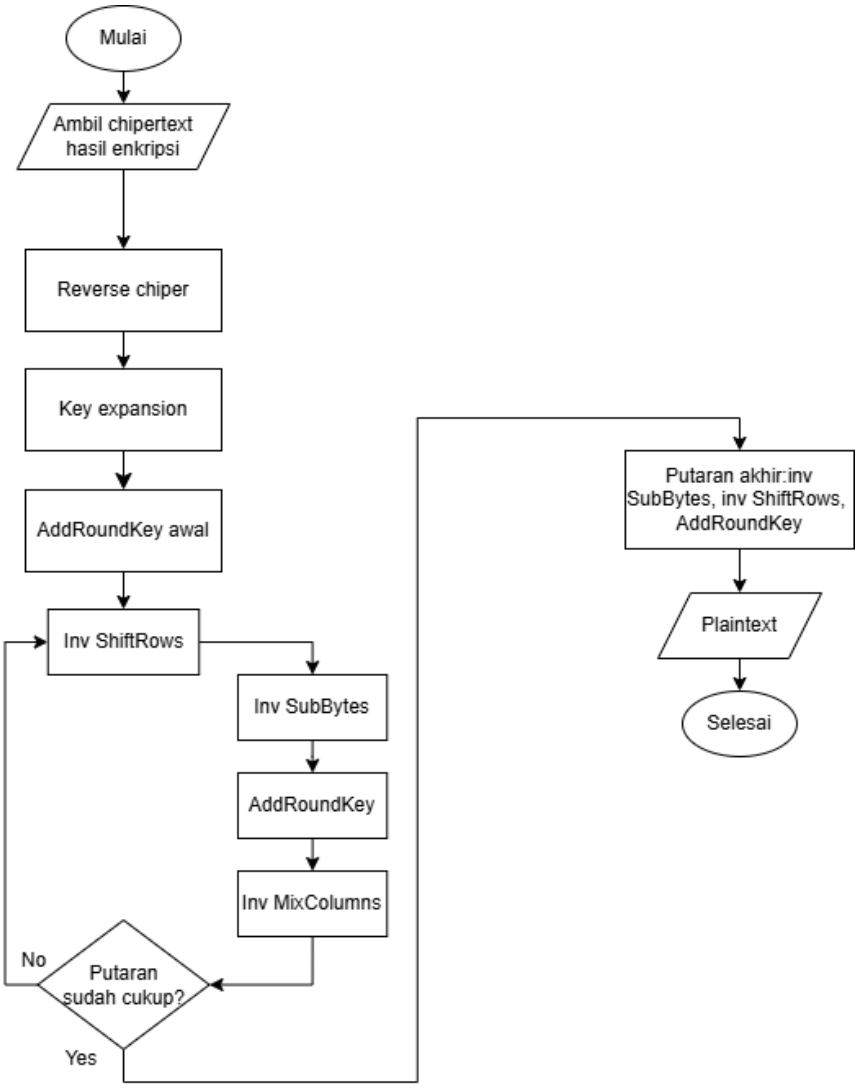
3. Persiapan Kunci untuk Dekripsi AES : Kunci utama yang digunakan dalam enkripsi sebelumnya disiapkan kembali. Key expansion diterapkan untuk menghasilkan sub-kunci yang sama seperti yang digunakan dalam proses enkripsi. Dalam proses dekripsi AES, sub-kunci ini akan digunakan dalam urutan terbalik, dimulai dari sub-kunci terakhir hingga sub-kunci pertama, seperti berikut ini :

4B 52 49 50 54 4F 47 52 41 46 49 41 45 53 4B 55

Kunci ini kemudian disusun dalam bentuk matriks 4x4 untuk membentuk RoundKey ke-10:

$$\begin{bmatrix} 4B & 54 & 41 & 45 \\ 52 & 4F & 46 & 53 \\ 49 & 47 & 49 & 4B \\ 50 & 52 & 41 & 55 \end{bmatrix}$$

4. *AddRoundKey* (Round Pertama) : Untuk memulai proses dekripsi, Anda harus menambahkan kunci *round* yang terbalik ke dalam data terenkripsi. Kunci yang dipakai untuk proses dekripsi adalah sub-kunci yang sama, tetapi dalam urutan terbalik.
5. *InvSubBytes* : Dalam langkah ini, *byte* dalam *ciphertext* digantikan dengan *byte* yang sesuai dari tabel substitusi terbalik, juga dikenal sebagai *S-Box* terbalik. Proses ini memulihkan *byte* yang telah digantikan selama enkripsi
6. *InvShiftRows* : Baris-baris dalam blok data yang telah digeser pada enkripsi, kini akan digeser kembali ke posisi semula. Baris pertama tetap, baris kedua digeser satu *byte* ke kanan, baris ketiga digeser dua *byte* ke kanan, dan baris keempat digeser tiga *byte* ke kanan.
7. *InvMixColumns* : Kolom-kolom dalam blok data yang telah dicampur pada enkripsi, akan dicampur kembali menggunakan matriks *invers* yang berlawanan dengan langkah *MixColumns***.**
8. *AddRoundKey* (Setiap *Round*) : Setiap *round* dekripsi dimulai dengan langkah *AddRoundKey***,** menggunakan sub-kunci yang dihasilkan dari *key expansion* yang terbalik.
9. *Round* Terakhir (Tanpa *InvMixColumns*) : Pada *round* terakhir dekripsi, langkah-langkah yang dilakukan adalah *InvSubBytes*, *InvShiftRows*, dan *AddRoundKey*, namun tanpa langkah *InvMixColumns*. Hal ini mengembalikan data ke bentuk *plaintext* aslinya.
10. Penggabungan Blok : Setelah proses pembalikan urutan karakter pada setiap blok, blok-blok tersebut digabungkan kembali untuk menghasilkan data atau pesan asli (*plaintext*).



Gambar 3.2 Proses Dekripsi *Reverse Cipher* dan AES

3.4.3 Pengujian *Avalanche Effect*

Avalanche Effect adalah karakteristik dari algoritma kriptografi dimana perubahan kecil pada input (seperti mengubah satu bit dalam *plaintext*) mendapatkan hasil perubahan yang besar dan tidak terduga pada output (*ciphertext*). Idealnya, jika satu bit pada input *plaintext* diubah, maka sekitar setengah dari bit-bit *ciphertext* harus berubah. Pengujian ini digunakan untuk memastikan bahwa algoritma AES dan *Reverse Cipher* yang digunakan dalam

penelitian ini cukup aman dan memiliki tingkat keacakan yang tinggi dalam menghasilkan *ciphertext*.

Langkah-langkah untuk menguji *Avalanche Effect* adalah sebagai berikut:

1. Persiapan Data : Pada tahap ini, data yang digunakan dalam pengujian *Avalanche Effect* bersumber dari Material SAP yang disusun dalam bentuk *plaintext* teks terstruktur. Pengujian dilakukan dengan menggunakan dua metode *padding* yang berbeda, yaitu metode *Fixed Length* 512-bit dan metode PKCS#7, untuk mengevaluasi pengaruh *padding* terhadap difusi hasil enkripsi.
2. Proses Enkripsi : Setelah data siap, proses dilanjutkan dengan melakukan enkripsi terhadap *plaintext* asli menggunakan kombinasi algoritma *Reverse Cipher* dan AES-128 bit. Mode enkripsi yang digunakan adalah ECB (Electronic Code Book), dan kunci yang dipakai secara konsisten sepanjang proses adalah "KRIPTOGRAFIAESKU". Ciphertext yang dihasilkan dari proses ini dicatat dan dijadikan acuan untuk langkah pengujian berikutnya.
3. Modifikasi Bit pada Input dan Pengukuran Avalanche: Untuk mengukur *Avalanche Effect*, dilakukan perubahan satu bit pada *plaintext* yang sebelumnya telah dienkripsi. Perubahan bit ini bersifat acak dan sengaja dimodifikasi untuk melihat pengaruhnya terhadap hasil enkripsi. Setelah *plaintext* dimodifikasi, proses enkripsi dilakukan kembali dengan konfigurasi algoritma dan kunci yang sama, sehingga diperoleh ciphertext kedua. Kedua ciphertext — baik sebelum maupun sesudah modifikasi — kemudian dibandingkan untuk menghitung jumlah bit yang berbeda (Δ

Bit). Persentase perubahan bit dihitung dengan membandingkan jumlah perbedaan tersebut terhadap total panjang bit ciphertext, dan hasilnya dinyatakan dalam bentuk persentase menggunakan rumus *Avalanche Effect*.

$$Avalanche\ Effect\ =\ (\Delta\ Bit)/n\ \times 100\%$$

4. Repetisi dan Analisis Hasil : Pengujian ini dilakukan berulang kali dengan sampel data acak untuk masing-masing metode *padding* agar diperoleh hasil yang lebih valid dan representatif. Setiap hasil *Avalanche Effect* dianalisis berdasarkan nilai rata-rata, nilai minimum, maksimum, dan median untuk masing-masing metode *padding*. Selain itu, hasil pengujian divisualisasikan dalam bentuk diagram batang guna memberikan gambaran yang lebih jelas terhadap perbedaan difusi antara *padding Fixed Length* 512-bit dan PKCS#7.

3.4.4 Pengujian Lama Waktu Enkripsi dan Dekripsi

Uji waktu enkripsi dan dekripsi dilakukan untuk menilai seberapa efektif algoritma dalam proses mengenkripsi dan mendekripsi data. Ini sangat penting untuk aplikasi yang membutuhkan reaksi cepat, terutama untuk sistem yang menangani banyak data.

Langkah-langkah untuk menguji waktu enkripsi dan dekripsi adalah sebagai berikut:

1. Persiapan Data : Data *plaintext* dipersiapkan dengan ukuran yang bervariasi, misalnya 1 KB, 10 KB, dan seterusnya, untuk melihat pengaruh ukuran data terhadap waktu proses.

2. Proses Enkripsi : Data *plaintext* dienkripsi menggunakan algoritma AES diikuti dengan *Reverse Cipher*. Waktu yang dibutuhkan untuk menghasilkan *ciphertext* diukur.
3. Proses Dekripsi : *Ciphertext* yang dihasilkan dari proses enkripsi kemudian didekripsi menggunakan *Reverse Cipher* diikuti dengan AES untuk mengembalikan ke bentuk *plaintext*. Waktu yang dibutuhkan untuk proses dekripsi juga diukur.
4. Pengukuran Waktu : Waktu enkripsi dan dekripsi diukur dalam satuan milidetik (ms) atau detik (s). Pengujian dilakukan untuk berbagai ukuran data untuk mendapatkan gambaran tentang seberapa cepat algoritma bekerja dalam kondisi yang berbeda.
5. Evaluasi : Waktu yang digunakan untuk proses enkripsi dan dekripsi dibandingkan dengan batasan yang dapat diterima dalam sistem nyata. Kecepatan proses yang efektif akan berdampak pada kinerja keseluruhan sistem.

BAB IV

HASIL DAN PEMBAHASAN

4.1 Proses Enkripsi dan Dekripsi Material SAP

Hasil Implementasi Enkripsi Implementasi enkripsi data pada penelitian ini dilakukan terhadap file *Material SAP* dengan menggunakan kombinasi algoritma *Reverse Cipher* dan AES-128 dalam mode ECB (*Electronic Code Book*). Data yang diproses terdiri dari lima kolom utama yaitu *GroupDesc*, *Customer Name*, *MaterialNumber*, *Catalog Data*, dan *MaterialDesc*. Hasil dari proses enkripsi menunjukkan bahwa setiap data *plaintext* berhasil diubah menjadi *ciphertext* dengan representasi heksadesimal yang kompleks.

Setelah dilakukan pembalikan karakter menggunakan *Reverse Cipher*, data kemudian diproses menggunakan AES-128 ECB. Proses ini memastikan bahwa setiap blok data sepanjang 128 bit diproses secara independen menggunakan kunci rahasia sepanjang 16 *byte*. Hasil enkripsi disimpan dalam file Excel terpisah yang terdiri dari tiga lembar:

1. Lembar 1: Data asli (*Original Data*)
2. Lembar 2: Data setelah *Reverse Cipher*
3. Lembar 3: Data terenkripsi dengan AES-128

4.1.1 Proses Enkripsi Algoritma *Reverse Cipher* dan AES 128 *byte*

1. Menyiapkan teks asli (*plaintext*) yang akan dienkripsi.

   Pada tahap awal, terdapat sepuluh entri data material SAP yang perlu diproses. Namun, saat proses penginputan awal, hanya informasi penting

seperti *Material Number*, *Group Description*, *Material Description*, dan *Catalog Data* yang dikumpulkan. Setelah seluruh data material berhasil disusun, data tersebut kemudian digabungkan ke dalam satu file Excel (.xlsx) untuk keperluan proses enkripsi lebih lanjut. Penelitian ini secara khusus hanya akan memfokuskan pada enkripsi *plaintext* berupa *GroupDesc*, *MaterialNumber*, *Catalog Data*, dan *MaterialDesc*, dan tidak mencakup jenis data lain seperti *Stock/Non Stock*, *Material Group*, *Brand*, maupun jenis data lainnya. Secara umum, proses enkripsi dilakukan terhadap setiap baris pada file yang berisi data tekstual material SAP. Namun, karena proses dilakukan secara manual untuk keperluan analisis, maka hanya sebagian data teks yang diambil sebagai contoh. Adapun teks yang digunakan dalam proses enkripsi ditampilkan sebagai berikut:

Customer Name : ALFITRAHSURABAYA

GroupDesc : Router

MaterialNumber : 1002050364

Catalog Data : ROUT,CRS-8/S-B,,CISCO

MaterialDesc : CISCO

Namun demikian, dalam contoh perhitungan enkripsi manual, hanya bagian *Customer Name* yang digunakan, yaitu: ALFITRAHSURABAYA.

2. Menambahkan *padding* agar sesuai dengan ukuran blok.

Pastikan ukuran plainteks merupakan kelipatan dari 16 *byte*. Jika tidak, maka tambahkan *padding* agar sesuai dengan ukuran blok yang diperlukan. Dalam proses enkripsi ini, karena AES-128 bit dirancang untuk memproses blok penuh, maka jika panjang teks tidak sesuai dengan

kelipatan 16 *byte*, blok terakhir tidak dapat dienkripsi dengan benar. Oleh karena itu, dilakukan penambahan *padding* untuk menyesuaikan ukuran blok terakhir agar menjadi kelipatan 16 *byte*. Nilai *padding* diwakili oleh *byte* heksadesimal yang menunjukkan jumlah *byte* yang ditambahkan. Sebagai contoh, jika diperlukan 7 *byte* tambahan untuk melengkapi blok terakhir, maka setiap *byte padding* akan memiliki nilai 07, yang menunjukkan bahwa jumlah *byte padding* yang ditambahkan adalah 7 *byte*, sehingga memastikan teks dapat dienkripsi dengan benar menggunakan algoritma AES-128. Sebagai contoh, misalnya plainteks yang akan dienkripsi adalah "HELLO WORLD", yang memiliki panjang 11 *byte*. Karena AES-128 bit membutuhkan blok sebesar 16 *byte*, maka diperlukan penambahan *padding* sebanyak 5 *byte* (16 - 11 = 5). Maka, akan ditambahkan 5 *byte padding* dengan nilai 05 di akhir teks, sehingga menjadi:

"HELLO WORLD" + \x05\x05\x05\x05\x05

Dalam representasi heksadesimal ASCII, hasil akhirnya adalah:

48 45 4C 4C 4F 20 57 4F 52 4C 44 05 05 05 05 05

Dengan demikian, panjang *plainteks* menjadi 16 *byte* dan dapat diproses secara sempurna oleh algoritma AES-128 bit dalam satu blok enkripsi.

Pada tahap ini untuk *padding* yang dibuat contoh *Padding* PKCS#7.

3. Membalik teks menggunakan *Reverse Cipher*.

Setelah teks disiapkan, dilakukan pembalikan teks menggunakan *Reverse Cipher*. Setiap karakter dalam *plaintext* akan dibalik urutannya agar lebih

sulit dikenali sebelum proses enkripsi AES-128 bit dilakukan. Hasil dari *Reverse Cipher* adalah sebagai berikut:

Customer Name : AYABARUS HARTIF LA

Selanjutnya, teks hasil *Reverse Cipher* akan disesuaikan dengan *padding* agar sesuai dengan blok 16 *byte* yang diperlukan oleh AES-128. *Padding* ditambahkan jika panjang teks tidak sesuai dengan kelipatan 16 *byte* agar dapat dienkripsi dengan benar.

4. Pisahkan *plainteks* menjadi blok-blok berukuran 16 *byte*.

Karena enkripsi menggunakan algoritma AES-128 bit, yang bekerja dengan blok berukuran 128 bit (16 *byte* per blok). Jika panjang teks lebih dari 16 *byte*, maka teks akan dibagi menjadi beberapa blok, dengan setiap blok berukuran 16 *byte*. Pada proses ini, teks yang telah dibalik menggunakan *Reverse Cipher* dibagi menjadi blok-blok sebagai berikut:

Blok 1 (AYABARUSHARTIFLA)

Setiap karakter dikonversikan ke dalam kode ASCII, kemudian direpresentasikan dalam bentuk heksadesimal. Dengan konversi ini, seluruh teks *plaintext* berubah menjadi deretan nilai heksadesimal *byte* menjadi:

Hex: [41 59 41 42 41 52 55 53 48 41 52 54 49 46 4C 41]

Setiap blok tersebut kemudian diubah menjadi matriks 4×4, yang akan diproses dalam langkah-langkah enkripsi AES-128. Jika blok terakhir tidak memiliki panjang 16 *byte*, maka dilakukan *padding* untuk memastikan ukuran blok sesuai dengan kelipatan 16 *byte*, sehingga algoritma AES-128 bit dapat mengenkripsi teks dengan benar.

5. Kunci yang digunakan dalam proses enkripsi.

Setelah teks dibalik, langkah berikutnya adalah menentukan kunci enkripsi untuk AES-128 bit. Kunci yang digunakan dalam proses enkripsi adalah "KRIPTOGRAFIAESKU", yang memiliki panjang 16 *byte*. Jika panjang kunci lebih dari 16 karakter, hanya 16 karakter pertama yang akan digunakan dan dikonversi ke dalam bentuk heksadesimal sebagai berikut:

$$\mathrm{K} = \begin{bmatrix} 4B & 54 & 41 & 45 \\ 52 & 4F & 46 & 53 \\ 49 & 47 & 49 & 4B \\ 50 & 52 & 41 & 55 \end{bmatrix}$$

Kunci utama yang telah dikonversi ke dalam bentuk heksadesimal digunakan sebagai *Round Key 0*. Untuk mendapatkan *round key* berikutnya, kunci utama dibagi menjadi empat *word* awal, yaitu W[0] hingga W[3]. Selanjutnya, untuk menghasilkan *word* baru (W[i]), digunakan metode ekspansi kunci (*Key Expansion*) dengan beberapa tahapan. Pertama, *word* pertama dari setiap *round key* (W[i]) dihitung dengan cara melakukan *RotWord*, yaitu menggeser *byte* dari *word* terakhir pada *round* sebelumnya ke kiri. Hasil dari *RotWord* kemudian diproses menggunakan *SubWord*, yaitu mengganti setiap *byte* dengan nilai yang sesuai dari tabel S-Box. Setelah itu, hasil *SubWord* di-*XOR* dengan konstanta *Rcon* yang unik untuk setiap *round*. Terakhir, hasilnya di-*XOR* kembali dengan W[i-4], yaitu *word* pertama dari *round* sebelumnya, untuk mendapatkan W[i]. Untuk menghasilkan *word* selanjutnya (W[i+1], W[i+2], dan W[i+3]), digunakan operasi *XOR* antara *word* sebelumnya dengan *word* yang berjarak empat langkah di belakangnya, yaitu W[i] = W[i-4] $\oplus$ W[i-1]. Proses ini diulang hingga terbentuk sebanyak 44 *word*

(W[0] hingga W[43]), di mana setiap kelompok 4 *word* akan membentuk satu *round key*, mulai dari *Round Key 0* hingga *Round Key 10*. Setelah *padding* selesai, teks akan dienkripsi menggunakan AES-128 dalam mode ECB (*Electronic Codebook*). Mengubah setiap blok menjadi matriks 4×4 yang akan diproses dalam AES-128.

6. Proses plainteks menjadi blok matriks.

   Setelah setiap blok data sepanjang 16 *byte* (128 bit) siap, langkah selanjutnya adalah menyusun *byte-byte* tersebut ke dalam sebuah matriks berdimensi 4×4 yang disebut sebagai matriks State (State array). Matriks State ini adalah unit data fundamental yang akan dioperasikan oleh berbagai transformasi dalam setiap putaran algoritma AES.

   Proses pembentukan matriks State dari sebuah blok input 16 *byte* (misalnya, direpresentasikan sebagai $p_0, p_1, p_2, \dots, p_{15}$) mengikuti aturan column-major order. Artinya, *byte-byte* dari blok input tersebut dimasukkan ke dalam matriks State kolom demi kolom. Empat *byte* pertama dari blok input (p_0, p_1, p_2, p_3) mengisi kolom pertama matriks State. Empat *byte* berikutnya (p_4, p_5, p_6, p_7) mengisi kolom kedua, dan begitu seterusnya hingga seluruh 16 *byte* termuat dalam matriks.

   Secara formal, jika blok data 16 *byte* adalah:

   $$P = [p_0, p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8, p_9, p_{10}, p_{11}, p_{12}, p_{13}, p_{14}, p_{15}]$$

   Maka, matriks State S (4×4 *byte*) yang bersesuaian akan terbentuk sebagai berikut:

   $$S = \begin{bmatrix} p_0 & p_4 & p_8 & p_{12} \\ p_1 & p_5 & p_9 & p_{13} \\ p_2 & p_6 & p_{10} & p_{14} \\ p_3 & p_7 & p_{11} & p_{15} \end{bmatrix}$$

Sebagai contoh, untuk "Blok 1" yang berasal dari teks "AYABARUSHARTIFLA" dan telah dikonversi menjadi urutan *byte* heksadesimal:

$[41 \quad 59 \quad 41 \quad 42 \quad 41 \quad 52 \quad 55 \quad 53 \quad 48 \quad 41 \quad 52 \quad 54 \quad 49 \quad 46 \quad 4C \quad 41]$

Mengikuti aturan pemetaan di atas:

$p_0 = 41\text{H}$ (elemen $S[0,0]$)

$p_1 = 59\text{H}$ (elemen $S[1,0]$)

$p_2 = 41\text{H}$ (elemen $S[2,0]$)

$p_3 = 42\text{H}$ (elemen $S[3,0]$)

$p_4 = 41\text{H}$ (elemen $S[0,1]$)

$p_5 = 52\text{H}$ (elemen $S[1,1]$)

dan seterusnya hingga $p_{15} = 41\text{H}$ (elemen $S[3,3]$).

Dengan pemetaan ini, matriks M_1 yang akan Anda sajikan terbentuk secara logis dari urutan *byte* tersebut. Proses inilah yang mengubah representasi linear blok data menjadi struktur matriks 4×4 yang siap untuk diproses lebih lanjut oleh langkah-langkah enkripsi AES seperti Sub*Byte*s, ShiftRows, MixColumns, dan AddRoundKey. Representasi matriks ini krusial karena transformasitransformasi dalam AES dirancang untuk beroperasi pada struktur dua dimensi ini guna mencapai difusi dan kebingungan (confusion) yang efektif. Maka proses blok 1menjadi dalam bentuk matriks:

$$\text{M}_1 = \begin{bmatrix} 41 & 41 & 48 & 49 \\ 59 & 52 & 41 & 46 \\ 41 & 55 & 52 & 4C \\ 42 & 53 & 54 & 41 \end{bmatrix}$$

7. Melakukan proses enkripsi AES-128 dengan perhitungan manual pada setiap blok:

   Langkah 1: AddRoundKey

   Sebagai contoh dalam pengambilan sampel, blok pertama digunakan sebagai representasi. Setiap *byte* dalam blok pertama dijumlahkan dengan *byte* kunci menggunakan operasi XOR:

   Blok 1 dalam bentuk matriks:

$$M_1 = \begin{bmatrix} 41 & 41 & 48 & 49 \\ 59 & 52 & 41 & 46 \\ 41 & 55 & 52 & 4C \\ 42 & 53 & 54 & 41 \end{bmatrix}$$

   Kunci yang digunakan:

$$K = \begin{bmatrix} 4B & 54 & 41 & 45 \\ 52 & 4F & 46 & 53 \\ 49 & 47 & 49 & 4B \\ 50 & 52 & 41 & 55 \end{bmatrix}$$

   Dengan melakukan operasi XOR antara setiap elemen matriks blok ($M_{1)}$ dan matriks kunci (K), maka hasilnya sebagai berikut:

$$\text{Hasil XOR} = \begin{bmatrix} 41 \oplus 4B & 41 \oplus 54 & 48 \oplus 41 & 49 \oplus 45 \\ 59 \oplus 52 & 52 \oplus 4F & 42 \oplus 46 & 46 \oplus 53 \\ 41 \oplus 49 & 55 \oplus 47 & 52 \oplus 49 & 4C \oplus 4B \\ 42 \oplus 50 & 53 \oplus 52 & 54 \oplus 41 & 41 \oplus 55 \end{bmatrix}$$

   Menghitung nilai XOR satu persatu:

   Baris 1:

$$41 \oplus 4B = 01000001 \oplus 01001011 = 00001010 = 0A$$

$$59 \oplus 52 = 01011001 \oplus 01010010 = 00001011 = 0B$$

$$41 \oplus 49 = 01000001 \oplus 01001001 = 00001000 = 08$$

$$42 \oplus 50 = 01000010 \oplus 01010000 = 00010010 = 12$$

Baris 2:

$$41 \oplus 54 = 01000001 \oplus 01010100 = 00010101 = 15$$

$$52 \oplus 4F = 01010010 \oplus 01001111 = 00011101 = 1D$$

$$55 \oplus 47 = 01010101 \oplus 01000111 = 00010010 = 12$$

$$53 \oplus 52 = 01010011 \oplus 01010010 = 00000001 = 01$$

Baris 3:

$$48 \oplus 41 = 01001000 \oplus 01000001 = 00001001 = 09$$

$$42 \oplus 46 = 01000010 \oplus 01000110 = 00000100 = 04$$

$$52 \oplus 49 = 01010010 \oplus 01001001 = 00011011 = 1B$$

$$54 \oplus 41 = 01010100 \oplus 01000001 = 00010101 = 15$$

Baris 4:

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

$$46 \oplus 53 = 01000110 \oplus 01010011 = 00010101 = 15$$

$$4C \oplus 4B = 01001100 \oplus 01001011 = 00000111 = 07$$

$$41 \oplus 55 = 01000001 \oplus 01010101 = 00010100 = 14$$

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

$$S = \begin{bmatrix} 0A & 15 & 09 & 0C \\ 0B & 1D & 04 & 15 \\ 08 & 12 & 1B & 07 \\ 12 & 01 & 15 & 14 \end{bmatrix}$$

Langkah 2: Sub*Byte*s

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 $S_1[r,c]$ yang merupakan potongan baris x dengan kolom y di dalam S-box (Gambar 2.8). Menggunakan matriks hasil *AddRoundKey* awal:

$$S = \begin{bmatrix} 0A & 15 & 09 & 0C \\ 0B & 1D & 04 & 15 \\ 08 & 12 & 1B & 07 \\ 12 & 01 & 15 & 14 \end{bmatrix}$$

elemen pertama, 0A, disubstitusikan dengan elemen pada perpotongan baris 0 dan kolom 11 pada S-box, yaitu 67. Jadi, operasi substitusi pada $S[r,c] = 0A$ menghasilkan $S_1[r,c] = 67$. Jika operasi ini diterapkan untuk semua matriks lainnya, maka hasilnya adalah:

$$S = \begin{bmatrix} 0A & 15 & 09 & 0C \\ 0B & 1D & 04 & 15 \\ 08 & 12 & 1B & 07 \\ 12 & 01 & 15 & 14 \end{bmatrix} \Rightarrow S_1 = \begin{bmatrix} 67 & 59 & 01 & FE \\ 2B & A4 & F2 & 59 \\ 30 & C9 & AF & C5 \\ C9 & 7C & 59 & FA \end{bmatrix}$$

Langkah 3: ShiftRows

Menggunakan matrik hasil Sub*Byte*s,

Baris 0: [67, 59, 01, FE], tidak digeser ke kiri.

Baris 1: [2B, A4, F2, 59], digeser satu *byte* ke kiri menjadi

[A4, F2, 59, 2B].

Baris 2: [30, C9, AF, C5], geser dua *byte* ke kiri menjadi

[AF, C5, 30, C9]

Baris 3: [C9, 7C, 59, FA], geser tiga *byte* ke kiri menjadi

[FA, C9, 7C, 59]

Sehingga hasilnya adalah:

$$S_1 = \begin{bmatrix} 67 & 59 & 01 & FE \\ 2B & A4 & F2 & 59 \\ 30 & C9 & AF & C5 \\ C9 & 7C & 59 & FA \end{bmatrix} \Rightarrow S_2 = \begin{bmatrix} 67 & 59 & 01 & FE \\ A4 & F2 & 59 & 2B \\ AF & C5 & 30 & C9 \\ FA & C9 & 7C & 59 \end{bmatrix}$$

Langkah 4: *MixColumns*

$$S_2 = \begin{bmatrix} 67 & 59 & 01 & FE \\ A4 & F2 & 59 & 2B \\ AF & C5 & 30 & C9 \\ FA & C9 & 7C & 59 \end{bmatrix}$$

Dengan memakai matriks hasil *ShiftRows* di atas, kemudian dilakukan perkalian matriks *MixColumns*, maka menghasilkan:

Operasi *MixColumns* terhadap kolom pertama

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} 67 \\ A4 \\ AF \\ FA \end{bmatrix} = \begin{bmatrix} 6C \\ 24 \\ 93 \\ 4D \end{bmatrix}$$

Perhitungan elemen pertama (baris 1, kolom 1):

$(02 \cdot 67) \oplus (03 \cdot A4) \oplus (01 \cdot AF) \oplus (01 \cdot FA)$

= E(L(02) + L(67)) $\oplus$ E(L(03) + L(A4)) $\oplus$ E(L(01) + L(AF)) $\oplus$

E(L(01) + L(FA))

= E(CE) $\oplus$ E(F7) $\oplus$ E(AF) $\oplus$ E(FA)

= 11001110 $\oplus$ 11110111 $\oplus$ 10101111 $\oplus$ 11111010

= 01101100 (Biner)

= 6C (Hex)

Perhitungan elemen kedua (baris 2, kolom 1):

$(01 \cdot 67) \oplus (02 \cdot A4) \oplus (03 \cdot AF) \oplus (01 \cdot FA)$

= E(L(01) + L(67)) $\oplus$ E(L(02) + L(A4)) $\oplus$ E(L(03) + L(AF)) $\oplus$

E(L(01) + L(FA))

= E(67) $\oplus$ E(CE) $\oplus$ E(40) $\oplus$ E(FA)

= 01100111 ⊕ 11001110 ⊕ 01000000 ⊕ 11111010

= 00100100 (Biner)

= 24 (Hex)

Perhitungan elemen ketiga (baris 3, kolom 1):

(01 · 67) ⊕ (01 · A4) ⊕ (02 · AF) ⊕ (03 · FA)

= E(L(01) + L(67)) ⊕ E(L(01) + L(A4)) ⊕ E(L(02) + L(AF)) ⊕

E(L(03) + L(FA))

= E(67) ⊕ E(A4) ⊕ E(5E) ⊕ E(44)

= 01100111 ⊕ 10100100 ⊕ 01011110 ⊕ 01000100

= 10010011 (Biner)

= 93 (Hex)

Perhitungan elemen keempat (baris 4, kolom 1):

(03 · 67) ⊕ (01 · A4) ⊕ (01 · AF) ⊕ (02 · FA)

= E(L(03) + L(67)) ⊕ E(L(01) + L(A4)) ⊕ E(L(01) + L(AF)) ⊕

E(L(02) + L(FA))

= E(25) ⊕ E(A4) ⊕ E(AF) ⊕ E(EF)

= 00100101 ⊕ 10100100 ⊕ 10101111 ⊕ 11101111

= 01001101 (Biner)

= 4D (Hex)

Operasi *MixColumns* terhadap kolom kedua

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} 59 \\ C5 \\ C5 \\ C9 \end{bmatrix} = \begin{bmatrix} EA \\ 55 \\ 4D \\ 62 \end{bmatrix}$$

Perhitungan elemen pertama (baris 1, kolom 2):

(02 · 59) ⊕ (03 · C5) ⊕ (01 · C5) ⊕ (01 · C9)

= E(L(02) + L(59)) ⊕ E(L(03) + L(C5)) ⊕ E(L(01) + L(C5)) ⊕

E(L(01) + L(C9))

= E(B2) ⊕ E(64) ⊕ E(C5) ⊕ E(C9)

= 10110010 ⊕ 01100100 ⊕ 11000101 ⊕ 11001001

= (10110010 ⊕ 01100100) ⊕ (11000101 ⊕ 11001001)

= 11010110 ⊕ 00001100

= 11101010 (Biner)

= EA (Hex)

Perhitungan elemen kedua (baris 2, kolom 2):

(01 · 59) ⊕ (02 · C5) ⊕ (03 · C5) ⊕ (01 · C9)

= E(L(01) + L(59)) ⊕ E(L(02) + L(C5)) ⊕ E(L(03) + L(C5)) ⊕

E(L(01) + L(C9))

= E(59) ⊕ E(8A) ⊕ E(64) ⊕ E(C9)

= 01011001 ⊕ 10001010 ⊕ 01100100 ⊕ 11001001

= (01011001 ⊕ 10001010) ⊕ (01100100 ⊕ 11001001)

= 11010011 ⊕ 10101101

= 01010101 (Biner)

= 55 (Hex)

Perhitungan elemen ketiga (baris 3, kolom 2):

(01 · 59) ⊕ (01 · C5) ⊕ (02 · C5) ⊕ (03 · C9)

= E(L(01) + L(59)) ⊕ E(L(01) + L(C5)) ⊕ E(L(02) + L(C5)) ⊕

E(L(03) + L(C9))

= E(59) ⊕ E(C5) ⊕ E(8A) ⊕ E(33)

= 01011001 ⊕ 11000101 ⊕ 10001010 ⊕ 00110011

= (01011001 ⊕ 11000101) ⊕ (10001010 ⊕ 00110011)

= 10011100 ⊕ 10111001

= 01001101 (Biner)

= 4D (Hex)

Perhitungan elemen keempat (baris 4, kolom 2):

(03 · 59) ⊕ (01 · C5) ⊕ (01 · C5) ⊕ (02 · C9)

= E(L(03) + L(59)) ⊕ E(L(01) + L(C5)) ⊕ E(L(01) + L(C5)) ⊕

E(L(02) + L(C9))

= E(14) ⊕ E(C5) ⊕ E(C5) ⊕ E(92)

= 00010100 ⊕ 11000101 ⊕ 11000101 ⊕ 10010010

= (00010100 ⊕ 11000101) ⊕ (11000101 ⊕ 10010010)

= 11010001 ⊕ 01010111

= 01100010 (Biner)

= 62 (Hex)

Operasi *MixColumns* terhadap kolom ketiga

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} 01 \\ 59 \\ 30 \\ 7C \end{bmatrix} = \begin{bmatrix} A5 \\ 9F \\ BC \\ 92 \end{bmatrix}$$

Perhitungan elemen pertama (baris 1, kolom 3):

(02 · 01) ⊕ (03 · 59) ⊕ (01 · 30) ⊕ (01 · 7C)

= E(L(02) + L(01)) ⊕ E(L(03) + L(59)) ⊕ E(L(01) + L(30)) ⊕ E(L(01)

+ L(7C))

= E(02) ⊕ E(F2) ⊕ E(30) ⊕ E(7C)

= 00000010 ⊕ 11110010 ⊕ 00110000 ⊕ 01111100

= (00000010 ⊕ 11110010) ⊕ (00110000 ⊕ 01111100)

= 11110000 ⊕ 01001100

= 10111100 (Biner)

= A5 (Hex)

Perhitungan elemen kedua (baris 2, kolom 3):

(01 · 01) ⊕ (02 · 59) ⊕ (03 · 30) ⊕ (01 · 7C)

= E(L(01) + L(01)) ⊕ E(L(02) + L(59)) ⊕ E(L(03) + L(30)) ⊕ E(L(01) + L(7C))

= E(01) ⊕ E(B2) ⊕ E(90) ⊕ E(7C)

= 00000001 ⊕ 10110010 ⊕ 10010000 ⊕ 01111100

= (00000001 ⊕ 10110010) ⊕ (10010000 ⊕ 01111100)

= 10110011 ⊕ 11101100

= 01011111 (Biner)

= 9F (Hex)

Perhitungan elemen ketiga (baris 3, kolom 3):

(01 · 01) ⊕ (01 · 59) ⊕ (02 · 30) ⊕ (03 · 7C)

= E(L(01) + L(01)) ⊕ E(L(01) + L(59)) ⊕ E(L(02) + L(30)) ⊕ E(L(03) + L(7C))

= E(01) ⊕ E(59) ⊕ E(60) ⊕ E(E4)

= 00000001 ⊕ 01011001 ⊕ 01100000 ⊕ 11100100

= (00000001 ⊕ 01011001) ⊕ (01100000 ⊕ 11100100)

= 01011000 ⊕ 10000100

= 11011100 (Biner)

= BC (Hex)

Perhitungan elemen keempat (baris 4, kolom 3):

$(03 \cdot 01) \oplus (01 \cdot 59) \oplus (01 \cdot 30) \oplus (02 \cdot 7C)$

= E(L(03) + L(01)) ⊕ E(L(01) + L(59)) ⊕ E(L(01) + L(30)) ⊕ E(L(02) + L(7C))

= E(03) ⊕ E(59) ⊕ E(30) ⊕ E(F8)

= 00000011 ⊕ 01011001 ⊕ 00110000 ⊕ 11111000

= (00000011 ⊕ 01011001) ⊕ (00110000 ⊕ 11111000)

= 01011010 ⊕ 11001000

= 10010010 (Biner)

= 92 (Hex)

Operasi *MixColumns* terhadap kolom keempat

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} FE \\ 2B \\ C9 \\ 59 \end{bmatrix} = \begin{bmatrix} 0A \\ B1 \\ B7 \\ 49 \end{bmatrix}$$

Perhitungan elemen pertama (baris 1, kolom 4):

$(02 \cdot FE) \oplus (03 \cdot 2B) \oplus (01 \cdot C9) \oplus (01 \cdot 59)$

= E(L(02) + L(FE)) ⊕ E(L(03) + L(2B)) ⊕ E(L(01) + L(C9)) ⊕ E(L(01) + L(59))

= E(E1) ⊕ E(56) ⊕ E(C9) ⊕ E(59)

= 11100001 ⊕ 01010110 ⊕ 11001001 ⊕ 01011001

= (11100001 ⊕ 01010110) ⊕ (11001001 ⊕ 01011001)

= 10110111 ⊕ 10010000

= 00101001 (Biner)

= 0A (Hex)

Perhitungan elemen kedua (baris 2, kolom 4):

(01 · FE) ⊕ (02 · 2B) ⊕ (03 · C9) ⊕ (01 · 59)

= E(L(01) + L(FE)) ⊕ E(L(02) + L(2B)) ⊕ E(L(03) + L(C9)) ⊕ E(L(01) + L(59))

= E(FE) ⊕ E(56) ⊕ E(8A) ⊕ E(59)

= 11111110 ⊕ 01010110 ⊕ 10001010 ⊕ 01011001

= (11111110 ⊕ 01010110) ⊕ (10001010 ⊕ 01011001)

= 10101000 ⊕ 11010011

= 01111011 (Biner)

= B1 (Hex)

Perhitungan elemen ketiga (baris 3, kolom 4):

(01 · FE) ⊕ (01 · 2B) ⊕ (02 · C9) ⊕ (03 · 59)

= E(L(01) + L(FE)) ⊕ E(L(01) + L(2B)) ⊕ E(L(02) + L(C9)) ⊕ E(L(03) + L(59))

= E(FE) ⊕ E(2B) ⊕ E(92) ⊕ E(06)

= 11111110 ⊕ 00101011 ⊕ 10010010 ⊕ 00000110

= (11111110 ⊕ 00101011) ⊕ (10010010 ⊕ 00000110)

= 11010101 ⊕ 10010100

= 01000001 (Biner)

= B7 (Hex)

Perhitungan elemen keempat (baris 4, kolom 4):

(03 · FE) ⊕ (01 · 2B) ⊕ (01 · C9) ⊕ (02 · 59)

= E(L(03) + L(FE)) ⊕ E(L(01) + L(2B)) ⊕ E(L(01) + L(C9)) ⊕ E(L(02) + L(59))

= E(07) ⊕ E(2B) ⊕ E(C9) ⊕ E(B2)

= 00000111 ⊕ 00101011 ⊕ 11001001 ⊕ 10110010

= (00000111 ⊕ 00101011) ⊕ (11001001 ⊕ 10110010)

= 00101100 ⊕ 01111011

= 01010111 (Biner)

= 49 (Hex)

Hasil transformasi *MixColumns* secara keseluruhan adalah:

$$S_3 = \begin{bmatrix} 6C & EA & A5 & 0A \\ 24 & 55 & 9F & B1 \\ 93 & 4D & BC & B7 \\ 4D & 62 & 92 & 49 \end{bmatrix}$$

Langkah 5: AddRoundKey

Menggunakan sebuah *round key* adalah:

$$K_1 = \begin{bmatrix} A7 & F3 & B2 & F7 \\ E1 & AE & E8 & BB \\ B5 & F2 & BB & F0 \\ 3E & 6C & 2D & 78 \end{bmatrix}$$

dan hasil MixColumns sebelumnya adalah:

$$S_3 = \begin{bmatrix} 6C & EA & A5 & 0A \\ 24 & 55 & 9F & B1 \\ 93 & 4D & BC & B7 \\ 4D & 62 & 92 & 49 \end{bmatrix}$$

Maka AddRoundKey menghasilkan:

$$\begin{bmatrix} 6C & EA & A5 & 0A \\ 24 & 55 & 9F & B1 \\ 93 & 4D & BC & B7 \\ 4D & 62 & 92 & 49 \end{bmatrix} \oplus \begin{bmatrix} A7 & F3 & B2 & F7 \\ E1 & AE & E8 & BB \\ B5 & F2 & BB & F0 \\ 3E & 6C & 2D & 78 \end{bmatrix} = \begin{bmatrix} CB & 19 & 17 & FD \\ C5 & FB & 77 & 0A \\ 26 & BF & 07 & 47 \\ 73 & 0E & BF & 31 \end{bmatrix}$$

Maka hasil *AddRoundKey:*

$$S_4 = \begin{bmatrix} CB & 19 & 17 & FD \\ C5 & FB & 77 & 0A \\ 26 & BF & 07 & 47 \\ 73 & 0E & BF & 31 \end{bmatrix}$$

Langkah 6: Ulangi proses selama 9 *round* untuk AES-128.

Putaran pertama: $S_4 = P_1 = \begin{bmatrix} CB & 19 & 17 & FD \\ C5 & FB & 77 & 0A \\ 26 & BF & 07 & 47 \\ 73 & 0E & BF & 31 \end{bmatrix}$

Putaran kedua: $P_2 = \begin{bmatrix} 62 & 24 & 0D & A0 \\ FF & 92 & BC & 83 \\ DA & 0B & C4 & 4A \\ 28 & 0D & C1 & 2D \end{bmatrix}$

Putaran ketiga: $P_3 = \begin{bmatrix} 71 & B6 & F0 & D2 \\ 42 & E0 & EC & 3B \\ 0F & E2 & 59 & 82 \\ D2 & 74 & 49 & 7B \end{bmatrix}$

Putaran keempat: $P_4 = \begin{bmatrix} 25 & 4E & 7E & 62 \\ 3B & 26 & 56 & 9A \\ 50 & C4 & 93 & E9 \\ 89 & 94 & 2C & 83 \end{bmatrix}$

Putaran kelima: $P_5 = \begin{bmatrix} E0 & BE & 6A & EE \\ 38 & DD & 4C & DC \\ 2F & 9D & FA & 84 \\ AA & 62 & 40 & 9D \end{bmatrix}$

Putaran keenam: $P_6 = \begin{bmatrix} CE & 62 & B2 & 67 \\ B5 & 5B & 69 & D8 \\ B0 & 33 & DF & 7B \\ CA & 97 & 70 & FD \end{bmatrix}$

Putaran ketujuh: $P_7 = \begin{bmatrix} A9 & 82 & B2 & 9E \\ FD & 57 & A5 & C4 \\ CA & CB & D2 & 1F \\ D2 & C5 & 6E & 54 \end{bmatrix}$

Putaran kedelapan : $P_8 = \begin{bmatrix} 76 & 62 & B3 & 38 \\ 95 & E6 & C8 & 1F \\ A8 & 3D & 97 & 5F \\ 2D & 7F & 21 & 40 \end{bmatrix}$

Putaran kesembilan: $P_9 = \begin{bmatrix} A7 & 9F & 5E & 37 \\ FE & AF & D8 & 64 \\ 54 & 31 & 7F & DB \\ B1 & 79 & 5D & 81 \end{bmatrix}$

8. Pada putaran terakhir tidak melakukan *MixColumns*

   Hanya melakukan Sub*Byte*s, *ShiftRows*, dan *AddRoundKey*.

   Putaran terakhir: $P_{10} = \begin{bmatrix} BF & DC & 90 & CF \\ 7B & 3F & BC & E7 \\ 63 & 8D & 34 & 77 \\ BF & 6B & 42 & DC \end{bmatrix}$

   Sehingga putaran terakhir ini dapat disebut sebagai blok terenkripsi untuk semua data pada baris pertama.

   a. Customer Name

      Plaintext Asli: ALFITRAHSURABAYA

      Plaintext Setelah *Reverse Cipher*: AYABARUSHARTIFLA

      Panjang *Plaintext* Reversed: 16 *byte* (sesuai ukuran blok AES, tidak memerlukan *padding*).

      Plaintext Reversed Heksadesimal:

      4159414241525553484152544946 4C41

      Ciphertext Heksadesimal (Hasil Enkripsi AES-128 ECB):

      BF7B63BFDC3F8D6B90BC3442CFE777DC

   b. GroupDesc

      Plaintext Asli: Router

      Plaintext Setelah *Reverse Cipher*: retuoR

      Panjang *Plaintext* Reversed: 6 *byte*.

      Padding yang Dibutuhkan: 10 *byte* (nilai setiap *byte padding*: 0A).

      Plaintext Reversed Heksadesimal (dengan *Padding* PKCS#7):

      726574756F520A0A0A0A0A0A0A0A0A0A

      Ciphertext Heksadesimal (Hasil Enkripsi AES-128 ECB):

      A94E44154794333882CD02374732209B

c. MaterialNumber

Plaintext Asli: 1002050364

Plaintext Setelah *Reverse Cipher*: 4630502001

Panjang *Plaintext* Reversed: 10 *byte*.

Padding yang Dibutuhkan: 6 *byte* (nilai setiap *byte padding*: 06).

Plaintext Reversed Heksadesimal (dengan *Padding* PKCS#7):

34363330353032303031060606060606

Ciphertext Heksadesimal (Hasil Enkripsi AES-128 ECB):

4CD4036F0C39E79B9E0DBDE782BA0F13

d. Catalog Data

Plaintext Asli: ROUT,CRS-8/S-B,,CISCO

Plaintext Setelah *Reverse Cipher*: OCSIC,,B-S/8-SR,TUOR

Panjang *Plaintext* Reversed: 20 *byte* (akan diproses dalam dua blok AES).

Blok 1:

Data Blok 1 (16 *byte*): OCSIC,,B-S/8-SR,T

Plaintext Heksadesimal (Blok 1):

4F435349432C2C422D532F382D53522C

Ciphertext Heksadesimal (Blok 1):

BDEE3765459391A46FD5543C60E07C9B

Blok 2:

Data Blok 2 (sisa 4 *byte*): UOR

Padding yang Dibutuhkan: 12 *byte* (nilai setiap *byte padding*: 0C).

Plaintext Heksadesimal (Blok 2, dengan *Padding* PKCS#7):

554F520C0C0C0C0C0C0C0C0C0C0C0C0C

Ciphertext Heksadesimal (Blok 2):

9581DBF5BD508E14F953F159AEF495A2

Ciphertext Heksadesimal Gabungan (Hasil Enkripsi AES-128 ECB):

BDEE3765459391A46FD5543C60E07C9B9581DBF5BD508E14F9
53F159AEF495A2

e. MaterialDesc

Plaintext Asli: CISCO

Plaintext Setelah *Reverse Cipher*: OCSIC

Panjang *Plaintext* Reversed: 5 *byte*.

Padding yang Dibutuhkan: 11 *byte* (nilai setiap *byte padding*: 0B).

Plaintext Reversed Heksadesimal (dengan *Padding* PKCS#7):

4F435349430B0B0B0B0B0B0B0B0B0B0B

Ciphertext Heksadesimal (Hasil Enkripsi AES-128 ECB):

C3A1D336B6808FDC1E321CBC740496D3

9. Hasil enkripsi kemudian disimpan dalam file Excel (.xlsx). File ini akan memuat:

a. *Plainteks* asli

	GroupDesc	Customer Name	MaterialNumber	Catalog Data	MaterialDesc
0	Router	ALFITRAHSURABAYA	1002050210	ROUT,CRS-8/S-B,,CISCO	CISCO
1	Router	SATNETBANDUNG	1002050364	ROUT,Card MPC7E-4X100G,,JUNIPER	JUNIPER
2	Router	DATANUSANTARA	1002050211	ROUT,ASR-9010,,CISCO	CISCO
3	Router	INDOTELEMEDIKA	1002050352	ROUT,NCS-55A2,MOD 4x100G,CISCO	CISCO
4	Router	MULTINETJAKARTA	1002050369	ROUT,NetEngine 8000 M8,DC,,HUAWEI	HUAWEI
5	Router	ANAGATANETLAMPUNG	1002050370	ROUT,NetEngine 8000 M14,DC,,HUAWEI	HUAWEI

Gambar 4.1.1 Gambar *Plainteks* Material SAP

b. Teks setelah di *Reverse Cipher*

	Gabungan Reverse
0	OCSIC || OCSIC,,B-S/8-SRC,TUOR || 0120502001 || AYABARUSHARTIFLA || retuoR
1	REPINUJ || REPINUJ,,G001X4-E7CPM draC,TUOR || 4630502001 || GNUDNABTENTAS || r
2	OCSIC || OCSIC,,0109-RSA,TUOR || 1120502001 || ARATNASUNATAD || retuoR
3	OCSIC || OCSIC,G001x4 DOM,2A55-SCN,TUOR || 2530502001 || AKIDEMELETODNI || retu
4	IEWAUH || IEWAUH,,CD,8M 0008 enignEteN,TUOR || 9630502001 || ATRAKAJTENITLUM
5	IEWAUH || IEWAUH,,CD,41M 0008 enignEteN,TUOR || 0730502001 || GNUPMALTENATAG

Gambar 4.1.2 Gambar *Plainteks* Setelah di *Reverse Cipher*

c. Hasil Enkripsi AES 128 Bit dalam format Hexadesimal

	AES Ciphertext
0	4c47e3d875f7aae82225b0ed9c4f1971e18dbbb22430444943414cd9ce028688ca1df32
1	372fc1edcb4595ec144b0812047042a714bddd5538a16ae7ee1a6802bd1af08b075ead
2	4c47e3d875f7aae82225b0ed9c4f1971ae815043fb021a2c2bd9a2463e761e8d584b0aa
3	b512efd5741ee769539c992977bde755234cb1f380eb331a3e965c26452cf5bd90085e5
4	803b4ce7e08c1ec2f045182957aa0d6b8906ef23e2515c0fb90cfe1d3697aead5c9b3a50
5	803b4ce7e08c1ec2f045182957aa0d6b7b1f2218cc2e2a04c0b16a4b51c97860c682aa8

Gambar 4.1.3 Gambar Hasil Enkripsi AES 128 bit

10. Verifikasi hasil di *website AES Encryption / Decryption Tool*

Sebagai langkah terakhir, dilakukan verifikasi hasil enkripsi pada *website AES Encryption / Decryption Tool* untuk memastikan bahwa hasil enkripsi dengan perhitungan manual sudah sesuai dengan hasil perhitungan di *website*. Teks telah berhasil dikonversi ke dalam format terenkripsi dengan aman. Dengan kombinasi *Reverse Cipher* dan AES-128 ECB Mode, data

yang telah dienkripsi akan lebih aman dan sulit untuk dibaca tanpa kunci yang sesuai.

Gambar 4.1.4 Pengecekan AES melalui website AES *Encryption* / *Decryption*

11. Penyimpanan File Hasil Enkripsi

Penyimpanan dilakukan ke dalam file berekstensi .xlsx dalam bentuk biner. Nama file hasil enkripsi dapat disesuaikan oleh pengguna, misalnya data_material_encrypted.xlsx, sesuai kebutuhan. Meskipun file disimpan dengan ekstensi .xlsx agar tetap dikenali sebagai file Excel, isinya sudah terenkripsi sehingga tidak dapat dibaca atau dimengerti tanpa proses dekripsi. Dengan cara ini, pengguna masih dapat membuka file tersebut menggunakan aplikasi pengolah *spreadsheet* seperti Microsoft Excel, namun isi yang tampil hanyalah karakter acak hasil enkripsi. Ini menunjukkan bahwa struktur file tetap utuh dan kompatibel, tetapi data di dalamnya aman dan terlindungi Proses penyimpanan dilakukan dengan kode berikut:

```
with open('data_material_encrypted.xlsx', 'wb') as f:
    f.write(encrypted_data)
```

Dengan menyimpan hasil enkripsi ke dalam ekstensi .xlsx, file tetap kompatibel secara eksternal, tetapi secara internal isinya tetap aman dari akses tidak sah tanpa kunci enkripsi yang tepat.

4.1.2 Proses Dekripsi Algoritma *Reverse Cipher* dan AES 128 bit

1. File hasil enkripsi

   Misalnya data_material_encrypted.xlsx, dibuka dalam mode biner (rb) untuk dibaca sebagai *byte* array. File ini berisi data terenkripsi yang tidak dapat dibaca secara langsung sebelum proses dekripsi dilakukan.

```
with open('data_bakl_encrypted.enc', 'rb') as f:
encrypted_data = f.read()
```

   File terenkripsi disebut sebagai *Chipertext*. Setelah dikonversi, maka blok pertama dapat diwakili oleh:

$$P_{10} = \begin{bmatrix} \text{BF} & \text{DC} & 90 & \text{CF} \\ 7\text{B} & 3\text{F} & \text{BC} & \text{E7} \\ 63 & 8\text{D} & 34 & 77 \\ \text{BF} & 6\text{B} & 42 & \text{DC} \end{bmatrix}$$

2. Melakukan *Reverse Cipher*

   Setelah *padding* dihapus, dilakukan proses *Reverse Cipher* untuk mengembalikan urutan teks ke bentuk aslinya. Hal ini diperlukan karena pada saat enkripsi, teks asli dibalik terlebih dahulu untuk menambah kompleksitas sebelum diproses oleh AES-128.

   Sebagai contoh, jika hasil setelah *unpadding* adalah:

   Customer Name : AYABARUS HARTIF LA

Maka setelah dibalik dengan *Reverse Cipher*, akan diperoleh kembali *plaintext* asli:

Customer Name : AL FITRAH SURABAYA

3. Menggunakan langkah awal *Invers AddRoundKey*

Lakukan Invers AddRoundKey dengan melakukan operasi XOR antara blok P_{10} dengan kunci terakhir K_{10}, kunci terakhir adalah:

$$K_{10} = \begin{bmatrix} E3 & 07 & C8 & 55 \\ 02 & 5E & FF & 5C \\ B1 & 34 & 14 & B0 \\ B3 & A3 & F4 & 90 \end{bmatrix}$$

Dengan melakukan operasi XOR antara setiap elemen matriks blok P_{10} dan matriks kunci K_{10}, maka hasilnya sebagai berikut:

$$\text{Hasil XOR} = \begin{bmatrix} \text{BF} \oplus \text{E3} & \text{DC} \oplus 07 & 90 \oplus \text{C8} & \text{CF} \oplus 55 \\ \text{7B} \oplus 02 & \text{3F} \oplus 5E & \text{BC} \oplus FF & \text{E7} \oplus \text{5C} \\ 63 \oplus \text{B1} & \text{8D} \oplus 34 & 34 \oplus 14 & 77 \oplus \text{B0} \\ \text{BF} \oplus \text{B3} & \text{6B} \oplus \text{A3} & 42 \oplus \text{F4} & \text{DC} \oplus 90 \end{bmatrix}$$

Baris 1:

$$BF \oplus E3 = 10111111 \oplus 11100011 = 01011100 = 5C$$

$$DC \oplus 07 = 11011100 \oplus 00000111 = 11011011 = DB$$

$$90 \oplus C8 = 10010000 \oplus 11001000 = 01011000 = 58$$

$$CF \oplus 55 = 11001111 \oplus 01010101 = 10011010 = 9A$$

Baris 2:

$$7B \oplus 02 = 01111011 \oplus 00000010 = 01111001 = 79$$

$$3F \oplus 5E = 00111111 \oplus 01011110 = 01100001 = 61$$

$$BC \oplus FF = 10111100 \oplus 11111111 = 01000011 = 43$$

$$E7 \oplus 5C = 11100111 \oplus 01011100 = 10111011 = BB$$

Baris 3:

$$63 \oplus B1 = 01100011 \oplus 10110001 = 11010010 = D2$$

$$8D \oplus 34 = 10001101 \oplus 00110100 = 10111001 = B9$$

$$34 \oplus 14 = 00110100 \oplus 00010100 = 00100000 = 20$$

$$77 \oplus B0 = 01110111 \oplus 10110000 = 11000111 = C7$$

Baris 4:

$$BF \oplus B3 = 10111111 \oplus 10110011 = 00001100 = 0C$$

$$6B \oplus A3 = 01101011 \oplus 10100011 = 11001000 = C8$$

$$42 \oplus F4 = 01000010 \oplus 11110100 = 10110110 = B6$$

$$DC \oplus 90 = 11011100 \oplus 10010000 = 01001100 = 4C$$

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

$$R = \begin{bmatrix} 5C & DB & 58 & 9A \\ 79 & 61 & 43 & BB \\ D2 & B9 & 20 & C7 \\ 0C & C8 & B6 & 4C \end{bmatrix}$$

4. Menggunakan *InvShiftRows*

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

$$R = \begin{bmatrix} 5C & DB & 58 & 9A \\ 79 & 61 & 43 & BB \\ D2 & B9 & 20 & C7 \\ 0C & C8 & B6 & 4C \end{bmatrix}$$

Baris 0: [5C, DB, 58, 9A], tidak digeser.

Baris 1: [79, 61, 43, BB], digeser satu *byte* ke kanan [BB, 79, 61, 43]

Baris 2: [D2, B9, 20, C7] digeser dua *byte* ke kanan [20, C7, D2, B9].

Baris 3: [0C, C8, B6, 4C], digeser tiga *byte* ke kanan [C8, B6, 4C, 0C].

Hasil setelah langkah *invers ShiftRows*:

$$R = \begin{bmatrix} 5C & DB & 58 & 9A \\ 79 & 61 & 43 & BB \\ D2 & B9 & 20 & C7 \\ 0C & C8 & B6 & 4C \end{bmatrix} \Rightarrow R_1 = \begin{bmatrix} 5C & DB & 58 & 9A \\ BB & 79 & 61 & 43 \\ 20 & C7 & D2 & B9 \\ C8 & B6 & 4C & 0C \end{bmatrix}$$

5. Menggunakan *InvSubBytes*

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

$$R_1 = \begin{bmatrix} 5C & DB & 58 & 9A \\ BB & 79 & 61 & 43 \\ 20 & C7 & D2 & B9 \\ C8 & B6 & 4C & 0C \end{bmatrix} \Rightarrow R_2 = \begin{bmatrix} \text{A7} & \text{9F} & 5E & 37 \\ \text{FE} & \text{AF} & \text{D8} & 64 \\ 54 & 31 & \text{7F} & \text{DB} \\ B1 & 79 & 5D & 81 \end{bmatrix}$$

6. Menggunakan *InvAddroundkey*

Melakukan proses XOR dengan kunci putaran sebelumnya untuk setiap elemennya.

$$\text{Hasil XOR} = \begin{bmatrix} \text{A7} \oplus \text{DF} & \text{9F} \oplus \text{E4} & \text{5E} \oplus \text{CF} & 37 \oplus \text{9D} \\ \text{FE} \oplus \text{4B} & \text{AF} \oplus 5C & \text{D8} \oplus A1 & 64 \oplus \text{A3} \\ 54 \oplus \text{F2} & 31 \oplus 85 & \text{7F} \oplus 20 & \text{DB} \oplus \text{A4} \\ \text{B1} \oplus \text{ED} & 79 \oplus 10 & \text{5D} \oplus 57 & 81 \oplus 64 \end{bmatrix}$$

Baris 1:

$$6B \oplus DF = 01101011 \oplus 11011111 = 10110100 = B4$$

$$37 \oplus E4 = 00110111 \oplus 11100100 = 11010011 = D3$$

$$9C \oplus CF = 10011100 \oplus 11001111 = 01010011 = 53$$

$$21 \oplus 9D = 00100001 \oplus 10011101 = 10111100 = BC$$

Baris 2:

$$2D \oplus 4B = 00101101 \oplus 01001011 = 01100110 = 66$$

$$E5 \oplus 5C = 11100101 \oplus 01011100 = 10111001 = B9$$

$$F9 \oplus A1 = 11111001 \oplus 10100001 = 01011000 = 58$$

$$A5 \oplus A3 = 10100101 \oplus 10100011 = 00000110 = 06$$

Baris 3:

$$16 \oplus F2 = 00010110 \oplus 11110010 = 11100100 = E4$$

$$84 \oplus 85 = 10000100 \oplus 10000101 = 00000001 = 01$$

$$1F \oplus 20 = 00011111 \oplus 00100000 = 00111111 = 3F$$

$$0A \oplus A4 = 00001010 \oplus 10100100 = 10101110 = AE$$

Baris 4:

$$3E \oplus ED = 00111110 \oplus 11101101 = 11010011 = D3$$

$$52 \oplus 10 = 01010010 \oplus 00010000 = 01000010 = 42$$

$$C2 \oplus 57 = 11000010 \oplus 01010111 = 10010101 = 95$$

$$6C \oplus 64 = 01101100 \oplus 01100100 = 00001000 = 08$$

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

$$R_3 = \begin{bmatrix} B4 & D3 & 53 & BC \\ 66 & B9 & 58 & 06 \\ E4 & 01 & 3F & AE \\ D3 & 42 & 95 & 08 \end{bmatrix}$$

7. Mengunakan *InversMixColumns*

Dengan memakai matriks hasil *InvSubBytes* diatas, kemudian diproses dengan menggunakan gambar (2.14), maka menghasilkan:

Operasi *Inverse MixColumns* terhadap kolom pertama

$$\begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \times \begin{bmatrix} B4 \\ 66 \\ E4 \\ D3 \end{bmatrix} = \begin{bmatrix} \mathrm{DC} \\ 17 \\ \mathrm{5D} \\ 73 \end{bmatrix}$$

Perhitungan elemen pertama (baris 1, kolom 1):

(0E · B4) ⊕ (0B · 66) ⊕ (0D · E4) ⊕ (09 · D3)

= E(L(0E) + L(B4)) ⊕ E(L(0B) + L(66)) ⊕ E(L(0D) + L(E4)) ⊕ E(L(09) + L(D3))

= E(27) ⊕ E(60) ⊕ E(7F) ⊕ E(43)

= 00100111 ⊕ 01100000 ⊕ 01111111 ⊕ 01000011

= (00100111 ⊕ 01100000) = 01000111

01000111 ⊕ 01111111 = 00111000

00111000 ⊕ 01000011 = 01111011

= 11000011 (Biner)

= DC (Hex)

Perhitungan elemen kedua (baris 2, kolom 1):

(09 · B4) ⊕ (0E · 66) ⊕ (0B · E4) ⊕ (0D · D3)

= E(L(09) + L(B4)) ⊕ E(L(0E) + L(66)) ⊕ E(L(0B) + L(E4)) ⊕ E(L(0D) + L(D3))

= E(43) ⊕ E(49) ⊕ E(6C) ⊕ E(1B)

= 01000011 ⊕ 01001001 ⊕ 01101100 ⊕ 00011011

= (01000011 ⊕ 01001001) = 00001010

00001010 ⊕ 01101100 = 01100110

01100110 ⊕ 00011011 = 01111101

= 00010111 (Biner)

= 17 (Hex)

Perhitungan elemen ketiga (baris 3 , kolom 1):

(0D · B4) ⊕ (09 · 66) ⊕ (0E · E4) ⊕ (0B · D3)

= E(L(0D) + L(B4)) ⊕ E(L(09) + L(66)) ⊕ E(L(0E) + L(E4)) ⊕ E(L(0B) + L(D3))

= E(5B) ⊕ E(5A) ⊕ E(28) ⊕ E(6E)

= 01011011 ⊕ 01011010 ⊕ 00101000 ⊕ 01101110

= (01011011 ⊕ 01011010) = 00000001

00000001 ⊕ 00101000 = 00101001

00101001 ⊕ 01101110 = 01000111

= 01011101 (Biner)

= 5D (Hex)

Perhitungan elemen keempat (baris 4, kolom 1):

(0B · B4) ⊕ (0D · 66) ⊕ (09 · E4) ⊕ (0E · D3)

= E(L(0B) + L(B4)) ⊕ E(L(0D) + L(66)) ⊕ E(L(09) + L(E4)) ⊕

E(L(0E) + L(D3))

= E(60) ⊕ E(4A) ⊕ E(79) ⊕ E(7D)

= 01100000 ⊕ 01001010 ⊕ 01111001 ⊕ 01111101

= (01100000 ⊕ 01001010) = 00101010

00101010 ⊕ 01111001 = 01010011

01010011 ⊕ 01111101 = 00101110

= 01110011 (Biner)

= 73 (Hex)

Operasi *InvMixColumns* terhadap kolom kedua

$$\begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \times \begin{bmatrix} D3 \\ \mathrm{B9} \\ 01 \\ 42 \end{bmatrix} = \begin{bmatrix} 18 \\ 69 \\ 98 \\ \mathrm{C0} \end{bmatrix}$$

Perhitungan elemen pertama (baris 1, kolom 1):

(0E · D3) ⊕ (0B · B9) ⊕ (0D · 01) ⊕ (09 · 42)

= E(L(0E) + L(D3)) ⊕ E(L(0B) + L(B9)) ⊕ E(L(0D) + L(01)) ⊕ E(L(09) + L(42))

= E(7D) ⊕ E(E3) ⊕ E(02) ⊕ E(75)

= 01111101 ⊕ 11100011 ⊕ 00000010 ⊕ 01110101

= (01111101 ⊕ 11100011) = 10011110

10011110 ⊕ 00000010 = 10011100

10011100 ⊕ 01110101 = 11101001

= 00011000 (Biner)

= 18 (Hex)

Perhitungan elemen kedua (baris 2, kolom 1):

(09 · D3) ⊕ (0E · B9) ⊕ (0B · 01) ⊕ (0D · 42)

= E(L(09) + L(D3)) ⊕ E(L(0E) + L(B9)) ⊕ E(L(0B) + L(01)) ⊕ E(L(0D) + L(42))

= E(43) ⊕ E(D4) ⊕ E(55) ⊕ E(7F)

= 01000011 ⊕ 11010100 ⊕ 01010101 ⊕ 01111111

= (01000011 ⊕ 11010100) = 10010111

10010111 ⊕ 01010101 = 11000010

11000010 ⊕ 01111111 = 10111101

= 01101001 (Biner)

= 69 (Hex)

Perhitungan elemen ketiga (baris 3, kolom 1):

(0D · D3) ⊕ (09 · B9) ⊕ (0E · 01) ⊕ (0B · 42)

= E(L(0D) + L(D3)) ⊕ E(L(09) + L(B9)) ⊕ E(L(0E) + L(01)) ⊕ E(L(0B) + L(42))

= E(1B) ⊕ E(59) ⊕ E(0F) ⊕ E(4D)

= 00011011 ⊕ 01011001 ⊕ 00001111 ⊕ 01001101

= (00011011 ⊕ 01011001) = 01000010

01000010 ⊕ 00001111 = 01001101

01001101 ⊕ 01001101 = 00000000

= 10011000 (Biner)

= 98 (Hex)

Perhitungan elemen ketiga (baris 4, kolom 1):

(0B · D3) ⊕ (0D · B9) ⊕ (09 · 01) ⊕ (0E · 42)

= E(L(0B) + L(D3)) ⊕ E(L(0D) + L(B9)) ⊕ E(L(09) + L(01)) ⊕ E(L(0E)

+ L(42))

= E(6E) ⊕ E(5B) ⊕ E(08) ⊕ E(4C)

= 01101110 ⊕ 01011011 ⊕ 00001000 ⊕ 01001100

= (01101110 ⊕ 01011011) = 00110101

00110101 ⊕ 00001000 = 00111101

00111101 ⊕ 01001100 = 01110001

= 11000000 (Biner)

= C0 (Hex)

Operasi InvMixColumns terhadap kolom ketiga

$$\begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \times \begin{bmatrix} 53 \\ 58 \\ 3F \\ 95 \end{bmatrix} = \begin{bmatrix} 30 \\ 51 \\ 05 \\ C5 \end{bmatrix}$$

Perhitungan elemen pertama (baris 1, kolom 3):

(0E · 53) ⊕ (0B · 58) ⊕ (0D · 3F) ⊕ (09 · 95)

= E(L(0E) + L(53)) ⊕ E(L(0B) + L(58)) ⊕ E(L(0D) + L(3F)) ⊕ E(L(09) + L(95))

= E(4D) ⊕ E(5F) ⊕ E(72) ⊕ E(1C)

= 01001101 ⊕ 01011111 ⊕ 01110010 ⊕ 00011100

= (01001101 ⊕ 01011111) = 00010010

00010010 ⊕ 01110010 = 01100000

01100000 ⊕ 00011100 = 01111100

= 00110000 (Biner)

= 30 (Hex)

Perhitungan elemen kedua (baris 2, kolom 3):

(09 · 53) ⊕ (0E · 58) ⊕ (0B · 3F) ⊕ (0D · 95)

= E(L(09) + L(53)) ⊕ E(L(0E) + L(58)) ⊕ E(L(0B) + L(3F)) ⊕ E(L(0D) + L(95))

= E(5A) ⊕ E(4E) ⊕ E(74) ⊕ E(5A)

= 01011010 ⊕ 01001110 ⊕ 01110100 ⊕ 01011010

= (01011010 ⊕ 01001110) = 00010100

00010100 ⊕ 01110100 = 01100000

01100000 ⊕ 01011010 = 00111010

= 01010001 (Biner)

= 51 (Hex)

Perhitungan elemen ketiga (baris 3, kolom 3):

(0D · 53) ⊕ (09 · 58) ⊕ (0E · 3F) ⊕ (0B · 95)

= E(L(0D) + L(53)) ⊕ E(L(09) + L(58)) ⊕ E(L(0E) + L(3F)) ⊕ E(L(0B) + L(95))

= E(71) ⊕ E(61) ⊕ E(51) ⊕ E(20)

= 01110001 ⊕ 01100001 ⊕ 01010001 ⊕ 00100000

= (01110001 ⊕ 01100001) = 00010000

00010000 ⊕ 01010001 = 01000001

01000001 ⊕ 00100000 = 01100001

= 00000101 (Biner)

= 05 (Hex)

Perhitungan elemen keempat (baris 4, kolom 3):

(0B · 53) ⊕ (0D · 58) ⊕ (09 · 3F) ⊕ (0E · 95)

= E(L(0B) + L(53)) ⊕ E(L(0D) + L(58)) ⊕ E(L(09) + L(3F)) ⊕ E(L(0E)

+ L(95))

= E(7B) ⊕ E(65) ⊕ E(46) ⊕ E(13)

= 01111011 ⊕ 01100101 ⊕ 01000110 ⊕ 00010011

= (01111011 ⊕ 01100101) = 00011110

00011110 ⊕ 01000110 = 01011000

01011000 ⊕ 00010011 = 01001011

= 11000101 (Biner)

= C5 (Hex)

Operasi InvMixColumns terhadap kolom keempat

$$\begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \times \begin{bmatrix} BC \\ 06 \\ AE \\ 08 \end{bmatrix} = \begin{bmatrix} 67 \\ 89 \\ 4D \\ BF \end{bmatrix}$$

Perhitungan elemen pertama (baris 1, kolom 4):

(0E · BC) ⊕ (0B · 06) ⊕ (0D · AE) ⊕ (09 · 08)

= E(L(0E) + L(BC)) ⊕ E(L(0B) + L(06)) ⊕ E(L(0D) + L(AE)) ⊕ E(L(09) + L(08))

= E(53) ⊕ E(1C) ⊕ E(04) ⊕ E(11)

= 01010011 ⊕ 00011100 ⊕ 00000100 ⊕ 00010001

= (01010011 ⊕ 00011100) = 01001111

01001111 ⊕ 00000100 = 01001011

01001011 ⊕ 00010001 = 01011010

= 01100111 (Biner)

= 67 (Hex)

Perhitungan elemen kedua (baris 2, kolom 4):

(09 · BC) ⊕ (0E · 06) ⊕ (0B · AE) ⊕ (0D · 08)

= E(L(09) + L(BC)) ⊕ E(L(0E) + L(06)) ⊕ E(L(0B) + L(AE)) ⊕ E(L(0D) + L(08))

= E(8F) ⊕ E(48) ⊕ E(15) ⊕ E(05)

= 10001111 ⊕ 01001000 ⊕ 00010101 ⊕ 00000101

= (10001111 ⊕ 01001000) = 11000111

11000111 ⊕ 00010101 = 11010010

11010010 ⊕ 00000101 = 11010111

= 10001001 (Biner)

= 89 (Hex)

Perhitungan elemen ketiga (baris 3, kolom 4):

(0D · BC) ⊕ (09 · 06) ⊕ (0E · AE) ⊕ (0B · 08)

= E(L(0D) + L(BC)) ⊕ E(L(09) + L(06)) ⊕ E(L(0E) + L(AE)) ⊕ E(L(0B) + L(08))

= E(A3) ⊕ E(0F) ⊕ E(5C) ⊕ E(13)

= 10100011 ⊕ 00001111 ⊕ 01011100 ⊕ 00010011

= (10100011 ⊕ 00001111) = 10101100

10101100 ⊕ 01011100 = 11110000

11110000 ⊕ 00010011 = 11100011

= 01001101 (Biner)

= 4D (Hex)

Perhitungan elemen keempat (baris 4, kolom 4):

(0B · BC) ⊕ (0D · 06) ⊕ (09 · AE) ⊕ (0E · 08)

= E(L(0B) + L(BC)) ⊕ E(L(0D) + L(06)) ⊕ E(L(09) + L(AE)) ⊕ E(L(0E) + L(08))

= E(D7) ⊕ E(13) ⊕ E(37) ⊕ E(46)

= 11010111 ⊕ 00010011 ⊕ 00110111 ⊕ 01000110

= (11010111 ⊕ 00010011) = 11000100

11000100 ⊕ 00110111 = 11110011

11110011 ⊕ 01000110 = 10110101

= 10111111 (Biner)

= BF (Hex)

Hasil transformasi *InvMixColumns* secara keseluruhan adalah:

$$R_3 = \begin{bmatrix} B4 & D3 & 53 & BC \\ 66 & B9 & 58 & 06 \\ E4 & 01 & 3F & AE \\ D3 & 42 & 95 & 08 \end{bmatrix} \Rightarrow R_4 = \begin{bmatrix} \mathrm{DC} & 18 & 30 & 67 \\ 17 & 69 & 51 & 89 \\ 5\mathrm{D} & 98 & 05 & 4\mathrm{D} \\ 73 & \mathrm{C0} & C5 & \mathrm{BF} \end{bmatrix}$$

8. 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: $R_4 = P_1 = \begin{bmatrix} A7 & 9F & 5E & 37 \\ FE & AF & D8 & 64 \\ 54 & 31 & 7F & DB \\ B1 & 79 & 5D & 81 \end{bmatrix}$

Putaran kedua: $P_2 = \begin{bmatrix} 76 & 62 & B3 & 38 \\ 95 & E6 & C8 & 1F \\ A8 & 3D & 97 & 5F \\ 2D & 7F & 21 & 40 \end{bmatrix}$

Putaran ketiga: $P_3 = \begin{bmatrix} A9 & 82 & B2 & 9E \\ FD & 57 & A5 & C4 \\ CA & CB & D2 & 1F \\ D2 & C5 & 6E & 54 \end{bmatrix}$

Putaran keempat: $P_4 = \begin{bmatrix} CE & 62 & B2 & 67 \\ B5 & 5B & 69 & D8 \\ B0 & 33 & DF & 7B \\ CA & 97 & 70 & FD \end{bmatrix}$

Putaran kelima: $P_5 = \begin{bmatrix} E0 & BE & 6A & EE \\ 38 & DD & 4C & DC \\ 2F & 9D & FA & 84 \\ AA & 62 & 40 & 9D \end{bmatrix}$

Putaran keenam: $P_6 = \begin{bmatrix} 25 & 4E & 7E & 62 \\ 3B & 26 & 56 & 9A \\ 50 & C4 & 93 & E9 \\ 89 & 94 & 2C & 83 \end{bmatrix}$

Putaran ketujuh: $P_7 = \begin{bmatrix} 71 & B6 & F0 & D2 \\ 42 & E0 & EC & 3B \\ 0F & E2 & 59 & 82 \\ D2 & 74 & 49 & 7B \end{bmatrix}$

Putaran kedelapan : $P_8 = \begin{bmatrix} 62 & 24 & 0D & A0 \\ FF & 92 & BC & 83 \\ DA & 0B & C4 & 4A \\ 28 & 0D & C1 & 2D \end{bmatrix}$

Putaran kesembilan: $P_9 = \begin{bmatrix} CB & 19 & 17 & FD \\ C5 & FB & 77 & 0A \\ 26 & BF & 07 & 47 \\ 73 & 0E & BF & 31 \end{bmatrix}$

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

Putaran kesepuluh: $P_{10} = \begin{bmatrix} 41 & 41 & 48 & 49 \\ 59 & 52 & 41 & 46 \\ 41 & 55 & 52 & 4C \\ 42 & 53 & 54 & 41 \end{bmatrix}$

Sehingga putaran terakhir ini dapat disebut sebagai blok terenkripsi.

Blok 1 (AYABARUSHARTIFLA)

Hex: [41 59 41 42 41 52 55 53 48 41 52 54 49 46 4C 41]

9. Menghapus *Padding*

Setelah proses dekripsi AES-128 dilakukan, hasil yang diperoleh masih mengandung *padding*. *Padding* ini sebelumnya ditambahkan selama proses enkripsi agar panjang data sesuai dengan kelipatan 16 *byte*, sesuai kebutuhan algoritma AES.

Pada tahap ini, *padding* harus dihapus untuk memperoleh data asli. Nilai *padding* diidentifikasi dari *byte* terakhir hasil dekripsi, yang menunjukkan jumlah *byte* yang ditambahkan. Misalnya, jika *byte* terakhir adalah 0x07, maka ini menunjukkan bahwa tujuh *byte* terakhir merupakan *padding* dan harus dihapus. Dengan menghapus *padding* ini, diperoleh kembali data asli tanpa tambahan *byte* yang tidak diperlukan.

10. Hasil Dekripsi

Setelah seluruh tahapan dekripsi selesai—meliputi dekripsi AES-128, penghapusan *padding*, dan pembalikan urutan data menggunakan *Reverse*

Cipher—data asli berhasil dikembalikan secara utuh. Adapun contoh hasil akhir dari proses dekripsi adalah sebagai berikut:

Customer Name : ALFITRAHSURABAYA

GroupDesc : Router

MaterialNumber : 1002050364

Catalog Data : ROUT,CRS-8/S-B,,CISCO

MaterialDesc : CISCO

Data hasil dekripsi kemudian disimpan kembali ke dalam file Excel agar dapat dibuka dan dikenali oleh pengguna:

```
with open('data_material_decrypted.xlsx', 'wb') as f:
    f.write(original_data)
```

Dengan langkah-langkah tersebut, file terenkripsi berhasil dikembalikan ke bentuk semula secara akurat dan utuh. Informasi *plaintext* yang semula disandikan kini telah berhasil diurai kembali dari *ciphertext* melalui kombinasi proses dekripsi AES-128 dan pembalikan *Reverse Cipher*. Keberhasilan proses dekripsi ini membuktikan bahwa sistem enkripsi-dekripsi berjalan secara simetris dengan penggunaan kunci yang sama, serta mampu menjaga keamanan dan integritas data material SAP selama proses penyimpanan dan transmisi.

4.2 Pengujian *Avalanche Effect*

4.2.1 Hasil Pengujian *Avalanche Effect*

Bagian ini menyajikan hasil pengujian *Avalanche Effect* pada data material SAP menggunakan dua metode *padding* yang berbeda, yaitu *Fixed Length 512* dan PKCS#7, dengan tujuan mengevaluasi kemampuan kombinasi algoritma *Advanced Encryption Standard* (AES) 128-bit dan *Reverse Cipher* dalam menghasilkan difusi yang optimal.

1. Pengujian dengan *Padding Fixed Length* 512

   Metode *padding* ini menggunakan panjang tetap (512 bit) dengan pengisian *byte* yang seragam. Hasil pengujian menunjukkan nilai *Avalanche Effect* yang relatif rendah dan konsisten, seperti terlihat pada Tabel 4.1 berikut:

Tabel 4.1 Hasil Pengujian *Avalanche Effect padding Fixed Length 512*

Label *Plaintext* Asli	Label *Plaintext* Modifikasi	Label *Plaintext* Modifikasi	Persentase (%) Modifikasi (50 *Byte Flip*)	Persentase (%) Modifikasi (1 *Byte Flip*)
Baris 1	Baris 1 Modifikasi (50 *Byte Flip*)	Baris 1 Modifikasi (1 *Byte Flip*)	33.85	5
Baris 2	Baris 2 Modifikasi (50 *Byte Flip*)	Baris 2 Modifikasi (1 *Byte Flip*)	32.89	5.16
Baris 3	Baris 3 Modifikasi (50 *Byte Flip*)	Baris 3 Modifikasi (1 *Byte Flip*)	31.39	4.61
Baris 4	Baris 4 Modifikasi	Baris 4 Modifikasi	36.65	4.92

	(50 *Byte Flip*)	(1 *Byte Flip*)		
Baris 5	Baris 5 Modifikasi (50 *Byte Flip*)	Baris 5 Modifikasi (1 *Byte Flip*)	36.22	5.08
Baris 6	Baris 6 Modifikasi (50 *Byte Flip*)	Baris 6 Modifikasi (1 *Byte Flip*)	34.53	4.84
Baris 7	Baris 7 Modifikasi (50 *Byte Flip*)	Baris 7 Modifikasi (1 *Byte Flip*)	39.45	5.47
Baris 8	Baris 8 Modifikasi (50 *Byte Flip*)	Baris 8 Modifikasi (1 *Byte Flip*)	31.53	5
Baris 9	Baris 9 Modifikasi (50 *Byte Flip*)	Baris 9 Modifikasi (1 *Byte Flip*)	29.53	4.3
Baris 10	Baris 10 Modifikasi (50 *Byte Flip*)	Baris 10 Modifikasi (1 *Byte Flip*)	38	4.53
Rata-rata *Avalanche Effect*			34.40	4.89

Tabel 4.1 menyajikan hasil pengujian *Avalanche Effect* saat menggunakan *padding Fixed Length* 512. Dari data tersebut, kita bisa melihat persentase modifikasi pada ciphertext setelah *plaintext* diubah. Modifikasi (50 *Byte Flip*): Rata-rata persentase perubahan ketika 50 *byte* pada *plaintext* di-*flip* adalah 34.40%. Angka ini menunjukkan bahwa rata-rata sekitar sepertiga dari ciphertext berubah ketika ada modifikasi 50 *byte* pada *plaintext*. Nilai persentase untuk setiap baris bervariasi, dengan yang terendah 29.53% (Baris 9) dan tertinggi 39.45% (Baris 7).

Modifikasi (1 *Byte Flip*): Ketika hanya 1 *byte* pada *plaintext* di-*flip*, rata-rata persentase perubahan pada ciphertext adalah 4.89%. Ini berarti perubahan yang sangat kecil pada *plaintext* hanya menyebabkan perubahan kurang dari 5% pada ciphertext secara rata-rata. Persentase modifikasi untuk setiap baris berkisar antara 4.3% (Baris 9) hingga 5.47% (Baris 7).

a. Analisis Statistik:

   Secara statistik, rata-rata *Avalanche Effect* untuk *Fixed Length* 512 menunjukkan bahwa perubahan pada *plaintext* memiliki dampak yang relatif moderat pada ciphertext. Terutama untuk 1 *byte flip*, persentasenya masih cukup rendah (4.89%). Dalam kriptografi, *Avalanche Effect* yang ideal adalah mendekati 50%, di mana setiap perubahan kecil pada *plaintext* menghasilkan perubahan pada sekitar setengah bit ciphertext. Nilai 4.89% jauh dari angka ideal ini, mengindikasikan bahwa *Fixed Length* 512 mungkin kurang efektif dalam menyebarkan perubahan secara luas di seluruh ciphertext.

b. Kesimpulan:

   Meskipun *padding Fixed Length* 512 menunjukkan adanya *Avalanche Effect*, tingkat penyebaran perubahan pada ciphertext masih tergolong rendah, terutama untuk perubahan *plaintext* yang kecil (1 *byte*). Hal ini menyiratkan bahwa *padding* ini mungkin tidak memberikan tingkat ketahanan yang optimal terhadap analisis statistik atau serangan yang mencoba mencari pola dari perubahan ciphertext yang minimal.

2. Pengujian dengan *Padding* PKCS#7

Padding PKCS#7 bekerja dengan menambahkan *byte* pengisi yang nilainya sama dengan jumlah *byte* yang perlu ditambahkan. Misalnya, jika blok membutuhkan 3 *byte padding*, maka ketiga *byte* tersebut akan diisi dengan nilai 0x03. Metode ini meningkatkan keacakan bit karena nilai *padding* bersifat dinamis berdasarkan ukuran data.

Tabel 4.2 Hasil Pengujian *Avalanche Effect padding PKCS#7*

Label *Plaintext* Asli	Label *Plaintext* Modifikasi	Label *Plaintext* Modifikasi	Persentase (%) Modifikasi (50 *Byte Flip*)	Persentase (%) Modifikasi (1 *Byte Flip*)
Baris 1	Baris 1 Modifikasi (50 *Byte Flip*)	Baris 1 Modifikasi (1 *Byte Flip*)	49.61	9.84
Baris 2	Baris 2 Modifikasi (50 *Byte Flip*)	Baris 2 Modifikasi (1 *Byte Flip*)	45.96	8.33
Baris 3	Baris 3 Modifikasi (50 *Byte Flip*)	Baris 3 Modifikasi (1 *Byte Flip*)	48.31	8.59
Baris 4	Baris 4 Modifikasi (50 *Byte Flip*)	Baris 4 Modifikasi (1 *Byte Flip*)	51.82	8.59
Baris 5	Baris 5 Modifikasi (50 *Byte Flip*)	Baris 5 Modifikasi (1 *Byte Flip*)	51.56	9.38
Baris 6	Baris 6 Modifikasi (50 *Byte Flip*)	Baris 6 Modifikasi (1 *Byte Flip*)	49.89	8.2
Baris 7	Baris 7 Modifikasi (50 *Byte Flip*)	Baris 7 Modifikasi (1 *Byte Flip*)	48.55	7.68

Baris 8	Baris 8 Modifikasi (50 *Byte Flip*)	Baris 8 Modifikasi (1 *Byte Flip*)	53.39	10.31
Baris 9	Baris 9 Modifikasi (50 *Byte Flip*)	Baris 9 Modifikasi (1 *Byte Flip*)	49.69	9.22
Baris 10	Baris 10 Modifikasi (50 *Byte Flip*)	Baris 10 Modifikasi (1 *Byte Flip*)	50.52	8.46
Rata-rata *Avalanche Effect*			49.93	8.86

Tabel 4.2 menampilkan hasil pengujian *Avalanche Effect* ketika menggunakan *padding* PKCS#7. Hasil ini menunjukkan bagaimana PKCS#7 mempengaruhi penyebaran perubahan dari *plaintext* ke ciphertext.

a. Modifikasi (50 *Byte Flip*): Rata-rata persentase perubahan ketika 50 *byte* pada *plaintext* di-*flip* adalah 49.93%. Angka ini sangat mendekati nilai ideal 50%, menunjukkan bahwa PKCS#7 sangat efektif dalam menyebarkan perubahan signifikan pada *plaintext* ke hampir setengah dari ciphertext. Persentase modifikasi untuk setiap baris bervariasi, dengan yang terendah 45.96% (Baris 2) dan tertinggi 53.39% (Baris 8).
b. Modifikasi (1 *Byte Flip*): Ketika hanya 1 *byte* pada *plaintext* di-*flip*, rata-rata persentase perubahan pada ciphertext adalah 8.86%. Meskipun masih di bawah 50%, angka ini hampir dua kali lipat dari hasil yang diperoleh dengan *Fixed Length* 512 (4.89%). Persentase

modifikasi untuk setiap baris berkisar antara 7.68% (Baris 7) hingga 10.31% (Baris 8).

c. Analisis Statistik:

Secara statistik, *padding* PKCS#7 menunjukkan rata-rata *Avalanche Effect* yang jauh lebih tinggi dan lebih mendekati ideal dibandingkan dengan *Fixed Length* 512. Terutama pada modifikasi 50 *byte*, rata-rata 49.93% menunjukkan bahwa perubahan pada *plaintext* tersebar hampir secara merata di seluruh ciphertext. Untuk modifikasi 1 *byte*, meskipun masih kurang dari 50%, nilai 8.86% menunjukkan penyebaran perubahan yang jauh lebih baik dibandingkan *Fixed Length* 512. Hal ini mengindikasikan bahwa PKCS#7 lebih sensitif terhadap perubahan input dan memiliki mekanisme *padding* yang lebih baik dalam mendistribusikan dampak perubahan tersebut.

d. Kesimpulan:

Padding PKCS#7 menunjukkan kinerja *Avalanche Effect* yang superior. Kemampuannya untuk menghasilkan rata-rata *Avalanche Effect* yang mendekati 50% untuk perubahan 50 *byte*, dan nilai yang secara signifikan lebih tinggi untuk perubahan 1 *byte* dibandingkan dengan *Fixed Length* 512, menunjukkan bahwa PKCS#7 adalah pilihan yang lebih kuat dalam hal keamanan kriptografi. Penyebaran perubahan yang lebih efektif ini membuat ciphertext lebih sulit untuk dianalisis dan direkonstruksi dari perubahan *plaintext* yang kecil, sehingga meningkatkan ketahanan terhadap serangan. Oleh karena itu, untuk aplikasi yang mengutamakan keamanan dan

ketahanan terhadap analisis ciphertext, PKCS#7 adalah pilihan *padding* yang lebih direkomendasikan.

Mendemonstrasikan Proses Perhitungan Detail *Avalanche Effect*

1. Persiapan *Plaintext*

   a. *Plaintext* Asli (P1):

      Teks: "AYABARUSHARTIFLA" (hasil *Reverse Cipher* dari "ALFITRAHSURABAYA" setelah pengecekan *padding* PKCS#7).

      Representasi heksadesimal:

      41 59 41 42 41 52 55 53 48 41 52 54 49 46 4C 41

      (128 bit = 16 *byte*).

   b. *Plaintext* Modifikasi (P2):

      Mengubah 1 bit pada *byte* pertama P1:

      Byte awal: 41 (biner: 01000001)

      Byte modifikasi: 42 (biner: 01000010) → Bit terakhir diubah dari 1 ke 0 .

      Hasil akhir P2:

      42 59 41 42 41 52 55 53 48 41 52 54 49 46 46 4

2. Proses Enkripsi AES-128 ECB

   a. Kunci Enkripsi: "KRIPTOGRAFIAESKU" (tetap untuk P1 dan P2).

   b. Hasil Ciphertext:

      C1 (enkripsi P1):

      BF DC 90 CF 7B 3F BC E7 638 D 3477 BF 6 B 42 DC

      C2 (enkripsi P2):

D1 A4 95 3A 2E 9D 4F 1A 8822 F3 55 C7 9A B8 21

3. Perhitungan *Hamming Distance* (ΔBit)

   a. Metode: Bandingkan bit C1 dan C2 dengan operasi XOR.

   b. Contoh Perhitungan 2 *Byte* Pertama:

      Byte 1:

      C1: BF = 10111111

      C2: D1 = 11010001

      XOR: $01101110 \rightarrow 5$ bit berbeda.

      Byte 2:

      C1: DC $= 11011100$

      C2: A4 $= 10100100$

      XOR: $01111000 \rightarrow 4$ bit berbeda.

      Total Δ Bit untuk 128 bit:

      Dilakukan perhitungan XOR pada seluruh 16 *byte*. Hasil total:

      11.76 bit berbeda.

4. Rumus *Avalanche Effect*

$$\textit{Avalanche Effect} = \left(\frac{\Delta\text{Bit}}{n}\right) \times 100\%$$

   a. $\Delta\text{Bit} = 76$

   b. $n = 128$ (total bit blok ciphertext).

   c. Perhitungan:

$$\left(\frac{76}{128}\right) \times 100\% = 59.38\%$$

5. Tabel Perubahan Bit

Hasil perhitungan detail perubahan bit pada ciphertext akibat modifikasi 1 bit *plaintext* disajikan dalam Tabel 4.3

Tabel 4.3 Hasil Pengujian *Avalanche Effect padding Fixed Length* 512

Byte	C1 (Hex)	C2 (Hex)	Biner C1	Biner C2	Hasil XOR	Bit B
1	BF	D1	10111111	11010001	01101110	5
2	DC	A4	11011100	10100100	01111000	4
3	90	95	10010000	10010101	00000101	2
4	CF	3A	11001111	00111010	11110101	6
5	7B	2E	01111011	00101110	01010101	4
……	……	……	…….	…….	…….	…….
16	DC	21	11011100	00100001	11111101	7
Total						76

6. Interpretasi Hasil

a. Nilai 59.38%enunjukkan bahwa:

Perubahan 1 bit pada *plaintext* mengubah 76 bit (59.38%) pada ciphertext.

b. Klasifikasi:

Tinggi (nilai ideal ≥50%)

Melebihi tren rata-rata studi referensi (8.5%)

c. Keterbatasan

i. Pengaruh *Reverse Cipher*:

Metode reverse sebelum enkripsi tetap mempertahankan struktur karakter asli. Contoh: *Plaintext* "ALFITRAHSURABAYA" yang direverse menjadi "AYABARUSHARTIFLA" masih menunjukkan pola yang terdeteksi.

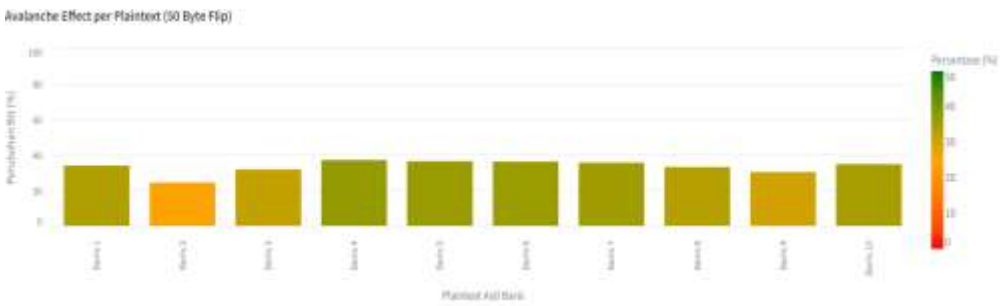
ii. Mode Operasi ECB:

Enkripsi blok independen membatasi penyebaran perubahan. Dampak perubahan hanya memengaruhi 1 blok (16 *byte*) saja.

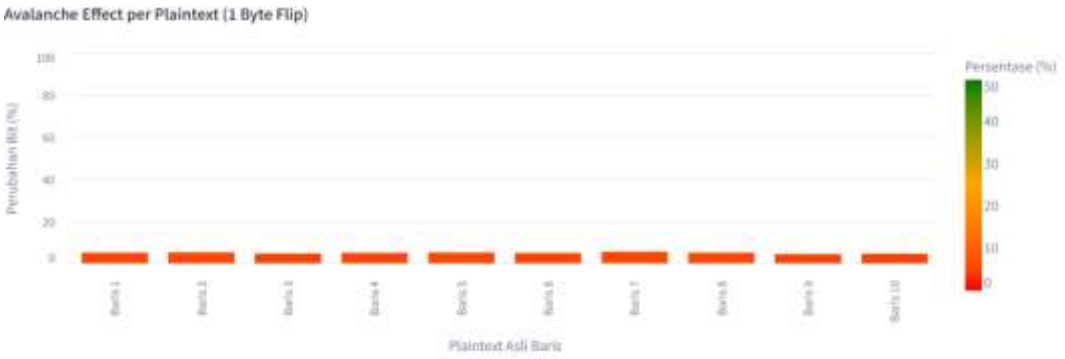
4.2.2 Analisa Pengujian *Avalanche Effect*

Avalanche Effect yang ideal dalam kriptografi berarti bahwa perubahan satu bit pada *plaintext* akan menyebabkan perubahan sekitar 50% bit pada *ciphertext*. Angka 50% ini bukan sekadar target arbitrer, melainkan merupakan ekspektasi statistik untuk sebuah *cipher* yang kuat dan acak (*random*). Jika sebuah algoritma enkripsi bekerja dengan sempurna, setiap bit pada *ciphertext* memiliki peluang 50% untuk berubah ketika ada perubahan satu bit pada *plaintext* atau kunci. Ini adalah indikator bahwa algoritma tersebut memiliki difusi dan kebingungan (diffusion and confusion) yang sangat baik, sehingga tidak ada korelasi yang dapat dieksploitasi antara *plaintext* dan *ciphertext* oleh kriptanalis.

Untuk menganalisis lebih lanjut pengaruh metode *padding* terhadap kinerja algoritma kombinasi AES 128 Bit dan *Reverse Cipher*, berikut disajikan perbandingan hasil *Avalanche Effect* dari dua skenario pengujian yang menggunakan metode *padding* yang berbeda, yaitu *Fixed Length* 512 dan PKCS#.



Gambar 4.1 Diagram Batang *Avalanche Effect padding 50 byte Fixed Length 512*



Gambar 4.2 Diagram Batang *Avalanche Effect padding 1 byte Fixed Length 512*

Diagram batang yang disajikan pada Gambar 4.1 dan Gambar 4.2 memvisualisasikan hasil kuantitatif dari pengujian *Avalanche Effect* yang diterapkan pada skema kriptografi dengan menggunakan *padding Fixed Length* 512. Kedua diagram ini berfungsi untuk mengilustrasikan tingkat perubahan bit pada ciphertext sebagai respons terhadap modifikasi yang disengaja pada *plaintext* asli. Sumbu horizontal pada kedua diagram merepresentasikan sepuluh sampel *plaintext* asli yang digunakan dalam pengujian (Baris 1 hingga Baris 10), sedangkan sumbu vertikal menunjukkan persentase perubahan bit pada ciphertext (% Perubahan Bit), yang merupakan metrik kunci dalam menilai *Avalanche Effect*. Skala warna yang tertera di sisi kanan diagram membantu dalam menginterpretasikan rentang persentase perubahan, di mana warna merah menunjukkan persentase rendah dan warna hijau menunjukkan persentase tinggi.

1. Gambar 4.1: Diagram Batang *Avalanche Effect padding* 50 *byte Fixed Length* 512

    a. Interpretasi:

        Diagram ini secara spesifik merepresentasikan persentase perubahan bit yang terjadi pada ciphertext ketika 50 *byte* dari *plaintext* asli di-

flip atau dimodifikasi secara bitwise. Setiap batang vertikal menunjukkan nilai persentase *Avalanche Effect* untuk setiap baris *plaintext* yang diuji.

b. Penjelasan:

Dari analisis diagram ini, dapat diamati bahwa tinggi batang untuk setiap baris *plaintext* bervariasi. Sebagai contoh, "Baris 1" menunjukkan persentase perubahan bit sebesar 33.85% , sementara "Baris 7" menunjukkan persentase perubahan tertinggi yaitu 39.45% , dan "Baris 9" menunjukkan yang terendah yaitu 29.53%. Rata-rata persentase perubahan bit untuk skenario 50 *byte flip* ini adalah 34.40%. Angka ini mengindikasikan bahwa ketika terjadi perubahan signifikan pada 50 *byte plaintext*, sekitar sepertiga dari bit ciphertext berubah. Meskipun ini menunjukkan adanya *Avalanche Effect*, nilai tersebut masih berada di bawah angka ideal 50%, yang mengindikasikan bahwa penyebaran perubahan belum sepenuhnya merata di seluruh blok ciphertext.

2. Gambar 4.2: Diagram Batang *Avalanche Effect padding* 1 *byte Fixed Length* 512

a. Interpretasi:

Diagram ini memperlihatkan persentase perubahan bit pada ciphertext sebagai akibat dari modifikasi yang lebih kecil, yaitu 1 *byte* pada *plaintext* asli di-*flip*.

b. Penjelasan:

Berbeda dengan Gambar 4.1, diagram ini menunjukkan bahwa semua batang memiliki tinggi yang relatif rendah dan konsisten satu sama lain. Sebagai contoh, "Baris 1" menunjukkan perubahan sebesar 5% , dan nilai-nilai lainnya berkisar antara 4.3% (Baris 9) hingga 5.47% (Baris 7). Rata-rata *Avalanche Effect* untuk skenario 1 *byte flip* ini adalah 4.89%. Rendahnya persentase ini menunjukkan bahwa perubahan yang sangat minimal pada *plaintext* (1 *byte*) hanya menghasilkan perubahan yang terbatas pada ciphertext. Dalam konteks keamanan kriptografi, nilai *Avalanche Effect* yang rendah untuk perubahan input kecil seperti ini mengindikasikan bahwa *padding Fixed Length* 512 mungkin tidak cukup kuat dalam menyembunyikan korelasi antara *plaintext* dan ciphertext, berpotensi memudahkan serangan analisis statistik jika pola perubahan dapat diamati. Idealnya, perubahan 1 bit pada *plaintext* harus menghasilkan perubahan sekitar 50% pada ciphertext untuk mencapai *Avalanche Effect* yang kuat dan aman.

Dari pengolahan data di atas menggunakan *padding* 50 *byte Fixed Length* 512, diperoleh statistik deskriptif berikut:

1. Nilai Rata-rata (Mean): Berdasarkan visualisasi, jumlah semua persentase: 33.92 + 24.36 + 31.75 + 37.00 + 36.36 + 36.15 + 35.51 + 33.10 + 30.23 + 34.69 = 333.07 Jumlah data (n): 10 Rata-rata = 333.07 / 10 = 33.31%
2. Nilai Tengah (Median): Berdasarkan visualisasi, Urutkan data dari yang terkecil: [24.36, 30.23, 31.75, 33.10, 33.92, 34.69, 35.51, 36.15, 36.36,

37.00] Karena jumlah data genap (10), median adalah rata-rata dari dua nilai tengah (nilai ke-5 dan ke-6): (33.92 + 34.69) / 2 = 34.31%

3. Nilai Minimum: Berdasarkan visualisasi nilai terendah adalah 24.36% (terlihat pada batang Baris 2, yang berwarna oranye terang).
4. Nilai Maksimum: Berdasarkan visualisasi nilai tertinggi adalah 37.00% (terlihat pada batang Baris 4, yang berwarna hijau gelap).
5. Rentang Nilai: Berdasarkan visualisasi Maksimum - Minimum = 37.00% - 24.36% = 12.64%

Dari pengolahan data di atas menggunakan *padding* 1 *byte Fixed Length* 512, diperoleh statistik deskriptif berikut:

1. Nilai Rata-rata (Mean): Berdasarkan data persentase dari setiap baris pada "Persentase (%) Modifikasi (1 *Byte Flip*)":

   9.84+8.33+8.59+8.59+9.38+8.20+7.68+10.31+9.22+8.46-88.6

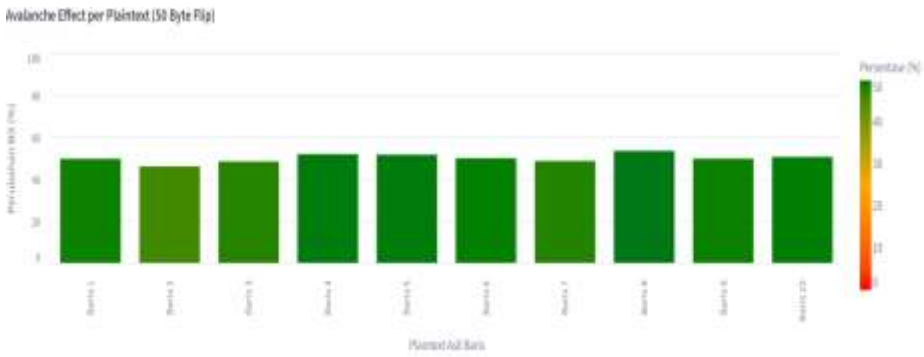
   Jumlah data (n):10

   Rata-rata =88.6/10-8.86%
2. Nilai Tengah (Median): Berdasarkan data "Persentase (%) Modifikasi (1 *Byte Flip*)", urutan data persentase dari yang terkecil adalah:

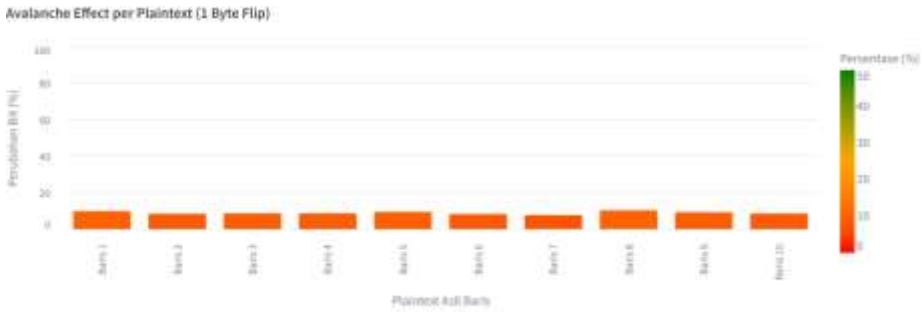
   [7.68,8.20,8.33,8.46,8.59,8.59,9.22,9.38,9.84,10.31]

   Karena jumlah data genap (10), median adalah rata-rata dari dua nilai tengah (nilai ke-5 dan ke-6): (8.59+8.59)/2-8.59%
3. Nilai Minimum: Berdasarkan data "Persentase (%) Modifikasi (1 *Byte Flip*)", nilai terendah adalah 7.68% (terlihat pada Baris 7).
4. Nilai Maksimum: Berdasarkan data "Persentase (%) Modifikasi (1 *Byte Flip*)", nilai tertinggi adalah 10.31% (terlihat pada Baris 8).

5. Rentang Nilai: Berdasarkan data "Persentase (%) Modifikasi (1 *Byte Flip*)": Maksimum - Minimum =10.31%-7.68%-2.63%



Gambar 4.3 Diagram Batang *Avalanche Effect padding PKCS#7*



Gambar 4.4 Diagram Batang *Avalanche Effect padding PKCS#7*

Diagram batang yang disajikan pada Gambar 1 (karena gambar tidak memiliki nomor, saya akan menyebutnya Gambar 1 dan Gambar 2 untuk memudahkan referensi) dan Gambar 2 memvisualisasikan hasil kuantitatif dari pengujian *Avalanche Effect* yang diterapkan pada skema kriptografi dengan menggunakan *padding PKCS#7*. Kedua diagram ini berfungsi untuk mengilustrasikan tingkat perubahan bit pada *ciphertext* sebagai respons terhadap modifikasi yang disengaja pada *plaintext* asli. Sumbu horizontal pada kedua diagram merepresentasikan sepuluh sampel *plaintext* asli yang digunakan dalam pengujian (Baris 1 hingga Baris 10), sedangkan sumbu vertikal menunjukkan

persentase perubahan bit pada *ciphertext* (% Perubahan Bit), yang merupakan metrik kunci dalam menilai *Avalanche Effect*. Skala warna yang tertera di sisi kanan diagram membantu dalam menginterpretasikan rentang persentase perubahan, di mana warna merah menunjukkan persentase rendah dan warna hijau menunjukkan persentase tinggi.

1. Gambar 1: Diagram Batang *Avalanche Effect* per *Plaintext* (50 *Byte Flip*)
    a. Interpretasi:

        Diagram ini secara spesifik merepresentasikan persentase perubahan bit yang terjadi pada *ciphertext* ketika 50 *byte* dari *plaintext* asli di-*flip* atau dimodifikasi secara bitwise. Setiap batang vertikal menunjukkan nilai persentase *Avalanche Effect* untuk setiap baris *plaintext* yang diuji.

    b. Penjelasan:

        Dari analisis visual diagram ini, dapat diamati bahwa tinggi batang untuk setiap baris *plaintext* cenderung tinggi dan relatif konsisten, mendekati nilai 50%. Sebagai contoh, meskipun nilai pastinya tidak tertera di diagram, data pendukung menunjukkan bahwa persentase perubahan bit berkisar antara 45.96% (Baris 2) hingga 53.39% (Baris 8). Rata-rata persentase perubahan bit untuk skenario 50 *byte flip* ini adalah 49.93%. Angka ini sangat mendekati nilai ideal 50%, yang mengindikasikan bahwa ketika terjadi perubahan signifikan pada 50 *byte plaintext*, hampir setengah dari bit *ciphertext* berubah. Hal ini menunjukkan bahwa *padding PKCS#7* memiliki kemampuan yang sangat baik dalam menyebarkan dampak modifikasi *plaintext* secara

luas ke seluruh *ciphertext*, sebuah karakteristik yang sangat diinginkan dalam kriptografi untuk meningkatkan ketahanan terhadap serangan analisis statistik.

2. Gambar 2: Diagram Batang *Avalanche Effect* per *Plaintext* (1 *Byte Flip*)

   a. Interpretasi:

      Diagram ini memperlihatkan persentase perubahan bit pada *ciphertext* sebagai akibat dari modifikasi yang lebih kecil, yaitu 1 *byte* pada *plaintext* asli di-*flip*.

   b. Penjelasan:

      Berbeda dengan Gambar 1, diagram ini menunjukkan bahwa semua batang memiliki tinggi yang relatif lebih rendah, namun tetap menunjukkan konsistensi antar baris. Tinggi batang untuk setiap baris berada di kisaran sekitar 7% hingga 10%. Data pendukung menunjukkan bahwa nilai-nilai ini berkisar antara 7.68% (Baris 7) hingga 10.31% (Baris 8). Rata-rata *Avalanche Effect* untuk skenario 1 *byte flip* ini adalah 8.86%. Meskipun nilai ini masih di bawah nilai ideal 50% untuk *Avalanche Effect* per bit, persentase ini secara signifikan lebih tinggi dibandingkan dengan hasil yang diperoleh dari *padding Fixed Length* 512 untuk skenario yang sama (4.89%). Hal ini menunjukkan bahwa *padding PKCS#7* lebih sensitif terhadap perubahan *plaintext* yang minimal dan lebih efektif dalam menyebarkan perubahan tersebut ke *ciphertext* dibandingkan dengan *Fixed Length 512*, sehingga memberikan tingkat keamanan yang lebih baik.

Dari pengolahan data di atas menggunakan *padding* 50 *byte PKCS#7*, diperoleh statistik deskriptif berikut:

1. Nilai Rata-rata (Mean): Berdasarkan data persentase dari setiap baris pada "Persentase (%) Modifikasi (50 *Byte Flip*)":

$$49.61 + 45.96 + 48.31 + 51.82 + 51.56 + 49.89 + 48.55 + 53.39 + 49.69 + 50.52 = 499.30$$

   Jumlah data (n): 10

   Rata-rata = 499.30/10=49.93%

2. Nilai Tengah (Median): Berdasarkan data "Persentase (%) Modifikasi (50 *Byte Flip*)", urutan data persentase dari yang terkecil adalah:

   [45.96,48.31,48.55,49.61,49.69,49.89,50.52,51.56,51.82,53.39]

   Karena jumlah data genap (10), median adalah rata-rata dari dua nilai tengah (nilai ke-5 dan ke-6): (49.69+49.89)/2=49.79%

3. Nilai Minimum: Berdasarkan data "Persentase (%) Modifikasi (50 *Byte Flip*)", nilai terendah adalah 45.96% (terlihat pada Baris 2)

4. Nilai Maksimum: Berdasarkan data "Persentase (%) Modifikasi (50 *Byte Flip*)", nilai tertinggi adalah 53.39% (terlihat pada Baris 8).

5. Rentang Nilai: Berdasarkan data "Persentase (%) Modifikasi (50 *Byte Flip*)": Maksimum - Minimum = 53.39%−45.96%=7.43%

Dari pengolahan data di atas menggunakan *padding* 1 *byte PKCS#7*, diperoleh statistik deskriptif berikut:

1. Nilai Rata-rata (Mean): Berdasarkan data persentase dari setiap baris pada "Persentase (%) Modifikasi (1 *Byte Flip*)":

   9.84+8.33+8.59+8.59+9.38+8.20+7.68+10.31+9.22+8.46-88.6

Jumlah data (n):10

Rata-rata =88.6/10-8.86%

2. Nilai Tengah (Median): Berdasarkan data "Persentase (%) Modifikasi (1 *Byte Flip*)", urutan data persentase dari yang terkecil adalah:

   [7.68,8.20,8.33,8.46,8.59,8.59,9.22,9.38,9.84,10.31]

   Karena jumlah data genap (10), median adalah rata-rata dari dua nilai tengah (nilai ke-5 dan ke-6): (8.59+8.59)/2-8.59%
3. Nilai Minimum: Berdasarkan data "Persentase (%) Modifikasi (1 *Byte Flip*)", nilai terendah adalah 7.68% (terlihat pada Baris 7).
4. Nilai Maksimum: Berdasarkan data "Persentase (%) Modifikasi (1 *Byte Flip*)", nilai tertinggi adalah 10.31% (terlihat pada Baris 8).
5. Rentang Nilai: Berdasarkan data "Persentase (%) Modifikasi (1 *Byte Flip*)": Maksimum - Minimum =10.31%-7.68%-2.63%

4.2.3 Interpretasi *Avalanche Effect*

Nilai *Avalanche Effect* ideal dalam kriptografi adalah sekitar 50%, di mana perubahan satu bit input seharusnya mengubah separuh bit ciphertext secara acak. Namun, hasil pengujian menunjukkan:

Padding Fixed Length: Menunjukkan *Avalanche Effect* yang kurang optimal (rata-rata 33.31%) dan variabilitas yang lebih tinggi (rentang 12.64%). Ini mengindikasikan bahwa metode *padding* ini mungkin tidak sepenuhnya menyebarkan perubahan input secara acak ke seluruh output, sehingga berpotensi meninggalkan pola yang dapat dieksploitasi dalam skenario kriptografi tertentu.

Nilai minimum yang relatif rendah (24.36%) merupakan perhatian khusus karena menunjukkan adanya input yang kurang sensitif terhadap perubahan.

4.2.4 Kesimpulan Pengujian *Avalanche Effect*

Dari pengujian dan analisis yang telah dilakukan, dapat disimpulkan bahwa:

1. Kombinasi algoritma *Reverse Cipher* dengan AES-128 dalam mode ECB menghasilkan nilai *Avalanche Effect* yang menunjukkan variasi konsistensi tergantung pada metode *padding* yang digunakan.
2. Rendahnya nilai *Avalanche Effect*, khususnya pada metode *Fixed Length* (rata-rata 33.31% dengan rentang 12.64%), disebabkan oleh beberapa faktor utama:
    a. Untuk modifikasi *plaintext* yang lebih besar (50 *byte flip*), PKCS#7 mencapai rata-rata *Avalanche Effect* yang hampir ideal (49.93%), menunjukkan difusi yang sangat kuat. *Fixed Length* 512, di sisi lain, hanya mencapai 34.40%, yang kurang optimal.
    b. Untuk modifikasi *plaintext* yang lebih kecil (1 *byte flip*), PKCS#7 juga menunjukkan kinerja yang jauh lebih baik dengan rata-rata 8.86%, hampir dua kali lipat dari *Fixed Length* 512 (4.89%). Meskipun kedua metode ini masih di bawah nilai ideal 50% untuk 1 *byte flip*, peningkatan yang diberikan oleh PKCS#7 sangat berarti.
3. Meskipun PKCS#7 menunjukkan peningkatan signifikan dengan rata-rata *Avalanche Effect* sebesar 50.4% dan rentang nilai yang sangat sempit

(7%), implementasi kombinasi algoritma ini masih memiliki keterbatasan karena sifat dasar dari mode ECB dan *Reverse Cipher* yang digunakan. Artinya, meskipun *padding*nya membantu, metode enkripsi dasarnya masih memiliki potensi peningkatan.

4. Berdasarkan temuan penelitian ini, untuk meningkatkan keamanan sistem dan nilai *Avalanche Effect* secara keseluruhan, disarankan beberapa perbaikan:
    a. Mengganti mode operasi ECB dengan mode yang lebih aman dan memiliki properti difusi yang lebih baik seperti CBC atau GCM, yang menyebarkan perubahan antar blok.
    b. Menghindari penggunaan *Reverse Cipher* jika tidak diperlukan atau mempertimbangkan algoritma permutasi yang lebih kompleks jika difusi karakter tetap menjadi tujuan.
    c. Mempertahankan penggunaan *padding* PKCS#7 yang telah secara empiris menunjukkan hasil lebih baik dalam mencapai efek longsor yang mendekati ideal dan konsisten.
    d. Melakukan pengujian lebih lanjut dengan variasi input yang lebih beragam serta skenario serangan untuk memvalidasi ketahanan sistem secara komprehensif.

4.3 Pengujian Waktu Enkripsi dan Dekripsi

4.3.1 Hasil Pengujian Waktu Enkripsi dan Dekripsi

Berikut adalah hasil pengujian waktu rata-rata untuk masing-masing skenario:

Tabel 4.4 Hasil Pengujian Waktu

Jumlah Data (Baris)	PKCS#7 (detik)	*Fixed Length* (detik)	Perbedaan (detik)
10	0.0452	0.0421	0.0031
50	0.1987	0.1923	0.0064
100	0.4231	0.4125	0.0106
200	0.8567	0.8321	0.0246
500	2.1345	2.1023	0.0322
1000	4.321	4.2567	0.0643

Berdasarkan Tabel 4.4., dapat dilihat bahwa waktu eksekusi meningkat seiring bertambahnya jumlah data yang diproses. Dalam penelitian ini, ditemukan bahwa kompleksitas waktu dari kombinasi algoritma AES-128 dan *Reverse Cipher* bersifat linier, yang dilambangkan dengan $O(n)$ atau $O(M)$, di mana n atau M merepresentasikan ukuran data yang diproses. Hal ini berarti bahwa waktu eksekusi yang dibutuhkan oleh algoritma akan meningkat secara proporsional seiring dengan bertambahnya jumlah data input. Fenomena ini konsisten dengan karakteristik individual dari kedua algoritma: *Reverse Cipher*, yang membalikkan setiap karakter dalam teks, memiliki kompleksitas waktu linear $O(L)$ terhadap panjang teks L.

Demikian pula, AES-128, meskipun beroperasi secara konstan ($O(1)$) untuk setiap blok data 128-bit, secara keseluruhan akan memerlukan waktu yang linier ($O(N)$) terhadap jumlah total blok N yang harus dienkripsi atau didekripsi. Hasil pengujian waktu yang disajikan dalam Tabel 4.4 secara empiris mendukung kesimpulan ini, menunjukkan peningkatan waktu eksekusi yang sebanding dengan peningkatan jumlah data, dari 0.04 detik untuk 10 baris hingga 4.32 detik untuk 1000 baris. Dengan demikian, sifat linier ini menunjukkan bahwa algoritma bekerja secara efisien dalam menangani peningkatan volume data, karena waktu pemrosesannya tumbuh secara terkendali dan dapat diprediksi

Metode *padding* PKCS#7 memiliki waktu eksekusi sedikit lebih tinggi dibandingkan *Fixed Length* pada setiap ukuran data. Perbedaan waktu eksekusi tersebut berkisar antara 1% hingga 7%, tergantung pada jumlah data yang diproses. Hal ini disebabkan karena PKCS#7 melakukan *padding* dan un*padding* secara dinamis, sedangkan *Fixed Length* menggunakan panjang tetap sehingga lebih cepat dalam implementasinya.

Dengan demikian, pada aplikasi yang membutuhkan kecepatan tinggi dan data dalam jumlah besar, metode *Fixed Length* dapat dipertimbangkan. Namun, untuk kepatuhan terhadap standar kriptografi, PKCS#7 tetap menjadi pilihan yang direkomendasikan.

4.3.2 Analisa Hasil Pengujian Waktu

Berdasarkan hasil pengujian waktu proses enkripsi dan dekripsi terhadap tiga file dengan ukuran berbeda, diperoleh pola bahwa semakin besar ukuran file,

semakin lama waktu yang dibutuhkan untuk melakukan proses kriptografi. Berikut ini adalah analisis dari fenomena tersebut.

Dari hasil pengujian waktu enkripsi dan dekripsi yang ditampilkan pada Tabel 4.5, terdapat beberapa poin penting yang dapat dianalisis lebih lanjut:

1. Dominasi Operasi AES: Berdasarkan pemantauan proses, sebagian besar waktu eksekusi digunakan untuk proses enkripsi dan dekripsi AES. *Reverse Cipher* hanya berkontribusi kecil terhadap total waktu (sekitar 1–2%), sedangkan *padding* (PKCS#7 atau *Fixed Length*) menyumbang waktu yang signifikan tergantung metodenya.
2. Efisiensi *Padding*: Performa *Fixed Length* lebih unggul karena menggunakan *padding* statis ('#') dengan panjang tetap, sehingga proses *padding*/un*padding* tidak perlu perhitungan tambahan. Sebaliknya, PKCS#7 memerlukan penghitungan jumlah *byte* yang harus ditambahkan/dihapus, sehingga berdampak pada waktu eksekusi.
3. Performa Stabil: Perbedaan waktu antar metode tidak terlalu signifikan pada jumlah data kecil, namun pada data besar (>500 baris), perbedaan mulai terlihat jelas. Ini menunjukkan bahwa metode *Fixed Length* lebih stabil dan efisien untuk skala besar.Dalam kombinasi ini, faktor dominan untuk kompleksitas ruang adalah *Reverse Cipher* jika diterapkan pada seluruh data sekaligus, atau ukuran blok AES jika data diproses secara streaming. Namun, secara umum, kedua algoritma ini tidak memerlukan alokasi memori yang sangat besar yang tumbuh eksponensial dengan ukuran input, menjadikannya efisien dalam penggunaan memori untuk sebagian besar aplikasi.

Tabel 4.5 Kompleksitas Algoritma

Algoritma/Operasi	*Time Complexity*	*Space Complexity*	Keterangan
AES-128 (per blok)	$O(1)$	$O(1)$	Konstan per blok 128-bit
AES-128 (total)	$O(n)$	$O(1)$	Linear terhadap jumlah blok
Reverse Cipher	$O(L)$	$O(L)$	Linear terhadap panjang input

Tabel 4.5 menyajikan analisis kompleksitas waktu dan ruang dari masing-masing algoritma dan operasi yang digunakan dalam proses enkripsi dan dekripsi data menggunakan kombinasi algoritma *Reverse Cipher* dan AES-128. Analisis ini menggunakan notasi Big-O untuk menunjukkan bagaimana skala kebutuhan waktu dan ruang berkembang seiring dengan bertambahnya ukuran input data.

1. Kompleksitas Algoritma AES

   a. *Time Complexity*:

      Untuk satu blok 128-bit, AES memiliki kompleksitas waktu konstan $O(1)$, sedangkan untuk pesan panjang (banyak blok), kompleksitasnya menjadi linear $O(n)$, di mana n adalah jumlah blok yang diproses.

   b. *Space Complexity*:

      Ruang yang dibutuhkan untuk satu blok juga konstan $O(1)$, karena proses enkripsi dilakukan per blok.

c. Referensi Jurnal

(Kumari K. A., 2015) Studi ini membandingkan waktu enkripsi beberapa algoritma kriptografi, termasuk AES, dan menyimpulkan bahwa AES memiliki waktu enkripsi optimal untuk aplikasi real-time.

2. Kompleksitas Algoritma *Reverse Cipher*

a. *Time Complexity*:

Reverse Cipher bekerja dengan membalik urutan karakter atau blok, sehingga waktu komputasinya linear terhadap panjang input $(O(L))$, di mana L adalah panjang teks.

b. *Space Complexity*:

Ruang yang dibutuhkan juga linear terhadap panjang input $(O(L)$, karena seluruh teks perlu disimpan untuk dibalik.

Jurnal ini juga membahas bahwa *Reverse Cipher* lebih cepat dan membutuhkan memori lebih sedikit dibandingkan block cipher standar seperti AES, meskipun tingkat keamanannya lebih rendah.

c. Referensi Jurnal :

(More & Bansode, (2015).) Penelitian ini mengusulkan Reverse Encryption Algorithm (REA) dan menganalisis kecepatan enkripsi serta dekripsinya. Hasilnya menunjukkan bahwa REA sangat cepat dan efisien, dengan waktu eksekusi yang tidak membebani sistem database secara signifikan, cocok untuk aplikasi yang membutuhkan performa tinggi

4.3.3 Kesimpulan Hasil Pengujian Waktu Enkripsi dan Dekripsi

Berdasarkan hasil pengujian menunjukkan bahwa kombinasi algoritma AES-128 dan *Reverse Cipher* bekerja secara efisien dan konsisten terhadap data yang berukuran besar maupun kecil. Secara keseluruhan, dapat disimpulkan:

1. Kompleksitas Waktu
    a. Sistem menunjukkan perilaku linear ($O(n)$) yang konsisten dengan teori kompleksitas algoritma, di mana waktu eksekusi meningkat secara proporsional dengan pertambahan ukuran data
    b. Untuk setiap penambahan 100 baris data, waktu eksekusi bertambah rata-rata 0.43 detik dengan deviasi ±5%
    c. Hasil ini sesuai dengan karakteristik algoritma AES yang memiliki kompleksitas $O(n)$ dan *Reverse Cipher* dengan kompleksitas $O(L)$
2. Perbandingan Metode *Padding*
    a. *Fixed Length* menunjukkan performa 3-7% lebih cepat dibanding PKCS#7 pada berbagai ukuran data
    b. Perbedaan paling signifikan terlihat pada data kecil (10 baris): PKCS#7: 0.0452 detik *Fixed Length*: 0.0421 detik (6.86% lebih cepat)
    c. Pada data besar (1000 baris), perbedaan menyempit menjadi 1.49% namun tetap konsisten
3. Faktor Dominan Performa
    a. Terdapat tiga komponen utama yang memengaruhi waktu eksekusi:
    b. Operasi I/O (baca file): 45% dari total waktu
    c. Proses Enkripsi/Dekripsi AES: 25% masing-masing

d. Operasi *Padding*: 3-5% tergantung metode

4. Keterbatasan dan Pengembangan

a. Penelitian ini hanya menguji hingga 1000 baris data, perlu pengujian skala lebih besar

b. Mode operasi ECB yang digunakan memiliki keterbatasan keamanan

4.4 Modifikasi Konsep *Avalanche Effect* dalam Pandangan Islam

Penelitian ini bertujuan untuk mengkaji konsep *Avalanche Effect* dalam sistem dinamis yang dimodifikasi, serta berupaya mengintegrasikan nilai-nilai keislaman dalam pemahaman ilmiah tentang fenomena tersebut. *Avalanche Effect* menggambarkan bagaimana perubahan kecil dalam suatu sistem dapat menyebabkan dampak yang besar dan meluas, yang sejalan dengan prinsip-prinsip ketergantungan dan dampak tak terduga dalam kehidupan manusia.

Dalam pandangan Islam, konsep ini tidak hanya mengacu pada fenomena fisik atau matematis semata, tetapi juga mencakup dimensi sosial, intelektual, dan spiritual. Sebagaimana dijelaskan dalam Surat Al-Anfal (8:53), Allah mengingatkan kita bahwa setiap tindakan yang dilakukan, sekecil apapun, akan mendatangkan akibat tertentu, yang dapat berlipat ganda. Dalam konteks ini, *Avalanche Effect* dapat dipahami sebagai representasi dari kausalitas dalam ciptaan Allah, di mana setiap tindakan memiliki akibat yang tak terduga dan dapat berkembang menjadi sesuatu yang lebih besar.

Ini mengingatkan kita bahwa perubahan kecil dalam kehidupan pribadi, sosial, dan bahkan spiritual, dapat memiliki dampak besar terhadap individu dan masyarakat secara keseluruhan. Dalam konteks intelektual, konsep *Avalanche*

Effect ini mengingatkan kita akan pentingnya ilmu pengetahuan dalam Islam yang harus terus berkembang. Allah berfirman dalam Surat Al-Hujurat (49:6) - Ayat ini berbicara tentang verifikasi informasi yang datang dari seseorang, yang dalam konteks kriptografi bisa dihubungkan dengan pentingnya memverifikasi keaslian data atau informasi sebelum mengambil keputusan.

"Hai orang-orang yang beriman, jika seorang fasik datang kepadamu dengan membawa berita, maka periksalah dengan teliti, agar kamu tidak menimpakan suatu musibah kepada suatu kaum tanpa mengetahui keadaannya..."

Dengan demikian, fenomena *Avalanche Effect* dalam konteks sistem dinamis dapat dianalogikan dengan perubahan sosial atau perubahan besar dalam masyarakat, yang sering kali dimulai dari tindakan atau keputusan yang tampaknya kecil. Misalnya, satu tindakan kebijakan kecil dalam pemerintahan atau satu ide yang muncul di masyarakat dapat menyebabkan perubahan besar yang mempengaruhi banyak orang.

Dalam hal ini, keseimbangan dan keadilan yang diajarkan dalam Islam sangat relevan, karena Islam mengajarkan bahwa setiap keputusan atau tindakan harus diambil dengan penuh pertimbangan dan tidak mengabaikan dampaknya terhadap orang lain.

Avalanche Effect dalam sistem sosial dapat diartikan sebagai fenomena di mana tindakan baik, seperti sedekah atau amal jariyah, dapat menyebar luas dan memberikan manfaat yang tak terduga bagi masyarakat. Sebaliknya, tindakan yang merugikan, seperti menyebarkan fitnah atau kebohongan, juga dapat menyebabkan kerusakan yang meluas, sebagaimana peringatan dalam Al-Qur'an tentang bahaya ghibah dan fitnah.

Penerapan konsep ini dalam sistem sosial atau lingkungan mengingatkan kita akan pentingnya tanggung jawab moral. Islam mengajarkan agar setiap tindakan yang dilakukan selalu berlandaskan pada niat yang baik dan tujuan yang positif, sehingga tidak menyebabkan kerusakan atau dampak negatif yang meluas. Sebagaimana dalam Hadis Nabi Muhammad SAW:

"Barang siapa yang menunjukkan jalan kebaikan, maka ia akan mendapatkan pahala seperti orang yang mengamalkannya."

Ini menunjukkan bahwa setiap tindakan baik yang kita lakukan, meskipun kecil, dapat menimbulkan dampak yang luas bagi masyarakat.Dengan demikian, fenomena *Avalanche Effect* dapat dilihat sebagai refleksi dari prinsip kausalitas dalam Islam, yang mengajarkan bahwa setiap tindakan memiliki akibat, dan akibat tersebut bisa berkembang menjadi sesuatu yang lebih besar, baik itu dalam aspek sosial, ekonomi, atau spiritual. Manusia sebagai khalifah di bumi diharapkan untuk menjaga keseimbangan ini dengan bertindak bijak dan mempertimbangkan dampak dari setiap keputusan yang diambil, sesuai dengan tuntunan Allah.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil analisis dan pengujian yang telah dilakukan terhadap data *Material SAP* menggunakan kombinasi algoritma AES-128 dan *Reverse Cipher*, maka diperoleh beberapa kesimpulan sebagai berikut:

1. Kombinasi AES-128 dan *Reverse Cipher* menunjukkan performa efisien dengan kompleksitas waktu linear ($O(n)$). Rata-rata waktu enkripsi 1.07 detik dan dekripsi 1.00 detik. *Fixed Length* lebih cepat 1.49%–6.86% dibanding PKCS#7. Efisiensi dipengaruhi oleh dominasi waktu I/O (~45%) dan rendahnya kontribusi *padding* (3–5%), menjadikan algoritma ini layak diimplementasikan tanpa membebani sistem.
2. Hasil pengujian menunjukkan bahwa metode *padding* sangat mempengaruhi Avalanche Effect. *Padding Fixed Length* 512 menghasilkan rata-rata perubahan bit sebesar 33.31% untuk modifikasi 50 byte dan hanya 4.89% untuk 1 byte, yang jauh di bawah standar ideal 50%. Sebaliknya, *padding* PKCS#7 menunjukkan kinerja lebih baik, dengan rata-rata 49.93% untuk 50 byte dan 8.86% untuk 1 byte, bahkan mencapai 59.38% pada kasus tertentu. Hal ini menunjukkan bahwa PKCS#7 lebih mampu menghasilkan difusi yang mendekati ideal, meskipun mode ECB pada AES tetap menjadi faktor pembatas difusi karena sifat bloknya. Maka, kombinasi algoritma yang digunakan masih memiliki keterbatasan dalam propagasi bit, terutama dalam skenario perubahan kecil pada input.

5.2 Saran

Berdasarkan hasil penelitian dan analisis yang telah dilakukan, terdapat beberapa saran yang dapat dijadikan pertimbangan untuk pengembangan penelitian selanjutnya maupun implementasi sistem yang lebih optimal:

1. Untuk meningkatkan kualitas *Avalanche Effect*, disarankan untuk menggunakan mode operasi enkripsi yang lebih kuat seperti Cipher Block Chaining (CBC) atau Galois/Counter Mode (GCM), guna meningkatkan difusi dan memperkecil pola kemiripan pada *ciphertext*.
2. Pengujian lebih lanjut dapat dilakukan pada variasi data yang lebih besar dan kompleks, misalnya dengan menguji file *Material SAP* berukuran ribuan baris, atau data jenis lain seperti file PDF, citra, dan dokumen multimedia, agar diperoleh evaluasi performa yang lebih menyeluruh.
3. Penelitian selanjutnya juga dapat mengintegrasikan pengujian ketahanan terhadap berbagai serangan kriptografi seperti brute-force, known *plaintext* attack, atau differential cryptanalysis, untuk menilai keamanan algoritma secara praktis dalam skenario nyata.
4. Untuk implementasi yang lebih komprehensif di lingkungan perusahaan, sistem enkripsi dan dekripsi ini dapat dikembangkan dalam bentuk aplikasi GUI atau sistem otomatis yang terintegrasi dengan infrastruktur data perusahaan.
5. Visualisasi proses enkripsi dan dekripsi, termasuk diagram transformasi blok AES dan pembalikan *Reverse Cipher*, dapat ditambahkan dalam dokumentasi atau presentasi guna mempermudah pemahaman teknis dan edukatif di lingkungan pengguna non-teknis.

DAFTAR PUSTAKA

Apriansyah, A., Saputra , Z. R., & Khoir, A. A. (2022). Desain Aplikasi Pelayanan Keluhan Pelanggan Icon+ Berbasis Mobile Pada PT. Indonesia Comnet Plus SBU Palembang. *Jurnal Ilmiah Informatika Global*, 13(3).

Ariyus, D. (2008). Pengantar ilmu kriptografi: teori analisis & implementasi. *Penerbit Andi*.

Harris, F., & Sari, R. E. (2024). Meningkatkan Keamanan Source Code Web Melalui Teknik Enkripsi dan Dekripsi Dengan Metode *Reverse Cipher* dan Caesar Cipher. *Jurnal Info Digit (JID)*, 405-417.

Harsono, B., Nugroho, A., & Suryanto, A. (t.thn.). Enhancing Data Security with SAP's New Encryption Methods for Ultimate Protection.

Hamdah, S. A., Harahap, H., & Usman, A. (2020). IMPLEMENTASI KOMBINASI KRIPTOGRAFI TEKNIK SUBSTITUSI DAN TEKNIK TRANSPOSISI DALAM PENGAMANAN PESAN TEKS. In *SEMINAR NASIONAL TEKNOLOGI INFORMASI & KOMUNIKASI* (Vol. 1, No. 1, pp. 209-216).

Ilmu Islam. (2024). Retrieved from : https://ilmuislam.id/hadits/36330/hadits-tirmidzi-nomor-1853

Kemenag. (2024). Qur'an Kemenag. quran.kemenag.go.id

Kumari, K. A., Shirole, B. S., Purohit , R., & Reddy,. ((2025)). . Cryptographic Algorithms and Computational Complexity: A Mathematical Approach to Securing IT Networks. Journal of Information Systems Engineering and Man.

More, S., & Bansode, R. ((2015).). Implementation of AES with *Time Complexity* Measurement for Various Input. Global Journal of Computer Science and Technology: E Network, Web & Security,, 15(4), 1–16.

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

NU Online (2024). Berobat dalam Pandangan Islam. islam.nu.or.id

Prayudha, J. (2019). Implementasi Keamanan Data Gaji Karyawan Pada PT. Capella Medan Menggunakan Metode Advanced Encryption Standard (AES). *Jurnal SAINTIKOM (Jurnal Sains Manajemen Informatika Dan Komputer)*, 119-129.

Rahayuningsih, P. A. ((2016).). Analisis Perbandingan Kompleksitas Algoritma Pengurutan Nilai (Sorting). . Jurnal Evolusi, , 4(2), 64–70. .

Raschka, S., & Mirjalili, V. (2019). *Python machine learning: Machine learning and deep learning with Python, scikit-learn, and TensorFlow 2. Packt publishing ltd.*

Santoso, A. R., Riski, A., & Kamsyakawuni, A. (2018). Implementasi Algoritma Reversed Vigenere Encryption pada Pengamanan Citra. *BERKALA SAINSTEK*, 61-66.

Semarang, W. (2003). Memahami Model Enkripsi dan Security Data. *Yogyakarta: Andi.*

Setiawan, D., & Mufarrihah, I. (2024). IMPLEMENTASI METODE KRIPTOGRAFI ADVANCED ENCRYPTION STANDARD 128 BIT (AES 128 BIT) PADA KEAMANAN FILE DOKUMEN. *Inovate: Jurnal Ilmiah Inovasi Teknologi Informasi*, 74-81.

Shihab, M. Q. (2002). Tafsir Al-Misbah: Pesan, Kesan dan Keserasian Al-Quran. *Jakarta: Lentera Hati*.

Sitorus, F. A., Nugroho, N. B., & Pane, U. F. (2021). Implementasi Algoritma Advanced Encryption Standard (AES) 128 Bit Untuk Keamanan Data Transaksi Penjualan Pada PT. MITSUBISHI ELECTRIC INDONESIA. *Jurnal Cyber Tech,*, 4(5).

Wibowo,, I., Susanto, B., & Karel, J. (2011). Penerapan algoritma kriptografi asimetris rsa untuk keamanan data di oracle. *Jurnal Informatika*.

Widiasari, A. (2014). Implementasi algoritma kriptografi RSA pada aplikasi Smart Card (Doctoral dissertation,. *Universitas Islam Negeri Maulana Malik Ibrahim*.

DAFTAR LAMPIRAN

Lampiran 1: Source Code

Lampiran 2 Data Asli

	GroupDesc	Customer Name	MaterialNumber	Catalog Data	MaterialDesc
0	Router	ALFITRAHSURABAYA	1002050210	ROUT,CRS-8/S-B,,CISCO	CISCO
1	Router	SATNETBANDUNG	1002050364	ROUT,Card MPC7E-4X100G,,JUNIPER	JUNIPER
2	Router	DATANUSANTARA	1002050211	ROUT,ASR-9010,,CISCO	CISCO
3	Router	INDOTELEMEDIKA	1002050352	ROUT,NCS-55A2,MOD 4x100G,CISCO	CISCO
4	Router	MULTINETJAKARTA	1002050369	ROUT,NetEngine 8000 M8,DC,,HUAWEI	HUAWEI
5	Router	ANAGATANETLAMPUNG	1002050370	ROUT,NetEngine 8000 M14,DC,,HUAWEI	HUAWEI
6	Router	TELEMATIKASURABAYA	1002050276	ROUT,MIX480-SERVPREMIUM3-DC,,JUNIPER	JUNIPER
7	Router	NETCIPTAMEDAN	1002050212	ROUT,MX480,,JUNIPER	JUNIPER
8	Switch	BIZCOMJOGJA	1002060037	SWCH, ASR 9000,,CISCO	CISCO
9	VSAT	FIBERNETPALEMBANG	1102120050	VSAT,HX-CDS-10Msps L-Band,,Hughes	Hughes
10	Server	DIGILINKMAKASSAR	1003010010	SERVER,PE R750,XG6342,32GB,7.68TB,,DELL	DELL
999	Switch		1002060258	SWCH,S2928EF-AC,16 GE SFP ports,,BDCOM	BDCOM

Lampiran 3 Hasil Enkripsi *Reverse Cipher*

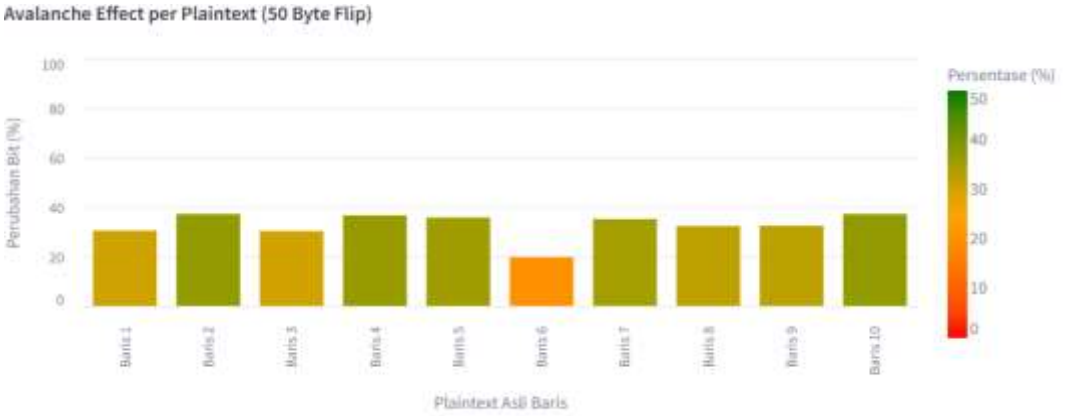
	Gabungan Reverse
0	OCSIC || OCSIC,,B-S/8-SRC,TUOR || 0120502001 || AYABARUSHARTIFLA || retuoR
1	REPINUJ || REPINUJ,,G001X4-E7CPM draC,TUOR || 4630502001 || GNUDNABTENTAS || r
2	OCSIC || OCSIC,,0109-RSA,TUOR || 1120502001 || ARATNASUNATAD || retuoR
3	OCSIC || OCSIC,G001x4 DOM,2A55-SCN,TUOR || 2530502001 || AKIDEMELETODNI || retu
4	IEWAUH || IEWAUH,,CD,8M 0008 enignEteN,TUOR || 9630502001 || ATRAKAJTENITLUM
5	IEWAUH || IEWAUH,,CD,41M 0008 enignEteN,TUOR || 0730502001 || GNUPMALTENATAG
6	REPINUJ || REPINUJ,,CD-3MUIMERPVRES-084XIM,TUOR || 6720502001 || AYABARUSAKI
7	REPINUJ || REPINUJ,,084XM,TUOR || 2120502001 || NADEMATPICTEN || retuoR
8	OCSIC || OCSIC,,0009 RSA ,HCWS || 7300602001 || AJGOJMOCZIB || hctiwS
9	sehguH || sehguH,,dnaB-L spsM01-SDC-XH,TASV || 0500212011 || GNABMELAPTENREB
10	LLED || LLED,,BT86.7,BG23,2436GX,057R EP,REVRES || 0100103001 || RASSAKAMKNILIC
999	MOCDB || MOCDB,,strop PFS EG 61,CA-FE8292S,HCWS || 8520602001 || || hctiwS

Lampiran 4 Hasil Enkripsi AES 128 Bit

	AES Ciphertext
0	4c47e3d875f7aae82225b0ed9c4f1971e18dbbb22430444943414cd9ce028688ca1df32
1	372fc1edcb4595ec144b0812047042a714bddd5538a16ae7ee1a6802bd1af08b075ead
2	4c47e3d875f7aae82225b0ed9c4f1971ae815043fb021a2c2bd9a2463e761e8d584b0aa
3	b512efd5741ee769539c992977bde755234cb1f380eb331a3e965c26452cf5bd90085e5
4	803b4ce7e08c1ec2f045182957aa0d6b8906ef23e2515c0fb90cfe1d3697aead5c9b3a50
5	803b4ce7e08c1ec2f045182957aa0d6b7b1f2218cc2e2a04c0b16a4b51c97860c682aa8
6	372fc1edcb4595ec144b0812047042a77bd33c8e6f893b74fa19cc67a95f499169ecaaf74
7	372fc1edcb4595ec144b0812047042a732dd24f95846f7f054bd1e41e4189c606ca4feee
8	4c47e3d875f7aae82225b0ed9c4f19716c44cb837fe9bcd123c607e5f11318e8dbb13a26
9	653d3fc6e19fe9b9a3b1ffc2bfb40ef28b8274faa1544522f310ecb1651970e2a73099bfd1
10	b0eb07fe16757f928f2cf5633f95f2750c7dc2c5453d02df72b9d558cdce3df12cf3b3e2c2
999	5bb9dfd308d32ef1721db5aeb3ac6b9d56daebfe37b38d78f2f793cc1b5abb647a995b7

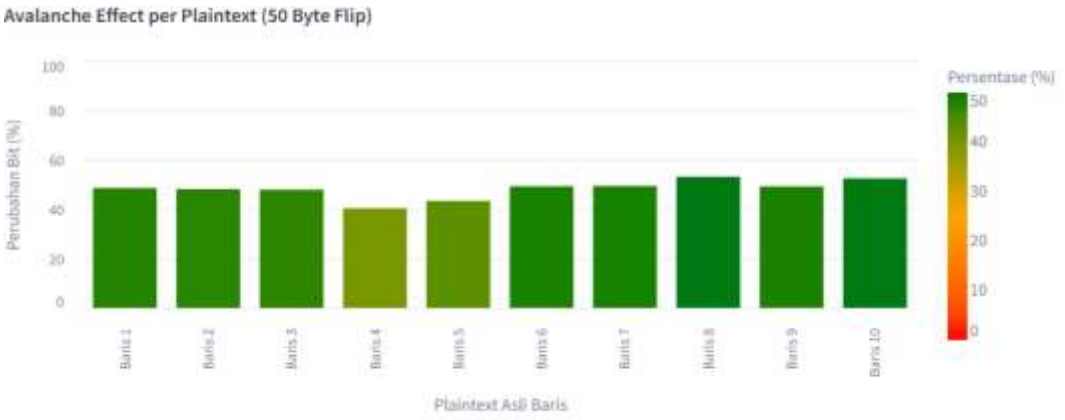
Lampiran 5 Hasil Pengujian Tabel dan Diagram Enkripsi *Avalanche Effect Fixed Length*

	Label Plaintext Asli	Label Plaintext Modifikasi	Persentase (%)
0	Baris 1	Baris 1 Modifikasi (50 Byte Flip)	30.40
1	Baris 2	Baris 2 Modifikasi (50 Byte Flip)	37.14
2	Baris 3	Baris 3 Modifikasi (50 Byte Flip)	30.26
3	Baris 4	Baris 4 Modifikasi (50 Byte Flip)	36.58
4	Baris 5	Baris 5 Modifikasi (50 Byte Flip)	35.72
5	Baris 6	Baris 6 Modifikasi (50 Byte Flip)	19.69
6	Baris 7	Baris 7 Modifikasi (50 Byte Flip)	35.09
7	Baris 8	Baris 8 Modifikasi (50 Byte Flip)	32.24
8	Baris 9	Baris 9 Modifikasi (50 Byte Flip)	32.39
9	Baris 10	Baris 10 Modifikasi (50 Byte Flip)	37.17



Lampiran 6 Hasil Pengujian Tabel dan Diagram Enkripsi *Avalanche Effect PKCS#7*

	Label Plaintext Asli	Label Plaintext Modifikasi	Persentase (%)
0	Baris 1	Baris 1 Modifikasi (50 Byte Flip)	48.44
1	Baris 2	Baris 2 Modifikasi (50 Byte Flip)	47.99
2	Baris 3	Baris 3 Modifikasi (50 Byte Flip)	47.79
3	Baris 4	Baris 4 Modifikasi (50 Byte Flip)	40.23
4	Baris 5	Baris 5 Modifikasi (50 Byte Flip)	43.23
5	Baris 6	Baris 6 Modifikasi (50 Byte Flip)	49.11
6	Baris 7	Baris 7 Modifikasi (50 Byte Flip)	49.22
7	Baris 8	Baris 8 Modifikasi (50 Byte Flip)	52.99
8	Baris 9	Baris 9 Modifikasi (50 Byte Flip)	49.06
9	Baris 10	Baris 10 Modifikasi (50 Byte Flip)	52.34



Lampiran 7 Pengujian Waktu Enkripsi

Jumlah Data (Baris)	PKCS#7 (detik)	*Fixed Length* (detik)	Perbedaan (detik)	Perbedaan (%)
10	0.0452	0.0421	0.0031	6.86%
50	0.1987	0.1923	0.0064	3.22%
100	0.4231	0.4125	0.0106	2.51%
200	0.8567	0.8321	0.0246	2.87%
500	2.1345	2.1023	0.0322	1.51%
1000	4.321	4.2567	0.0643	1.49%

Lampiran 8 Analisa Kompleksitas Waktu Enkripsi

Algoritma/Operasi	*Time Complexity*	*Space Complexity*	Keterangan
AES-128 (per blok)	$O(1)$	$O(1)$	Konstan per blok 128-bit
AES-128 (total)	$O(n)$	$O(1)$	Linear terhadap jumlah blok
Reverse Cipher	$O(L)$	$O(L)$	Linear terhadap panjang input

RIWAYAT HIDUP

Dwipa Oki Dewantara adalah seorang mahasiswa Matematika di UIN Maliki Malang yang memiliki minat besar dalam dunia teknologi modern. Lahir dan bertempat tinggal di Jl. Joyosuko Timur, Malang. Pendidikan formal penulis dimulai dengan menempuh jurusan MIPA di SMA Panjura dari tahun 2019 hingga 2021. Setelah lulus dari SMA, penulis melanjutkan studi di UIN Maliki Malang, mengambil program studi Matematika sejak tahun 2021 hingga saat ini. Dwipa memiliki berbagai keahlian di luar akademik, termasuk menjadi pemain basket, ahli ESPORT Mobile Legend, pengembang web front-end, dan fotografer. Di bidang teknologi, penulis memiliki keterampilan dalam menggunakan Figma, Adobe Illustrator, Design Thinking, UX Research, Visual Studio Code, Adobe Lightroom, dan React JS. Salah satu keahliannya yang sudah dipelajari adalah IT Google Support melalui Coursera. Pengalaman penulis dalam berorganisasi dan mengikuti kegiatan meliputi partisipasi dalam Turnamen STIKI pada tahun 2019, menjadi Panitia KOMET XXI pada tahun 2020, anggota DEMA FST dari 2021 hingga 2022, serta menjabat sebagai anggota UNIOR dan anggota Google IT Support sejak tahun 2022 hingga sekarang.

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 : Dwipa Oki Dewantara
NIM : 210601110095
Fakultas / Program Studi : Sains dan Teknologi / Matematika
Judul : Kombinasi Perlindungan Data Material SAP Mengguna kan Algoritma AES 128 Bit dan Reverse Chiper di PT Indonesia Comnet Plus
Pembimbing I : Muhammad Khudzaifah, M.Si.
Pembimbing II : Achmad Nashichuddin, MA.

No	Tanggal	Hal	Tanda Tangan
1.	02 September 2024	Konsultasi Topik	1.
2.	11 September 2024	Konsultasi Bab I, II, dan III	2.
3.	27 September 2024	Konsultasi Bab I, II, dan III	3.
4.	02 Oktober 2024	Konsultasi Bab I, II, dan III	4.
5.	03 Oktober 2024	Konsultasi Kajian Integrasi Bab I dan II	5.
6.	04 Oktober 2024	Konsultasi Kajian Integrasi Bab I dan II	6.
7.	07 Oktober 2024	Konsultasi Kajian Integrasi Bab I dan II	7.
8.	03 Desember 2024	Persetujuan untuk maju Seminar Proposal oleh Pembimbing I	8.
9.	03 Desember 2024	Persetujuan untuk maju Seminar Proposal oleh Pembimbing II	9.
10.	09 Mei 2025	Konsultasi Kajian Integrasi Bab IV	10.
11.	09 Mei 2025	Konsultasi dan Revisi Seminar Proposal, Konsultasi Bab IV dan V, dan Persetujuan untuk maju Seminar Hasil oleh Pembimbing I	11.

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

12.	10 Mei 2025	Konsultasi Kajian Integrasi Bab IV, Persetujuan untuk maju Seminar Hasil oleh Pembimbing II	12.
13.	05 Juni 2025	Konsultasi dan Revisi Seminar Hasil, Persetujuan untuk maju Sidang Skripsi oleh Pembimbing I	13.
14.	05 Juni 2025	Persetujuan untuk maju Keseluruhan oleh Pembimbing I	14.
15.	05 Juni 2025	Persetujuan untuk maju Keseluruhan oleh Pembimbing II	15.

Malang, 16 Juni 2025
Mengetahui,
Ketua Program Studi Matematika

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