Studi Algoritma AES dan SPECK: Enkripsi, Dekripsi, Efisiensi Penyimpanan, dan Uji NIST

Ahmad Muflih Izfatara[1,*)], Rahadian Ronggo Kusumo[2)], Zamir Achmad Sachio[3)], Achmad Fatih Binasrillah[4)], Ni Putu Aishwara Kusumaningtyas Sudana Putri[5)] , Rio Nelson Caesar Simorangkir[6)] , Hermawan Setiawan[7)] , Moch Radhit Julian Randito[8)]

1) Program Studi Rekayasa Kriptografi, Politeknik Siber dan Sandi Negara, ahmad.muflih@student.poltekssn.ac.id

2) Program Studi Rekayasa Keamanan Siber, Politeknik Siber dan Sandi Negara, rahadian.ronggo@student.poltekssn.ac.id

3) Program Studi Rekayasa Perangkat Keras Kriptografi, Politeknik Siber dan Sandi Negara, zamir.achmad@student.poltekssn.ac.id

4) Program Studi Rekayasa Kriptografi, Politeknik Siber dan Sandi Negara, achmad.fatih@student.poltekssn.ac.id

5) Program Studi Rekayasa Kriptografi, Politeknik Siber dan Sandi Negara, niputu.aishwara@student.poltekssn.ac.id

6) Program Studi Rekayasa Kriptografi, Politeknik Siber dan Sandi Negara, rio.nelson@student.poltekssn.ac.id

7) Program Studi Rekayasa Kriptografi, Politeknik Siber dan Sandi Negara, hermawan.setiawan@poltekssn.ac.id

8) Program Studi Teknik Sipil, Sekolah Tinggi Teknologi Pekerjaan Umum Jakarta, julianradhit@gmail.com

Riwayat Artikel

Dikirim 24 Feb 2026
Diterima 26 Agu 2026
Diterbitkan 25 Sep 2026

Kata kunci:

AES-128
Kriptografi
Mode *Counter* (CTR)
NIST SP 800-22
SPECK64/128

Keywors:

AES-128
Counter (CTR) *Mode Cryptography*
NIST SP 800-22
SPECK64/128

Abstrak

Penelitian ini membandingkan kinerja AES-128 berstruktur *Substitution-Permutation Network* (SPN) dan *cipher* ringan SPECK64/128 berstruktur *Add-Rotate-Xor* (ARX) yang diimplementasikan dalam bahasa C murni tanpa akselerasi perangkat keras pada mode *Counter* (CTR). Pengujian dilakukan pada lima ukuran data (500 KB hingga 10 MB) dengan sepuluh kali pengulangan untuk mengukur waktu eksekusi, *throughput*, dan ukuran berkas keluaran. AES-128 diukur pada implementasi naif dan implementasi teroptimasi berbasis *xtime* yang menghasilkan *ciphertext* yang identik. Keacakan *ciphertext* dievaluasi menggunakan NIST SP 800-22 Rev. 1a pada 20 berkas dari sepuluh kategori plaintext (55 baris × 1.000.000 bit, α = 0,01, 15 subtes). SPECK64/128 mencapai *throughput* enkripsi rata-rata 510,95 MB/s, sedangkan AES-128 teroptimasi 60,67 MB/s dan AES-128 naif 7,48 MB/s. Selisih 8,4 kali antara SPECK64/128 dan AES-128 teroptimasi mencerminkan perbedaan arsitektur, sedangkan selisih 8,1 kali antara kedua implementasi AES mencerminkan kualitas implementasi. Kedua algoritma tidak menyebabkan ekspansi ukuran berkas. Uji NIST mencatat 5 dan 4 baris gagal dari masing-masing 1.880 baris, tidak berbeda nyata dari nilai harapannya (P = 0,373 dan 0,616) maupun antar-algoritma (uji eksak Fisher, P = 1,000), sehingga seluruh berkas dan kategori subtes lulus; kelulusan ini merupakan syarat perlu, bukan syarat cukup, bagi keamanan.

Abstract

This study compares AES-128, based on a Substitution-Permutation Network (SPN), and the lightweight SPECK64/128 cipher, based on an Add-Rotate-Xor (ARX) structure, both implemented in pure C without hardware acceleration under Counter (CTR) mode. Benchmarking covered five data

sizes (500 KB to 10 MB) with ten repetitions, measuring execution time, throughput, and output size. AES-128 was measured in a naive and an xTime-based optimized implementation that produced identical ciphertext. Randomness was evaluated with NIST SP 800-22 Rev. 1a over 20 ciphertext files from ten plaintext categories (55 sequences × 1,000,000 bits, a = 0.01, 15 test categories). SPECK64/128 reached an average encryption throughput of 510.95 MB/s, compared with 60.67 MB/s for optimized AES-128 and 7.48 MB/s for naive AES-128. The 8.4x gap between SPECK64/128 and optimized AES-128 reflects architecture, whereas the 8.1x gap between the two AES implementations reflects implementation quality. Neither cipher caused ciphertext expansion. The NIST evaluation recorded 5 and 4 failing rows out of 1,880 rows, not significantly different from their expected values (P = 0.373 and 0.616) or from each other (Fisher's exact test, P = 1.000), so all files and test categories passed; passing is a necessary, not a sufficient, condition for security.

1. PENDAHULUAN

Kriptografi simetris merupakan fondasi pengamanan kerahasiaan data dalam komunikasi dan penyimpanan berkecepatan tinggi. *Advanced Encryption Standard* (AES), yang distandardisasi oleh NIST melalui FIPS 197 pada tahun 2001 dan diperbarui pada tahun 2023 tanpa perubahan teknis pada algoritma inti [1], diakui secara luas karena ketahanannya terhadap berbagai serangan kriptanalisis [2][3]. Namun, perangkat *Internet of Things* (IoT), jaringan sensor nirkabel, dan *smart card* memiliki keterbatasan memori, daya, dan kemampuan komputasi, sehingga penerapan AES berpotensi menghambat kinerja [4]. Kriptografi ringan (*Lightweight Cryptography*) hadir untuk mengoptimalkan efisiensi komputasi, jejak memori, dan penggunaan energi tanpa mengorbankan batas keamanan minimum [5].

National Security Agency (NSA) memperkenalkan keluarga *block cipher* ringan SIMON dan SPECK pada tahun 2013 [6]. SPECK memanfaatkan struktur ARX yang tidak memerlukan tabel substitusi (S-Box), sehingga unggul dalam implementasi perangkat lunak [7][8]. SPECK diajukan ke ISO/IEC JTC 1/SC 27 pada tahun 2015, tetapi ditolak pada April 2018 terutama karena keraguan terhadap transparansi NSA, bukan karena ditemukannya kelemahan kriptanalitik yang signifikan [9]. Standar kriptografi ringan NIST saat ini adalah keluarga *Ascon* yang dipilih pada tahun 2023 dan diterbitkan sebagai NIST SP 800-232 pada Agustus 2025 [10][11][12]. Oleh karena itu, SPECK dalam penelitian ini tidak diposisikan sebagai standar kriptografi ringan terkini, melainkan sebagai representasi *block cipher* ARX yang efisien pada perangkat lunak dan tetap relevan untuk dibandingkan dengan AES [13][14].

Sebagian besar studi komparatif terdahulu berfokus pada analisis teoretis, implementasi perangkat keras, atau hanya mengevaluasi satu metrik kinerja secara parsial [15][16]. Penelitian ini mengisi celah tersebut melalui evaluasi empiris yang secara bersamaan mengukur waktu eksekusi, *throughput*, efisiensi penyimpanan, dan keacakan *ciphertext* AES-128 (SPN) dan SPECK64/128 (ARX) pada implementasi C murni dalam mode CTR. Kontribusi penelitian ini meliputi: (1) data *benchmarking* pada lima ukuran beban tanpa AES-NI dengan interval terukur yang bebas dari latensi I/O; (2) pemisahan pengaruh arsitektur dari pengaruh kualitas implementasi melalui dua implementasi AES-128 yang menghasilkan ciphertext identik; dan (3) validasi keacakan sesuai protokol lengkap NIST SP 800-22 Rev. 1a [17]. Hipotesis yang diajukan adalah: (1) SPECK64/128 menghasilkan *throughput* perangkat lunak yang lebih tinggi daripada AES-128; (2) *throughput* kriptografis murni, yaitu *throughput* yang diukur di luar operasi I/O, bersifat konstan terhadap ukuran masukan; dan (3) *ciphertext* dari kedua algoritma memenuhi kriteria kelulusan NIST SP 800-22 pada seluruh 15 kategori subtes.

2. LANDASAN TEORI

2.1. Mode CTR, SPECK64/128, dan AES-128

Mode *Counter* (CTR) yang distandardisasi dalam NIST SP 800-38A mengubah *block cipher* menjadi pembangkit *keystream* [18]. Untuk blok ke-i, blok pencacah CTRi = Nonce || i dienkripsi dengan kunci K, lalu *ciphertext* diperoleh melalui Ci = Pi ⊕ EK(CTRi); dekripsi dilakukan dengan fungsi enkripsi yang sama, yaitu Pi = Ci ⊕ EK(CTRi) [19][20]. Mode CTR mendukung pemrosesan paralel dan akses acak serta tidak memerlukan *padding*, sehingga |C| = |P|. Syarat keamanannya adalah pasangan (K, Nonce) tidak boleh digunakan ulang, karena penggunaan ulang *keystream* dapat merusak kerahasiaan data [19][20].

SPECK64/128 menggunakan blok 64 bit (w = 32 bit), kunci 128 bit, dan 27 putaran [6][7]. Setiap putaran mentransformasikan (xi, yi) melalui xi+1 = ((xi ⋙ 8) + yi) ⊕ ki dan yi+1 = (yi ⋘ 3) ⊕ xi+1, dengan penjadwalan kunci yang menggunakan fungsi putaran yang serupa. Karena hanya menggunakan penjumlahan modular, rotasi, dan XOR, operasi SPECK dapat dipetakan langsung ke instruksi prosesor tanpa memerlukan tabel *lookup* [7][8].

AES-128 berbasis *Substitution-Permutation Network* (SPN) memproses blok 128 bit dalam *state* 4×4 *byte* melalui 10 putaran [3][15]. Setiap putaran terdiri dari *SubBytes* (substitusi nonlinier dengan S-Box berbasis inversi di $GF(2^8)$), *ShiftRows* (permutasi siklik baris), *MixColumns* (difusi linier berupa perkalian matriks di $GF(2^8)$), dan *AddRoundKey* (XOR dengan subkunci putaran). Tanpa AES-NI, pembacaan tabel dan perkalian medan Galois pada *SubBytes* dan *MixColumns* menimbulkan *overhead* instruksi yang lebih tinggi dibandingkan dengan struktur ARX [15][16].

2.1. Uji Keacakan NIST SP 800-22

Keluaran *block cipher* yang aman seharusnya tidak dapat dibedakan dari permutasi acak [19], sehingga penyimpangan statistik pada *ciphertext* dapat menjadi indikasi awal adanya kelemahan dalam difusi. Namun, NIST *Statistical Test Suite* (STS) hanya menguji sifat statistik barisan bit dan tidak membuktikan ketahanan terhadap kriptanalisis; kelulusan uji merupakan syarat perlu, bukan syarat cukup, bagi keamanan [17]. Setiap subtes merupakan uji hipotesis dengan H_0 bahwa barisan tersebut acak. Barisan dinyatakan lulus apabila *p-value* ≥ α dengan α = 0,01, sehingga sekitar satu dari seratus barisan yang benar-benar acak tetap akan ditolak (galat Tipe I). *P-value* juga tidak mengukur derajat keacakan, karena di bawah H_0 nilainya tersebar secara seragam pada interval [0, 1].

Subtes *Overlapping Template Matching*, *Linear Complexity*, *Random Excursions*, dan *Random Excursions Variant* menuntut panjang barisan $n \geq 10^6$ bit [17]. Pada tingkat kumpulan m barisan, NIST menerapkan dua kriteria secara serentak: proporsi barisan lulus tidak boleh kurang dari $\hat{p} - 3\sqrt{(\hat{p}(1-\hat{p})/m)}$ dengan $\hat{p} = 1 - \alpha$, dan sebaran *p-value* harus seragam berdasarkan uji *chi-square* atas sepuluh subinterval, dengan batas $P\text{-value}_t \geq 0{,}0001$.

3. METODE PENELITIAN

3.1. Lingkungan Implementasi dan Protokol Pengukuran

Seluruh eksperimen dijalankan pada satu komputer dengan spesifikasi seperti pada Tabel 1. Kode C murni kedua algoritma dikompilasi dengan perintah yang seragam; opsi *-march=native* dan *-maes* tidak diaktifkan dan tidak ada pustaka pihak ketiga seperti OpenSSL, sehingga yang terukur adalah kinerja intrinsik algoritma pada perangkat lunak, bukan kemampuan akselerator AES-NI. Perangkat lunak terdiri atas modul SPECK64/128 (27 subkunci 32-bit, blok 8 *byte*, pencacah 64-bit), modul AES-128 (11 subkunci putaran, blok 16 *byte*, pencacah 128-bit), serta modul pengujian dan I/O.

AES-128 diimplementasikan dalam dua versi yang menghasilkan *ciphertext* identik bit demi bit. Implementasi naif menghitung setiap perkalian $GF(2^8)$ pada *MixColumns* melalui gelung delapan iterasi, sehingga satu blok 16 *byte* menuntut 144 pemanggilan rutin perkalian *Galois*, sedangkan implementasi teroptimasi memakai identitas 2a = *xtime*(a) dan 3a = *xtime*(a) ⊕ a yang cukup diselesaikan dengan satu pergeseran bit dan satu operasi XOR bersyarat.

Tabel 1. Spesifikasi Lingkungan Eksperimen

Komponen	Spesifikasi
Prosesor	Intel Core i7-14650HX, 16 inti (8 *P-core* + 8 *E-core*) / 24 utas, 2,20–5,20 GHz
Memori utama	32 GB DDR5-5600, *dual channel*
Sistem operasi	Microsoft Windows 11 Home 25H2 (build 26200), 64-bit
Kompilator dan standar	GCC 15.2.0 (MSYS2), C99 dengan pustaka standar C saja
Perintah kompilasi	gcc -std=c99 -O2 -Wall -o benchmark.exe aes.c speck.c test.c
Pustaka eksternal	Tidak ada (tanpa OpenSSL); AES-NI tidak diaktifkan
Instrumen waktu	Windows API *QueryPerformanceCounter*, resolusi 10.000.000 tick/detik
Interval terukur	Ekspansi kunci dan gelung pemrosesan blok, di luar operasi I/O berkas
Pengulangan dan daya	N = 10 pengulangan setelah 1 pemanasan; profil daya *Balanced*, tersambung listrik

Waktu diukur dengan *QueryPerformanceCounter* yang dibaca tepat sebelum dan sesudah ekspansi kunci serta gelung pemrosesan blok, sehingga waktu baca-tulis berkas tidak termasuk. Setiap skenario diulang N = 10 kali setelah satu eksekusi pemanasan, dengan komputer terhubung ke daya listrik, tanpa aplikasi lain di latar depan, dan pembaruan otomatis dinonaktifkan. Waktu rata-rata dihitung sebagai $T_{avg} = (1/N) \sum T_k$ dengan simpangan baku berpenyebut $N - 1$, *throughput* sebagai TP = ukuran berkas (MB) / T_{avg} dengan 1 MB = 1.000.000 *byte*, dan efisiensi penyimpanan sebagai rasio S_{output} / S_{input}.

3.2. *Dataset* dan Protokol Uji NIST

Uji kinerja menggunakan lima berkas berukuran 500 KB, 2 MB, 5 MB, 6,875 MB, dan 10 MB dengan kunci 128 bit yang sama untuk kedua algoritma. Uji keacakan menggunakan sepuluh kategori plaintext heterogen dengan ukuran awal sekitar 140.000 byte, yaitu data acak CSPRNG, teks alami bahasa Indonesia, citra bitmap tak terkompresi, citra PNG berderau, potongan biner ELF, audio PCM 16-bit, dokumen PDF, data tabular CSV, serta dua kasus ekstrem: seluruh *byte* 0x00 dan seluruh *byte* 0xFF. Setiap *plaintext* diperbesar hingga menghasilkan *ciphertext* 55.000.000 bit dan dienkripsi dengan kedua algoritma, sehingga diperoleh 20 berkas *ciphertext*.

Pengujian menggunakan NIST STS versi 2.1.2 yang dikompilasi dari arsip resmi tanpa modifikasi, dengan n = 1.000.000 bit, m = 55 baris per berkas, $\alpha = 0{,}01$, masukan biner, dan seluruh 15 kategori subtes. Satu berkas menghasilkan 188 baris pengujian (*Non-overlapping Template Matching* 148 baris, *Random Excursions Variant* 18, *Random Excursions* 8, *Cumulative Sums* dan Serial masing-masing 2, serta sepuluh kategori lain masing-masing 1). Satu baris dinyatakan gagal apabila cacah barisan lulus kurang daripada ambang 52 dari 55 atau $P\text{-}value_t < 0{,}0001$. Kedua kriteria dihitung ulang secara mandiri pada 3.760 baris dan cocok 100% dengan tanda yang dicetak NIST STS pada finalAnalysisReport.txt.

Di bawah hipotesis *keacakan*, peluang satu baris gagal adalah peluang cacah barisan gagal melampaui ambang proporsi ditambah 0,0001, yaitu 0,002372 untuk m = 55; pada *Random Excursions* dan *Random Excursions Variant,* peluang ini dihitung dari m′ aktual pada tiap berkas (27–44 baris). Banyaknya baris yang gagal mengikuti sebaran *Poisson-binomial* yang dihitung secara eksak [21], dengan nilai harapan 0,394–0,473 baris per berkas. Nilai $P(X \geq 4)$ tertinggi adalah 0,0014 dan $P(X \geq 5)$ tertinggi adalah 0,00013, sehingga dengan koreksi *Bonferroni* untuk 20 berkas (0,01 / 20 = 0,0005), nilai kritisnya adalah 5 baris gagal: 0–1 baris gagal dinyatakan konsisten dengan hipotesis keacakan, 2–4 baris dalam batas ragam acak, dan ≥ 5 baris menyimpang secara nyata.

Selain penilaian per berkas, hasil juga direkapitulasi berdasarkan kategori subtes. Untuk setiap kategori dan algoritma dilaporkan $P\text{-}value_t$ minimum, proporsi lulus minimum, jumlah baris gagal beserta nilai harapannya, serta peluang eksak $P(X \geq x)$ memperoleh sekurang-kurangnya x baris gagal; kategori dinyatakan lulus apabila $P(X \geq x) \geq 0{,}01$. Perbedaan jumlah baris yang gagal antara kedua algoritma diuji dengan uji eksak Fisher [22]. Perhitungan tersebut mengasumsikan antar baris saling bebas, sehingga nilai P yang diperoleh merupakan pendekatan.

4. HASIL DAN PEMBAHASAN

4.1. Waktu Eksekusi dan *Throughput*

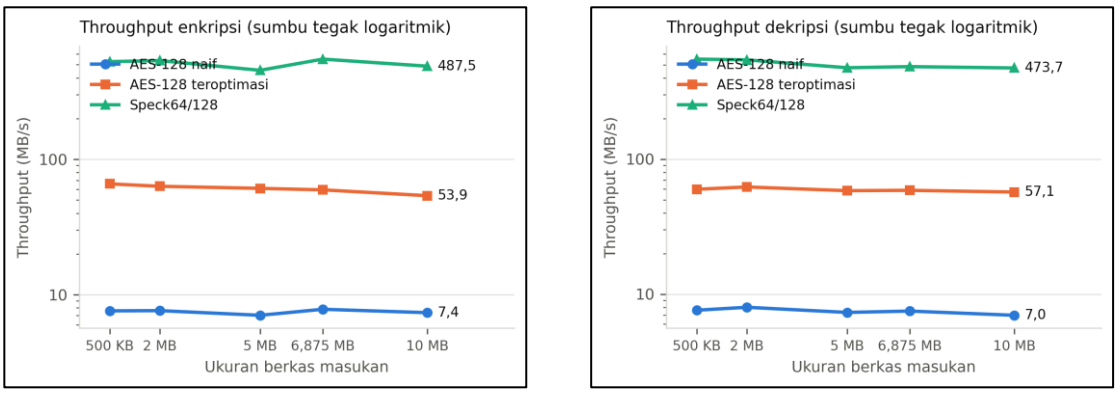
Tabel 2 dan Tabel 3 menyajikan waktu eksekusi (rata-rata ± simpangan baku dari sepuluh pengulangan) serta *throughput* enkripsi dan dekripsi, sedangkan Gambar 1 memperlihatkan *throughput* pada skala logaritmik. Waktu eksekusi ketiga implementasi tumbuh linier terhadap ukuran masukan, sesuai dengan sifat mode CTR yang memproses setiap blok secara mandiri. Pada beban 10 MB, enkripsi SPECK64/128 selesai dalam 0,0205 detik, AES-128 teroptimasi dalam 0,1857 detik, dan AES-128 naif dalam 1,3581 detik. *Throughput* enkripsi rata-rata SPECK64/128 mencapai 510,95 MB/s, AES-128 teroptimasi 60,67 MB/s, dan AES-128 naif 7,48 MB/s; pada dekripsi, nilainya berturut-turut 505,79 MB/s, 59,26 MB/s, dan 7,47 MB/s. Selisih enkripsi dan dekripsi berada di bawah 2 persen dan masih tercakup dalam simpangan baku, karena mode CTR menggunakan fungsi enkripsi yang sama untuk kedua arah.

Tabel 2. Waktu Eksekusi dan *Throughput* Enkripsi AES-128 dan SPECK64/128

Ukuran berkas	AES-128 naif waktu (s)	AES-128 teroptimasi waktu (s)	SPECK64/128 waktu (s)	AES-128 naif (MB/s)	AES-128 teroptimasi (MB/s)	SPECK64/128 (MB/s)
500 KB	0,065865 ± 0,015772	0,007593 ± 0,000730	0,000949 ± 0,000123	7,59	65,85	526,82
2 MB	0,262472 ± 0,019999	0,031650 ± 0,003403	0,003730 ± 0,000692	7,62	63,19	536,22
5 MB	0,709743 ± 0,068381	0,081979 ± 0,011102	0,011006 ± 0,001511	7,04	60,99	454,32
6,875 MB	0,881059 ± 0,052896	0,115604 ± 0,018551	0,012502 ± 0,001029	7,80	59,47	549,90
10 MB	1,358106 ± 0,088999	0,185694 ± 0,025834	0,020513 ± 0,001482	7,36	53,85	487,48
Rata-rata *throughput*	–	–	–	7,48	60,67	510,95

Tabel 3. Waktu Eksekusi dan *Throughput* Dekripsi AES-128 dan SPECK64/128

Ukuran berkas	AES-128 naif waktu (s)	AES-128 teroptimasi waktu (s)	SPECK64/128 waktu (s)	AES-128 naif (MB/s)	AES-128 teroptimasi (MB/s)	SPECK64/128 (MB/s)
500 KB	0,065751 ± 0,010106	0,008356 ± 0,002792	0,000907 ± 0,000084	7,60	59,84	551,12
2 MB	0,250363 ± 0,016763	0,032138 ± 0,004573	0,003678 ± 0,000643	7,99	62,23	543,83
5 MB	0,683102 ± 0,042121	0,085585 ± 0,009515	0,010514 ± 0,001668	7,32	58,42	475,55
6,875 MB	0,918227 ± 0,067163	0,116994 ± 0,016046	0,014184 ± 0,003523	7,49	58,76	484,72
10 MB	1,435648 ± 0,138283	0,175216 ± 0,015218	0,021109 ± 0,002628	6,97	57,07	473,72
Rata-rata *throughput*	–	–	–	7,47	59,26	505,79



(a) (b)

Gambar 1. *Throughput* AES-128 dan SPECK64/128 Mode CTR (MB/s, sumbu tegak logaritmik): (a) Enkripsi, (b) Dekripsi

Rasio *throughput* SPECK64/128 terhadap AES-128 teroptimasi adalah 8,4 kali, sedangkan rasio antara kedua implementasi AES adalah 8,1 kali. Hanya rasio pertama yang dapat diatribusikan pada perbedaan arsitektur ARX terhadap SPN; rasio kedua sepenuhnya berasal dari cara MixColumns dihitung, karena kedua implementasi AES menjalankan algoritma yang sama dan menghasilkan

ciphertext yang identik. Dengan demikian, perbandingan kinerja antar-algoritma kriptografi hanya bermakna apabila tingkat optimasi implementasi yang dibandingkan setara.

Sejalan dengan hipotesis kedua, *throughput* enkripsi ketiga implementasi praktis mendatar pada seluruh rentang beban: AES-128 naif berkisar 7,04–7,80 MB/s, AES-128 teroptimasi 53,85–65,85 MB/s, dan SPECK64/128 454,32–549,90 MB/s, dengan variasi yang sebanding dengan simpangan baku antarpengulangan. Kenaikan *throughput* terhadap ukuran berkas yang lazim dilaporkan pada pengukuran menyeluruh bersumber dari teramortisasinya biaya tetap inisialisasi dan I/O berkas, bukan dari perilaku algoritma kriptografi.

4.2. Efisiensi Penyimpanan

Tabel 4 membandingkan ukuran berkas masukan dan *ciphertext*. Pada kelima ukuran uji, kedua algoritma menghasilkan keluaran yang berukuran sama persis dengan masukan, sehingga rasio S_{output} / S_{input} bernilai 1,000 tanpa kecuali. Hasil tersebut mengonfirmasi karakteristik mode CTR: *block cipher* hanya menghasilkan keystream yang di-XOR-kan dengan plaintext, sehingga tidak memerlukan *padding*. Sifat ini berlaku terlepas dari ukuran blok internal, yaitu 64 bit pada SPECK64/128 dan 128 bit pada AES-128, karena *keystream* dipotong tepat sesuai panjang *plaintext*. Perbedaan ukuran keluaran yang teramati pada pengujian sebelumnya bersumber dari mode ECB yang menuntut *padding* PKCS#7.

Tabel 4. Ukuran Berkas Masukan dan Keluaran Pasca Enkripsi Mode CTR

Ukuran berkas	S_input (byte)	S_output AES-128 (byte)	S_output SPECK64/128 (byte)	Rasio S_output / S_input
500 KB	500.000	500.000	500.000	1,000
2 MB	2.000.000	2.000.000	2.000.000	1,000
5 MB	5.000.000	5.000.000	5.000.000	1,000
6,875 MB	6.875.000	6.875.000	6.875.000	1,000
10 MB	10.000.000	10.000.000	10.000.000	1,000

4.3. Keacakan *Ciphertext* Berdasarkan NIST SP 800-22

Tabel 5 menyajikan hasil per berkas beserta peluang eksak $P(X \geq x)$. AES-128-CTR mencatat 5 baris gagal dari 1.880 baris (0,266%), sedangkan SPECK64/128-CTR mencatat 4 baris gagal dari 1.880 baris (0,213%). Tidak ada berkas yang mencapai nilai kritis lima baris gagal: 18 berkas tergolong konsisten dengan hipotesis keacakan, dan 2 berkas, yaitu AES-128-CTR 08 dan SPECK64/128-CTR 02, masih dalam batas ragam acak dengan $P(X \geq 2)$ sebesar 0,060 dan 0,082; keduanya di atas $\alpha = 0,01$.

Tabel 5. Hasil Uji NIST SP 800-22 Rev. 1a per Berkas *Ciphertext* ($n = 10^6$ bit, $m = 55$, $\alpha = 0,01$, 188 baris per berkas)

Kategori *plaintext*	AES-128-CTR baris gagal (P)	AES-128-CTR status agregat	SPECK64/128-CTR baris gagal (P)	SPECK64/128-CTR status agregat
01 Data acak (CSPRNG)	0 (1,000)	Konsisten	1 (0,326)	Konsisten
02 Teks alami	0 (1,000)	Konsisten	2 (0,082)	Dalam batas ragam
03 Citra BMP	0 (1,000)	Konsisten	1 (0,329)	Konsisten
04 Citra PNG berderau	1 (0,338)	Konsisten	0 (1,000)	Konsisten
05 Biner ELF	1 (0,327)	Konsisten	0 (1,000)	Konsisten
06 Audio PCM 16-bit	0 (1,000)	Konsisten	0 (1,000)	Konsisten
07 Dokumen PDF	1 (0,336)	Konsisten	0 (1,000)	Konsisten
08 Data CSV	2 (0,060)	Dalam batas ragam	0 (1,000)	Konsisten
09 Seluruh *byte* 0x00	0 (1,000)	Konsisten	0 (1,000)	Konsisten
10 Seluruh *byte* 0xFF	0 (1,000)	Konsisten	0 (1,000)	Konsisten
Jumlah (1.880 baris)	5 dari 1.880 (0,373)	10 dari 10 lulus	4 dari 1.880 (0,616)	10 dari 10 lulus

Keterangan: angka dalam kurung adalah $P(X \geq x)$ eksak untuk x baris gagal di bawah hipotesis keacakan. "Konsisten" = 0–1 baris gagal; "Dalam batas ragam" = 2–4 baris gagal; "Menyimpang nyata" = ≥ 5 baris gagal.

Tabel 6 merinci hasil yang sama untuk setiap kategori subtes. Dari sembilan baris yang gagal, tujuh berasal dari kriteria proporsi Uji *Non-overlapping Template Matching*, satu dari kriteria proporsi

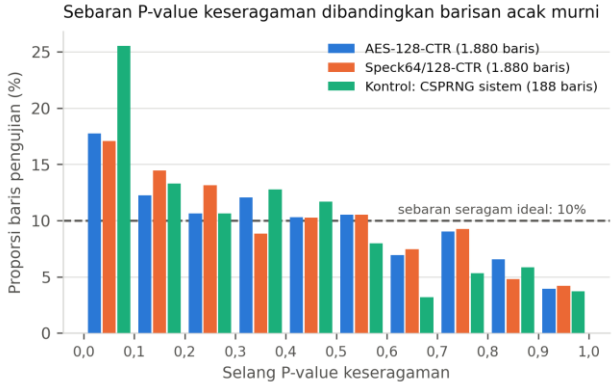
Uji *Block Frequency* (AES-128-CTR 07, 51/55), dan satu dari kriteria keseragaman Uji *Non-overlapping Template Matching* (SPECK64/128-CTR 03, *P-value*$_t$ = 0,000012). Pada *Non-overlapping Template Matching*, 4 baris gagal per algoritma sebanding dengan nilai harapan 3,51 dari 1.480 baris (P = 0,466). Kegagalan tunggal *Block Frequency* pada AES-128-CTR memiliki P = 0,023, yang tidak signifikan pada α = 0,01 maupun setelah koreksi Bonferroni untuk 30 perbandingan (0,00033). Kategori lainnya tidak mencatat kegagalan, sehingga seluruh 15 kategori pada kedua algoritma berstatus lulus.

Tabel 6. *P-value* Keseragaman, Proporsi Lulus, dan Status per Kategori Subtes NIST (Gabungan 10 Berkas per Algoritma)

Subtes (baris)	AES-128-CTR					SPECK64/128-CTR				
	P-value$_t$ min	Proporsi min	Gagal (E)	P(X ≥ x)	Status	*P-value*$_t$ min	Proporsi min	Gagal (E)	P(X ≥ x)	Status
Frequency (10)	0,0146	53/55	0 (0,02)	1,000	Lulus	0,1025	54/55	0 (0,02)	1,000	Lulus
Block Frequency (10)	0,0220	51/55	1 (0,02)	0,023	Lulus	0,0062	54/55	0 (0,02)	1,000	Lulus
Cumulative Sums (20)	0,0329	53/55	0 (0,05)	1,000	Lulus	0,0220	53/55	0 (0,05)	1,000	Lulus
Runs (10)	0,0046	53/55	0 (0,02)	1,000	Lulus	0,0554	53/55	0 (0,02)	1,000	Lulus
Longest Run (10)	0,0554	53/55	0 (0,02)	1,000	Lulus	0,0288	53/55	0 (0,02)	1,000	Lulus
Rank (10)	0,0712	53/55	0 (0,02)	1,000	Lulus	0,1025	54/55	0 (0,02)	1,000	Lulus
DFT (*Spectral*) (10)	0,0110	54/55	0 (0,02)	1,000	Lulus	0,0288	54/55	0 (0,02)	1,000	Lulus
Non-overlapping Template (1.480)	0,000474	50/55	4 (3,51)	0,466	Lulus	0,000012	51/55	4 (3,51)	0,466	Lulus
Overlapping Template (10)	0,1626	52/55	0 (0,02)	1,000	Lulus	0,0805	52/55	0 (0,02)	1,000	Lulus
Universal (10)	0,0554	54/55	0 (0,02)	1,000	Lulus	0,0376	54/55	0 (0,02)	1,000	Lulus
Approximate Entropy (10)	0,0487	53/55	0 (0,02)	1,000	Lulus	0,0487	53/55	0 (0,02)	1,000	Lulus
Random Excursions (80)	0,0018	36/39	0 (0,05)	1,000	Lulus	0,0011	25/27	0 (0,13)	1,000	Lulus
Random Excursions Variant (180)	0,0011	37/40	0 (0,12)	1,000	Lulus	0,000163	28/30	0 (0,29)	1,000	Lulus
Serial (20)	0,0054	54/55	0 (0,05)	1,000	Lulus	0,0909	53/55	0 (0,05)	1,000	Lulus
Linear Complexity (10)	0,000883	54/55	0 (0,02)	1,000	Lulus	0,0012	54/55	0 (0,02)	1,000	Lulus
Seluruh subtes (1.880)	0,000474	50/55	5 (4,01)	0,373	Lulus	0,000012	51/55	4 (4,26)	0,616	Lulus

Keterangan: *P-value*$_t$ min = *P-value* keseragaman terkecil (baris gagal bila < 0,0001); proporsi min = rasio lulus terkecil (ambang 52/55 untuk m = 55, ditetapkan NIST STS untuk m′ < 55); E = nilai harapan baris gagal; E dan P(X ≥ x) dihitung secara eksak menggunakan sebaran *Poisson-binomial*; status lulus bila P(X ≥ x) ≥ 0,01.

Secara keseluruhan, jumlah baris gagal tidak berbeda nyata dari nilai harapannya, yaitu 5 berbanding 4,01 pada AES-128-CTR (P = 0,373) dan 4 berbanding 4,26 pada SPECK64/128-CTR (P = 0,616). Uji eksak Fisher pada 5/1.880 dan 4/1.880 baris gagal menghasilkan P = 1,000, sehingga tidak terdapat perbedaan yang signifikan antara kedua algoritma. Proporsi barisan lulus gabungan sebesar 0,9903 pada kedua algoritma, sesuai dengan nilai nominal 1 − α = 0,99. Kegagalan muncul pada berkas berentropi tinggi (01 data acak) maupun rendah (02 teks alami), sedangkan kasus 0x00 dan 0xFF tidak mencatat kegagalan; pada mode CTR, *ciphertext* atas *plaintext* bernilai nol identik dengan *keystream*, sehingga kasus ini menguji pembangkitan *keystream* secara langsung.



Gambar 2. Sebaran *P-value* Keseragaman Seluruh Baris Pengujian Dibandingkan Barisan Acak Murni sebagai Kontrol

Gambar 2 memperlihatkan sebaran *P-value*$_t$ seluruh baris pengujian. Proporsi *P-value*$_t$ pada selang [0; 0,1) adalah 17,8% pada AES-128-CTR dan 17,1% pada SPECK64/128-CTR, di atas 10% pada sebaran seragam, dengan median masing-masing 0,367 dan 0,335. Barisan kontrol 6.875.000 *byte* dari CSPRNG sistem operasi yang diuji dengan protokol yang sama menunjukkan proporsi sekitar 25% pada selang tersebut dan 2 baris gagal dari 188, sehingga kecondongan ini merupakan

perilaku perangkat uji pada m = 55, bukan cacat keluaran *cipher*. Uji kecocokan sebaran pada histogram gabungan ini tidak valid karena baris pengujian tidak saling bebas: 148 baris Non-overlapping Template Matching dijalankan pada barisan yang sama, frekuensi harapan chi-square pada m = 55 hanya 5,5 per subinterval, dan karena kunci serta nonce yang sama, ciphertext kasus 0x00 dan 0xFF saling berkomplemen sehingga subtes yang simetris menghasilkan *P-value* yang identik. Penilaian kelulusan didasarkan pada Tabel 5 dan Tabel 6, bukan pada bentuk histogram. Selain itu, Kowalska dkk. [23] menunjukkan kekeliruan perhitungan peluang pada Uji *Overlapping Template Matching* di NIST STS 2.1.2; kekeliruan ini berlaku sama bagi kedua algoritma sehingga tidak memengaruhi perbandingan. Kelulusan seluruh 20 berkas menunjukkan bahwa keluaran kedua algoritma tidak memperlihatkan penyimpangan statistik yang terdeteksi oleh NIST SP 800-22, sehingga memenuhi persyaratan yang diperlukan untuk keluaran *cipher* blok yang aman. Hasil ini tidak membuktikan ketahanan terhadap kriptanalisis diferensial, kriptanalisis linier, maupun serangan kanal samping; bukti mengenai hal tersebut berada dalam ranah kriptanalisis publik terhadap kedua algoritma [6][15][9].

4.4. Sintesis Komparatif dan *Trade-Off Arsitektural*

Keunggulan *throughput* SPECK64/128 sebesar 8,4 kali dibandingkan AES-128 teroptimasi berakar pada operasi ARX yang dieksekusi langsung pada register CPU tanpa pembacaan tabel S-Box [7][8]. Sebaliknya, AES-128 memiliki margin keamanan yang telah teruji luas sebagai standar global [2][3][15] dan dapat memanfaatkan AES-NI yang tidak tersedia pada SPECK64/128, sehingga urutan kinerja keduanya dapat berbalik pada lingkungan yang mendukung instruksi tersebut. Selisih 8,1 kali antara dua implementasi AES-128 dengan *output* yang identik menegaskan bahwa angka *throughput* dalam literatur perlu dibaca bersama keterangan tingkat optimasi implementasinya.

Kedua algoritma memenuhi kriteria NIST SP 800-22 [17] tanpa perbedaan jumlah kegagalan yang signifikan, tetapi kelulusan ini hanya menunjukkan tidak adanya cacat difusi yang terdeteksi secara statistik dan bukan jaminan terhadap serangan kanal samping pada implementasi yang tidak berwaktu tetap. Rekomendasi SPECK64/128 didasarkan pada efisiensi perangkat lunak ARX, bukan sebagai alternatif terhadap standar kriptografi ringan NIST yang kini dipegang oleh *Ascon* [11][12]. SPECK64/128 sesuai untuk transmisi data berkecepatan tinggi pada perangkat tertanam dan IoT yang membutuhkan efisiensi *throughput* perangkat lunak, sedangkan AES-128 tetap menjadi standar utama pada lingkungan komputasi yang didukung akselerasi perangkat keras [2][3].

5. KESIMPULAN

Berdasarkan eksperimen komparatif AES-128 dan SPECK64/128 pada mode CTR dalam implementasi C murni, diperoleh kesimpulan sebagai berikut.

1. SPECK64/128 mengungguli AES-128 teroptimasi sebesar 8,4 kali, dengan *throughput* enkripsi 510,95 MB/s berbanding 60,67 MB/s dan *throughput* dekripsi 505,79 MB/s berbanding 59,26 MB/s, berkat efisiensi operasi ARX yang tidak memerlukan S-Box.
2. Dua implementasi AES-128 yang menghasilkan *ciphertext* identik menunjukkan perbedaan kinerja 8,1 kali (60,67 MB/s berbanding 7,48 MB/s), sehingga perbandingan antaralgoritma hanya bermakna pada tingkat optimasi implementasi yang setara.
3. *Throughput* kriptografis murni ketiga implementasi konstan pada rentang 500 KB hingga 10 MB; kenaikan *throughput* pada pengukuran menyeluruh berasal dari amortisasi biaya inisialisasi dan I/O, bukan dari perilaku algoritma.
4. Pada mode CTR, kedua algoritma tidak menimbulkan ekspansi berkas ($|C| = |P|$, rasio 1,000) pada seluruh ukuran uji.
5. Seluruh 20 berkas *ciphertext* dan 15 kategori subtes lulus NIST SP 800-22. Jumlah baris gagal, yaitu 5 dan 4 dari masing-masing 1.880 baris, tidak berbeda nyata dari nilai harapannya (P = 0,373 dan

0,616) maupun antara kedua algoritma (P = 1,000). Kelulusan ini merupakan syarat perlu, bukan syarat cukup, bagi keamanan.

6. SPECK64/128 direkomendasikan untuk perangkat lunak berkecepatan tinggi pada sistem tertanam dan IoT, sedangkan AES-128 tetap menjadi pilihan standar pada komputasi umum, terlebih pada lingkungan yang menyediakan AES-NI.

REFERENSI

[1] National Institute of Standards and Technology, *Advanced Encryption Standard (AES)*, Federal Information Processing Standards Publication 197 (Update 1), U.S. Department of Commerce, Gaithersburg, MD, USA, 2023, doi: 10.6028/NIST.FIPS.197-upd1.

[2] A. M. Abdullah, "Advanced Encryption Standard (AES) algorithm to encrypt and decrypt data," *International Journal of Scientific & Engineering Research*, vol. 8, no. 6, pp. 1–11, 2017.

[3] National Institute of Standards and Technology, *Advanced Encryption Standard (AES)*, Federal Information Processing Standards Publication 197, U.S. Department of Commerce, Gaithersburg, MD, USA, 2001, doi: 10.6028/NIST.FIPS.197.

[4] V. A. Thakor, M. A. Razzaque, and M. R. A. Khandaker, "Lightweight cryptography for IoT: A state-of-the-art survey," *IEEE Access*, vol. 9, pp. 28168–28193, 2021, doi: 10.1109/ACCESS.2021.3056678.

[5] A. Biryukov and L. Perrin, "State of the art in lightweight symmetric cryptography," Cryptology ePrint Archive, Report 2017/511, International Association for Cryptologic Research, 2017. [Daring]. Tersedia: https://eprint.iacr.org/2017/511

[6] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers, "The SIMON and SPECK families of lightweight block ciphers," Cryptology ePrint Archive, Report 2013/404, International Association for Cryptologic Research, 2013. [Daring]. Tersedia: https://eprint.iacr.org/2013/404

[7] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers, "SIMON and SPECK: Block ciphers for the Internet of Things," in *Proc. 52nd Annu. Design Automation Conf. (DAC '15)*, San Francisco, CA, USA, 2015, pp. 1–6, doi: 10.1145/2744769.2747946.

[8] P. Ahir, M. Mozaffari-Kermani, and R. Azarderakhsh, "Lightweight architectures for reliable and fault detection Simon and Speck cryptographic algorithms on FPGA," *ACM Transactions on Embedded Computing Systems*, vol. 16, no. 4, Art. no. 109, 2017, doi: 10.1145/3063380.

[9] T. Ashur and A. Luykx, "An Account of the ISO/IEC Standardization of the Simon and Speck Block Cipher Families," in Security of Ubiquitous Computing Systems: Selected Topics, G. Avoine and J. Hernandez-Castro, Eds. Cham, Switzerland: Springer, 2021, pp. 63–78, doi: 10.1007/978-3-030-10591-4_4.

[10] C. Dobraunig, M. Eichlseder, F. Mendel, and M. Schläffer, "Ascon v1.2: Lightweight authenticated encryption and hashing," *Journal of Cryptology*, vol. 34, no. 3, Art. no. 33, 2021, doi: 10.1007/s00145-021-09398-9.

[11] National Institute of Standards and Technology, "NIST selects 'Ascon' algorithm for lightweight cryptography standard," *NIST News*, 2023. [Daring]. Tersedia: https://www.nist.gov/news-events/news/2023/02/nist-selects-ascon-algorithm-lightweight-cryptography-standard

[12] M. Sönmez Turan, K. A. McKay, D. Chang, J. Kang, and J. Kelsey, Ascon-Based Lightweight Cryptography Standards for Constrained Devices, NIST Special Publication 800-232, National Institute of Standards and Technology, Gaithersburg, MD, USA, 2025, doi: 10.6028/NIST.SP.800-232.

[13] W. E. Putri and F. Darwiynes, "Analisis perbandingan block cipher Simon-Speck, Simeck, Skinny pada komunikasi berbasis LoRa," *Multinetics*, vol. 8, no. 2, pp. 143–151, 2022, doi: 10.32722/multinetics.v8i2.4925.

[14] M. A. Taqwim, A. Kusyanti, and R. A. Siregar, "Implementasi algoritme Speck untuk enkripsi one-time password pada two-factor authentication," *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, vol. 5, no. 7, pp. 3103–3111, 2021.

[15] J. Daemen and V. Rijmen, *The Design of Rijndael: AES - The Advanced Encryption Standard*. Berlin, Germany: Springer-Verlag, 2002, doi: 10.1007/978-3-662-04722-4.

[16] F. K. Wardhana, A. Kurniawan, B. R. Seto, and I. A. Saputro, "Analisis perbandingan kinerja enkripsi algoritma RC4 dan AES," in *Prosiding Seminar Nasional AMIKOM Surakarta (SEMNASA 2023)*, Surakarta, Indonesia, 2023, pp. 1–7.

[17] A. Rukhin *et al.*, "A statistical test suite for random and pseudorandom number generators for cryptographic applications," NIST Special Publication 800-22 Rev. 1a, National Institute of Standards and Technology, Gaithersburg, MD, USA, 2010, doi: 10.6028/NIST.SP.800-22r1a.

[18] M. Dworkin, "Recommendation for block cipher modes of operation: Methods and techniques," NIST Special Publication 800-38A, National Institute of Standards and Technology, Gaithersburg, MD, USA, 2001, doi: 10.6028/NIST.SP.800-38A.

[19] J. Katz and Y. Lindell, *Introduction to Modern Cryptography*, 3rd ed. Boca Raton, FL, USA: CRC Press, 2020, doi: 10.1201/9781351133036.

[20] P. Rogaway, "Evaluation of some blockcipher modes of operation," Cryptography Research and Evaluation Committees (CRYPTREC), 2004, pp. 1–43.

[21] Y. Hong, "On computing the distribution function for the Poisson binomial distribution," *Computational Statistics & Data Analysis*, vol. 59, pp. 41–51, 2013, doi: 10.1016/j.csda.2012.10.006.

[22] A. Agresti, *Categorical Data Analysis*, 3rd ed. Hoboken, NJ, USA: John Wiley & Sons, 2013.

[23] A. Kowalska, D. Fogliano, and J. Garcia Coello, "On the revision of NIST 800-22 test suites," Cryptology ePrint Archive, Report 2022/540, International Association for Cryptologic Research, 2022. [Daring]. Tersedia: https://eprint.iacr.org/2022/540