Studi Performa Algoritma AES dan SPECK dalam Pengelolaan Kecepatan Enkripsi, Dekripsi, dan Alokasi Memori

Ni Putu Aishwara Kusumaningtyas Sudana Putri[1] , Rio Nelson Caesar Simorangkir[2]

Rekayasa Kriptografi[1,2]
Politeknik Siber dan Sandi Negara[1,2]
https://poltekssn.ac.id/[1,2]
niputu.aishwara@student.poltekssn.ac.id[1] ; rio.nelson@student.poltekssn.ac.id[2]

***Abstrak*—** Keamanan data sangat penting untuk mencegah akses tidak sah yang dapat menyebabkan kerugian finansial dan penyalahgunaan informasi. Data yang tidak terproteksi dapat diakses oleh orang yang tidak berwenang dengan cara yang merugikan, dapat menyebabkan pembocoran rahasia komersial yang dapat menyebabkan kerugian finansial, atau memungkinkannya untuk menggunakan informasi tersebut terhadap subjek yang sama. Oleh karena itu, teknik kriptografi yang dirancang untuk mengamankan kerahasiaan informasi selama penyimpanan dan transmisi sangat penting. Algoritma AES dan Speck merupakan teknik kriptografi simetris yang banyak digunakan, meskipun keduanya bertujuan melindungi data dari akses yang tidak sah, masing-masing memiliki karakteristik yang berbeda. Penelitian ini bertujuan untuk membandingkan kecepatan waktu eksekusi enkripsi dan dekripsi serta penggunaan memori Algoritma AES dan Speck dalam pengembangan perangkat lunak berbasis C. Penelitian ini dilakukan dua macam pengujian, yaitu pengujian kecepatan waktu eksekusi dan pengujian alokasi memori. Hasil penelitian dapat disimpulkan bahwa Algoritma Speck lebih cepat dalam proses enkripsi dibandingkan Algoritma AES. Secara keseluruhan, ukuran file dapat mempengaruhi kecepatan proses enkripsi dan dekripsi. Semakin besar file input maka semakin cepat proses enkripsi dan dekripsi yang dilakukan. Algoritma Speck memiliki keunggulan dalam hal kecepatan enkripsi dan dekripsi, khususnya pada file berukuran besar dibandingkan Algoritma AES. Pada file besar, Algoritma Speck dan AES memiliki efisiensi memori yang seimbang dalam hal alokasi penyimpanan hasil enkripsi. Kesimpulannya, Algoritma Speck unggul dalam kecepatan, sedangkan Algoritma AES lebih kompleks namun tetap efisien dalam alokasi memori.

Kata kunci : Kriptografi, Algoritma AES, Algoritma Speck, Kecepatan Enkripsi, Kecepatan Dekripsi, Alokasi Memori, Efisiensi Penyimpanan.

***Abstract*—** *Data security is crucial to prevent unauthorized access that may lead to financial losses and misuse of information. Unprotected data can be accessed by unauthorized individuals in a harmful manner, potentially causing commercial secret leaks, financial losses, or enabling its misuse against the same subject. Therefore, cryptographic techniques designed to secure information confidentiality during storage and transmission are essential. AES and Speck algorithms are widely used symmetric cryptographic techniques. Although both aim to protect data from unauthorized access, each has distinct characteristics. This study aims to compare the execution speed of encryption and decryption, as well as memory usage, of the AES and Speck algorithms in software development using the C programming language. Two types of testing were conducted: execution time measurement and memory allocation analysis. The results indicate that the Speck algorithm is faster in encryption processing compared to the AES algorithm. Overall, file size affects encryption and decryption speed, where larger input files result in faster encryption and decryption processes. The Speck algorithm demonstrates superiority in encryption and decryption speed, particularly for large-sized files, compared to AES. For large files, both Speck and AES exhibit balanced memory efficiency in terms of encrypted data storage. In conclusion, the Speck algorithm excels in speed, while the AES algorithm is more complex but remains efficient in memory allocation.*

Keywords: Cryptography, AES Algorithm, Speck Algorithm, Encryption Speed, Decryption Speed, Memory Allocation, Storage Efficiency

PENDAHULUAN

Salah satu prioritas utama di hampir semua sektor pada era digital saat ini adalah keamanan informasi. Data yang tidak terproteksi dapat diakses oleh orang yang tidak berwenang dengan cara yang merugikan, dapat menyebabkan pembocoran rahasia komersial yang dapat menyebabkan kerugian finansial, atau memungkinkannya untuk menggunakan informasi tersebut terhadap subjek yang sama. Oleh karena itu, teknik kriptografi yang dirancang untuk mengamankan kerahasiaan informasi selama penyimpanan dan transmisi sangat penting. Salah satu teknik kriptografi ini adalah kriptografi simetris. Secara khusus, kriptografi simetris mengacu pada skema enkripsi dan dekripsi yang memanfaatkan satu set kunci yang sama atau setara. Masuk akal: orang yang ingin mengakses lagi dokumen tersebut hanya akan dapat mencari kunci-muncul untuk melakukannya. Ada banyak teknik kriptografi simetris, tetapi yang paling banyak adalah Speck dan Advanced Encryption Standard. NSA memperkenalkan Speck pada tahun 2013 sebagai proses kriptografi yang efisien di semua platform terutama perangkat sumber daya terbatas, contoh untuk perangkat IOT. Setidaknya salah satu aspek dari Speck adalah kemampuan untuk mengenkripsi dokumen dengan kecepatan tinggi, disertai dengan sedikit penggunaan memori. Di sisi lain, AES menjadi standar nasional Amerika Serikat pada tahun 2001 dan memiliki tingkat kerumitan yang paling tinggi di semua ukuran data. Meskipun keduanya bertujuan melindungi data dari akses yang tidak sah, Speck dan AES memiliki karakteristik yang berbeda. AES berdasarkan struktur blok, yang berarti bahwa itu terbaik untuk pengaturan yang menggunakan pekerjaan yang sama ukuran yang sudah diterapkan, sementara Speck bit processing melalui. Ini adalah solusi efisien yang terbaik untuk semua operasi berkecepatan tinggi, sedangkan AES terbaik untuk keamanan optimal. Keduanya dapat memakan waktu, tergantung pada kasus pengguna yang akan melakukan enkripsi atau dekripsi. Oleh karena itu studi ini penting untuk melihat lebih mendalam speck dan AES. Tujuan penelitian ini adalah untuk menguji kecepatan enkripsi dan dekripsi serta alokasi memori antara Speck dan AES. Memberikan informasi yang lebih besar kepada masyarakat adalah penting. Ini sebabnya, Studi ini dilakukan tentu saja menguji kecepatan kedua algoritma. Studi ini diharapkan untuk memberikan pemahaman tentang seluk beluk kedua algoritma ini.

Oleh karena itu, penelitian ini bertujuan untuk membandingkan kecepatan enkripsi serta penggunaan memori algoritma AES dan Speck pada pengembangan perangkat lunak yang dibangun dengan bahasa pemrograman C. Penelitian ini bertujuan untuk memberikan pemahaman yang lebih baik tentang kinerja algoritma yang beredar pada perangkat yang memiliki sumber daya terbatas. Selain itu, penelitian ini juga memusatkan perhatian terhadap pengaruh ukuran blok dan panjang kunci terhadap kecepatan enkripsi algoritma dan alokasi memori yang digunakan. Secara lebih spesifik, pengaruh penambahan panjang kunci pada AES atau perubahan ukuran blok pada algoritma Speck akan mempengaruhi kinerja dan keamanan algoritma, yang merupakan aspek penting untuk dinilai dalam aplikasi dunia nyata. Pentingnya studi ini sama sekali tidak bisa diabaikan, mengingat tuntutan akan perangkat yang dapat mengelola data dengan cara yang efisien dalam, terutama sistem perangkat lunak. Sistem ini sering dihadapkan pada sumber daya terbatas, termasuk memori, daya pemrosesan, dan sumber daya listrik. Dengan memilih algoritma yang tepat, kita dapat mempengaruhi kinerja keseluruhan sistem dalam, termasuk kecepatan enkripsi dan keamanan data yang dienkripsi. Karena AES dan Speck beroperasi pada spektrum yang sangat berbeda dalam hal desain dan efisiensi, penelitian ini akan memberikan pemahaman yang jauh lebih baik tentang kapan menggunakan algoritma mana. Studi ini dapat memberikan wawasan yang lebih baik tentang kapan membuat trade off antara kecepatan dan keamanan data dan kapan menjangkau salah satu. Kategori ini mungkin akan bergantung pada ketersediaan sumber daya sistem, jenis aplikasi, dan prioritas keamanan.

KERANGKA TEORI

Istilah "lightweight" sering digunakan untuk menggambarkan algoritma yang dirancang agar dapat beroperasi secara efisien pada platform dengan keterbatasan sumber daya. Namun, karakteristik yang membuat suatu algoritma unggul pada mikrokontroler 8-bit tidak selalu menjamin bahwa algoritma tersebut dapat diimplementasikan secara efisien dalam sirkuit berukuran kecil. Oleh karena itu, diperlukan pendekatan yang lebih umum dan tidak bergantung pada platform tertentu dalam mendefinisikan konsep "lightweight", sehingga dapat digunakan sebagai acuan dalam perancangan algoritma yang optimal untuk berbagai lingkungan komputasi. Berawal dari sebuah perancangan hingga pada tahap implementasi sistem dan didapatkan hasil pengujian. Hasil pengujian yang dihasilkan kemudian dianalisis untuk menilai kinerja algoritma Speck dan algoritma AES dalam proses enkripsi dan dekripsi.

A. Algoritma Speck

Algoritma Speck adalah salah satu keluarga cipher blok ringan yang dikembangkan oleh National Security Agency (NSA) Amerika Serikat. Cipher ini dirancang untuk memberikan keamanan yang efisien pada perangkat dengan sumber daya terbatas seperti mikrokontroler dan perangkat IoT (Internet of Things) (Beaulieu et al., 2013). Speck menggunakan struktur berbasis Feistel yang dimodifikasi dengan operasi sederhana seperti penjumlahan modular, rotasi, dan XOR (ARX). Algoritma ini memiliki beberapa varian dengan ukuran blok dan kunci yang berbeda, misalnya SPECK96/144 yang memiliki ukuran blok 96 bit dan ukuran kunci 144 bit (Beaulieu et al., 2015). Implementasi algoritma Speck dilakukan dengan memasukkan kode yang sudah dibuat ke dalam aplikasi DEV-C++ dan dijalankan melalui *Command Prompt*. Pada perancangan algoritma Speck menjelaskan proses realisasi untuk sistem enkripsi dan dekripsi pada algoritma Speck berdasarkan perancangan kode yang telah disusun. Pelaksanaan implementasi algoritma ini merupakan proses pengkodean yang dilakukan dengan menggunakan bahasa pemrograman C kemudian program yang terbentuk dan disimpan dalam aplikasi DEV-C++ untuk menjalankan kode melalui *Command Prompt.*

B. Algoritma AES

Algoritma Advanced Encryption Standard (AES) merupakan standar enkripsi yang dikembangkan untuk menggantikan Data Encryption Standard (DES). AES dikembangkan oleh Joan Daemen dan Vincent Rijmen dengan menggunakan prinsip Substitution-Permutation Network (SPN), yang berbeda dengan struktur Feistel yang digunakan pada DES (Daemen & Rijmen, 2002).AES telah diuji ketahanannya terhadap berbagai jenis serangan kriptografi, termasuk serangan diferensial, linear, dan serangan brute-force. Hingga saat ini, tidak ada metode praktis yang mampu menembus AES dalam jumlah putaran penuh, menjadikannya salah satu algoritma enkripsi yang paling aman (Liu et al., 2017). Impelementasi algoritma AES dilakukan dengan mengunduh kode yang telah dibentuk ke dalam komputer dan dimasukkan dalam aplikasi DEV-C++ dan dijalankan melalui *Command Prompt*. Pada algoritma AES menjelaskan proses realisasi untuk sistem enkripsi dan dekripsi dengan membandingkan kecepatan proses dengan algoritma Speck yang telah dirancang sebelumnya. Implementasi algoritma ini merupakan proses pengkodean yang dilakukan dengan mengguanakan bahasa pemrograman C yang disimpan dalam aplikasi DEV-C++ untuk menjalankan kode melalui *Command Prompt.*

C. Block Cipher

Kriptanalisis merupakan cabang dalam kriptologi yang berfokus pada analisis dan upaya untuk memecahkan algoritma kriptografi. Meskipun secara umum dikaitkan dengan upaya membobol sistem keamanan, kriptanalisis justru memainkan peran penting dalam pengembangan cipher yang lebih kuat. Seorang kriptografer harus memahami berbagai teknik kriptanalisis guna mengidentifikasi potensi kelemahan dalam suatu algoritma, sehingga dapat merancang sistem yang lebih tahan terhadap serangan (Stallings, 2020). Proses pengujian ketahanan cipher terhadap berbagai metode serangan, seperti kriptanalisis diferensial, linear, maupun serangan brute-force, merupakan bagian penting dalam standar evaluasi keamanan algoritma (Schneier, 1996). Cipher blok merupakan salah satu metode enkripsi simetris yang mengolah data dalam blok-blok berukuran tetap. Berbeda dengan cipher stream, yang mengenkripsi bit atau byte secara individual, cipher blok mengenkripsi seluruh blok data sekaligus (Menezes et al., 1996). Cipher blok menggunakan fungsi putaran (round function), yaitu serangkaian operasi sederhana yang dieksekusi berulang kali dalam sejumlah putaran tertentu, yang disebut rounds. Proses enkripsi dalam cipher blok dimulai dengan input, yaitu blok plaintext berukuran tetap (nnn bit) yang dimasukkan pada putaran pertama. Selanjutnya, setiap putaran menggunakan fungsi putaran yang bergantung pada subkunci (round key), yang diperoleh dari kunci rahasia utama (kkk bit).

D. Mode Counter (CTR) dalam Block Cipher

Mode Counter (CTR) adalah salah satu mode operasi yang paling efisien dalam block cipher. Dalam mode ini, nilai counter yang berbeda dihasilkan untuk setiap blok pesan yang akan dienkripsi. Block cipher digunakan untuk mengenkripsi counter tersebut, dan hasil enkripsi ini kemudian dipadukan dengan plaintext menggunakan operasi XOR untuk menghasilkan ciphertext. Keunggulan utama dari CTR adalah kemampuannya untuk melakukan enkripsi secara paralel, yang meningkatkan kecepatan dan efisiensi (Rogaway, 2004). Mode Counter (CTR) dikenal dengan tingkat keamanan dan efisiensinya yang tinggi, karena tidak terpengaruh oleh pola dalam plaintext dan mendukung operasi enkripsi dan dekripsi secara paralel. Namun, penting untuk memastikan bahwa nilai counter yang digunakan tidak pernah tumpang tindih dalam aplikasi yang sama, karena ini bisa menurunkan tingkat keamanan" (Katz & Lindell, 2007).

METODOLOGI

Metode Penelitian

Penelitian ini menggunakan metode eksperimen dengan pendekatan benchmarking untuk mengukur performa algoritma kriptografi AES dan Speck. Tujuan dari metode ini adalah untuk membandingkan kecepatan enkripsi dan dekripsi kedua algoritma berdasarkan waktu eksekusi. Selain itu, penelitian ini juga menggunakan metode pengembangan perangkat lunak untuk mendukung proses eksperimen. Pada tahap ini, dilakukan perancangan perangkat lunak yang diimplementasikan menggunakan bahasa pemrograman C. Proses ini melibatkan tahap implementasi untuk membangun program enkripsi dan dekripsi, serta unit testing guna memastikan setiap komponen program berfungsi dengan baik dan sesuai dengan spesifikasi yang telah direncanakan.

Alat penunjang yang digunakan dalam peniltian ini melibatkan perangkat lunak dan perangkat keras. Adapun spesfikasi perangkat keras yang digunakan penilitian sebagai berikut:

a. Operating system: Windows 11 Home Single Language
b. Processor: 11th Gen Intel(R) Core(TM) i7-1165G7

Adapun spesifikasi perangkat lunak yang digunakan penelitian sebagai berikut:

a. Dev C++ Version 5.11: Digunakan untuk menulis, mengedit, dan mengompilasi kode sumber dalam bahasa pemrograman C.
b. Command Prompt (cmd.exe): Digunakan untuk menjalankan program hasil kompilasi, mengelola file, serta melakukan pengujian performa secara manual.
c. PowerShell: Dimanfaatkan untuk melakukan *benchmarking* waktu eksekusi enkripsi dan dekripsi menggunakan perintah seperti Measure-Command, sehingga mempermudah analisis performa algoritma.
d. Notepad: Digunakan sebagai editor teks sederhana untuk membuat dan mengedit kode sumber program sebelum dikompilasi.
e. GCC (GNU Compiler Collection): Digunakan sebagai kompiler untuk menerjemahkan kode sumber C menjadi file eksekusi (.exe) yang dapat dijalankan di Command Prompt atau PowerShell.

Bahan Sampling

Bahan sampling yang dibutuhkan untuk penelitian ini adalah teks yang akan diterjemahkan dan kunci yang digunakan untuk enkripsi dan dekripsi. Kunci yang digunakan dalam algoritma Speck sepanjang 64 bit dan algoritma AES sepanjang 128 bit. Kunci untuk algoritma Speck dan AES disimpan dalam file key.txt. Teks terang

akan dimasukkan secara langsung ketika program algoritma AES dan Speck dijalankan

Paramater Kerja

a. Penyusunan program

Program akan disusun dalam aplikasi DEV C++ berisikan kode yang menggunakan implementasi algoritma Speck dan AES. Pada tahap pengembangan desain sistem, langkah pertama yang dilakukan adalah menyusun *source code* untuk proses enkripsi dan dekripsi. *Source code* yang digunakan disusun berdasarkan prinsip kerja yang sesuai dengan mekanisme *round function* dan *key expansion* yang terdapat pada algoritma yang dipelajari. *Round function* merupakan serangkaian operasi yang dilakukan pada setiap putaran (*round*) algoritma untuk mengacak dan mengubah data, seperti penggantian byte (substitution), pergeseran baris data (*shift rows*), dan operasi lainnya yang memperkuat keamanan algoritma. Sementara itu, *key expansion* adalah proses yang menghasilkan serangkaian kunci putaran dari kunci utama yang diberikan, memastikan setiap putaran menggunakan kunci yang berbeda untuk meningkatkan tingkat keamanan.

b. *Running Time*

Running time mengacu pada waktu yang diperlukan oleh sebuah algoritma untuk menyelesaikan proses enkripsi atau dekripsi. Dalam konteks algoritma AES dan Speck, *running time* mengukur efisiensi algoritma dalam menyelesaikan operasi pada data yang diberikan. Waktu ini dihitung dari saat algoritma dimulai hingga selesai menjalankan seluruh proses, dan biasanya diukur dalam satuan detik (s) atau milidetik (ms). Sistem pemrograman yang telah disusun akan disimpan dalam aplikasi DEV C++ dan menyimpannya dalam satu folder agar dapat dijalankan dengan menggunakan GCC. Program akan dijalankan dengan GCC dengan memasukkan file yang berisikan teks terang ketika program dijalankan. Kemudian, dilanjutkan dengan memasukkan kunci yang digunakan untuk enkripsi dan dekripsi. Program akan melakukan proses enkripsi dan dekripsi dengan menghasilkan hasil enkripsi dan dekripsi dalam bentuk teks yang memiliki bentuk sama seperti teks terang. Proses ini akan dilakukan dengan alur yang sama antara algoritma AES dan Speck.

c. Kecepatan

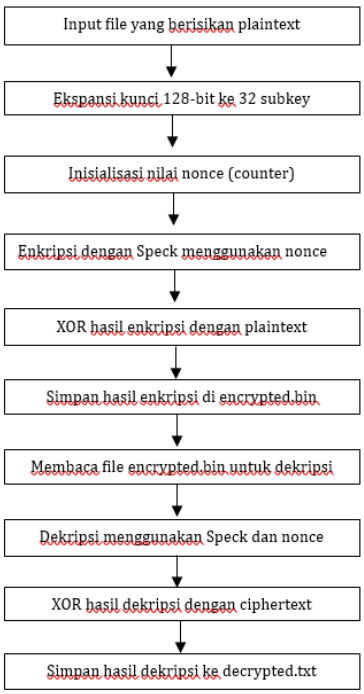
Dalam mengevaluasi kecepatan enkripsi dan dekripsi dari algoritma AES dan Speck pada platform berbasis PowerShell, pengujian dilakukan dengan menghitung waktu eksekusi dari masing-masing algoritma menggunakan instruksi waktu yang tersedia dalam PowerShell. Waktu eksekusi untuk setiap operasi enkripsi dan dekripsi diukur dengan menggunakan cmdlet Measure-Command yang memberikan waktu total yang dibutuhkan untuk mengeksekusi sebuah blok perintah. Pengujian dilakukan dengan menggunakan data acak sepanjang 128-bit untuk AES dan 64-bit untuk Speck.

Desain Program Algoritma Speck

Program ini mengimplementasikan algoritma enkripsi dan dekripsi Speck 64-bit menggunakan kunci 128-bit. Algoritma Speck adalah algoritma kriptografi simetris yang memiliki struktur yang lebih sederhana dibandingkan dengan algoritma block cipher lainnya, dan dirancang untuk perangkat dengan keterbatasan sumber daya. Desain program ini mengimplementasikan algoritma enkripsi dan dekripsi Speck 64-bit dengan kunci 128-bit menggunakan mode CTR (Counter Mode). Algoritma Speck yang dirancang untuk perangkat dengan sumber daya terbatas, memiliki struktur sederhana dan efisien.

Proses dimulai dengan membaca file input dalam bentuk biner dan memperluas kunci 128-bit menjadi 32 subkey menggunakan ekspansi kunci Speck. Enkripsi dilakukan dengan mengenkripsi nonce, yang kemudian di-XOR dengan plaintext untuk menghasilkan ciphertext. Setiap blok terenkripsi secara unik dengan menambah nilai nonce, mencegah pola berulang dalam ciphertext. Dekripsi dilakukan dengan cara yang sama, mengembalikan ciphertext ke plaintext.

Ketika program dijalankan, fungsi crypto_stream_ speck128128ctr_ref_xor(out, in, inlen, n, k) dipanggil untuk mengenkripsi plaintext yang diberikan sebagai input. Pertama, program akan membaca dan menganalisis file input (in), menentukan panjang plaintext (inlen), serta menggunakan nonce (n) agar karakter tetap unik dan kunci (k) untuk proses enkripsi. Jika panjang plaintext (inlen) lebih dari 16 byte, maka enkripsi dilakukan dalam blok-blok 16 byte menggunakan mode counter (CTR) dengan perulangan *while*. Setiap proses perulangan, nonce akan dienkripsi menggunakan algoritma Speck untuk menghasilkan keystream, yang kemudian diproses dengn operasi XOR dengan plaintext untuk menghasilkan ciphertext. Setelah blok 16 byte dienkripsi, nonce akan diperbarui agar keystream tetap unik di setiap blok. Jika setelah perulangan utama masih terdapat sisa plaintext kurang dari 16 byte, maka sisa tersebut tetap dienkripsi menggunakan keystream baru yang dihasilkan dari nonce yang telah diperbarui. Hasil enkripsi disimpan dalam file encrypted.bin, dan hasil dekripsi dalam decrypted.txt, memastikan data dapat dipulihkan dengan benar.

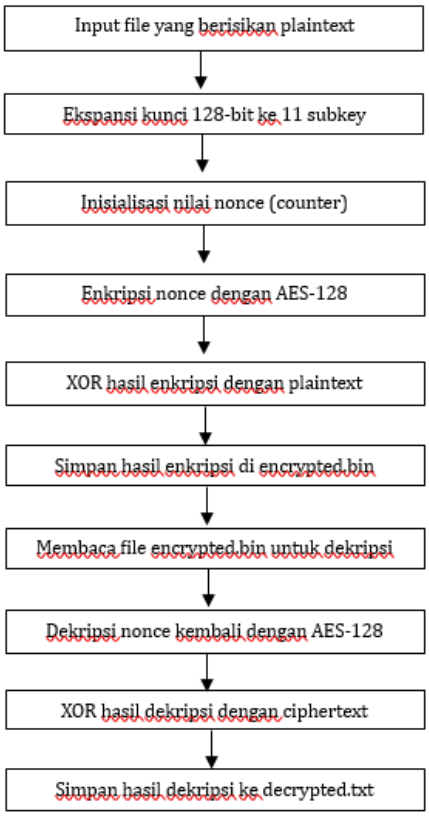


Gambar 1. Diagram Desain Algroitma Speck

Desain Program Algoritma AES

Program ini mengimplementasikan algoritma enkripsi dan dekripsi AES-128 menggunakan panjang kunci 128-bit dengan mode CTR (Counter Mode). AES-128 merupakan algoritma kriptografi simetris yang memiliki keamanan tinggi dan efisiensi dalam pemrosesan data. Implementasi ini diawali dengan inisialisasi parameter utama, termasuk panjang kunci, jumlah putaran, dan struktur matriks state 4×4. Proses ekspansi kunci dilakukan untuk menghasilkan subkunci yang digunakan pada setiap putaran enkripsi dan dekripsi. Enkripsi dilakukan dengan mengenkripsi nonce yang kemudian di-XOR dengan plaintext untuk menghasilkan ciphertext. Setiap blok terenkripsi secara unik dengan menambah nilai nonce, sehingga pola berulang dalam ciphertext dapat dicegah. Dekripsi dilakukan dengan cara yang sama, mengembalikan ciphertext ke plaintext.

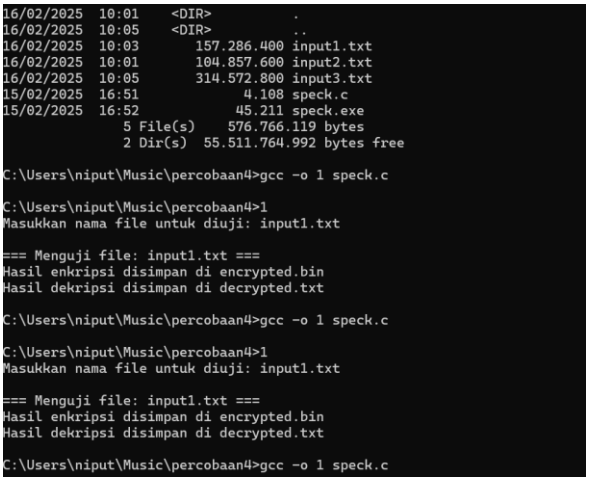
Implementasi ini menggunakan dua modul utama: aes.c yang berisi fungsi enkripsi dan dekripsi, serta test.c yang menangani pembacaan file input, pemrosesan enkripsi, dan penyimpanan hasil. Hasil enkripsi disimpan dalam file encrypted.bin, sedangkan hasil dekripsi disimpan dalam decrypted.txt, memastikan data dapat dipulihkan dengan benar. Mode CTR dipilih karena kemampuannya dalam menangani blok data yang lebih fleksibel dan meningkatkan keamanan terhadap serangan pola.



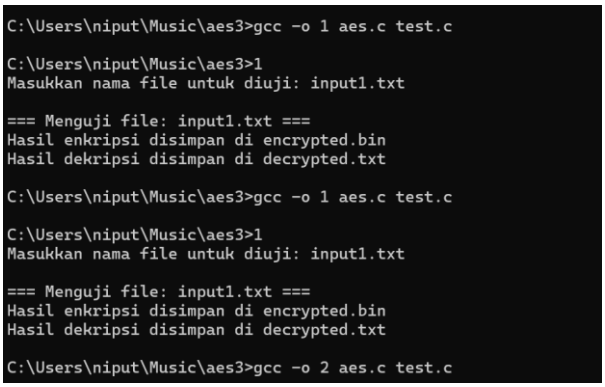
Gambar 2. Diagram Desain Algroitma AES

HASIL PEMBAHASAN

Sistem yang dirancang untuk menunjang penelitian dalam enkripsi dan dekripsi adalah algoritma AES dan Algoritma Speck. Program untuk melakukan enkripsi menggunakan bahasa pemrograman C dimana untuk mengerjakannya menggunakan aplikasi DEV C++. Pengujian ini dilakukan dengan menggunakan sistem operasi Windows 10. Dalam aplikasi ini juga dapat terlihat pengamatan waktu eksekusi proses enkripsi dan dekripsi serta alokasi memori dari masing-masing file yang diuji dengan algoritma AES dan algoritma Speck. Pada penelitian ini dilakukan dua macam pengujian, yaitu pengujian kecepatan waktu eksekusi dan pengujian alokasi memori. Hasil pengujian program yang dibangun bisa berjalan sesuai perencanaan. Proses running program menggunakan aplikasi Dev C++ dan *Command Prompt* seperti ditunjukkan pada Gambar 3 dan Gambar 4 dibawah.



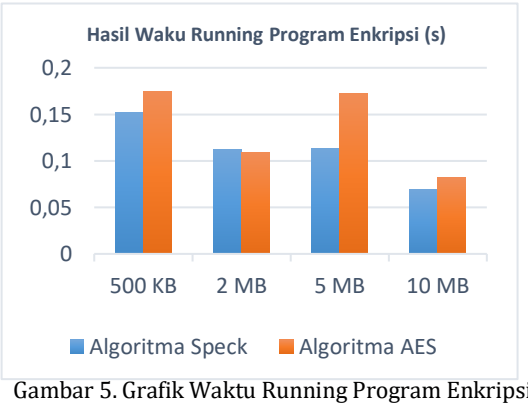
Gambar 3. Running Program Algoritma Speck



Gambar 4. Running Program Algoritma AES

A. Hasil Running Program

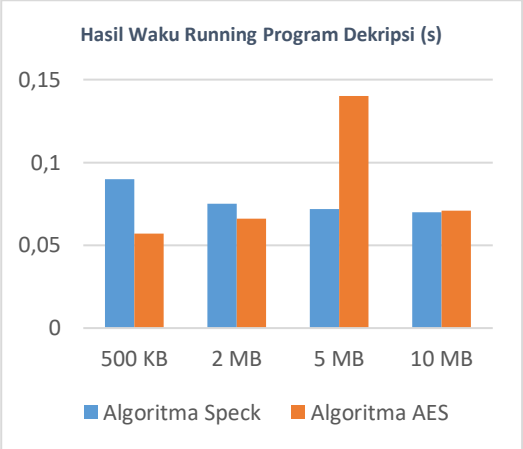
1) *Running* Program Enkripsi



Gambar 5. Grafik Waktu Running Program Enkripsi

Gambar 5 menunjukkan perbandingan waktu proses saat menjalankan program enkripsi pada Algoritma AES dan Algoritma Speck. Dalam gambar menunjukkan waktu enkripsi file input.txt yang memiliki ukuran terbesar menjadi program enkripsi yang dijalankan tercepat. Dalam grafik juga menunjukkan bahwa waktu yang dibutuhkan saat enkripsi pada Algoritma Speck lebih cepat dibandingkan Algoritma AES. Terlihat ketika file dengan ukuran 5 MB dijalankan Algoritma Speck membutuhkan waktu yang lebih singkat dibandingkan Algoritma AES.

2) *Running* Program Dekripsi



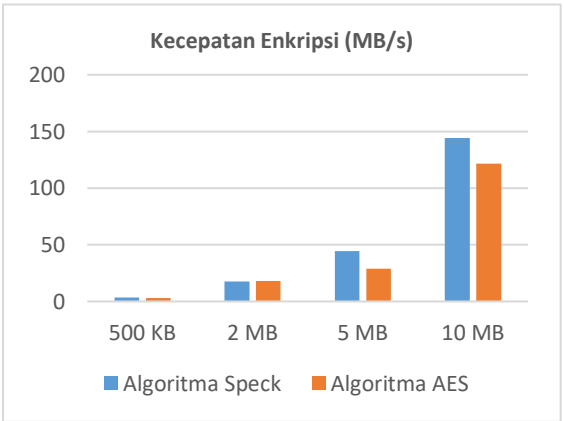
Gambar 6. Grafik Waktu Running Program Dekripsi

Gambar 6 menunjukkan perbandingan waktu proses saat menjalankan program dekripsi pada Algoritma AES dan Algoritma Speck. Dalam gambar menunjukkan waktu tercepat ketika program dekripsi dijalankan terdapat pada Algoritma AES dengan ukuran 500 KB dan 2 MB. Dalam situasi lain, terlihat bahwa Algoritma AES memiliki waktu yang lebih cepat dalam hal dekripsi dibandingkan Algoritma Speck. Berdasarkan rata-rata waktu yang dibutuhkan dalam program dekripsi, waktu tercepat terletak pada Algoritma Speck dengan rata-rata waktu 0,077.

B. Pengujian Kecepatan Waktu Eksekusi Program

Tujuan pengujian kecepatan waktu eksekusi adalah untuk melihat perbandingan kecepatan Algoritma Speck dan Algoritma AES.

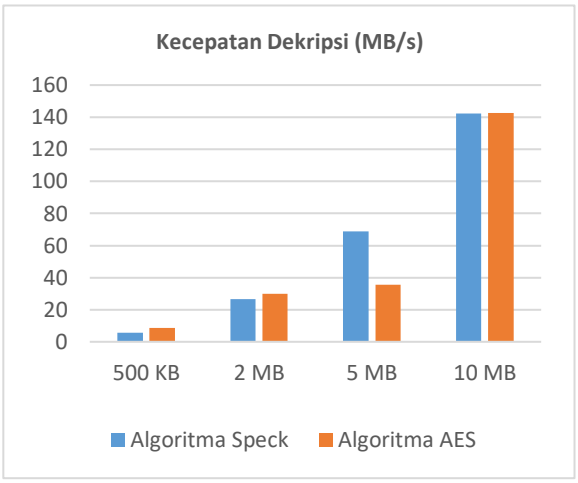
1) Kecepatan Enkripsi



Gambar 7. Grafik Waktu Running Program Dekripsi

Berdasarkan Gambar 7 menunjukkan bahwa kecepatan Algoritma Speck dalam menjalan program Enkripsi lebih cepat dibandingkan Algoritma AES. Dalam gambar 7 juga menunjukkan semakin bertambahnya ukuran file maka kecepatan dalam menjalankan program enkripsi semakin cepat. Hal ini berlaku bagi Algoritma Speck maupun Algoritma AES, apabila ukuran file semakin besar maka proses enkripsi mengalami peningkatan kecepatan.

2) Kecepatan Dekripsi



Gambar 8. Grafik Waktu Running Program Dekripsi

Berdasarkan Gambar 8 menunjukkan bahwa kecepatan Algoritma Speck dalam menjalan program dekripsi lebih cepat dibandingkan Algoritma AES. Dalam gambar tersebut juga menunjukkan semakin bertambahnya ukuran file maka kecepatan dalam menjalankan program dekripsi semakin cepat. Sama halnya dengan proses enkripsi, kecepatan proses dekripsi algoritma Speck lebih tinggi dibandingkan Algoritma AES. Dalam satu kasus, file dengan ukuran 10 MB dijalankan dengan algoritma Speck dan Algoritma AES memiliki kecepatan yang hampir sama.

C. Perbandingan Waktu dan Kecepatan

Gambar berikut menunjukkan perbandingan waktu dan kecepatan yang dibutuhkan dalam program enkripsi dan dekripsi yang dilakukan dengan Algoritma AES dan Algoritma Speck.

PROSES ENKRIPSI

Ukuran (KB)	Waktu Proses (Second)		Selisih waktu	Kecepatan (MB/s)		Selisih Kecepatan
	Speck	AES		Speck	AES	
500 KB	0,152	0,175	-0,023	3,27	2,78	0,49
2000 KB	0,112	0,109	0,003	17,63	18,24	-0,61
5000 KB	0,113	0,172	-0,059	44,61	28,99	15,62
1000 KB	0,069	0,082	-0,013	144,22	121,72	22,5
Rata-rata	0,1115	0,1345		52,4325	42,9325	

Gambar 9. Grafik Waktu Running Program Dekripsi

Berdasarkan Gambar 9 dapat ditunjukkan bahwa hasil waktu eksekusi program enkripsi tercepat dari kedua algoritma tersebut adalah Algoritma Speck dengan rata-

rata waktu 0,1115 detik dengan kecepatan rata-rata enkripsi 52,4325 MB/s. Namun, terdapat beberapa kasus dalam file ukuran 2000 KB waktu proses lebih cepat dan memiliki kecepatan enkripsi yang tercepat adalah Algoritma AES.

PROSES DEKRIPSI

Ukuran (KB)	Waktu Proses (Second)		Selisih waktu	Kecepatan (MB/s)		Selisih Kecepatan
	Speck	AES		Speck	AES	
500 KB	0,09	0,057	0,033	5,52	8,51	-2,99
2000 KB	0,075	0,066	0,009	26,55	29,93	-3,38
5000 KB	0,072	0,14	-0,068	68,94	35,55	33,39
1000 KB	0,071	0,07	0,001	142,33	142,72	-0,39
Rata-rata	0,077	0,08325		60,835	54,1775	

Gambar 10. Grafik Waktu Running Program Dekripsi

Berdasarkan Gambar 10 dapat ditunjukkan bahwa hasil waktu eksekusi program dekripsi tercepat dari kedua algoritma tersebut adalah Algoritma Speck dengan rata-rata waktu 0,077 detik. Berdasarkan rata-rata kecepatan dekripsi Algoritma Speck lebih unggul dibandingkan Algoritma AES dengan kecepatan rata-rata 60,835 MB/s. Selain itu, terlihat dalam file berukuran 5000 KB bahwa Algoritma Speck lebih unggul dibandingkan Algoritma AES.

D. Perbandingan Alokasi Memori

Gambar berikut menunjukkan perbandingan ukuran output antara algoritma Speck dan AES setelah proses enkripsi dengan berbagai ukuran file input

Input (KB)	Output (KB)	
	Speck	AES
500 KB	512.04 KB	498.05 KB
2000 KB	2048 KB	2048 KB
5000 KB	5120 KB	5120 KB
1000 KB	10240 KB	10240 KB

Gambar 11. Grafik Waktu Running Program Dekripsi

Berdasarkan Gambar, Algoritma Speck menunjukkan peningkatan ukuran output pada ukuran file 500 KB menjadi 512,04 KB sedangkan Algoritma AES menunjukkan ukuran output yang lebih kecil pada ukuran file 500 KB, yaitu 498,05 KB. Untuk file input 2000 KB, 5000 KB, dan 10000 KB, Algoritma Speck dan Algoritma AES menghasilkan ukuran output yang sama dengan ukuran input. Dalam hal ini Algoritma Speck dapat menghasilkan output yang sedikit lebih besar pada file input dengan ukuran kecil dan stabil untuk file input berkuran besar, sedangkan Algoritma AES lebih konsisten dalam mempertahankan ukuran file output, meskipun terdapat file output yang lebih kecil dibandingkan file inputnya.

Algoritma Speck lebih hemat memori saat proses enkripsi dan dekripsi karena desainnya yang ringan, tetapi bisa menghasilkan output lebih besar pada file kecil, yang berarti membutuhkan sedikit lebih banyak ruang penyimpanan. Algoritma AES menggunakan lebih banyak memori selama proses enkripsi/dekripsi, tetapi lebih konsisten dalam menjaga ukuran output, terutama pada file kecil.

KESIMPULAN

Berdasarkan hasil pengujian, dapat disimpulkan bahwa Algoritma Speck lebih cepat dalam proses enkripsi dibandingkan Algoritma AES. Hal ini ditunjukkan pada file input dengan ukuran besar. Kecepatan enkripsi Algoritma Speck meningkat seiring dengan bertambahnya ukuran file. Dalam hal proses dekripsi, Algoritma Speck juga lebih cepat dibandingkan Algoritma AES. Algoritma Speck menunjukkan kecepatan dekripsi yang lebih tinggi pada file berukuran besar. Hal ini menunjukkan bahwa Algoritma Speck memiliki kecepatan lebih tinggi dibandingkan Algoritma AES saat proses enkripsi maupun dekripsi.

Dalam perbandingan alokasi memori, Algoritma Speck cenderung menghasilkan ukuran output yang sedikit lebih besar pada file input berukuran kecil yaitu ukuran 500 KB, namun stabil pada file dengan ukuran besar. Selain itu, Algoritma AES lebih konsisten dalam mempertahankan ukuran output. Secara keseluruhan, ukuran file dapat mempengaruhi kecepatan proses enkripsi dan dekripsi. Semakin besar file input maka semakin cepat proses enkripsi dan dekripsi yang dilakukan dan Algoritma Speck memiliki keunggulan dalam hal kecepatan enkripsi dan dekripsi, khususnya pada file berukuran besar dibandingkan Algoritma AES. Pada file besar, Algoritma Speck dan AES memiliki efisiensi memori yang seimbang dalam hal alokasi penyimpanan hasil enkripsi. Alokasi memori selama eksekusi, Algoritma Speck lebih efisien. Namun, jika yang dimaksud alokasi memori dalam penyimpanan hasil enkripsi, AES lebih unggul untuk file kecil.

REFERENSI

Abdullah, A. M. (2017). Advanced Encryption Standard (AES) Algorithm to Encrypt and Decrypt Data. *Cryptography and Network Security*.

Christophe DeCannière, A. B. (2006). An introduction to block cipher cryptanalysis. *Proceedings of the IEEE*.

Dyah Pipit Muliyah, S. S. (2020). *Journal GEEJ*.

Feri Kusuma Wardhana, A. K. (2023). Analisis Perbandingan Kinerja Enkripsi Algoritma RC4 Dan AES. *Prosiding SEMINAR NASIONAL AMIKOM SURAKARTA (SEMNASA) 2023*.

Karmila Dewi Sulistyowati, A. K. (2019). Analisis Kinerja Algoritme Speck Pada Keamanan File Teks. *Pelayanan Kesehatan*.

Katz, J. &. (2017). Introduction to Modern Cryptography. CRC Press.

Moh. Ahsani Taqwim, A. K. (2021). Implementasi Algoritme Speck Untuk Enkripsi One-Time Password Pada Two-Factor Authentication. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*.

Phillip Rogaway, D. W. (2000). 1 Review of Counter-Mode Encryption.

Ray Beaulieu, D. S.-C. (2013). The Simon and Speck Families of Lightweight Block Ciphers.

Ray Beaulieu, D. S.-C. (2015). SIMON and SPECK: Block Ciphers for the Internet of Things. *IACR Cryptol. ePrint Arch*, 585.

Rogaway, P. (2004). The CTR Mode of Encryption. Journal of Cryptographic Engineering. 185-208.

Sudarta. (n.d.). *2022*, 1-23.

Winda Ezranata Putri, F. D. (2022). Analisis Perbandingan Block Cipher Simon-Speck, Simeck, Skinny pada Komunikasi Berbasis LoRa. *Multinetics*.

Ditandatangani oleh: Nama Anda

Tanggal: 2025-07-22 15:16