Full Paper

Dashboard Interaktif untuk Evaluasi Komprehensif Kinerja dan Statistik Pseudo Random Number Generators (PRNG)

Arisman[1], Nurhayati[2], Taufiq Azmi Harahap[3]
[1,2]Teknik Informatika, [3]Bisnis Universitas Mikroskil

Informasi Artikel

Terbit: Juli 2025

Kata Kunci:

Pseudo Random Number Generator
AES-CTR
ChaCha20
MT19937
Entropy

ABSTRAK

Masa era digital ditandai dengan adanya pertukaran informasi secara masif, kebutuhan terhadap sistem keamanan yang tangguh menjadi krusial. Salah satu komponen fundamental dari sistem keamanan tersebut adalah pembangkitan bilangan acak (Pseudo Random Number Generation - PRNG) yang tidak terprediksi dan tidak menampilkan pola statistik. Kualitas dan ketidakterdugaan dari bilangan acak menjadi penentu utama kekuatan sebuah sistem enkripsi. Penelitian ini menyajikan evaluasi komparatif terhadap tiga algoritma PRNG yaitu AES-CTR, ChaCha20, dan Mersenne Twister (MT). Evaluasi dilakukan menggunakan output data berukuran 50 MB dengan simulasi sebanyak 20 kali pada setiap algoritma berdasarkan lima parameter utama: waktu pemrosesan, penggunaan memori, nilai entropi, nilai chi-square, dan standar deviasi dari distribusi byte. Hasil pengujian menunjukkan bahwa ketiga algoritma menghasilkan keluaran dengan tingkat keacakan yang sangat tinggi, ditunjukkan oleh nilai entropi yang identik dan ideal (8.0000). Algoritma ChaCha20 menunjukkan performa paling efisien, dengan waktu proses tercepat (0.1009 detik) dan konsumsi memori paling rendah (70.6113 MB). Di sisi lain, AES-CTR menghasilkan performa statistik yang sangat baik namun dengan konsumsi memori tertinggi (130.6746 MB. Mersenne Twister, menunjukkan performa distribusi byte yang sangat stabil, dengan nilai chi-square terendah (249.9215) dan standar deviasi terkecil (446.8977), mengindikasikan penyebaran data yang sangat merata.

Corresponding Author:

Arisman
Email: arisman@mikroskil.ac.id

1. PENDAHULUAN

Pembangkitan bilangan acak Semu (Pseudeo Random Number Generation - RNG) merupakan pilar fundamental yang menopang dua domain krusial dalam komputasi modern yaitu keamanan kriptografi dan simulasi ilmiah[1][2]. Dalam kriptografi, ketidakterdugaan (unpredictability) sebuah urutan bilangan adalah penentu utama kekuatan protokol keamanan. Kunci enkripsi, nonce, dan token yang dihasilkan dari generator yang lemah dapat menciptakan celah fatal yang memungkinkan kompromi total sebuah sistem. Di sisi lain, dalam dunia komputasi ilmiah seperti metode Monte Carlo atau fisika statistik, kualitas yang dicari adalah properti statistik yang nyaris sempurna, seperti distribusi seragam dan ketiadaan korelasi, untuk memastikan hasil simulasi yang akurat dan absah.

Secara umum, generator bilangan acak diklasifikasikan berdasarkan tujuan penggunaannya, Pseudorandom Number Generators (PRNG) seperti Mersenne Twister yang dirancang untuk kecepatan dan kualitas statistik[1]. Cryptographically Secure Pseudorandom Number Generators (CSPRNG) seperti AES-CTR dan ChaCha20 yang dirancang untuk menahan serangan prediktif. Meskipun klasifikasi ini jelas, tantangan praktis muncul saat memilih generator yang tepat untuk sebuah aplikasi, sebuah keputusan yang dikenal sebagai fitness for purpose[3]. Kesalahan dalam pemilihan dapat berakibat fatal, penggunaan PRNG statistik dalam konteks kriptografi adalah resep bencana keamanan, sementara penggunaan CSPRNG yang lebih lambat untuk simulasi ilmiah dapat menyebabkan overhead kinerja yang tidak perlu.

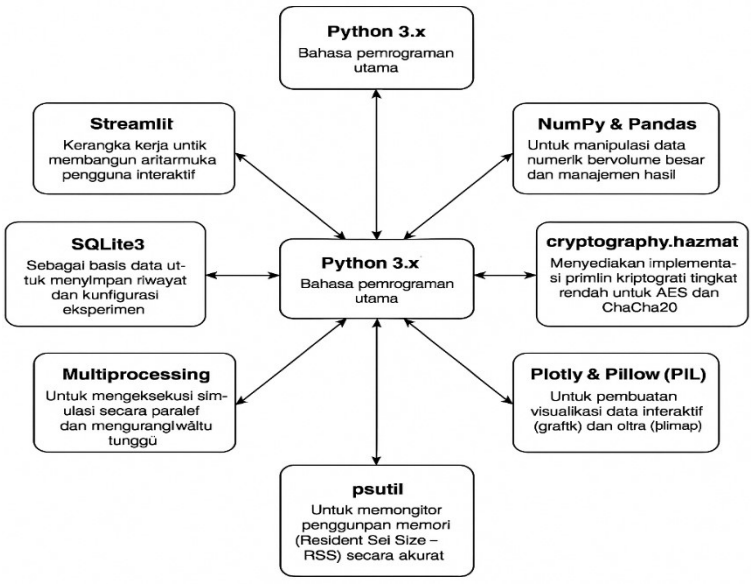
Masalah ini diperparah oleh fakta bahwa proses evaluasi untuk membuat keputusan tersebut seringkali tidak efisien dan terfragmentasi[2]. Pengembang dan peneliti harus menggunakan berbagai perangkat lunak terpisah untuk melakukan benchmarking kinerja, menjalankan uji statistik, dan memvisualisasikan data. Ketiadaan platform terpadu yang memungkinkan analisis komparatif secara holistik mencakup kinerja, kualitas statistik, dan visualisasi menciptakan sebuah kesenjangan (gap) signifikan. Untuk menjembatani kesenjangan ini, penelitian ini merancang dan mengimplementasikan sebuah dasbor interaktif berbasis web. Platform ini berfungsi sebagai laboratorium virtual untuk melakukan evaluasi komprehensif terhadap performa dan kualitas statistik dari berbagai jenis generator. Dengan menyediakan antarmuka yang intuitif dan terintegrasi, penelitian ini bertujuan untuk menyederhanakan proses evaluasi dan memberikan data empiris yang solid. Secara spesifik, studi ini menyajikan analisis komparatif mendalam terhadap tiga algoritma representatif, AES-CTR, ChaCha20, dan Mersenne Twister untuk memberikan wawasan yang jelas tentang kekuatan dan kelemahan masing-masing dalam berbagai skenario penggunaan algoritma. Berdasarkan latar belakang tersebut, penelitian ini menyajikan pendekatan baru dalam mengevaluasi algoritma PRNG menggunakan sistem interaktif berbasis web. Sistem ini mengintegrasikan fungsi kriptografi, statistik, dan visualisasi ke dalam satu antarmuka terpadu. Penelitian ini membandingkan tiga algoritma yang relevan, yakni AES-CTR: CSPRNG representatif yang menggunakan cipher blok AES yang distandarkan secara global, ChaCha20: CSPRNG representatif berdasarkan cipher aliran modern, yang diadopsi secara luas karena kinerjanya yang tinggi dalam implementasi perangkat lunak, Mersenne Twister: PRNG non-kriptografi representatif yang telah menjadi standar dalam aplikasi ilmiah dan statistik..Tujuan utama dari penelitian ini adalah untuk membangun sebuah dasbor interaktif sebagai platform terpadu untuk analisis kinerja, statistik, dan visualisasi algoritma PRNG, melakukan analisis komparatif mendalam terhadap kinerja (waktu, memori) dan skalabilitas dari ketiga generator, memvalidasi kualitas statistik output dari setiap generator menggunakan serangkaian uji standar. menyediakan alat bantu sumber terbuka yang dapat digunakan oleh komunitas untuk penelitian dan pendidikan.

2. METODE PENELITIAN

Penelitian ini dilakukan dengan pendekatan eksperimental konstruktif, di mana sebuah sistem interaktif dibangun untuk memecahkan masalah yang telah diidentifikasi.

2.1. Arsitektur Sistem dan Teknologi

Sistem dibangun sebagai aplikasi web interaktif menggunakan Streamlit. Arsitektur perangkat lunak didukung oleh tumpukan teknologi (*technology stack*) sebagai berikut:



Gambar 1. Disain Penelitian

2.2. Implementasi Generator

Tiga algoritma generator diimplementasikan dan dievaluasi berdasarkan **AES-CTR** yang memanfaatkan cipher blok AES-256 dalam mode Counter (CTR) untuk menghasilkan keystream pseudorandom dengan mengenkripsi nilai counter yang berurutan[3]. Keystream blok ke-*i*, dinotasikan sebagai O_i, dihasilkan dengan:

$O_i = E_K(\text{nonce}||\text{ctr}_i)$

Ciphertext (C_i) dihasilkan dengan melakukan operasi XOR antara plaintext (P_i) dan keystream (O_i). Proses dekripsi identik.

$C_i = P_i \oplus O_i$
$P_i = C_i \oplus O_i$
di mana:
Oi: Blok keystream (output) ke-*i*.
C*i*: Blok ciphertext ke-*i*.
P*i*: Blok plaintext ke-*i.*
E_K: Fungsi enkripsi AES dengan kunci *K*.
Nonce: *Number Used Once*, sebuah nilai acak unik untuk setiap sesi enkripsi.
ctr_i: Nilai *counter* untuk blok ke-i (misalnya, 0, 1, 2, ...).
||: Operasi konkatenasi (penggabungan bit).
⊕: Operasi bitwise XOR

1. ChaCha20

Sebuah stream cipher modern yang dirancang untuk performa tinggi pada CPU tanpa akselerasi perangkat keras. Inti dari ChaCha20 adalah fungsi **Quarter Round** yang beroperasi pada state 32-bit. Fungsi ini adalah operasi **Add-Rotate-XOR (ARX)**.
Fungsi Quarter Round: QR (a, b, c, d), Fungsi ini mengubah empat input (a,b,c,d) menjadi empat output baru.:
a = a + b; d = d ⊕ a; d = (d <<< 16)
c = c + d; b = b ⊕ c; b = (b <<< 12)
a = a + b; d = d ⊕ a; d = (d <<< 8)
c = c + d; b = b ⊕ c; b = (b <<< 7)
Output dari fungsi ini adalah (*a'',b'',c'',d'').*
di mana:
a,b,c,d: State internal (angka integer 32-bit).
⊞: Operasi penjumlahan modulo 2^{32}.
⊕: Operasi bitwise XOR.
⋘*n*: Operasi **rotasi bit ke kiri** sebanyak *n* posisi.
Proses ini diulang dalam beberapa putaran untuk menghasilkan aliran kunci yang sangat tersebar. Desain ChaCha20 memberikan ketahanan yang sangat baik terhadap kriptoa alisis sambil mempertahankan kinerja yang efisien dalam implementasi perangkat lunak.

2. Mersenne Twister (MT19937)

Sebuah PRNG klasik yang diimplementasikan melalui numpy.random.default_rng, digunakan sebagai baseline non-kriptografis. Algoritma ini didasarkan pada relasi rekurensi linier atas field biner[4]. Setiap state baru x_k dihasilkan dari state sebelumnya dengan rumus:

$$x_{k+n} = x_{k+m} \oplus \left(\left(x_k^{\text{upper}} \mid x_{k+1}^{\text{lower}} \right) A \right) \quad (1)$$

Output dari relasi rekurensi di atas kemudian ditempering yaitu mengubah angka dari internal state menjadi angka acak yang berkualitas tinggi untuk meningkatkan kualitas distribusinya sebelum diberikan sebagai hasil akhir (z).

$$\begin{aligned} y &\leftarrow x \oplus ((x \gg u) \;\&\; d) \\ y &\leftarrow y \oplus ((y \ll s) \;\&\; b) \\ y &\leftarrow y \oplus ((y \ll t) \;\&\; c) \\ z &\leftarrow y \oplus (y \gg l) \end{aligned} \quad (2)$$

di mana:
x_k: State ke-k dalam barisan (sebuah vektor 32-bit).
n,m: Parameter integer dari generator (untuk MT19937, n=624,m=397).
A: Matriks transformasi (twisting matrix) 32×32.

x_k^{upper}: Bit-bit paling signifikan dari x_k.
x_{k+1}^{lower}: Bit-bit paling tidak signifikan dari x_{k+1}.
u,s,t,l: Parameter pergeseran bit (bit shifts).
b,c,d: Bitmask untuk tempering.
⊕: Operasi bitwise XOR.
|: Operasi bitwise OR.
&: Operasi bitwise AND.
≫,≪: Operasi pergeseran bit ke kanan dan ke kiri.

2.3. Protokol Pengujian Kinerja.

Untuk setiap simulasi, dua metrik kinerja utama diukur berdasarkn waktu Eksekusi: diukur menggunakan time.perf_counter(), yang menyediakan jam monotonik dengan resolusi tertinggi yang tersedia pada sistem, sehingga ideal untuk mengukur interval waktu yang singkat secara akurat. Pengukuran dimulai tepat sebelum pemanggilan fungsi generator dan dihentikan tepat setelahnya, konsumsi Memori: dihitung menggunakan metrik RSS (Resident Set Size) dari modul (kode: psutil. Process(os.getpid()).memory_info().rss). Metrik ini dipilih karena secara akurat merepresentasikan porsi memori fisik (RAM) yang dialokasikan untuk proses pada saat pengukuran, tidak termasuk memori yang di swap, skalabilitas: Pengujian dilakukan secara iteratif pada berbagai ukuran output (1, 5, 10, 20, hingga 50 MB) untuk memodelkan hubungan antara beban kerja dan penggunaan sumber daya. Ukuran output yang besar dipilih untuk memperjelas perbedaan performa antar algoritma yang mungkin tertutup oleh *noise* sistem pada ukuran kecil.

2.4. Protokol Uji Statistik

Kualitas keacakan dari data output divalidasi melalui serangkaian uji: Pengujian pada ukuran data yang besar, misal: >1 MB penting karena distribusi byte akan lebih merata dan estimasi statistik menjadi lebih akurat dan representatif terhadap kualitas generator yang sebenarnya.

1. Distribusi Frekuensi Byte & Entropi Shannon

Frekuensi setiap nilai byte (0-255) dihitung. Dari distribusi probabilitas ini, Entropi Shannon (H(X)) dihitung untuk mengukur tingkat keacakan. Nilai yang mendekati 8.0 menunjukkan distribusi yang mendekati seragam sempurna dihitung menggunakan rumus:

$$H = -\sum_{i=0}^{255} p_i \cdot \log_2(p_i) \quad (3)$$

- di mana *pi* adalah probabilitas kemunculan byte ke-*i*.

Nilai yang mendekati 8.0 mengindikasikan distribusi yang sangat seragam dan acak.

2. Uji Chi-Square (chi2)

Uji goodness of fit ini secara kuantitatif membandingkan distribusi frekuensi byte yang diamati dengan distribusi seragam yang diharapkan. Nilai p-value yang tinggi (> 0.01) dari uji ini mengindikasikan tidak ada alasan untuk menolak hipotesis bahwa data terdistribusi secara seragam.

$$\chi^2 = \sum_{i=0}^{255} \frac{(O_i - E_i)^2}{E_i} \quad (4)$$

3. Uji Monobit

Uji frekuensi fundamental yang memastikan jumlah bit '0' dan '1' dalam aliran data berada dalam proporsi yang seimbang.

4. Deviasi standar distribusi frekuensi

Meskipun bukan uji keacakan primer, metrik ini dihitung untuk melihat sebaran frekuensi kemunculan byte di sekitar nilai rata-rata. Nilai yang rendah mengindikasikan bahwa frekuensi setiap byte tidak terlalu jauh dari ekspektasi.

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{256}(f_i - \mu_f)^2} \quad (5)$$

- di mana *fi* adalah frekuensi byte ke-*i* dan
- μf adalah rata-rata frekuensi.

Nilai deviasi standar yang lebih rendah menunjukkan bahwa frekuensi terdistribusi lebih merata.

Analisis Visual

Untuk melengkapi analisis kuantitatif, dua metode visualisasi kualitatif diimplementasikan:

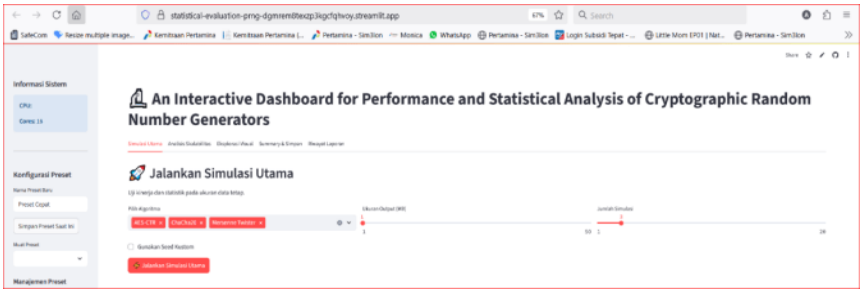
5. Citra Bitmap: Data biner dirender menjadi sebuah gambar grayscale. Output yang benar-benar acak akan menghasilkan citra yang tampak seperti *white noise* tanpa pola, struktur, atau tekstur yang dapat dikenali.
6. Plot Korelasi: Scatter plot dibuat dengan memetakan setiap byte B_n pada sumbu-x terhadap byte berikutnya Bn+1 pada sumbu-y. Pola apa pun pada plot ini (misalnya, garis diagonal, cluster) akan mengindikasikan adanya korelasi serial orde pertama yang tidak diinginkan.

3. HASIL DAN ANALISIS

Platform interaktif berhasil dikembangkan dan dapat diakses secara lokal. Menggunakan CLI untuk menjalankan aplikasi terlebih dahulu menggunakan url lokal: http://localhost: 8501.

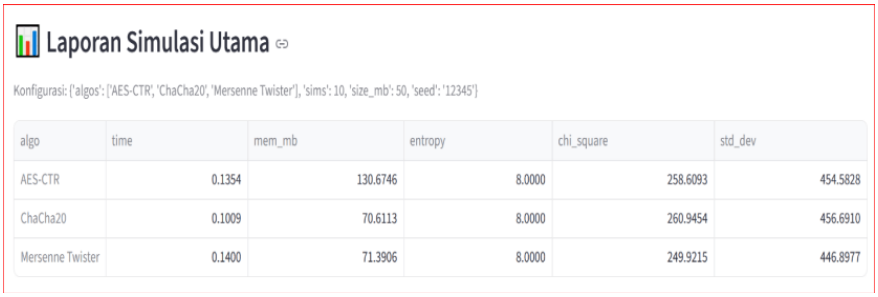
Gambar 2. CLI Interface For Running the Application Locally

Setelah tampil pada halaman utama, pengguna dapat memilih algoritma, menentukan ukuran data (1-50 MB), dan jumlah simulasi (1-20) untuk memulai pengujian. Hasilnya disajikan dalam beberapa laporan terstruktur. Tampilan Dashboard System adalah sebagai berikut:



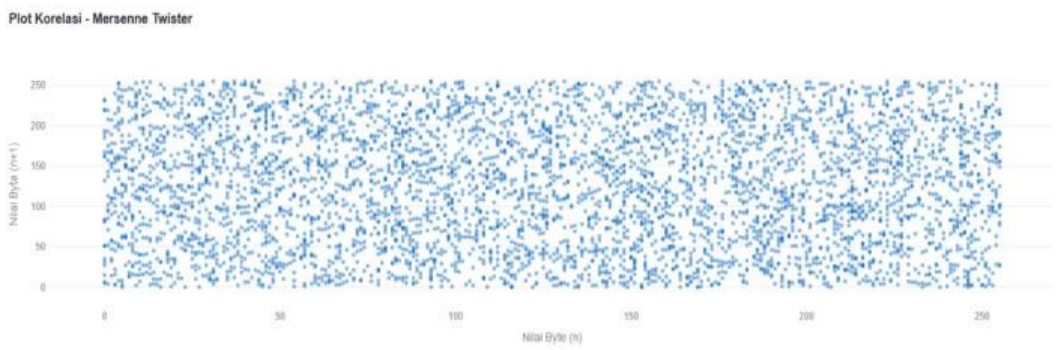
Gambar 3. Main Simulation Dashboard Display

Seluruh source code dan implementasi simulasi terkait penelitian ini dapat diakses di github pada alamat: https://statistical-evaluation-prng-dgmrem8texzp3kgcfqhvoy.streamlit.app. Pada halaman utama user dapat memahami secara intuisi dengan langsung memilih tipe algoritma yang akan dilakukan pengujian. Selain itu, platform ini memungkinkan pengguna untuk dapat menentukan parameter penting seperti ukuran data keluaran mulai dari sampel kecil hingga data berskala besar dan jumlah simulasi ingin dilakukan. Fleksibilitas ini mendukung pengujian eksplorasi cepat dan studi pembandingan yang lebih efisien, sehingga aplikasi ini cocok untuk tujuan pendidikan, penelitian, dan validasi kinerja dalam berbagai kebutuhan. Platform ini juga menggabungkan visualisasi interaktif dan laporan ringkasan secara rinci yang diperbarui secara real time saat simulasi dijalankan. Desain ini memungkinkan peneliti, pendidik, dan praktisi mampu mengamati dampak berbagai konfigurasi pada metrik kinerja dan properti statistik. Dengan menggabungkan pemilihan algoritma, pengaturan parameter, eksekusi, dan analisis hasil ke dalam alur kerja terintegrasi, sistem ini menyederhanakan proses evaluasi dan mendorong eksperimen dengan berbagai generator angka pseudorandom dalam kondisi yang konsisten. Laporan Simulasi Utama adalah sebagai berikut:



Gambar 4. Main Simulation Results Display

Laporan simulasi utama menunjukkan ringkasan performa untuk satu kali pengujian (output 50 MB dan jumlah simulasi sebanyak 20 kali) berdasarkan **ChaCha20** menjadi algoritma dengan rata-rata waktu proses tercepat, yaitu 0.1009 detik, **AES-CTR** menunjukkan penggunaan memori tertinggi (130.6746 MB), namun performa waktu tetap kompetitif (0.1354 detik), **Mersenne Twister** masih lebih unggul dalam efisiensi memori (71.3906 MB), namun memiliki waktu proses lebih lambat (0.1400 detik).
Plot Korelasi-Marsenne Twister



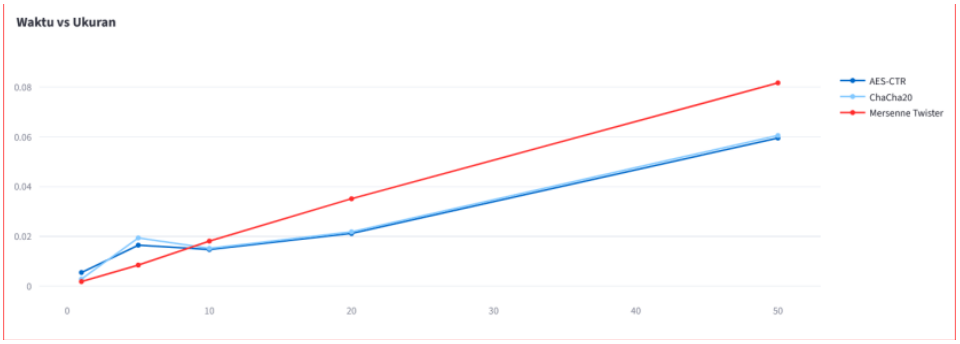
Gambar 5. Correlation Plots-MT19937

Laporan simulasi utama menunjukkan ringkasan performa untuk satu kali pengujian (misal, pada ukuran 50 MB): Untuk ketiga algoritma, plot korelasi byte menunjukkan sebaran titik yang acak dan merata tanpa adanya pola signifikan, mengonfirmasi tidak adanya korelasi serial orde pertama[2]. Semua algoritma menghasilkan nilai entropi mendekati ideal (8.0), yang mengindikasikan distribusi bit yang acak dan tidak dapat diprediksi.

3.1. Laporan Analisis Skalabilitas

Analisis skalabilitas dapat dijalankan secara paralel dengan ukuran data 1,10 dan 50 MB. Menghasilkan perbandingan antara penggunaan waktu vs ukuran data serta memori yang digunakan terhadap ukuran data.

Gambar 6. Waktu vs Ukuran AES-CTR, Cha-Cha20 and MT19937



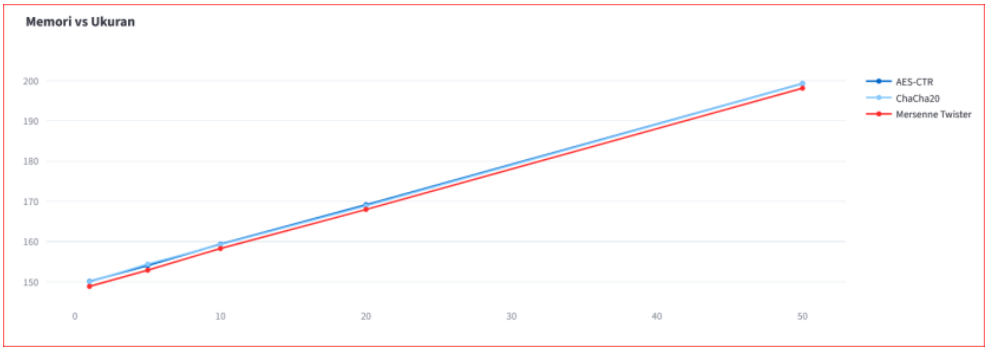
Gambar 7. Skalabilitas Waktu Eksekusi

3.2. Skalabilitas Waktu

Terlihat jelas adanya hubungan **linear** antara ukuran data dan waktu eksekusi. Pada ukuran data terbesar (50 MB), **AES-CTR** tetap menjadi yang tercepat. AES-CTR memiliki karakteristik linier terbaik terhadap pertambahan ukuran data, diikuti oleh ChaCha20. Mersenne Twister menunjukkan kurva pertumbuhan waktu yang lebih tajam.

3.3. Skalabilitas Memori

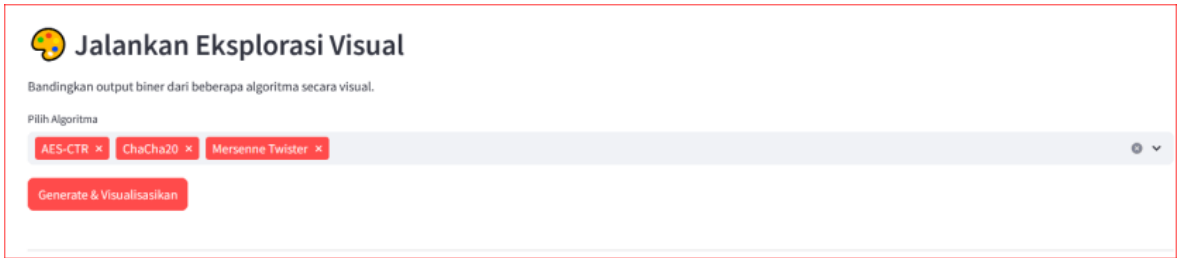
Grafik penggunaan memori menunjukkan semua algoritma memiliki jejak memori yang hampir identik dan berskala linear. Semua algoritma menunjukkan konsumsi memori yang meningkat secara linier dan hampir identik, mengindikasikan efisiensi implementasi.



Gambar 8. Memory Scalability

3.4. Laporan Eksplorasi Visual

Eksplorasi visual merepresentasikan output dari AES-CTR, ChaCha20, Mersenne Twister. Hasil yang tampak seperti 'noise' acak tanpa pola adalah konfirmasi visual dari data yang memiliki entropi tinggi.



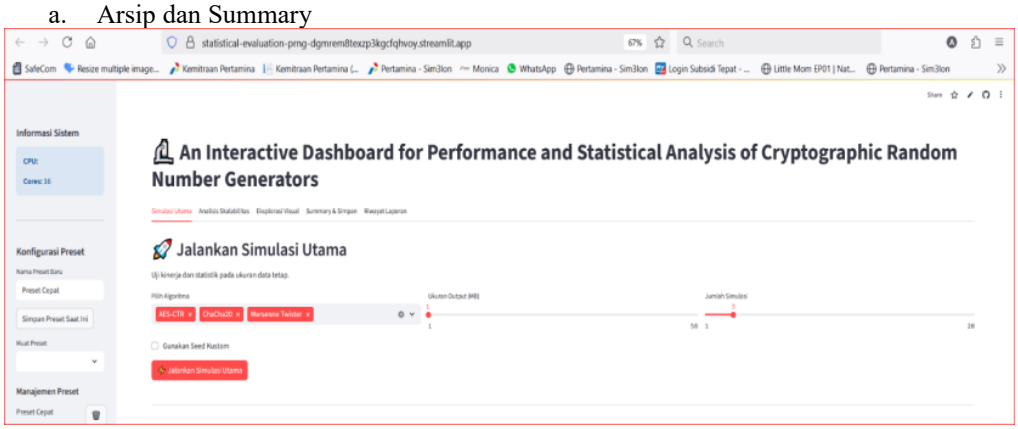
Gambar 9. Run Visual Exploration as noise

Gambar bitmap di berikut secara visual merepresentasikan output dari AES-CTR, ChaCha20, Mersenne Twister. Hasil yang tampak seperti 'noise' acak tanpa pola adalah konfirmasi visual dari data yang memiliki entropi tinggi. Visualisasi bitmap juga menampilkan pola noise acak yang konsisten, mendukung validitas statistik dari output CRNG.

Gambar 10. Bitmap Visualizations

3.5. Laporan Riwayat Pengujian

a. Arsip dan Summary



Gambar 11. History Report

Modul riwayat pengujian menyediakan arsip yang merangkum semua eksperimen yang dilakukan. Hasil dapat diekspor dalam format Excel untuk analisis eksternal lebih lanjut. Semua parameter dan output konfigurasi disimpan dalam database SQLite, yang memungkinkan fungsi muat ulang dan ekspor.

3.6. Kinerja Komputasi

Hasil benchmarking menunjukkan tren yang konsisten di seluruh pengujian. ChaCha20 secara signifikan mengungguli AES-CTR dan Mersenne Twister dalam hal kecepatan eksekusi. Hal ini sejalan dengan desain ChaCha20 yang dioptimalkan untuk operasi aritmatika sederhana pada CPU general-purpose. AES-CTR, meskipun sangat cepat, kinerjanya dapat lebih dioptimalkan pada sistem dengan instruksi AES-NI (hardware acceleration). Ketiga algoritma menunjukkan skalabilitas linear yang sempurna, di mana waktu eksekusi dan konsumsi memori meningkat secara proporsional dengan ukuran data yang dihasilkan.

3.7. Kualitas Statistik dan Visual

Semua generator yang diuji menunjukkan kualitas statistik yang sangat baik dan nyaris identik. Nilai Entropi Shannon secara konsisten mendekati nilai teoritis maksimal 8.0. Seluruh output berhasil melewati Uji Chi-Square dan Uji Monobit dengan margin yang besar, mengonfirmasi tidak adanya bias statistik yang terdeteksi. Hasil ini diperkuat oleh analisis visual, di mana citra bitmap dari ketiga generator tampak sebagai noise yang acak sempurna dan plot korelasi tidak menunjukkan adanya struktur tersembunyi. Temuan ini menyoroti konsep krusial: "kesesuaian untuk tujuan" (fitness for purpose). Meskipun Mersenne Twister menghasilkan output yang secara statistik berkualitas tinggi, ia tidak boleh digunakan untuk kriptografi karena sifatnya yang prediktif. Di sisi lain, ChaCha20 dan AES-CTR menyediakan jaminan keamanan yang diperlukan[2]. Pilihan antara ChaCha20 dan AES-CTR pada dasarnya adalah soal teknis. Praktiknya, untuk implementasi berbasis software, ChaCha20 terbukti menawarkan performa yang lebih cepat. Berdasarkan data empiris, untuk aplikasi yang diimplementasikan murni pada perangkat lunak tanpa akselerasi perangkat keras, ChaCha20 menunjukkan kinerja yang lebih unggul. Platform interaktif ini menyediakan sarana bagi para pengembang untuk tidak hanya sekedar mempelajari perbedaan teoretis, tetapi juga untuk melakukan pengujian

dan validasi secara langsung terhadap asumsi kinerja pada arsitektur yang digunakan. Hal ini memungkinkan untuk membuat keputusan yang lebih tepat dengan berlandaskan data.

4. KESIMPULAN

Penelitian ini berhasil mengembangkan sebuah platform interaktif yang fungsional dan komprehensif untuk analisis dan evaluasi generator bilangan acak. Sistem secara efektif mengintegrasikan tolok ukur kinerja, validasi statistik, dan eksplorasi visual ke dalam satu alur kerja yang intuitif. Ketiga algoritma yang diuji berhasil memenuhi syarat sebagai generator bilangan acak berkualitas tinggi dari sisi statistik. ChaCha20 adalah pilihan terbaik untuk kebutuhan performa cepat dan efisiensi memori[6]. AES-CTR unggul dalam kestabilan skalabilitas dan hasil distribusi yang kuat[7], sementara Mersenne Twister layak dipertimbangkan untuk aplikasi non-kriptografi yang memerlukan distribusi data yang merata dan akurat[4]. Platform yang dihasilkan memberikan kontribusi signifikan sebagai alat bantu praktis bagi komunitas akademisi dan industri untuk tujuan pendidikan, validasi algoritma, dan penelitian di bidang keamanan siber dan komputasi. Untuk pengembangan di masa depan, fungsionalitas sistem dapat diperluas dengan mengintegrasikan suite uji yang lebih lengkap dan menambahkan lebih banyak varian generator untuk dianalisis.

DAFTAR PUSTAKA

[1] W. Zhao, Z. Chang, C. Ma, and Z. Shen, “A Pseudorandom Number Generator Based on the Chaotic Map and Quantum Random Walks,” *Entropy*, vol. 25, no. 1, pp. 1–25, 2023, doi: 10.3390/e25010166.

[2] K. P. Widiatmika, “No 主観的健康感を中心とした在宅高齢者における 健康関連指標に関する共分散構造分析Title,” *Etika Jurnalisme Pada Koran Kuning Sebuah Stud. Mengenai Koran Lampu Hijau*, vol. 16, no. 2, pp. 39–55, 2015.

[3] L. Malliga, D. L. S. Kavya, G. Jahnavi, G. Vanajakshi, and E. Rachana, “A New Secure Data Hiding AES-CTR Key Modulation,” vol. 23, no. 12, pp. 15–22, 2023.

[4] F. Cannizzo, “”VMT19937: A SIMD-Friendly Pseudo Random Number Generator based on Mersenne Twister 19937”,” pp. 1–13, 2023.

[5] M. B. Imtiaz and R. Kamran, “Mitigating Transmission Errors: A Forward Error Correction-Based Framework for Enhancing Objective Video Quality,” *Sensors*, vol. 25, no. 11, p. 3503, 2025, doi: 10.3390/s25113503.

[6] O. Sabri, B. Al-Shargabi, A. Abuarqoub, and T. A. Hakami, “A Lightweight Encryption Method for IoT-Based Healthcare Applications: A Review and Future Prospects,” *IoT*, vol. 6, no. 2, p. 23, 2025, doi: 10.3390/iot6020023.

[7] S. S. Lee, J. S. Moon, Y. J. Choi, D. Kim, and S. Lee, “DTR-SHIELD: Mutual Synchronization for Protecting against DoS Attacks on the SHIELD Protocol with AES-CTR Mode,” *Sensors*, vol. 24, no. 13, pp. 1–19, 2024, doi: 10.3390/s24134163.

[8] I. Zerraza, “Lightweight Authentication for IOT Edge Devices Authentication scheme,” vol. 48, pp. 15–20, 2024.

[9] C. P. Roca *et al.*, “A cross entropy test allows quantitative statistical comparison of t-SNE and UMAP representations,” *Cell Reports Methods*, vol. 3, no. 1, 2023, doi: 10.1016/j.crmeth.2022.100390.

[10] M. Irfan and M. A. Khan, “Cryptographically Secure Pseudo-Random Number Generation (CS-PRNG) Design using Robust Chaotic Tent Map (RCTM),” pp. 1–11, 2024, [Online]. Available: http://arxiv.org/abs/2408.05580

[11] Y. Akkem, B. S. Kumar, and A. Varanasi, “Streamlit Application for Advanced Ensemble Learning Methods in Crop Recommendation Systems – A Review and Implementation,” *Indian J. Sci. Technol.*, vol. 16, no. 48, pp. 4688–4702, 2023, doi: 10.17485/ijst/v16i48.2850.

[12] O. Valikhanli and F. Abdullayeva, “Securing UAV Flight Data Using Lightweight Cryptography and Image Steganography,” *Int. J. Adv. Comput. Sci. Appl.*, vol. 16, no. 5, pp. 278–289, 2025, doi: 10.14569/IJACSA.2025.0160527.

[13] P. Singh, P. Pranav, and S. Dutta, “Optimizing cryptographic protocols against side channel attacks using WGAN-GP and genetic algorithms,” *Sci. Rep.*, vol. 15, no. 1, p. 86118, 2025, doi: 10.1038/s41598-025-86118-4.

[14] H. Park and J. Lee, “LMSA: A Lightweight Multi-Key Secure Aggregation Framework for Privacy-Preserving Healthcare AIoT,” *C. - Comput. Model. Eng. Sci.*, vol. 143, no. 1, pp. 827–847, 2025, doi: 10.32604/cmes.2025.061178.

[15] D. Berger, M. Lemoudden, and W. J. Buchanan, “Post-Quantum Migration of the Tor Application,” *J. Cybersecurity Priv.*, vol. 5, no. 2, p. 13, 2025, doi: 10.3390/jcp5020013.

[16] H. A. Park, “The DtMin Protocol: Implementing Data Minimization Principles in Medical Information Sharing,” *Electron.*, vol. 14, no. 8, 2025, doi: 10.3390/electronics14081501.