JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 9, Nomor 1, Januari 2025, Page 37-42
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI 10.30865/mib.v9i1.8154

Implementasi Permuted Congruential Generator untuk Pembangkitan Kunci pada Enkripsi Berbasis One Time Pad

Taronisokhi Zebua
Teknologi Informasi,Fakultas Ilmu Komputer & Teknologi Informasi, Universitas Budi Darma, Medan, Indonesia
Email: yunaze2018@gmail.com
Email korespondensi: yunaze2018@gmail.com

Abstrak- Pembangkitan kunci merupakan komponen penting dalam implementasi sistem enkripsi, khususnya pada mekanisme penyandian yang membutuhkan deret kunci sepanjang pesan. Penelitian ini mengimplementasikan Permuted Congruential Generator (PCG) sebagai pembangkit deret byte yang digunakan dalam operasi penyandian aditif modulo 256 menyerupai One Time Pad (OTP). Prosedur mencakup pengubahan teks ke representasi UTF-8, inisialisasi PCG32, pembangkitan deret kunci, enkripsi, dekripsi, dan pengujian statistik deskriptif. Parameter verifikasi menggunakan state 64-bit, multiplier 6364136223846793005, dan increment 1442695040888963407. Lima pesan utama dengan panjang 22–73 byte berhasil dikembalikan secara identik setelah dekripsi. Sebanyak 100 uji tambahan dengan ukuran 16–1.024 byte juga menghasilkan 100% keberhasilan dekripsi; seluruh pasangan seed berbeda pada pengujian tersebut menghasilkan deret keluaran berbeda. Pada sampel kunci 256–8.192 byte, entropy teramati meningkat dari 7,1545 menjadi 7,9758 bit per byte. Hasil ini membuktikan konsistensi prosedural pada skenario yang diuji, bukan keamanan kriptografisnya. PCG merupakan pseudorandom number generator deterministik untuk penggunaan umum, sehingga deret kuncinya tidak dapat diasumsikan setara dengan kunci OTP ideal yang benar-benar acak dan rahasia. Kontribusi penelitian dibatasi pada implementasi dan evaluasi fungsional yang dapat diperiksa ulang; implementasi keamanan nyata memerlukan cryptographically secure pseudorandom number generator dan analisis manajemen kunci.

Kata Kunci: Permuted Congruential Generator; One Time Pad; pembangkitan kunci; kriptografi; pseudorandom number generator

Abstract- Key generation is a critical component of encryption implementations, particularly mechanisms requiring a key stream as long as the protected message. This study implements a Permuted Congruential Generator (PCG) to produce byte sequences used in an additive modulo-256 encryption scheme resembling the One Time Pad (OTP) operation. The procedure includes converting plaintext to UTF-8 bytes, initializing PCG32, generating a key stream, performing encryption and decryption, and examining descriptive sequence statistics. The verification configuration uses a 64-bit state, a multiplier of 6364136223846793005, and an increment of 1442695040888963407. All five primary messages, containing 22 to 73 bytes, were correctly recovered by decryption. An additional 100 tests with message sizes ranging from 16 to 1,024 bytes also achieved complete recovery; different seeds yielded different tested streams in all 100 comparisons. Observed byte entropy increased from 7.1545 to 7.9758 bits per byte for samples of 256 to 8,192 bytes. These results demonstrate functional consistency for the reported test cases, not cryptographic security. PCG is a deterministic general-purpose pseudorandom generator; its output cannot be considered equivalent to the truly random, secret key required for an ideal OTP. The article therefore positions the implementation as a reproducible functional experiment rather than a secure OTP construction. Deployment for actual confidentiality would require a cryptographically secure generator, carefully specified seed handling, and a separate security assessment.

Keywords: Permuted Congruential Generator; One Time Pad; key generation; cryptography; pseudorandom number generator

1. PENDAHULUAN

Keamanan pertukaran informasi digital bergantung pada pemilihan algoritma dan pengelolaan material kunci. Pesan yang diproses menjadi ciphertext belum tentu terlindungi apabila pihak yang tidak berwenang dapat memperoleh atau menebak kunci yang digunakan. Pada algoritma penyandian berbasis deret kunci, proses pembangkitan serta pemakaian kunci menjadi bagian yang harus dijelaskan secara eksplisit. Penelitian terdahulu mengenai pengamanan pesan dan data menunjukkan variasi pendekatan simetris serta mekanisme pembentukan kunci [1]–[4], [7]–[12].

One Time Pad ideal mempunyai karakteristik keamanan informasi-teoretis apabila kuncinya benar-benar acak, rahasia, sepanjang plaintext, dan tidak pernah dipakai ulang. Prinsip tersebut berakar pada kajian teori kerahasiaan Shannon [15]. Operasi penjumlahan modular atau XOR tidak cukup untuk menyebut sebuah implementasi sebagai OTP yang aman apabila kunci dibentuk dari deret deterministik berentropi rendah. Perbedaan antara operasi enkripsi dan asumsi keamanan ini penting untuk mencegah klaim yang terlalu luas.

Deretan pseudorandom dapat dibentuk dengan generator deterministik yang menerima seed dan mempertahankan state internal. Permuted Congruential Generator (PCG) menggabungkan pembaruan state kongruensial dan transformasi permutasi keluaran sehingga menghasilkan sifat statistik yang baik untuk berbagai keperluan komputasi. O'Neill memperkenalkan keluarga PCG sebagai metode pembangkitan bilangan acak semu yang efisien [13]. Namun, kualitas statistik yang baik tidak identik dengan sifat tahan prediksi yang disyaratkan pada pembangkit kunci untuk aplikasi keamanan.

Submitted: **01/12/2024**; Accepted: **27/12/2024**; Published: **28/01/2025**

JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 9, Nomor 1, Januari 2025, Page 37-42
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI 10.30865/mib.v9i1.8154

Standar NIST SP 800-90A mendeskripsikan mekanisme deterministic random bit generators berbasis primitif kriptografi [14]. Literatur evaluasi keacakan juga membedakan uji statistik dari penilaian keamanan generator [16], [17]. Uji distribusi byte, entropy, dan korelasi berguna untuk menggambarkan karakteristik keluaran, tetapi tidak dengan sendirinya membuktikan ketahanan terhadap rekonstruksi state atau penebakan seed. Dengan demikian, interpretasi eksperimen PCG memerlukan batasan klaim yang jelas.

Penelitian sumber membahas pembangkitan kunci PCG untuk operasi OTP dan menguji konsistensi enkripsi–dekripsi, perbedaan seed, entropy, keunikan byte, serta korelasi serial. Di dalam proposal awal yang melandasi laporan terdapat penyebutan Lagged Fibonacci Generator pada sebagian kajian pustaka, sedangkan judul dan implementasi berfokus pada PCG. Artikel ini mempertahankan PCG sebagai objek kajian, menyelaraskan terminologi, dan menyajikan hasil pada batas yang benar-benar didukung laporan.

Celah kajian yang ditargetkan bukan pembuktian bahwa PCG lebih aman daripada generator kriptografis, melainkan dokumentasi terstruktur hubungan konfigurasi PCG32, panjang deret kunci, operasi modulo 256, dan hasil pengujian fungsional. Kontribusi yang dapat ditunjukkan adalah prosedur verifikasi yang terperinci dan pembacaan kritis atas batas interpretasi statistik. Penelitian tidak mengklaim keunggulan performa terhadap algoritma lain karena data komparatif tidak tersedia.

Tujuan penelitian adalah menjelaskan integrasi PCG32 pada pembangkitan keystream, memverifikasi pemulihan plaintext pada sejumlah ukuran pesan, dan mendeskripsikan statistik keluaran yang tersedia. Artikel juga mengidentifikasi risiko dari penggunaan generator nonkriptografis dalam konstruksi yang diberi label OTP. Penilaian tersebut diperlukan agar pemanfaatan studi pada pendidikan atau pengembangan prototipe tidak disalahartikan sebagai rekomendasi produksi untuk perlindungan informasi sensitif.

2. METODOLOGI PENELITIAN

2.1 Tahapan Penelitian

Tahapan penelitian terdiri atas identifikasi kebutuhan keamanan kunci, studi literatur, penetapan data dan parameter, pembangkitan byte PCG, operasi enkripsi, operasi dekripsi, serta pengujian fungsional dan statistik. Rangkaian ini ditunjukkan pada Gambar 1. Hasil pengujian dianalisis secara deskriptif. Karena laporan sumber tidak menyertakan kode program, data mentah lengkap, dan prosedur pengulangan secara independen, angka pengujian dilaporkan sebagai temuan sumber yang belum diaudit ulang menggunakan eksekusi terpisah.



Gambar 1. Tahapan Penelitian PCG dan Penyandian Modulo 256

2.2 Konfigurasi Generator dan Pembentukan Kunci

Penelitian menggunakan konfigurasi verifikasi PCG32 dengan state 64-bit. Nilai multiplier dan increment dipertahankan sebagaimana tercatat di laporan. Algoritma diinisialisasi dengan seed yang ditentukan untuk setiap skenario. Keluaran 32-bit dipecah menjadi sejumlah byte sampai panjang deret kunci setara dengan panjang

Submitted: **01/12/2024**; Accepted: **27/12/2024**; Published: **28/01/2025**

JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 9, Nomor 1, Januari 2025, Page 37-42
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI 10.30865/mib.v9i1.8154

plaintext dalam representasi UTF-8. Karena seed dan parameter sama akan membentuk keluaran deterministik, proses dekripsi wajib memperoleh keystream yang identik.

Tabel 1. Konfigurasi Generator untuk Verifikasi

Parameter	Nilai
Varian	PCG32
State	64-bit
Multiplier	6364136223846793005
Increment	1442695040888963407
Keluaran	32-bit, dipecah menjadi byte
Seed	Bervariasi menurut skenario
Panjang kunci	Sama dengan jumlah byte plaintext

2.3 Operasi Enkripsi dan Dekripsi

Untuk setiap posisi byte i, nilai plaintext Pi, byte kunci Ki, dan ciphertext Ci berada pada himpunan bilangan 0 hingga 255. Enkripsi menggunakan Ci = (Pi + Ki) mod 256, sedangkan dekripsi menggunakan Pi = (Ci − Ki + 256) mod 256. Kedua persamaan tersebut menjelaskan inversi modular yang memungkinkan pemulihan byte. Kesuksesan dekripsi terutama memverifikasi konsistensi implementasi dua operasi dan kesamaan keystream, bukan kualitas kerahasiaan atau resistansi terhadap serangan.

2.4 Skenario dan Metrik Pengujian

Pengujian utama mencakup lima pesan dengan panjang 22, 23, 42, 58, dan 73 byte serta seed yang berbeda. Pengujian tambahan berjumlah 100 skenario dengan ukuran 16 sampai 1.024 byte. Hasil dicatat berdasarkan kecocokan plaintext sebelum enkripsi dengan plaintext setelah dekripsi. Penelitian sumber juga membandingkan perubahan deret pada seed berbeda. Untuk deret kunci 256, 1.024, 4.096, dan 8.192 byte, dihitung entropy Shannon, jumlah nilai byte unik, dan korelasi serial sederhana.

Entropy empiris dihitung dengan $H = -\Sigma\, p(b) \log_2 p(b)$, dengan p(b) merupakan frekuensi relatif byte b. Untuk byte delapan bit, batas maksimum teoritisnya adalah 8 bit per byte. Kenaikan entropy sampel dapat dipengaruhi ukuran data dan tidak boleh dianggap sebagai jaminan ketidakmampuan adversary memprediksi deret. Uji keamanan formal, pengukuran periode, deteksi hubungan antarstream, dan penilaian kebocoran state berada di luar data yang disediakan.

2.5 Ancaman Validitas dan Batas Analisis

Secara metodologis, laporan menyajikan tabel keluaran sebagian keystream dan ciphertext, bukan berkas data atau kode yang dapat dieksekusi langsung. Oleh sebab itu, artikel ini tidak menambah skor keamanan, tingkat resistansi terhadap brute force, maupun hasil uji NIST yang tidak dilaporkan. Seed demonstrasi juga bukan secret key berkualitas kriptografi. Tujuan pengujian didefinisikan sebagai verifikasi operasional pada contoh, bukan pembuktian perfect secrecy, keamanan semantik, ataupun kelayakan deployment.

3. HASIL DAN PEMBAHASAN

3.1 Implementasi pada Lima Pesan

Tabel 2 memuat lima pengujian utama yang berasal dari laporan. Seluruh pesan diproses sebagai byte UTF-8 sehingga panjang kunci mengacu pada banyaknya byte, bukan jumlah karakter. Setiap baris memperlihatkan seed, cuplikan 12 byte awal kunci dan ciphertext, serta status pemulihan pesan. Lima status “Sesuai” menunjukkan bahwa keluaran dekripsi identik dengan byte masukan pada kelima skenario yang dicatat.

Tabel 2. Pengujian Enkripsi dan Dekripsi pada Lima Pesan

No	Byte	Seed	12 Byte Kunci (Hex)	12 Byte Cipher (Hex)	Hasil
1	22	20250801	1D51227251EBB5DAB1B2EC6B	729F6BC8963D082305F33F8B	Sesuai
2	23	20250802	4D5FD78BB820CB0CD1C4B3BC	91A02BCCD870105A1610FC10	Sesuai
3	42	20250803	0440F11CEE6403F579CE016F	4F8532692FB2444399174FB5	Sesuai
4	58	20250804	F0DFA1523322150345BD6C9E	4024EF978563654493DDBBEC	Sesuai
5	73	20250805	FA931437866657C2D3003EBE	43E15A86D8B398151C207F09	Sesuai

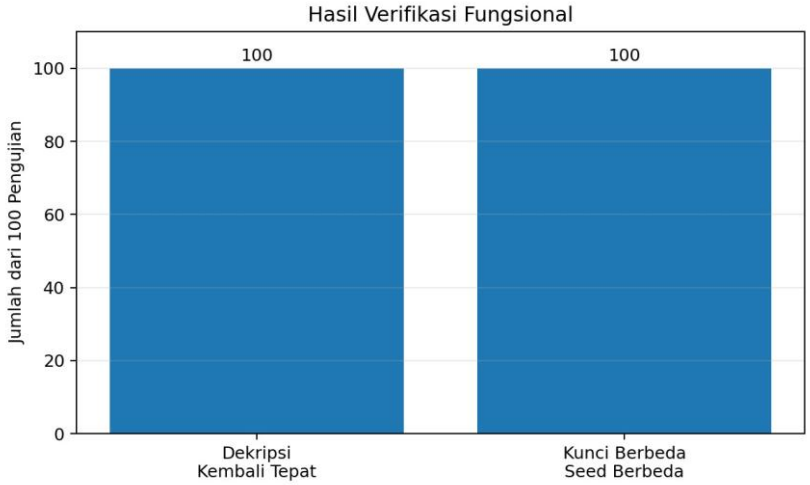
Kesesuaian hasil dapat dipahami dari sifat invers operasi modular: penambahan kunci pada tahap enkripsi dibatalkan oleh pengurangan kunci pada tahap dekripsi apabila byte kunci yang digunakan sama. Keberhasilan ini juga mengindikasikan panjang kunci yang dihasilkan cukup untuk mengimbangi seluruh byte masukan. Akan tetapi, kesalahan pengelolaan seed pada implementasi nyata dapat menyebabkan tidak sinkronnya urutan byte dan kegagalan pemulihan, bahkan ketika formula matematis yang digunakan benar.

Submitted: **01/12/2024**; Accepted: **27/12/2024**; Published: **28/01/2025**

JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 9, Nomor 1, Januari 2025, Page 37-42
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI 10.30865/mib.v9i1.8154

3.2 Pengujian Berulang dan Variasi Seed

Pada pengujian tambahan, laporan mencatat 100 dari 100 skenario berhasil dikembalikan ke plaintext asli. Demikian pula, 100 dari 100 skenario pembandingan seed menghasilkan deret yang berbeda. Persentase 100% dalam kedua pengujian adalah hasil pada himpunan uji yang diberikan, bukan estimasi probabilitas keberhasilan yang berlaku universal. Tidak tersedia ukuran variasi seed, pemilihan sampel acak yang terdokumentasi, ataupun kode yang memungkinkan audit independen atas seluruh 100 perbandingan.



Gambar 2. Ringkasan Verifikasi Fungsional Berdasarkan Laporan

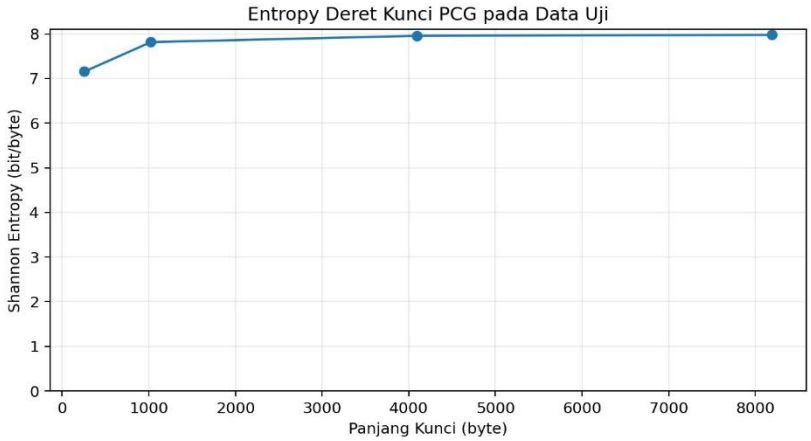
Variasi seed menghasilkan keluaran berbeda merupakan perilaku yang diharapkan dari generator deterministik. Namun perbedaan dua deret tidak mengukur tingkat kesulitan menebak seed. Jika ruang seed terbatas atau seed dapat diketahui pihak lain, seluruh keystream dapat dihasilkan ulang. Ketahanan terhadap tebakan juga bergantung pada cara seed dibangkitkan, disimpan, didistribusikan, dan dirotasi—faktor yang belum diuji dalam laporan.

3.3 Karakteristik Statistik Deret Kunci

Tabel 3 memperlihatkan ringkasan statistik untuk empat panjang sampel yang dilaporkan. Pada 256 byte, entropy empiris 7,1545 bit per byte dengan 159 nilai unik. Pada sampel 8.192 byte, entropy meningkat menjadi 7,9758 bit per byte dan seluruh 256 kemungkinan nilai byte teramati. Nilai korelasi serial yang rendah pada sebagian sampel mengindikasikan hubungan linear sederhana antartetangga tidak dominan dalam statistik tersebut, tanpa meniadakan kemungkinan pola tingkat lanjut.

Tabel 3. Statistik Deskriptif Deret Kunci PCG

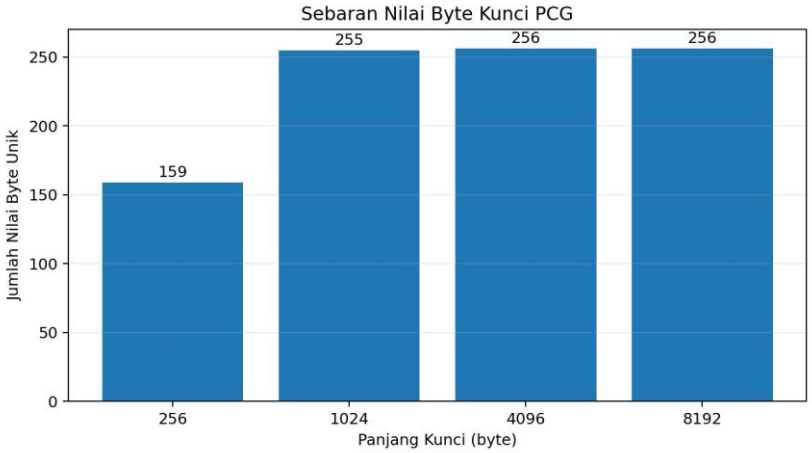
Panjang (byte)	Entropy (bit/byte)	Byte Unik	Korelasi Serial
256	7,1545	159	0,0345
1.024	7,8148	255	0,0707
4.096	7,9556	256	0,0012
8.192	7,9758	256	−0,0164



Gambar 3. Entropy Shannon pada Empat Panjang Sampel

Submitted: **01/12/2024**; Accepted: **27/12/2024**; Published: **28/01/2025**

JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 9, Nomor 1, Januari 2025, Page 37-42
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI 10.30865/mib.v9i1.8154



Gambar 4. Jumlah Nilai Byte Unik pada Empat Panjang Sampel

Naiknya entropy teramati ketika panjang sampel bertambah konsisten dengan semakin lengkapnya frekuensi kategori byte dalam sampel. Meski demikian, nilai entropy mendekati 8 tidak sama dengan sumber acak sempurna. Generator deterministik memiliki state yang menentukan setiap keluaran berikutnya; apabila parameter internal atau seed dapat direkonstruksi, penyerang mungkin memprediksi keluaran berikutnya terlepas dari hasil statistik satu sampel. Standar evaluasi generator untuk keamanan memerlukan desain dan asumsi yang berbeda [14], [16], [17].

3.4 Interpretasi Ketepatan Istilah One Time Pad

Secara operasional, penelitian memanfaatkan aturan penjumlahan modular yang serupa dengan konstruksi OTP berbasis byte. Namun, kunci dihasilkan oleh PCG dari seed sehingga seluruh deret merupakan fungsi deterministik dari informasi awal tersebut. Konstruksi itu lebih tepat dinamai penyandian berbasis keystream PCG atau simulasi operasi OTP daripada OTP ideal. Hal ini penting karena klaim keamanan OTP ideal bertumpu pada kunci yang benar-benar acak, rahasia, sepanjang pesan, dan tidak digunakan kembali [15], [18].

Penggunaan ulang keystream pada lebih dari satu pesan dapat membocorkan hubungan antarpesan. Untuk penyandian aditif modulo 256, informasi yang terungkap berbeda bentuknya dari XOR dua ciphertext, tetapi prinsip yang sama berlaku: keamanan bergantung pada kerahasiaan dan kebaruan material kunci, bukan sekadar keberhasilan inversi matematis. Pengujian laporan belum mencakup eksperimen reuse, penilaian entropi seed, ancaman adversary yang mengamati beberapa ciphertext, dan analisis kompromi state.

3.5 Pembahasan terhadap Literatur dan Kontribusi

Penelitian sumber mengenai modifikasi pembangkitan kunci pada kriptografi, penggunaan OTP untuk teks, dan pembangkit generator lain memberikan dasar kontekstual [3], [4], [9], [11], [12]. Studi PCG [13] menjelaskan tujuan performa dan kualitas statistik keluarga generator. Akan tetapi, standar generator untuk aplikasi keamanan menekankan konstruksi yang dirancang untuk konteks kriptografi [14]. Perbandingan tersebut menegaskan bahwa penelitian fungsional PCG dan persyaratan implementasi enkripsi aman harus diperlakukan sebagai dua pertanyaan ilmiah yang berbeda.

Novelty artikel dibatasi pada demonstrasi integrasi PCG32 dengan penyandian modulo 256 dalam konfigurasi dan kumpulan pengujian sumber, disertai eksplisitnya pemisahan antara keberhasilan dekripsi dan keamanan kriptografis. Hasil ini tidak membuktikan keunggulan dibandingkan generator lain, tidak menyediakan dataset serangan, dan belum menunjukkan pengujian performa komputasi. Untuk penyempurnaan publikasi ilmiah, peneliti perlu menyertakan kode, input contoh, konfigurasi PCG yang tidak ambigu, dan prosedur pembangkitan seed agar eksperimen dapat direplikasi.

3.6 Rancangan Pengujian Lanjutan

Pengembangan lanjutan perlu menggunakan vektor uji keluaran PCG yang telah diverifikasi dan data input dengan representasi byte yang berbeda, termasuk teks Unicode, file biner, pesan kosong, dan ukuran panjang. Pemulihan plaintext sebaiknya diperiksa pada setiap ukuran yang didukung. Perbandingan sumber acak umum, PCG, dan CSPRNG dapat menggunakan tingkat pengulangan yang sama; keluaran uji yang dilaporkan harus meliputi waktu pembangkitan, throughput, penggunaan memori, karakteristik statistik, dan ancaman prediksi.

Submitted: **01/12/2024**; Accepted: **27/12/2024**; Published: **28/01/2025**

JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 9, Nomor 1, Januari 2025, Page 37-42
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI 10.30865/mib.v9i1.8154

Jika penelitian dimaksudkan sebagai konstruksi keamanan sesungguhnya, mekanisme yang direkomendasikan adalah menggunakan primitive standar dan manajemen kunci yang sesuai, bukan menjadikan PCG sebagai keystream rahasia. NIST memberikan rekomendasi khusus mengenai deterministic random bit generators untuk konteks kriptografi [14]. Pengembangan sistem juga perlu memperhatikan autentikasi ciphertext, nonce atau IV yang benar, dan larangan penggunaan ulang keystream yang sama. Semua langkah tersebut merupakan saran dan bukan hasil implementasi pada penelitian sumber.

4. KESIMPULAN

Penelitian menunjukkan bahwa deret byte hasil Permuted Congruential Generator dapat digunakan secara fungsional sebagai kunci pada prosedur penjumlahan dan pengurangan modulo 256. Lima skenario utama dengan ukuran 22 sampai 73 byte serta 100 skenario tambahan dengan ukuran 16 sampai 1.024 byte dilaporkan berhasil mengembalikan plaintext hasil dekripsi secara identik. Perubahan seed pada 100 skenario yang diuji juga menghasilkan deret berbeda. Untuk panjang sampel 256 hingga 8.192 byte, Shannon entropy meningkat dari 7,1545 menjadi 7,9758 bit per byte, sementara jumlah nilai byte unik bertambah dari 159 menjadi 256. Angka tersebut merupakan ukuran pengamatan statistik dan konsistensi proses, bukan bukti keamanan kunci. Karena PCG adalah generator deterministik untuk penggunaan umum, hasilnya tidak boleh dinyatakan memenuhi syarat One Time Pad ideal yang memerlukan kunci sungguh-sungguh acak, sepanjang pesan, rahasia, dan digunakan hanya sekali. Kontribusi utama naskah terletak pada penyajian tahapan implementasi, hasil uji, dan batas interpretasi metodologis. Sebelum diterapkan pada perlindungan informasi yang benar-benar sensitif, perlu dilakukan perbandingan terhadap CSPRNG, pengujian keamanan yang sesuai, verifikasi data dan kode eksperimen, serta perancangan manajemen seed dan kunci. Dengan demikian, hasil penelitian paling tepat ditempatkan sebagai demonstrasi pendidikan dan pengujian prosedural pembangkitan keystream, bukan sebagai produk kriptografi siap digunakan.

REFERENCES

[1] D. A. Fauzan et al., “Perancangan Aplikasi Pengamanan Data Menggunakan Algoritma RSA dan AES Berbasis Web,” Journal of Information and Information Security, vol. 4, no. 1, pp. 91–104, 2023. [Periksa kelengkapan nama penulis]
[2] R. Rivaldi et al., “Implementasi Pengamanan Data Arsitektur Menggunakan Metode Kriptografi dengan Algoritma RC4,” SKANIKA, vol. 4, no. 2, pp. 139–143, 2021. [Periksa kelengkapan nama penulis]
[3] M. Sianturi and T. Zebua, “Modifikasi Pembangkit Kunci Algoritma Vigenere Cipher Berdasarkan Pembangkit Bilangan Acak CSPRNG untuk Mengamankan Citra Digital,” JUSSI, vol. 2, no. 4, pp. 136–142, 2023.
[4] H. Alam, A. K. Habibi, and H. Widya, “Penggunaan Algoritma Vegenere Cipher dan One Time Pad untuk Keamanan Pesan Teks,” SEMNASTEK-UISU, 2022.
[5] I. Sunni, “Aplikasi Lagged Fibonacci Generator dan Bilangan Irrasional dalam Stream Cipher,” Institut Teknologi Bandung, 2011.
[6] Junianto et al., “Penerapan Algoritme Lagged Fibonacci Generator pada Edugame Puzzle Pengenalan Hewan Berbasis Android,” STMIK MDP, tahun tidak dicantumkan pada sumber. [Metadata perlu dilengkapi]
[7] N. M. S. Sianturi, N. B. Nugroho, and W. R. Maya, “Implementasi Kriptografi untuk Pengamanan Data Aset Perusahaan ... AES 192,” Jurnal CyberTech, 2020.
[8] D. H. Pane, “Implementasi Kriptografi Keamanan Data Resi pada PT JNE Perbaungan Menggunakan Metode Merkle Hellman,” Device, vol. 1, no. 1, pp. 6–10, 2020.
[9] T. Zebua, “Penerapan Multiply With Carry Generator pada Proses Pembangkitan Kunci Algoritma Beaufort Cipher,” Journal of Information System Research, vol. 4, no. 2, pp. 607–613, 2023.
[10] M. Z. Solihin et al., “Implementasi Kriptografi Menggunakan Metode Algoritma RSA pada Aplikasi Pengamanan Data Berbasis Java Desktop,” SENAFTI, 2022.
[11] M. K. Harahap and N. Khairina, “Analisis Algoritma One Time Pad dengan Algoritma Cipher Transposisi sebagai Pengamanan Pesan Teks,” Sinkron, vol. 1, no. 2, pp. 58–62, 2017.
[12] S. A. Saputri et al., “Penerapan Algoritma One Time Pad pada Pesan Rahasia Menggunakan QR Code Berbasis Android,” Jurnal FTIK, vol. 1, no. 1, pp. 1193–1206. [Tahun dan detail penulis perlu diverifikasi]
[13] M. E. O’Neill, “PCG: A Family of Simple Fast Space-Efficient Statistically Good Algorithms for Random Number Generation,” Harvey Mudd College, Tech. Rep. HMC-CS-2014-0905, 2014.
[14] E. Barker and J. Kelsey, “Recommendation for Random Number Generation Using Deterministic Random Bit Generators,” NIST SP 800-90A Rev. 1, 2015, doi: 10.6028/NIST.SP.800-90Ar1.
[15] C. E. Shannon, “Communication Theory of Secrecy Systems,” Bell System Technical Journal, vol. 28, no. 4, pp. 656–715, 1949, doi: 10.1002/j.1538-7305.1949.tb00928.x.
[16] A. Rukhin et al., “A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications,” NIST SP 800-22 Rev. 1a, 2010, doi: 10.6028/NIST.SP.800-22r1a.
[17] E. B. Barker and J. M. Kelsey, “Recommendation for the Entropy Sources Used for Random Bit Generation,” NIST SP 800-90B, 2018, doi: 10.6028/NIST.SP.800-90B.
[18] J. Katz and Y. Lindell, Introduction to Modern Cryptography, 3rd ed. Boca Raton, FL, USA: CRC Press, 2020.
[19] D. Eastlake, J. Schiller, and S. Crocker, “Randomness Requirements for Security,” RFC 4086, 2005, doi: 10.17487/RFC4086.
[20] Y. Nir and A. Langley, “ChaCha20 and Poly1305 for IETF Protocols,” RFC 8439, 2018, doi: 10.17487/RFC8439.

Submitted: **01/12/2024**; Accepted: **27/12/2024**; Published: **28/01/2025**