LAPORAN KUIS 3

Fungsi Enkripsi Menggunakan Algoritma *Data Encryption Standard* (DES)

Oleh:

Amanda Putri Aprilliani	105222001
Raihan Akira Rahmaputra	105222040
Ni Putu Merta Bhuana Ningsih	105222008

MATA KULIAH KRIPTOGRAFI

PROGRAM STUDI ILMU KOMPUTER

FAKULTAS SAINS DAN ILMU KOMPUTER

UNIVERSITAS PERTAMINA

2025

BAB I PENDAHULUAN

1.1 Latar Belakang

Dalam era digital saat ini, keamanan informasi menjadi perhatian utama, terutama dengan berkembangnya teknologi dan internet yang memungkinkan akses dan transmisi data dengan mudah. Hal ini diperkuat oleh pandangan pakar Digitalisasi Kampus dan Pendidikan Tinggi seperti Dr. Dandi, yang memaparkan strategi untuk memastikan keamanan siber di lingkungan akademik dalam seminar nasional beberapa hari lalu [1]. Pentingnya keamanan siber ini menunjukkan bahwa perlindungan informasi telah menjadi prioritas utama di berbagai sektor. Untuk melindungi data sensitif, metode enkripsi menjadi salah satu solusi utama yang digunakan untuk menjaga kerahasiaan dan integritas data [2]. Enkripsi merupakan proses atau sebuah cara untuk mengubah data yang semula mudah dimengerti menjadi bentuk yang sulit dipahami [3].

Salah satu algoritma enkripsi yang populer adalah Data Encryption Standard (DES), yang dikembangkan pada tahun 1970-an oleh International Business Machines Corporation dan kemudian diadopsi oleh pemerintah Amerika Serikat sebagai standar enkripsi untuk data sensitif. Algoritma ini dapat mengenkripsi dan mendekripsi informasi dalam format biner. DES menggunakan kunci sepanjang 64 bit yang terdiri dari 56 bit efektif untuk proses enkripsi dan 8 bit tambahan sebagai bit paritas untuk deteksi kesalahan. Data dapat didekripsi jika memiliki dan menggunakan kunci yang sama dengan yang digunakan untuk mengenkripsinya [4].

Meskipun saat ini DES dianggap tidak cukup aman karena kemajuan kekuatan komputasi dan kemunculan algoritma enkripsi yang lebih kuat seperti AES [5], DES tetap menjadi dasar penting dalam pemahaman kriptografi simetris. Algoritma ini memiliki beberapa keterbatasan, terutama panjang kunci yang hanya 56 bit yang rentan terhadap serangan *brute force* [6]. Namun demikian, DES memberikan kontribusi besar dalam perkembangan kriptografi modern dan menjadi dasar bagi pengembangan algoritma yang lebih aman. Oleh karena itu, dalam laporan ini tim kami akan mengimplementasikan algoritma DES menggunakan bahasa pemrograman Python untuk memahami cara kerja algoritma ini secara praktis.

1.2 Tujuan

Tujuan laporan ini dibuat adalah untuk memahami cara algoritma *Data Encryption Standard* (DES) bekerja, serta bagaimana proses enkripsi dilakukan dalam *python*, serta untuk menyelesaikan Kuis 3.

1.3 Batasan Masalah

Dalam laporan dan code yang telah dibuat, algoritma DES ini hanya mencakup proses enkripsi saja. Proses dekripsi, yang diperlukan untuk mengembalikan data yang telah dienkripsi ke bentuk semula, belum diimplementasikan dalam kode ini. Oleh karena itu, analisis yang dilakukan terbatas pada pengujian enkripsi dan validasi hasil enkripsi terhadap kunci yang diberikan. Hasil enkripsi dalam bentuk biner juga belum diuji dalam aplikasi yang lebih luas, hanya sampai implementasi dalam *python*.

BAB II LANDASAN TEORI

2.1 Kriptografi Simetris

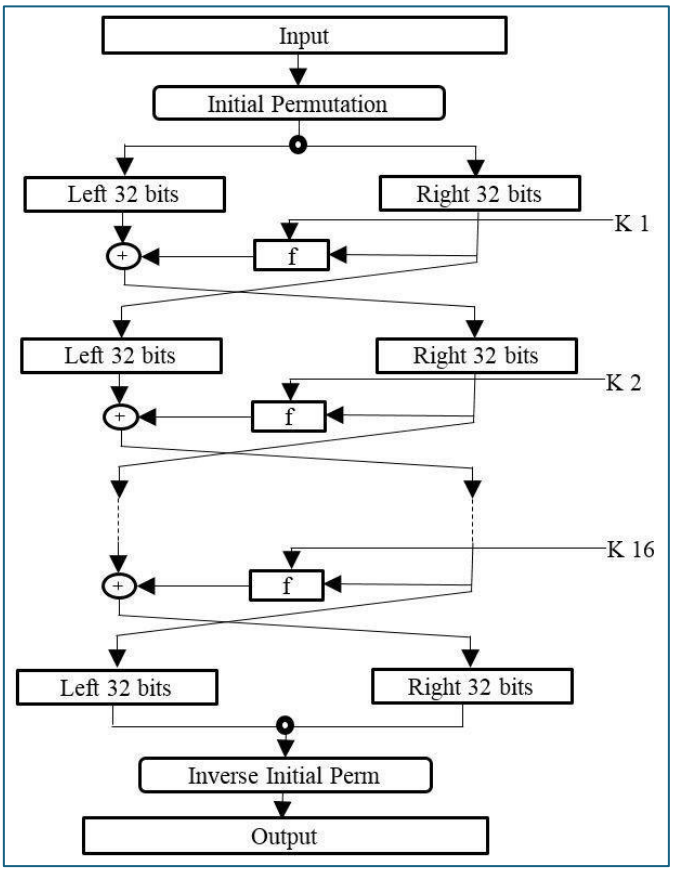
Kriptografi simetris merupakan salah satu teknik enkripsi fundamental yang menggunakan kunci yang sama untuk proses enkripsi dan dekripsi data. Dalam era transformasi digital yang berkembang pesat, keamanan informasi menjadi aspek krusial terutama ketika data sensitif ditransmisikan melalui jaringan yang tidak aman [7]. Kriptografi simetris menawarkan solusi efisien dengan algoritma yang ringan secara komputasi, sehingga cocok untuk memproses data dalam jumlah besar dengan kecepatan tinggi [8].

Berbagai algoritma kriptografi simetris telah dikembangkan untuk memenuhi kebutuhan keamanan data, mulai dari algoritma klasik seperti Caesar cipher hingga algoritma modern seperti *Advanced Encryption Standard (AES)*, dengan masing-masing memiliki karakteristik arsitektur, fleksibilitas, reliabilitas, dan tingkat keamanan yang berbeda [9].

2.2 Data Encryption Standard (DES)

Data Encryption Standard (DES) merupakan algoritma enkripsi simetris berbasis block cipher yang dikembangkan oleh IBM pada tahun 1970-an dan diadopsi sebagai standar enkripsi nasional Amerika Serikat pada tahun 1977. DES mengoperasikan data dalam blok 64-bit dengan menggunakan kunci efektif sepanjang 56-bit, meskipun kunci eksternal berukuran 64-bit [10]. Algoritma ini menggunakan struktur jaringan Feistel dengan 16 putaran enkripsi, dimana setiap putaran melibatkan operasi permutasi, substitusi menggunakan S-box, dan operasi XOR. Meskipun DES telah dianggap tidak aman untuk aplikasi modern karena panjang kunci yang relatif pendek dan rentan terhadap serangan *brute force*, algoritma ini tetap memiliki nilai historis penting sebagai katalis dalam pengembangan studi kriptografi akademis dan menjadi dasar bagi pengembangan algoritma enkripsi yang lebih kuat seperti Triple DES dan AES.

2.3 Struktur Proses DES



Gambar 1 Alur DES

Algoritma DES mengoperasikan data dalam blok64-bit dan menggunakan kunci 56-bit (dari kunci eksternal 64-bit). Proses enkripsi DES terdiri dari beberapa tahap utama yang dijalankan secara berurutan:

a. Permutasi awal

Tahap pertama dalam proses DES adalah melakukan permutasi awal terhadap blok plainteks 64-bit. Permutasi ini menggunakan matriks permutasi awal (IP) yang telah ditentukan, bertujuan untuk mengacak urutan bit dalam plainteks sehingga posisi bit-bit terendah berubah sesuai dengan pola yang telah ditetapkan. Hasil dari permutasi awal ini akan menjadi input untuk tahap selanjutnya.

b. Pembagian data menjadi dua bagian (kiri dan kanan)

Setelah permutasi awal, data 64-bit dibagi menjadi dua bagian yang sama:

- **L_0 (Left)**: 32-bit bagian kiri
- **R_0 (Right)**: 32-bit bagian kanan

Kedua bagian ini akan menjadi input untuk 16 putaran enkripsi menggunakan jaringan Feistel.

c. Ekspansi dan XOR

Dalam setiap putaran, blok R (32-bit) akan diperluas menjadi 48-bit menggunakan matriks ekspansi E. Fungsi ekspansi ini memperluas 32-bit menjadi 48-bit dengan menduplikasi beberapa bit tertentu. Hasil ekspansi kemudian di-XOR dengan kunci internal Ki (48-bit) untuk putaran ke-i, menghasilkan vektor A berukuran 48-bit.

d. Substitusi dengan S-box

Vektor A (48-bit) hasil XOR dibagi menjadi 8 kelompok, masing-masing 6-bit. Setiap kelompok 6-bit kemudian diproses melalui kotak substitusi (S-box) yang berbeda, yaitu S_1 sampai S_8. Setiap S-box mengubah input 6-bit menjadi output 4-bit berdasarkan tabel substitusi yang telah ditentukan. Hasil dari 8 S-box ini digabungkan menjadi 32-bit.

e. Permutasi P (P-box)

Output 32-bit dari proses substitusi S-box kemudian dipermutasi menggunakan matriks permutasi P. Tujuan permutasi ini adalah untuk **mengacak** hasil substitusi sehingga output dari fungsi f dapat tersebar secara merata. Hasil permutasi P ini merupakan output final dari fungsi f.

f. Pertukaran Blok Kiri dan Kanan

Setelah fungsi f menghasilkan output 32-bit, dilakukan operasi:

- $\mathbf{L_i = R_{i-1}}$ (blok kiri baru sama dengan blok kanan sebelumnya)
- $\mathbf{R_i = L_{i-1} \oplus f(R_{i-1}, K_i)}$ (blok kanan baru adalah hasil XOR antara blok kiri sebelumnya dengan output fungsi f)

Proses ini diulang sebanyak 16 putaran dengan menggunakan kunci internal yang berbeda untuk setiap putaran.

g. Permutasi Akhir (Inverse Initial Permutation – IP^{-1})

Setelah 16 putaran selesai, hasil akhir berupa (R_{16}, L_{16}) digabungkan menjadi 64-bit. Perhatikan bahwa urutan penggabungannya adalah R_{16} diikuti L_{16}, bukan $L_{16}R_{16}$. Blok 64-bit ini kemudian dipermutasi menggunakan matriks permutasi awal balikan (IP^{-1}) untuk menghasilkan ciphertext final 64-bit.

h. Pembangkitan Kunci Internal

Secara paralel dengan proses enkripsi, algoritma DES juga membangkitkan 16 kunci internal (K_1 sampai K_{16}) dari kunci eksternal 64-bit melalui proses:

1. Permutasi PC-1 (64-bit → 56-bit)
2. Pembagian menjadi C_0 dan D_0 (masing-masing 28-bit)
3. Left shift sesuai tabel untuk setiap putaran
4. Permutasi PC-2 untuk menghasilkan kunci internal 48-bit

Struktur proses DES ini mengikuti pola jaringan Feistel yang memungkinkan proses dekripsi menggunakan algoritma yang sama dengan urutan kunci yang dibalik.

BAB III PEMBAHASAN

3.1 Implementasi Algoritma DES

Bab ini membahas implementasi spesifik algoritma DES yang telah dikembangkan dalam penelitian ini, bukan penjelasan umum tentang implementasi DES secara global. Fokus pembahasan adalah pada arsitektur, struktur kode, dan fitur-fitur khusus yang diterapkan dalam program yang telah dibuat.

3.1.1 Arsitektur Program

Program DES diimplementasikan menggunakan Python dengan struktur modular dalam Jupyter Notebook. Kode dibagi menjadi beberapa cell untuk memudahkan eksekusi dan debugging. Arsitektur terdiri dari:

- Konstanta DES (matriks permutasi dan S-Box)
- Fungsi utilitas untuk manipulasi bit
- Modul pembangkitan kunci internal
- Fungsi enkripsi utama
- Interface input/output

3.1.2 Fitur Utama

a) Input Fleksibel

Program menyediakan dua mode kunci - default dan custom 64-bit binary dari pengguna dengan validasi otomatis.

b) Output Detail

Menampilkan setiap tahap proses enkripsi termasuk pembangkitan 16 kunci internal, detail setiap putaran Feistel, dan hasil dalam format binary serta hexadecimal.

c) Error Handling

Validasi input untuk memastikan kunci 64-bit *binary* yang valid dan penanganan kesalahan yang informatif.

3.1.3 Struktur Implementasi

Python

```
# Struktur fungsi utama yang diimplementasikan
def des_encrypt(plaintext, key_binary):
    key_bits = binary_string_to_bits(key_binary)
    keys = generate_keys(key_bits)
    plaintext_bits = string_to_bits(plaintext)

# Enkripsi per blok 64-bit
ciphertext_bits = []
for i in range(0, len(plaintext_bits), 64):
    block = plaintext_bits[i:i+64]
    encrypted_block = des_encrypt_block(block, keys)
    ciphertext_bits.extend(encrypted_block)

return ciphertext_bits
```

Program menggunakan representasi bit sebagai list integer untuk memudahkan operasi *bitwise* dan *debugging*. Setiap fungsi dilengkapi dengan dokumentasi dan output yang informatif untuk keperluan analisis.

3.1.4 Keunggulan Implementasi

a) Modular: Setiap komponen dapat dijalankan dan diuji secara terpisah
b) Transparan: Menampilkan detail proses internal untuk verifikasi
c) User-friendly: Interface sederhana dengan validasi input otomatis
d) Portable: Format Jupyter Notebook memudahkan eksekusi dan sharing

3.1.5 Komponen Implementasi

a) Konstanta DES

Semua matriks yang diperlukan (IP, IP^{-1}, PC-1, PC-2, E, P, dan 8 S-Box) diimplementasikan sebagai array Python sesuai standar.

b) Fungsi Utilitas

Mencakup validate_binary(), string_to_bits(), permute(), left_shift(), xor(), dan fungsi formatting lainnya.

c) Pembangkitan Kunci

Fungsi generate_keys() menghasilkan 16 kunci internal melalui tahapan PC-1, left shift, dan PC-2.

d) Enkripsi

Fungsi des_encrypt_block() melakukan enkripsi per blok 64-bit dengan function_f() sebagai fungsi transformasi Feistel.

e) Interface

Program utama menyediakan menu pilihan kunci dan menampilkan hasil enkripsi dalam berbagai format.

Program ini dirancang khusus untuk keperluan edukasi dan demonstrasi dengan transparansi proses yang tinggi, sehingga setiap langkah algoritma DES dapat diamati dan diverifikasi secara *real-time*.

3.2 Analisis dan Pengujian Hasil

Implementasi algoritma DES ini telah diuji dengan melakukan enkripsi pada sebuah plaintext (teks asli) dan kunci yang telah ditentukan. Dalam pengujian ini, plaintext yang digunakan adalah 'KOMPUTER', yang memiliki panjang 8 karakter atau 64 bit. Sebagai kunci, dipilih kunci default biner 64-bit yaitu 10100100 10000010 10011100 10001110 10001110

10000010 10001110 10011100. Proses enkripsi berjalan melalui serangkaian tahapan yang transparan, memungkinkan setiap langkah untuk diamati dan diverifikasi.

Pada tahap pembangkitan kunci internal, kunci eksternal 64-bit di-permutasi menggunakan PC-1 menjadi 56-bit, kemudian dibagi menjadi dua bagian 28-bit (C0 dan D0). Selanjutnya, 16 kunci internal (K1 hingga K16) dihasilkan melalui proses left shift dan permutasi PC-2. Setiap kunci internal yang dihasilkan memiliki panjang 48-bit dan unik untuk setiap putaran. Misalnya, K1 adalah 00011111 01001001 00010001 10111011 01011110 11000111, dan K16 adalah 00011110 01000001 01010001 10110011 01111100 00111001.

Setelah kunci internal siap, proses enkripsi blok dimulai. Plainteks 'KOMPUTER' dikonversi ke dalam bentuk biner 64-bit: 01001011 01001111 01001101 01010000 01010101 01010100 01000101 01010010. Blok ini kemudian menjalani Initial Permutation (IP), menghasilkan 11111111 10111000 01110110 01010111 00000000 00000000 00000111 10000011. Hasil IP ini dibagi menjadi L0 dan R0, masing-masing 32-bit.

Selama 16 putaran Feistel, blok kiri (L) dan kanan (R) mengalami transformasi. Setiap putaran melibatkan operasi ekspansi R menjadi 48-bit, XOR dengan kunci internal yang sesuai (Ki), substitusi melalui S-Box, dan permutasi P. Sebagai contoh, pada putaran pertama, f(R0, K1) menghasilkan 00010011 00001001 00111010 00011001. R1 kemudian dihitung sebagai L0 XOR f(R0, K1), dan L1 menjadi R0. Proses ini diulang hingga putaran ke-16, di mana L16 dan R16 dihasilkan.

Setelah 16 putaran, R16 dan L16 digabungkan (dengan urutan R16L16) dan kemudian di-permutasi balik menggunakan IP^-1. Hasil akhir dari proses enkripsi untuk plaintext 'KOMPUTER' dan kunci yang diberikan adalah ciphertext dalam format biner: 00001111 01101100 00101000 10001110 01000110 10010000 00101001 01001000. Dalam format heksadesimal, ciphertext ini adalah 0F6C288E46902948. Validasi input kunci dan penanganan kesalahan juga telah diimplementasikan untuk memastikan bahwa kunci yang dimasukkan pengguna sesuai dengan standar 64-bit biner yang disyaratkan. Proses yang transparan dan output yang detail ini menunjukkan bahwa implementasi algoritma DES telah berhasil mengenkripsi data sesuai dengan spesifikasi.

BAB IV PENUTUP

4.1 Kesimpulan

Implementasi algoritma *Data Encryption Standard* (DES) yang dilakukan pada praktik ini, kami menyimpulkan bahwa algoritma DES merupakan algoritma enkripsi simetris yang berperanan penting dalam sejarah kriptografi modern. Walau saat ini dianggap kurang aman untuk aplikasi yang membutuhkan perlindungan tingkat tinggi, terutama jika algoritma ini mengalami serangan brute force, karena itu AES akan menjadi solusi untuk menggantikan DES.

Dengan menggunakan bahasa pemrograman Python, kami hanya mencoba proses enkripsi, dengan fokus pada pembangkitan kunci dan transformasi bit menggunakan jaringan Feistel. Walaupun proses dekripsi belum diimplementasikan, kode yang dibuat sudah dapat memvalidasi hasil enkripsi dengan menggunakan kunci yang diberikan, serta menampilkan output dalam format biner dan *hexadecimal*.

DAFTAR PUSTAKA

[1] SEVIMA, "Dr. Dandi Darmadi: Strategi Jitu Keamanan Data Digital Kampus," *kumparan*, May 20, 2025. https://kumparan.com/komunitas-sevima/dr-dandi-darmadi-strategi-jitu-keamanan-data-digital-kampus-256bsf0TBpl.

[2] Fahri Husaini, Akim M.H Pardede, and Imeldawaty Gultom, "Penerapan Enkripsi Menggunakan Metode Elgamal guna Meningkatkan Keamanan Data," *JUKI : Jurnal Komputer dan Informatika*, vol. 4, no. 1, pp. 67–73, 2022, doi: https://doi.org/10.53842/juki.v4i1.104.

[3] I. W. Wulandari and H. Hwihanus, "PERAN SISTEM INFORMASI AKUNTANSI DALAM PENGAPLIKASIAN ENKRIPSI TERHADAP PENINGKATAN KEAMANAN PERUSAHAAN," *Jurnal Kajian dan Penalaran Ilmu Manajemen*, vol. 1, no. 1, pp. 11–25, Jan. 2023, doi: https://doi.org/10.59031/jkpim.v1i1.46.

[4] J. Siahaan, "Data Encryption Standard (DES)," *Encyclopedia of Cryptography and Security*, 2011. https://www.academia.edu/81345451/Data_Encryption_Standard_DES_

[5] R. Sood and H. Kaur, "A Literature Review on RSA, DES and AES Encryption Algorithms," *Emerging Trends in Engineering and Management*, pp. 57–63, Feb. 2023, Available: https://www.publications.scrs.in/chapter/978-81-955020-3-5/7

[6] "Strength of Data encryption standard (DES)," *GeeksforGeeks*, Jan. 31, 2020. https://www.geeksforgeeks.org/strength-of-data-encryption-standard-des/

[7] M. Sharma and R. B. Garg, "DES: The oldest symmetric block key encryption algorithm," Jan. 2016, doi: https://doi.org/10.1109/sysmart.2016.7894489.

[8] S. Kumar, M. S. Gaur, P. Sagar Sharma, and D. Munjal, "A Novel Approach of Symmetric Key Cryptography," *IEEE Xplore*, Apr. 01, 2021. https://ieeexplore.ieee.org/abstract/document/9445343/

[9] S. S. Ghosh, H. Parmar, P. Shah, and K. Samdani, “A Comprehensive Analysis Between Popular Symmetric Encryption Algorithms,” *2018 IEEE Punecon*, Nov. 2018, doi: https://doi.org/10.1109/punecon.2018.8745324.

[10] N. M. M. Alhag and Y. A. Mohamed, “An Enhancement of Data Encryption Standards Algorithm (DES),” *2018 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE)*, 2018, doi: https://doi.org/10.1109/ICCCEEE.2018.8515843.

LAMPIRAN

Link Repository: https://github.com/akiraraihaan/kuis-3-kriptografi