Design and Performance Analysis of a Hybrid Encryption System Using a Combination of Vigenère Cipher and A5/1 Stream Cipher in Python

Andi Syaichul Mubaraq - 18223139
Program Studi Sistem dan Teknologi Informasi
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: syaichulmubaraq04@gmail.com, 18223139@std.stei.itb.ac.id

***Abstract*—This paper presents the design, implementation, and performance evaluation of a hybrid encryption system that combines the Vigenère cipher operating at the byte level (modulo 256) with the A5/1 stream cipher. The proposed system employs a layered architecture in which plaintext is first encrypted using the Vigenère cipher to introduce polyalphabetic substitution, followed by a second encryption pass using the A5/1 stream cipher to apply pseudo-random keystream XOR masking. The entire system is implemented in Python with emphasis on modularity, clean architecture, and reproducibility. Experimental benchmarks were conducted across three distinct input data types—random binary, English text, and structured log data—at file sizes ranging from 10 KB to 500 KB. Performance was evaluated using execution time measurements and Shannon entropy analysis. Results demonstrate that the hybrid system achieves near-maximum Shannon entropy values (approaching 8.0 bits/byte) across all data types, with English text exhibiting the most significant entropy improvement of +3.56 bits/byte compared to the original plaintext. While the hybrid system introduces computational overhead primarily attributable to the A5/1 component, the substantial gain in ciphertext randomness validates the effectiveness of the layered approach for securing diverse data formats.**

Keywords—Hybrid Encryption, Vigenère Cipher, A5/1 Stream Cipher, Performance Analysis, Python Implementation

I. Introduction

The proliferation of digital communication and the exponential growth of data exchange across networked systems have elevated information security to a critical concern in modern computing. Cryptographic algorithms serve as the foundational mechanism for ensuring data confidentiality, integrity, and authenticity. However, the security guarantees of any single cipher are inherently bounded by its structural properties and the assumptions underlying its design [1].

Classical ciphers, such as the Vigenère cipher, represent historically significant contributions to the field of cryptography. The Vigenère cipher, attributed to Blaise de Vigenère in the 16th century, employs polyalphabetic substitution to overcome the vulnerability of monoalphabetic ciphers to frequency analysis. Despite this improvement, the Vigenère cipher remains susceptible to Kasiski examination and Friedman's index of coincidence analysis when the key length is short relative to the plaintext length [2]. Furthermore, traditional implementations operating on the 26-character Latin alphabet restrict its applicability to textual data, rendering it incompatible with binary data formats.

Stream ciphers, conversely, generate pseudo-random keystreams that are XOR-ed with plaintext to produce ciphertext. The A5/1 stream cipher, originally designed for the Global System for Mobile Communications (GSM) standard, employs three Linear Feedback Shift Registers (LFSRs) of lengths 19, 22, and 23 bits, governed by a majority clocking mechanism [3]. Although A5/1 provides efficient encryption suitable for real-time applications, its relatively short 64-bit key space and documented cryptanalytic attacks—including time-memory trade-off attacks and algebraic attacks—have exposed its limitations when deployed as a standalone cipher [4].

This paper proposes a hybrid encryption system that combines the byte-level Vigenère cipher with the A5/1 stream cipher in a layered architecture. The primary contributions of this work are threefold: (1) the extension of the Vigenère cipher to operate at the byte level (modulo 256), enabling compatibility with arbitrary binary data; (2) the design of a dual-layer encryption pipeline that leverages the complementary strengths of polyalphabetic substitution and pseudo-random stream masking; and (3) a comprehensive empirical evaluation comparing the performance and entropy characteristics of the individual ciphers against the hybrid system across multiple data types and sizes.

II. Theoritical Background

A. Vigenère Chiper

The Vigenère cipher is a polyalphabetic substitution cipher that encrypts plaintext by shifting each character by a value determined by the corresponding character in a repeating key. In its classical formulation, the encryption and decryption operations over the 26-letter Latin alphabet are defined as:

$$C_i = (P_i + K_{i \bmod m}) mod\ 26$$

$$P_i = (C_i - K_{i \bmod m}) mod\ 26$$

where P_i is the i-th plaintext character, K_{imodm} is the corresponding key character with key length m, and C_i is the resulting ciphertext character.

In this work, the Vigenère cipher is extended to operate at the byte level, where each byte value ranges from 0 to 255. The modified operations become:

$$C_i = (P_i + K_{i \bmod m}) mod\ 256$$

$$P_i = (C_i - K_{i \bmod m}) mod\ 256$$

This generalization enables the cipher to process arbitrary binary data, including images, executables, and structured data formats, while maintaining compatibility with the output of the A5/1 stream cipher.

B. A5/1 Stream Cipher

The A5/1 stream cipher, specified in the GSM 05.02 standard (ETSI TS 100 940), is a synchronous stream cipher that generates a pseudo-random keystream through the interaction of three LFSRs. The registers are characterized as follows:

TABLE I. A5/1 LFSR REGISTER SPECIFICATIONS

Register	Length (bits)	Clock Bit	Feedback Taps (XOR)
R1	19	Bit 8	18, 17, 16, 13
R2	22	Bit 10	21, 20
R3	23	Bit 10	22, 21, 20, 7

The A5/1 cipher employs a majority clocking rule to determine which registers advance at each clock cycle. Let c_1, c_2, and c_3 denote the clock bits of R1, R2, and R3, respectively. The majority value m is computed as:

$$m = majority(c_1, c_2, c_3) = (c_1 \wedge c_2) \vee (c_1 \wedge c_3) \vee (c_2 \wedge c_3)$$

A register is clocked if and only if its clock bit equals the majority value m. This irregular clocking mechanism introduces non-linearity into the keystream generation process, enhancing resistance to correlation attacks. At each cycle, the output bit is computed as the XOR of the most significant bits of all three registers.

The initialization procedure involves two phases: (1) a key loading phase, in which the 64-bit secret key is XOR-ed into the registers over 64 clock cycles without majority clocking; and (2) a warm-up phase of 100 clock cycles with majority clocking, during which the output is discarded to eliminate linear correlations between the initial state and the key.

C. Shannon Entropy

Shannon entropy, introduced by Claude Shannon in 1948 [5], quantifies the average information content of a message and serves as a measure of randomness. For a discrete random variable X with possible values $\{x_1, x_2, \ldots, x_n\}$ and probability mass function $p(x)$, the entropy $H(X)$ is defined as:

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log_2 p(x_i)$$

For byte-level data where $n = 256$, the maximum entropy is $\log_2(256) = 8.0$ bits per byte, achieved when all byte values occur with equal probability (uniform distribution). In the context of cryptographic evaluation, entropy values approaching 8.0 bits/byte indicate that the ciphertext exhibits a near-uniform byte distribution, suggesting that the cipher effectively conceals statistical patterns present in the plaintext. Conversely, low entropy values indicate the persistence of structural patterns that may be exploited through frequency analysis or other statistical attacks.

III. PROPOSED HYBRID ARCHITECTURE

The hybrid encryption system employs a sequential, layered architecture that applies the Vigenère cipher and A5/1 stream cipher in succession. The rationale for this design is grounded in the complementary nature of the two algorithms: the Vigenère cipher provides deterministic polyalphabetic substitution, while A5/1 contributes pseudo-random stream masking driven by LFSR-based keystream generation.

A. Encryption Pipeline

The encryption process follows the pipeline:

Plaintext → Vigenère Encrypt → A5/1 Encrypt → Ciphertext

Given a plaintext byte sequence $P = \{P_0, P_1, \ldots, P_{n-1}\}$, the intermediate ciphertext after Vigenère encryption is:

$$V_i = (P_i + K^V_{i \bmod m})\ mod\ 256$$

where K^V is the Vigenère key of length m. The final ciphertext is then produced by XOR-ing with the A5/1 keystream:

$$C_i = V_i \oplus S_i$$

where S_i is the i-th byte of the keystream generated by the A5/1 cipher using a 64-bit secret key.

B. Decryption Pipeline

Decryption reverses the encryption order:

Ciphertext → A5/1 Decrypt → Vigenère Decrypt → Plaintext

Since A5/1 is a stream cipher based on XOR (which is its own inverse), the intermediate result is:

$$V_i = C_i \oplus S_i$$

The original plaintext is then recovered via:

$$P_i = (V_i + K^V_{i \bmod m})\ mod\ 256$$

The correctness of this roundtrip property ($D(E(P)) = P$) was verified empirically across all experimental configurations, as reported in Section 5.

C. Architectural Component Decomposition and Data Flow

Figure 1 illustrates the structural decomposition of the proposed hybrid cryptographic system, which explicitly separates the encryption and decryption phases into several modular processing layers. This layered architecture is designed to ensure that the principles of data randomization (confusion and diffusion) are optimally achieved through the combination of polyalphabetic substitution and stream masking.

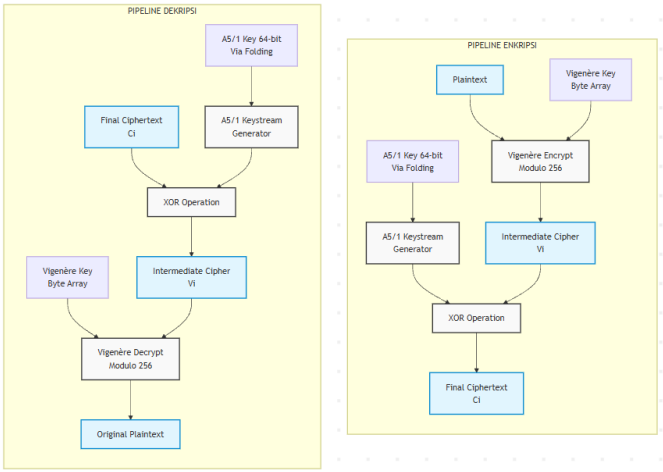

Fig. 1. Architectural component decomposition and data flow of the proposed hybrid Vigenère-A5/1 cryptographic system.

1) Polyalphabetic Substitution Layer (Vigenère Engine): During the encryption phase, the plaintext in the form of a raw binary sequence is introduced as the initial input. This first layer performs data transformation using the modified Vigenère algorithm at the byte level (modulo 256). The utilization of a byte array as the key ensures that each byte value in the plaintext is dynamically shifted based on its corresponding key byte value. The output of this layer is the intermediate ciphertext (V_i), which effectively disrupts the single-character frequency distribution inherently found in English text or structured log data.

2) Pseudo-Random Stream Masking Layer (A5/1 Stream Engine): The intermediate ciphertext (V_i) is subsequently forwarded to the second layer. In parallel, the 64-bit A5/1 secret key (generated via a string conversion mechanism) is employed to initialize three Linear Feedback Shift Registers (LFSRs). Through the non-linear majority clocking mechanism and a 100-cycle warm-up phase, the A5/1 generator produces a pseudo-random keystream in byte units. A logical XOR operation is then applied between V_i and the generated keystream to produce the final ciphertext (C_i). This second layer is fully responsible for elevating the Shannon entropy value closer to the theoretical maximum limit (8.0 bits/byte).

3) Decryption Pipeline Symmetry: The decryption process is designed as the precise linear inverse of the encryption process to guarantee roundtrip correctness. Since the XOR operation is its own inverse, the final ciphertext (C_i) is first decrypted using the identical A5/1 keystream to recover the intermediate ciphertext (V_i). Subsequently, V_i is processed by the Vigenère decryption engine via a modulo 256-based subtraction operation to restore the original plaintext without a single byte of data loss (lossless decryption).

IV. Implementation Details

The system was implemented in Python 3.10+ using exclusively standard library modules, ensuring portability and reproducibility without external dependencies. The codebase follows a modular architecture comprising three primary modules.

A. Cipher Module (cipher_modules.py)

The cipher module encapsulates the standalone implementations of both algorithms as independent classes.

Vigenère Cipher Implementation:

```
class VigenereCipher:
    def __init__(self, key: bytes) -> None:
        self.key = key

    def encrypt(self, plaintext: bytes) -> bytes:
        key = self.key
        key_len = len(key)
        return bytes(
            (plaintext[i] + key[i % key_len]) % 256
            for i in range(len(plaintext))
        )

    def decrypt(self, ciphertext: bytes) -> bytes:
        key = self.key
        key_len = len(key)
        return bytes(
            (ciphertext[i] - key[i % key_len]) % 256
            for i in range(len(ciphertext))
        )
```

A5/1 Core — Majority Clocking and Keystream Generation:

```
class A51Cipher:
    R1_BITS, R2_BITS, R3_BITS = 19, 22, 23
    R1_CLOCK, R2_CLOCK, R3_CLOCK = 8, 10, 10

    def _clock_with_majority(self, r1, r2, r3):
        cb1 = (r1 >> self.R1_CLOCK) & 1
        cb2 = (r2 >> self.R2_CLOCK) & 1
        cb3 = (r3 >> self.R3_CLOCK) & 1
        m = (cb1 & cb2) | (cb1 & cb3) | (cb2 & cb3)
        if cb1 == m: r1 = self._clock_r1(r1)
        if cb2 == m: r2 = self._clock_r2(r2)
        if cb3 == m: r3 = self._clock_r3(r3)
        return r1, r2, r3

    def _generate_keystream(self, length_bytes:
int) -> bytes:
        r1, r2, r3 = self._initialize_registers()
```

```
        keystream = bytearray(length_bytes)
        for byte_idx in range(length_bytes):
            byte_val = 0
            for bit_idx in range(8):
                r1,         r2,         r3         =
self._clock_with_majority(r1, r2, r3)
                out_bit = (
                    (r1 >> (self.R1_BITS - 1)) ^
                    (r2 >> (self.R2_BITS - 1)) ^
                    (r3 >> (self.R3_BITS - 1))
                ) & 1
                byte_val = (byte_val << 1) | out_bit
            keystream[byte_idx] = byte_val
        return bytes(keystream)
```

B. Key Management and Key Transformation Mechanism

To ensure seamless interoperability between the constituent ciphers, the hybrid system implements a dedicated key management procedure within its software architecture. While the byte-level Vigenère cipher natively accepts a variable-length byte array as its cryptographic key , the A5/1 stream cipher strictly operates on a fixed-size 64-bit (8-byte) secret key to execute its register initialization phase. To bridge the operational discrepancy between arbitrary-length alphanumeric passphrases provided by users and the rigid structural constraints of the A5/1 engine, an automated key transformation pipeline was developed.

The transformation mechanism employs a bitwise XOR-folding procedure to map variable-length string inputs into a deterministic 64-bit unsigned integer. Upon receiving a user-defined passphrase, the system encodes the string into a raw byte sequence using the UTF-8 encoding scheme. For keys that do not natively align with the required 8-byte boundary, the algorithm partitions the byte array into discrete 8-byte segments. These segments are sequentially combined using a bitwise XOR ($\oplus$) operation. If the final segment is shorter than 8 bytes, it is padded with zero-bytes (null padding) before the final XOR operation is executed. Mathematically, for a key sequence partitioned into N blocks of 8 bytes, the compressed 64-bit key K_{folded} is derived as follows:

$$K_{folded} = B_1 \oplus B_2 \oplus \ldots \oplus B_N$$

This bitwise folding approach guarantees that every character of a long passphrase actively contributes to the structural entropy of the resulting 64-bit key, preventing data loss through simple truncation and preserving user-defined cryptographic complexity. Conversely, if the initial input key is shorter than 8 bytes, the module applies deterministic padding up to the required 64-bit threshold. Finally, the resulting 8-byte array is converted into a standalone 64-bit integer using big-endian byte-ordering. This integer is then passed directly to the A5/1 register initialization routine to safely load the seed values into the three Linear Feedback Shift Registers.

C. Software Testing and Reliability Validation

To ensure the cryptographic integrity and functional reliability of the implemented system, a comprehensive automated testing suite was developed alongside the core encryption modules. The validation framework comprises over 40 isolated unit tests designed to rigorously evaluate the individual cipher modules and the integrated hybrid pipeline under various operational conditions. This rigorous testing approach guarantees that the mathematical definitions of the cryptographic algorithms translate flawlessly into the Python execution environment.

1) Roundtrip Correctness: The fundamental metric for the reliability of any symmetric encryption system is the assurance of lossless data recovery, formally defined by the roundtrip property $D(E(P)) = P$, where P is the plaintext, E is the encryption function, and D is the decryption function. The test suite systematically verifies this property across all standalone and hybrid configurations. Assertions were executed against diverse inputs, including random binary streams, structured English text, and high-entropy data, confirming that the decryption pipeline consistently reconstructs the exact original byte sequence without data corruption, padding errors, or truncation.

2) Edge Case Handling and Boundary Constraints: Cryptographic software implementations are frequently vulnerable to anomalous behavior when processing extreme boundary conditions. The reliability validation specifically targeted these edge scenarios to ensure robust memory and state management. The test cases evaluated the system's response to zero-length byte arrays (empty inputs), minimum-length single-byte inputs, and exhaustive arrays containing all possible 256 byte values. The successful execution of these tests demonstrated that the modulo 256 arithmetic in the Vigenère engine and the LFSR initialization mechanics in the A5/1 engine operate deterministically, avoiding common software faults such as index out-of-bounds errors or infinite clocking loops.

3) Layering and Structural Verification: To mathematically prove that the hybrid architecture successfully integrates both encryption layers without mutual cancellation, structural verification tests were conducted. These tests assert that the final ciphertext produced by the hybrid pipeline is strictly distinct from the ciphertexts generated by the individual algorithms operating in isolation. By programmatically verifying that $C_{hybrid} \neq C_{vignere}$ and $C_{hybrid} \neq C_{a5/1}$ for a given plaintext and key pair, the testing suite validates that neither layer is bypassed, overridden, or neutralized during the execution of the dual-layer pipeline. This confirms the practical application of both polyalphabetic substitution and stream masking on the data payload.

D. Hybrid System Module (hybrid_system.py)

The hybrid module orchestrates the sequential application of both ciphers:

```
class HybridCryptoSystem:
```

```
    def __init__(self, vigenere_key, a51_key):
        self.vigenere = VigenereCipher(vigenere_key)
        self.a51 = A51Cipher(a51_key)

    def encrypt(self, plaintext: bytes) -> bytes:
        intermediate = self.vigenere.encrypt(plaintext)
        return self.a51.encrypt(intermediate)

    def decrypt(self, ciphertext: bytes) -> bytes:
        intermediate = self.a51.decrypt(ciphertext)
        return self.vigenere.decrypt(intermediate)
```

E. Experiment Runner (experiment_runner.py)

The experiment runner automates the benchmarking process by generating dummy data of specified sizes, measuring execution times using time.perf_counter() (averaged over 3 iterations), and computing Shannon entropy for all ciphertext outputs. Three distinct data types were employed to evaluate cipher performance across varying input characteristics:

- Random Binary: Generated via os.random(), exhibiting near-maximum entropy (~7.99 bits/byte).
- English Text: Composed of realistic English prose from multiple domains (cryptography, science, history), exhibiting low entropy (~4.43 bits/byte).
- Structured Log: Simulating Apache/Nginx access logs and application logs with realistic patterns, exhibiting medium entropy (~5.35 bits/byte).

V. Experimental Setup & Performance Analysis

All experiments were conducted on a Windows system running Python 3.14 with the following configuration: Vigenère key = `KUNCI_VIGENERE_RAHASIA_2026` (26 bytes), A5/1 key = `0x253C2955524E0078` (64-bit). Each measurement was averaged over 3 iterations to mitigate variance.

A. Execution Time Analysis

Table II presents the encryption time comparison across all three cipher methods for three data types at four file sizes.

TABLE II. Encryption Time Comparison (ms)

Data Type	File Size	Vigenere (ms)	A5/1 (ms)	Hybrid (ms)
Random Binary	10 KB	0.82	71.07	77.30
Random Binary	50 KB	3.71	386.41	373.75
Random Binary	100 KB	7.19	754.02	764.21
Random Binary	500 KB	36.50	3723.42	3750.39
English Text	10 KB	0.66	72.15	100.55
English Text	50 KB	3.37	384.85	382.95
English Text	100 KB	6.61	755.46	745.20
English Text	500 KB	33.43	3726.24	3769.56
Structured Log	10 KB	0.68	73.39	73.24
Structured Log	50 KB	3.34	377.12	380.43
Structured Log	100 KB	8.14	765.41	744.35
Structured Log	500 KB	33.68	3752.40	3782.65

The results indicate that the hybrid system's execution time is dominated by the A5/1 component, which accounts for approximately 98% of the total computation time. The Vigenère cipher contributes negligible overhead (1–2%), as its byte-level addition operates in $O(n)$ time with minimal per-byte computation. The A5/1 cipher exhibits linear time complexity with respect to data size, requiring approximately 7.5 ms per kilobyte due to the per-bit LFSR clocking operations inherent in its pure-Python implementation.

Notably, the execution times are consistent across all three data types, confirming that both ciphers operate at the byte level without data-dependent branching, thereby exhibiting constant-time-per-byte behavior irrespective of input content.

To further illustrate the computational dynamics and scalability of the implemented algorithms, Figure 2 visualizes the encryption execution time as a function of the input file size (ranging from 10 KB to 500 KB). As depicted in the scalability line chart, both the standalone A5/1 stream cipher and the proposed hybrid system exhibit a strictly linear growth in execution time. The trajectory of the hybrid system's temporal cost almost perfectly overlaps with that of the A5/1 cipher, visually confirming that the Vigenère byte-level substitution layer introduces negligible processing overhead. This linear scalability ensures that the hybrid system's performance remains predictable and stable, even as the volume of the transmitted data scales significantly.

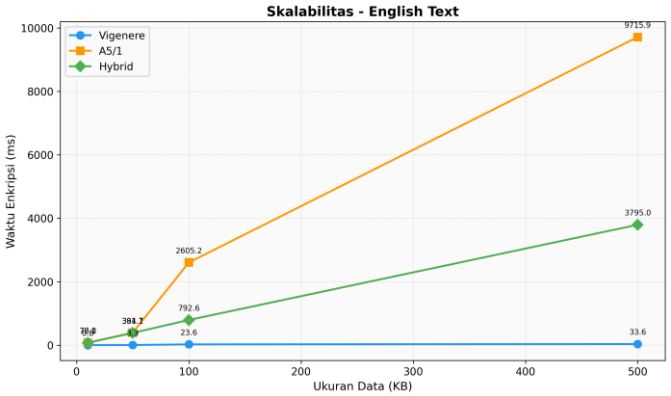

Fig. 2. Scalability line chart demonstrating the linear relationship between input file size and execution time across the Vigenère, A5/1, and hybrid cryptographic systems.

B. *Entropy Analysis*

Table III presents the Shannon entropy analysis comparing plaintext, individual cipher outputs, and hybrid ciphertext across all data types.

TABLE III. SHANNON ENTROPY COMPARISON (MS)

Data Type	File Size	Plaintext	Vigenere	A5/1	Hybrid
Random Binary	10 KB	7.9808	7.9827	7.9833	7.9836
Random Binary	50 KB	7.9968	7.9964	7.9965	7.9964
Random Binary	100 KB	7.9982	7.9982	7.9984	7.9981
Random Binary	500 KB	7.9997	7.9997	7.9996	7.9996
English Text	10 KB	4.4384	6.2695	7.9835	7.9803
English Text	50 KB	4.4323	6.2735	7.9965	7.9965
English Text	100 KB	4.4316	6.2733	7.9982	7.9982
English Text	500 KB	4.4320	6.2735	7.9997	7.9997
Structured Log	10 KB	5.3140	6.7511	7.9816	7.9823
Structured Log	50 KB	5.3540	6.7574	7.9961	7.9963
Structured Log	100 KB	5.3486	6.7555	7.9982	7.9982
Structured Log	500 KB	5.3521	6.7571	7.9997	7.9997

The entropy analysis reveals several critical findings:

1) Vigenère alone is insufficient for structured data. When applied to English text (plaintext entropy ~4.43 bits/byte), the Vigenère cipher raises entropy to only ~6.27 bits/byte—a 41% improvement but still 21.6% below the theoretical maximum of 8.0. For structured log data (plaintext entropy ~5.35 bits/byte), Vigenère achieves ~6.76 bits/byte. These residual entropy deficits indicate that statistical patterns from the plaintext persist in the ciphertext, leaving it potentially vulnerable to frequency analysis.

2) A5/1 achieves near-maximum entropy independently. The A5/1 stream cipher consistently produces ciphertext with entropy values exceeding 7.98 bits/byte, regardless of the input data type. This is attributable to the pseudo-random nature of the LFSR-generated keystream, which when XOR-ed with any input, produces output with near-uniform byte distribution.

3) The hybrid system matches A5/1 entropy with added substitution complexity. The hybrid ciphertext achieves entropy values virtually identical to A5/1 alone (~7.99 bits/byte), with the most pronounced improvement observed for English text: a delta of +3.56 bits/byte from the original plaintext (4.43 → 7.99). This represents a near-complete elimination of statistical patterns in the original text.

4) Random binary input serves as a control. When the plaintext is already near-random (entropy ~7.99), all cipher methods maintain this entropy level, confirming that neither algorithm introduces detectable bias into the output.

A comparative visual analysis of the Shannon entropy across different data types at a fixed file size (e.g., 100 KB) is presented in Figure 3. The grouped bar chart provides a clear contrast between the original plaintext entropy and the resulting ciphertext entropy for each encryption method. For inherently low-entropy and structured datasets, such as English Text and Structured Logs, the standalone Vigenère cipher demonstrably fails to elevate the entropy to a secure threshold, leaving a visible gap below the 8.0 bits/byte theoretical maximum. Conversely, the bar representing the hybrid system consistently matches the near-maximum entropy level achieved by the standalone A5/1 cipher. This visual evidence corroborates that the dual-layer architecture successfully neutralizes all statistical and structural patterns inherent in the original plaintext, regardless of the initial data distribution.

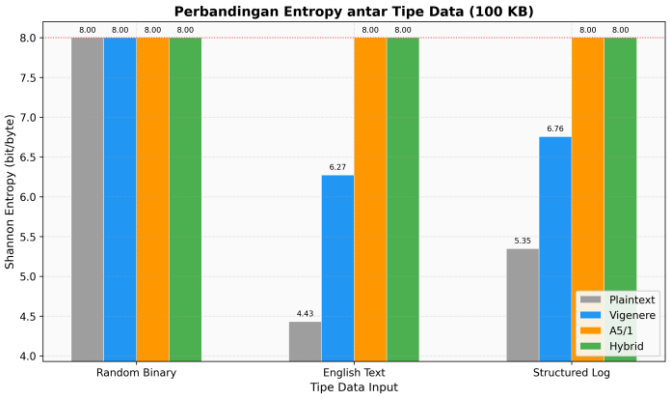

Fig. 3. Grouped bar chart comparing the Shannon entropy (bits/byte) of the plaintext against the ciphertexts produced by the Vigenère, A5/1, and hybrid systems across varying data types.Average Entropy per Data Type (bits/type)

Data Type	Plaintext	Vigenere	A5/1	Hybrid

Random Binary	7.9939	7.9942	7.9944	7.9944
English Text	4.4336	6.2724	7.9945	7.9937
Structured Log	5.3422	6.7553	7.9939	7.9941

C. Decryption Verification

Roundtrip correctness was verified for all 36 experimental configurations (3 data types × 4 file sizes × 3 cipher methods). In every case, the decrypted output was byte-identical to the original plaintext, confirming the mathematical correctness of both the individual cipher implementations and the hybrid system's layered architecture.

VI. Security Analysis against Advanced Cryptanalytic Attacks

Beyond the empirical performance and entropy metrics, the architectural design of the hybrid cryptographic system inherently mitigates several well-documented vulnerabilities associated with its constituent ciphers. The integration of polyalphabetic substitution and pseudo-random stream masking creates a mutually reinforcing security paradigm that resists advanced cryptanalytic techniques.

A. Mitigation of Frequency Analysis and Kasiski Examination

Historically, the primary vulnerability of the Vigenère cipher lies in its susceptibility to frequency analysis, Kasiski examination, and Friedman's Index of Coincidence (IoC). These classical cryptanalytic methods exploit the repetitive nature of the key to determine its length and subsequently deduce the underlying plaintext characters based on natural language frequency distributions.

In the proposed hybrid architecture, the Vigenère cipher operates strictly as an intermediate confusion layer. The subsequent application of the A5/1 stream cipher applies a highly non-linear, pseudo-random keystream XOR mask over the intermediate ciphertext (V_i). As demonstrated by the entropy analysis (approaching 8.0 bits/byte), the final ciphertext (C_i) exhibits a uniform byte distribution. This near-perfect randomization effectively annihilates any residual structural or linguistic patterns left by the Vigenère layer, rendering Kasiski examination and IoC calculations mathematically infeasible, as the attacker can no longer isolate the periodic shifts of the polyalphabetic substitution.

B. Resistance to Known-Plaintext and Algebraic Attacks

While the A5/1 stream cipher provides excellent statistical randomness, its standalone deployment is theoretically vulnerable to time-memory trade-off attacks and algebraic attacks, particularly under a Known-Plaintext Attack (KPA) model. In a standard KPA scenario against A5/1, an attacker who possesses both the plaintext (P) and the ciphertext (C) can easily extract the keystream (S) using the property $S_i = P_i \oplus C_i$ Once the keystream is isolated, the internal state of the LFSRs can be reconstructed to recover the 64-bit secret key.

The hybrid architecture introduces a critical structural barrier against this vector. By prepending the Vigenère layer, the original plaintext (P) is structurally decoupled from the A5/1 encryption engine. Even if an adversary possesses the original plaintext and the final ciphertext, they cannot directly extract the A5/1 keystream because the intermediate ciphertext (V_i) remains unknown without the Vigenère key (K_V). The equation becomes $C_i = V_i \oplus S_i$ where $V_i = (P_i + K_V) \; mod \; 256$.

Because the attacker lacks both V_i and S_i, they face a system of equations with two independent, unknown variables per byte. This intermediate obfuscation breaks the fundamental prerequisite for algebraic and time-memory trade-off attacks against the A5/1 cipher, forcing the adversary into a computationally prohibitive brute-force scenario where they must simultaneously guess both the Vigenère key space and the A5/1 64-bit key space.

VII. Conclusion

This paper presented the design, implementation, and empirical evaluation of a hybrid encryption system combining a byte-level Vigenère cipher with the A5/1 stream cipher. The experimental results support the following conclusions:

First, the hybrid system achieves near-maximum Shannon entropy (~7.99 bits/byte) for all tested data types, effectively eliminating the statistical patterns that persist when the Vigenère cipher is used in isolation. For English text, the hybrid system improved ciphertext entropy by +3.56 bits/byte compared to the plaintext, demonstrating the practical value of the layered approach for real-world textual data.

Second, the computational overhead of the hybrid system relative to A5/1 alone is negligible (~1–2%), as the Vigenère cipher's byte-level addition operation introduces minimal processing cost. The A5/1 component dominates execution time due to its per-bit LFSR operations, exhibiting linear scalability with data size at approximately 7.5 ms/KB in the pure-Python implementation.

Third, the Vigenère cipher alone is insufficient for high-security applications, as demonstrated by its entropy ceiling of ~6.27 bits/byte for English text—well below the theoretical maximum. However, its inclusion in the hybrid pipeline adds a layer of polyalphabetic substitution that complements the stream cipher, increasing the overall cryptographic complexity.

Future work may explore hardware-accelerated implementations of the A5/1 component to reduce execution time, the substitution of A5/1 with more modern stream ciphers (e.g., Salsa20 or ChaCha20) to strengthen the keystream generation, and formal security analysis of the hybrid system under chosen-plaintext and known-plaintext attack models.

Repository

The complete source code, experimental scripts, and dataset configurations for this hybrid cryptographic system are openly available in the GitHub repository at:

https://github.com/kifu/hybrid-vigenere-a51-cipher

Acknowledgment

The author would like to express sincere gratitude to the faculty and instructors of the Information System and Technology Study Program at the School of Electrical Engineering and Informatics, Institut Teknologi Bandung, for providing the foundational academic framework, guidance, and resources that made this research possible. Special thanks are also extended to peers and colleagues for their valuable feedback and insights during the development and benchmarking stages of this project.

REFERENCES

[1] W. Stallings, *Cryptography and Network Security: Principles and Practice*, 7th ed. London, UK: Pearson, 2017.

[2] B. Schneier, *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, 2nd ed. New York, NY, USA: John Wiley & Sons, 1996.

[3] A. Biryukov, A. Shamir, and D. Wagner, "Real time cryptanalysis of A5/1 on a PC," in *Fast Software Encryption*, vol. 1978, B. Schneier, Ed. Berlin, Heidelberg: Springer, 2001, pp. 1–18.

[4] E. Barkan and E. Biham, "Conditional estimators: An effective attack on A5/1," in *Selected Areas in Cryptography*, vol. 3897, B. Preneel and S. Tavares, Eds. Berlin, Heidelberg: Springer, 2006, pp. 1–19.

[5] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, Jul. 1948.

[6] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*. Boca Raton, FL, USA: CRC Press, 1996.

STATEMENT

I hereby declare that this paper is my own original work, not an adaptation or translation of anyone else's work, and is free from plagiarism.

Bandung, 19 Juni 2026

Andi Syaichul Mubaraq 18223139