JIRBDAI

Journal of Institutional Research, Big Data Analytics and Innovation

Enhancing the Vigenère Cipher for Unicode Compatibility: A Modern Approach to Polyalphabetic Encryption

***Olumide Chris Ajogbeje, Adebusola Modupeola Tope-Oke, Toyin Okebule, Olanike Christianah Akinduyite**

Department of Computing, College of Sciences, Afe Babalola University, Ado-Ekiti, Ekiti State, Nigeria.
Corresponding Author's email: ajogbejeolumide@abuad.edu.ng

ABSTRACT: In today's digital age, the significance of strong encryption in communication has significantly increased. The Vigenère cipher, a classic polyalphabetic encryption substitution method, offers both effectiveness and simplicity; however, traditional methods only accommodate the uppercase alphabet of mode 26, which is insufficient for contemporary text encryption and security. This work advances the field of cryptography by introducing a robust and adaptable encryption technique suitable for modern computing environments. It suggests an updated version of the Vigenère cipher that supports Unicode, enabling a modern approach to polyalphabetic encryption through a system that can encrypt a wider range of characters beyond the ASCII (256) limit, including those from non-Latin scripts. This study also includes a flexible key generation system capable of dynamically managing different character sets while preserving the core principles of the cipher. Not only does the method reveal the enhanced Vigenère cipher, but it also considers capability and security implications. Finally, by integrating the modernize Vigenère cipher for Unicode compatibility, it creates a robust and versatile polyalphabetic encryption solution equipped to tackle the complexities of contemporary computing environments, securing text data across various applications such as messaging platforms, online storage services, and confidential document exchange.

KEYWORDS: Vigenère Cipher, Unicode Compatibility, Polyalphabetic encryption, Modern Cryptography, Text Encryption

I. INTRODUCTION

Cryptography is extensively employed to solve information security issues that arise when data is transmitted across the internet. A well-known and generally robust cryptographic algorithm is the Vigenère Cipher [2]. The Vigenère Cipher is a classic encryption technique that encodes alphabetic messages by applying a sequence of Caesar ciphers determined by the letters of a chosen keyword [10]. The Vigenère cipher is a cryptographic technique that uses a character-based key to encrypt the original message (plaintext) [11]. The Vigenère Cipher is based on a simple type of polyalphabetic substitution, which means it's a substitution cipher that employs multiple substitution alphabets. Encryption is carried out using the Vigenère square (or table). This table is essentially a 26×26 grid where the alphabet is written out in 26 different rows. In each row, the alphabet is cyclically shifted one position to the left compared to the row above it, creating a visual representation of all 26 possible Caesar Ciphers [4]. The Vigenère Cipher is a symmetrical, polyalphabetic encryption method. It was once regarded as highly secure until it was cryptanalyzed through the Kaisiki Test. There are three primary limitations of the Vigenère cipher. First, it relies on an encryption table created by shifting the 26 letters of the Roman alphabet regularly. Second, it uses a keystream generated by repeating a chosen password, which can be a vulnerability. Third, the decryption process is slower because it does not rely on the table for direct searching and matching; instead, it requires more time. The original and generalized forms of the Vigenère Cipher utilize a square cipher

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.17666872

table, known as the Vigenère Square, which is a 26x26 grid formed by shifting the 26 uppercase letters of the Roman alphabet [15].

Recent rapid advancements in technology, particularly in communication technology, have led to a growing demand for quick access to information. This trend is especially evident in electronic media, where the internet significantly contributes to this phenomenon. Today, individuals can utilize portable devices like smartphones or tablets, as well as technological tools such as personal computers, to connect to the online world [5]. The primary goal of cryptography is to ensure that plaintext remains confidential and hidden from eavesdroppers. Third parties might attempt to alter messages within the data, especially when they have unrestricted access to the communication channel. As a result, cryptography is designed to verify the integrity and authenticity of the message, ensuring its truthfulness [14]. Cryptography methods are generally classified into two categories: classical cryptography and modern cryptography. Classical cryptography is an ancient message encoding technique that predates the advent of computers and modern technology, having been used for thousands of years. It relies on mathematical techniques and the manipulation of letters or characters in the original message to generate an encrypted version that is challenging for anyone without the correct key or method to decipher or interpret. Conversely, modern cryptography is a more advanced and intricate method of message encoding developed to address the vulnerabilities of classical cryptography and to safeguard information against more powerful attacks. These techniques generally employ more sophisticated and secure algorithms, along with longer and more complex keys to enhance protection [12].

Polyalphabetic substitution is a cipher technique in which each letter of the plaintext is replaced by a different letter according to a particular pattern or key. Some well-known examples of polyalphabetic ciphers include the Vigenère Cipher, Autokey Cipher, Running Key Cipher, Beaufort Cipher, Hill Cipher, Playfair Cipher, Two-Square Cipher, Gronsfeld Cipher, Porta Cipher, and Digraph Cipher. These represent just a selection of the numerous polyalphabetic substitution ciphers available. Each cipher type has its own advantages and vulnerabilities, with some offering greater security than others. The purpose of the paper is to update the traditional Vigenère cipher by modifying it for compatibility with Unicode. This enhancement will enable it to support the wide variety of characters utilized in contemporary digital communication. Expanding the character length of the Vigenère cipher, which is traditionally limited to mod26, to encompass the full Unicode range and support complex password combinations using a Salt key. This involves systematically integrating the key with an additional string (a "salt"). Message encryption, also known as cryptography, has a history spanning thousands of years. It entails transforming understandable data (plaintext) into an unintelligible form (ciphertext) to ensure its secrecy. This procedure depends on specific algorithms and keys that only authorized individuals possess, enabling them to decrypt and access the original message.

II. RELATED WORK

The Vigenère cipher, a well-known classic in cryptography, is appreciated for its sophisticated polyalphabetic substitution. However, its original implementation is restricted to the Latin alphabet, rendering it inadequate for contemporary digital communication that utilizes Unicode. This paper, titled "Enhancing the Vigenère Cipher for Unicode Compatibility: A Modern Approach to Polyalphabetic Encryption," tackles this challenge by proposing a new method to adapt the cipher to support the entire range of Unicode characters. This modernization ensures that the Vigenère cipher remains a relevant and effective encryption tool in an era that demands a broader character set. Nahar *et al.,* [9] introduced a modified version of the Vigenère cipher utilizing a 95 × 95 table. They expanded the traditional Vigenère table to accommodate a broader set of characters. In their approach, plain text messages containing both

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.17666872

uppercase and lowercase letters are converted to uppercase prior to encryption, as the original algorithm did not account for case sensitivity. Additionally, it lacked the capability to encode numbers, spaces, special characters, or symbols commonly used in English. To address these limitations, the researchers proposed an extended Vigenère cipher that employs a 95 × 95 table, enabling the encryption of lowercase letters (a–z), digits (0–9), and 33 special characters, including spaces, found on a keyboard. This enhanced algorithm is similar to the original, but instead of using modulo 26, it applies modulo 95 to handle the larger character set.

Shoukat *et al.,* [16] proposed an enhanced method of the Vigenère cipher aimed at securely compressing text by utilizing relative frequency analysis. This approach enhances the confidentiality of the ciphertext by incorporating the relative frequencies of alphabetic letters. The method involves arranging the letters based on their relative frequencies in ascending order and then making modifications to the traditional Vigenère cipher using this frequency information, followed by data compression. The proposed technique strengthens message security through multiple layers, including encryption, decryption, and compression. It employs relative letter frequency analysis to determine the likelihood of each alphabetic character, leveraging the principle that text or data contains characters with varying frequencies. Each plaintext letter is matched with a key character according to the increasing order of their relative frequency, thereby adding an extra layer of security through frequency-based matching. Hassan [4] proposed an analysis and modification of the Vigenère cipher aimed at enhancing its security. He improved the traditional Vigenère cipher by modifying its square table to incorporate alphanumeric ciphertext, expanding the character set from 26 letters to 31 characters, including both letters and digits. This increase in complexity strengthens the cipher's resistance to frequency analysis and brute-force attacks.

Mir *et al.,* [7] introduced a combined public and private key image encryption method based on a modified Vigenère cipher integrated with chaotic maps. During the diffusion phase, pixel values are scrambled using the Vigenère cipher with a modified key expansion and updating process, resulting in a partially encrypted image. This image then undergoes further encryption through a confusion phase that employs the chaotic Baker map to randomize the data. The encryption process takes as inputs the original image and key parameters {Ki} for i = 1 to 4, where each Ki is an 8-tuple vector. Ultimately, the encrypted image, along with the secret keys and the public keys {Ki} for i = 1 to 4—which are also encrypted—is transmitted to the receiver. Subandi *et al.,*[17] proposed a modification to the Vigenère cipher algorithm by integrating RC6 key expansion and implementing a double encryption process. The modification leverages RC6's key expansion technique to generate new keys for encryption, with RC6 being a well-known modern cryptographic algorithm. In this approach, the RC6 key expansion produces an S key that matches the length of the plaintext, since it typically generates 2r+4 keys; here, the S key is extended to correspond with the plaintext length. The encryption is performed twice, with each iteration using a completely different key sequence derived through this method. The Vigenère cipher is thus enhanced by utilizing the RC6 key expansion process to obscure the user's original key used for encryption. All operations are conducted using 8-bit ASCII characters and implemented in Microsoft Visual Basic 2005 Express Edition. The plaintext and key are entered for processing, with the RC6 key expansion generating the initial ciphertext, followed by a second round of encryption using the same key to produce the final ciphertext. The same procedures are applied during decryption to recover the original plaintext. This process ensures that the key expansion produces an S key matching the plaintext length, which is then used for both encryption and decryption of the Vigenère cipher. The encryption and decryption processes follow these steps:

$Ci = E(Pi + Si)\ mod\ 256$

$Pi = D(Ci - Si)\ mod\ 256$

Modifying the Vigenère Cipher algorithm with the triple transposition technique. This technique involves repeating the transposition process three times to randomize the order of characters in an encrypted message. Thus, attacks against the Vigenère Cipher can be more difficult to carry out.

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.17666872

A. Cryptography

Cryptography is employed to protect the privacy of data or messages by transforming them into a form that is difficult to understand [3]. To enhance data security, cryptography is a technique used to encode and decode information, ensuring safe data transfer over the internet [6]. Cryptography serves as a robust safeguard for data security by transforming plain text into unreadable cipher text, preventing unauthorized users from understanding it. This encryption process employs algorithms and mathematical techniques to ensure the data remains protected from unauthorized access or interpretation [14].

B. Vigenère Cipher

Cryptography methods include techniques such as the Vigenère cipher and the Caesar cipher. Both methods involve substituting each original character with another character or letter [3]. The Vigenère cipher, notable for its historical importance and technical complexity, offers a basic understanding of how classical cryptography has influenced contemporary security methods [10]. The main goals of computer security are ensuring integrity, confidentiality, availability, authenticity, and accountability. The Caesar cipher is one of the earliest known classical encryption techniques, developed by Julius Caesar for communication with his army [8]. The Vigenère Cipher is a type of polyalphabetic substitution cipher introduced by the French diplomat and cryptologist Blaise de Vigenère in 1586. It encodes text by applying a series of Caesar cipher shifts determined by the letters in a keyword. The Vigenère cipher's substitution process is typically carried out in two forms: using numbers (similar to the Caesar cipher) or letters (the Vigenère cipher itself) [3]. At one point, the Vigenère cipher was regarded as highly secure and believed to be unbreakable, earning it the nickname "le chiffre indéchiffrable," meaning "the unbreakable cipher" in French. Although it was not completely invulnerable to decryption, it served as an effective encryption method. The Vigenère cipher works by substituting letters through a system that employs different alphabets for various parts of the message. It utilizes a 26-by-26 matrix of shifts, building upon the principles of the Caesar cipher but with a broader range of shifts from 0 to 25. Named after Blaise de Vigenère, who lived during the reign of Henry III in France, the cipher encrypts each letter of the plaintext by shifting it according to its position and the corresponding letter in the password, using the Vigenère square or tableau [14].

C. Unicode

Unicode is a character encoding standard that assigns a unique number to every character, symbol, and emoji used in languages worldwide. Several Unicode-based techniques employ a similar approach of modifying the Unicode value of specific letters within a text [1]. The entire Unicode range extends from U+0000 to U+10FFFF, encompassing:

- Basic Multilingual Plane (BMP): U+0000 to U+FFFF (65,536 code points)
- Supplementary Multilingual Plane (SMP): U+10000 to U+1FFFF (65,536 code points)
- Supplementary Ideographic Plane (SIP): U+20000 to U+2FFFF (65,536 code points)
- Tertiary Ideographic Plane (TIP): U+30000 to U+3FFFF (65,536 code points)
- Supplementary Special-purpose Plane (SSP): U+E0000 to U+EFFFF (65,536 code points)
- Supplementary Private Use Area-A (PUA-A): U+F0000 to U+FFFFF (65,536 code points)
- Supplementary Private Use Area-B (PUA-B): U+100000 to U+10FFFF (65,536 code points)

In total, Unicode contains approximately 1,114,112 code points across 17 planes, each with 65,536 code points.

III. METHODOLOGY

The Vigenère cipher is a technique of polyalphabetic substitution that employs a keyword to encode a message. Typically, it operates using a restricted alphabet, like the 26 letters of the English alphabet. This project seeks to modernize this traditional cipher by expanding its capabilities to support the Unicode

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.17666872

character set, which encompasses characters from almost all of the world's writing systems. The main focus of the system design and methodology is to develop a strong and flexible encryption process that is not limited by a fixed character set, making it adaptable to a wide range of characters.

A. System Design

The system design in figure 1 for this project will be modular, comprising separate components for managing plaintext, the keyword, and the primary encryption/decryption processes. The traditional Vigenère cipher is susceptible to attacks such as the Kasiski examination and Friedman test, and is limited to a 26-character alphabet. This modern adaptation improves upon it by: supporting Unicode (extending the algebraic operations across the entire Unicode code space, such as UTF-8), strengthening the key through a Key Derivation Function (KDF) to generate a secure, pseudo-random key stream from a human-memorable password, ensuring encoding awareness by separating cryptographic operations from string encoding and handling data as bytes for increased robustness, and maintaining modularity to facilitate integration into larger applications or security systems.

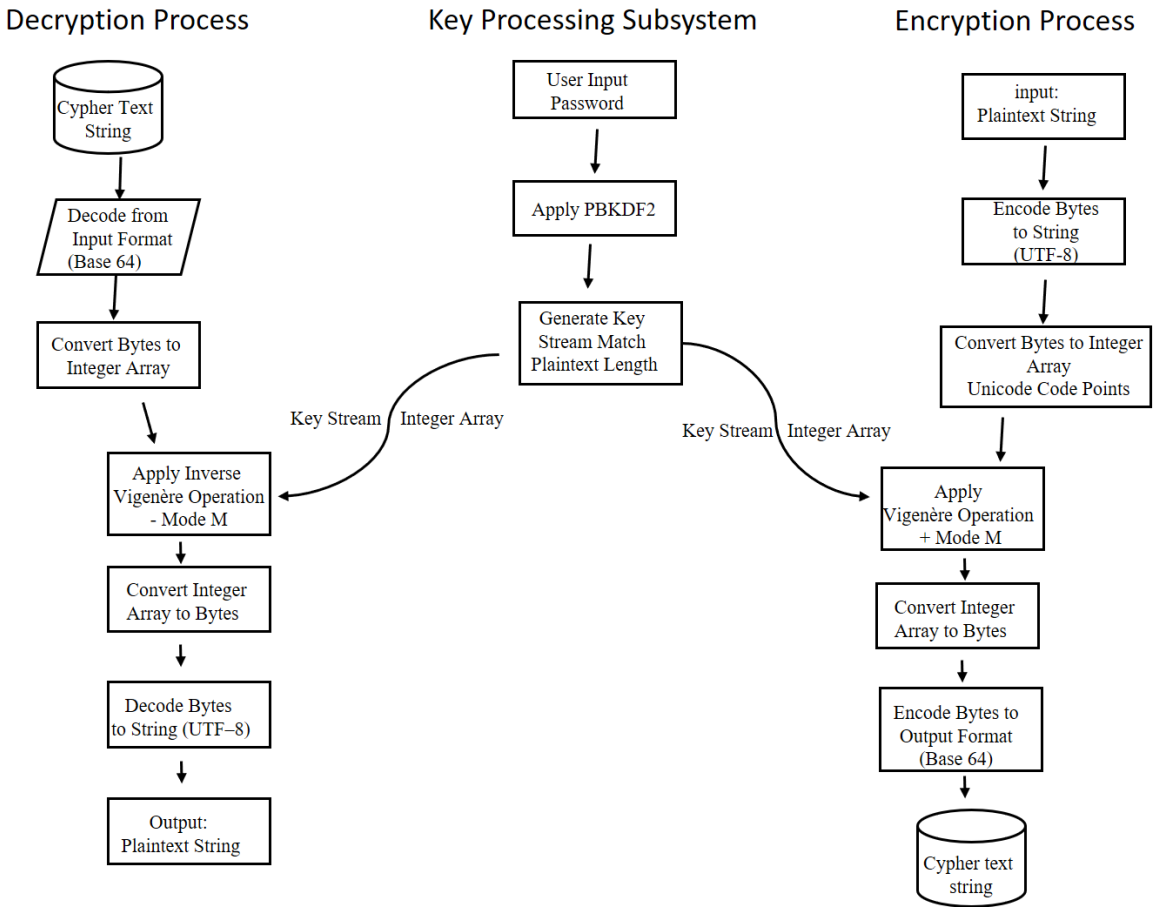

Figure 1: System Architecture

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.17666872

(a). User Interface (UI) Module

Purpose: Facilitates user interaction by providing an interface for data input and command execution.
Inputs: Operation type (encrypt or decrypt), inputText (plaintext or ciphertext), and password (user's key).
Outputs: Sends these parameters to the Core Engine for processing; receives the resulting data and displays it to the user.
Format: Implemented as a graphical user interface (GUI).

(b). Key Processing Module

Purpose: Converts a short, variable-length password into a cryptographically secure key stream.
Components:
Key Derivation Function (KDF): Accepts the user's password and a randomly generated salt. Utilizes PBKDF2 to produce a master key, enhancing security against brute-force and dictionary attacks. Generates a unique, random salt (at least 128 bits or more) for each password or secret. Combines the salt with the password during KDF computation, typically hashing or processing (password || salt) multiple times to slow down attacks. Stores the salt openly alongside the derived key to enable future verification, ensuring it is not kept secret.
Output: An array of integers representing the generated key stream.

(c). **Data Handling and Character Mapping**

Rather than depending on a basic A-Z table, the system should assign a distinct integer to each character in the Unicode set by directly using the character's Unicode code point as its numerical value. This numerical representation will serve as the foundation for all cryptographic processes.

(d). **Keyword Handling**

The keyword would similarly be transformed into a sequence of numerical values using the same character mapping. When the keyword is shorter than the plaintext, a key design choice in the Vigenère cipher is to repeat the keyword cyclically. This system would implement a continuous repetition of the keyword's numerical sequence. Employing the entire Unicode range ensures maximum entropy, prevents ASCII bias, and simplifies calculations. If the output includes non-printable characters, they are simply valid code points at the internal level.

(e). Error Handling

Any code point exceeding Unicode's maximum valid value (0x10FFFF) is discarded, as modulo operations will wrap values back into the valid range. The mathematical model of the Vigenère cipher encryption process can be expressed by equation (1):

$$Ci = (Pi + Ki) mod 26 \quad (1)$$

where:

- Ci is the encrypted character at position i
- Pi is the plaintext character at position i
- Ki is the i^{th} character in the keyword, shifted down by 'a' to get a number
- 26 is the number of letters in the alphabet

In this study, a modified version of the Vigenère cipher is used, altering the standard algorithm. The modification involves changing the modulus value to encompass the full Unicode range. Equation (2) reflects this adjustment:

$$C[i] = (P[i] + K[i mod |K|]) mod M \quad (2)$$

where:
$C[i]$: This represents the ith element of the output (ciphertext) array.
$P[i]$: This represents the ith element of the input (plaintext) array that we want to encrypt

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.17666872

$K[i \ mod \ |K|]$: indicates the character at position i in the keyword, with |K| being the length of the key; the modulo operation ensures that if the plaintext is longer than the key, the key repeats cyclically M: represents the maximum Unicode value, which is 1,114,112, corresponding to the full Unicode range up to U+10FFFF
The approach adheres to a conventional software development lifecycle, emphasizing rigorous testing and validation to verify that the improved cipher is accurate and secure. Python 3.13 is utilized as the programming language for the implementation. The graphical user interface (GUI) is built using the tkinter package, which provides a lightweight, object-oriented wrapper around the cross-platform tcl/Tk toolkit. The software was executed on a Windows 10 Pro system, version 20H2.

```
import tkinter as tk
from tkinter import messagebox
```

The Vigenère cipher is a traditional encryption method that improves plaintext security by shifting characters based on a keyword. The Python implementation presented illustrates how to perform encryption using this approach, highlighting the conversion between characters and their Unicode equivalents. With Unicode compatibility, the cipher can be developed using contemporary programming languages and libraries that natively support Unicode. This support allows the cipher to process non-ASCII characters, surrogate pairs, and other special symbols, thereby increasing its versatility and suitability for a wide range of applications.

B. Research and Design Phase

The Vigenère cipher is a polyalphabetic substitution cipher derived from an altered Caesar cipher. Among polyalphabetic ciphers, it is regarded as one of the most secure encryption methods. In this cipher, the keys are user-input characters [11]. As a form of modern cryptography, it employs more advanced and intricate message encoding techniques developed to address the vulnerabilities of traditional cryptographic methods and to safeguard information against more powerful attacks [12]. Figure 2 is the traditional square table for Vigenère cipher.

C. Implementation Phase

The system is developed using a programming language with robust Unicode support, such as Python. Python is widely favored for its simplicity, readability, versatility, strong community backing, and extensive libraries, making it an ideal choice for both beginners and experienced programmers. The implementation is divided into modules as specified in the system design: a character mapping utility, a keyword processing function, and the main encryption/decryption routines.

D. Testing and Validation Phase

A comprehensive testing process includes:

(a). Unit Tests: Confirming that individual modules, like character mapping and modular arithmetic functions, function correctly in isolation.
(b). Integration Tests: Ensuring that all modules work together smoothly without issues.
(c). End-to-End Testing: Encrypting various Unicode text samples—including texts in different languages, emojis, and special characters—and verifying that decrypting them restores the original plaintext exactly.
(d). Security Analysis: Although the Vigenère cipher is not considered secure by modern standards, the analysis examines its vulnerabilities when applied to Unicode data. This involves testing against common cryptanalytic techniques such as frequency analysis and methods like the Kasiski examination, which identifies repeated sequences to determine the key length. The evaluation also considers how the expanded character set impacts the effectiveness of brute-force attacks on the keyword.

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.17666872

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Figure 2: Vigenère Cipher square table

IV. RESULTS AND DISCUSSIONS

We successfully created an enhanced Vigenère cipher that handles all Unicode characters, dramatically improving its security and real-world usefulness. By leveraging the vast Unicode keyspace, the cipher can encrypt multilingual text and emojis with the same strong polyalphabetic encryption as the original, but without data loss. It runs efficiently nearly as fast as the old ASCII version proving that this classic method is now ready for modern, global communication needs.

A. Decryption

The decryption phase is explained below:

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.17666872

```
def Vigenère_encrypt(plaintext, key):
    """Encrypt plaintext using the Vigenère cipher with the given key."""
    encrypted_text = []
    key_length = len(key)
    key_as_int = [ord(i) for i in key]  # Convert key characters to their Unicode integer
values
    plaintext_int = [ord(i) for i in plaintext] # Convert plaintext characters to their Unicode
nteger values
    for i in range(len(plaintext_int)):
        # Calculate the encrypted character using the Vigenère formula
        value = (plaintext_int[i] + key_as_int[i % key_length]) % 0x110000  # Wrap around
he full Unicode range
        encrypted_text.append(chr(value))  # Convert the integer back to a character

    return ''.join(encrypted_text)  # Join the list into a single string and return
```

Figure 3: The Vigenère Cipher Encryption Code

Table 1: Views on how the Program Generate its Cipher

	A a	B b	C c	D d	E e	F f	G g	H h	I i	J j	K k	L l	M m	N n	O o	P p	Q q	R r	S s	T t	U v	V v	W w	X x	Y y	Z z	..
a	¢ Â	£ Ã	¤ Ä	¥ Å	¦ Æ	§ Ç	¨ È	© É	ª Ê	« Ë	¬ Ì	- Í	® Î	¯ Ï	° Đ	± Ñ	² Ò	³ Ó	´ Ô	µ Õ	¶ Ö	· ×	¸ Ø	¹ Ù	º Ú	» Û	
b	£ Ã	¤ Ä	¥ Å	¦ Æ	§ Ç	¨ È	© É	ª Ê	« Ë	¬ Ì	- Í	® Î	¯ Ï	° Đ	± Ñ	² Ò	³ Ó	´ Ô	µ Õ	¶ Ö	· ×	¸ Ø	¹ Ù	º Ú	» Û	¼ Ü	
c	¤ Ä	¥ Å	¦ Æ	§ Ç	¨ È	© É	ª Ê	« Ë	¬ Ì	- Í	® Î	¯ Ï	° Đ	± Ñ	² Ò	³ Ó	´ Ô	µ Õ	¶ Ö	· ×	¸ Ø	¹ Ù	º Ú	» Û	¼ Ü	½ Ý	
d	¥ Å	¦ Æ	§ Ç	¨ È	© É	ª Ê	« Ë	¬ Ì	- Í	® Î	¯ Ï	° Đ	± Ñ	² Ò	³ Ó	´ Ô	µ Õ	¶ Ö	· ×	¸ Ø	¹ Ù	º Ú	» Û	¼ Ü	½ Ý	¾ Þ	
e	¦ Æ	§ Ç	¨ È	© É	ª Ê	« Ë	¬ Ì	- Í	® Î	¯ Ï	° Đ	± Ñ	² Ò	³ Ó	´ Ô	µ Õ	¶ Ö	· ×	¸ Ø	¹ Ù	º Ú	» Û	¼ Ü	½ Ý	¾ Þ	¿ ß	
f	§ Ç	¨ È	© É	ª Ê	« Ë	¬ Ì	- Í	® Î	¯ Ï	° Đ	± Ñ	² Ò	³ Ó	´ Ô	µ Õ	¶ Ö	· ×	¸ Ø	¹ Ù	º Ú	» Û	¼ Ü	½ Ý	¾ Þ	¿ ß	À à	

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.17666872

g	¨ È	© É	ª Ê	« Ë	¬ Ì	- Í	® Î	¯ Ï	° Ð	± Ñ	² Ò	³ Ó	´ Ô	µ Õ	¶ Ö	· ×	¸ Ø	¹ Ù	º Ú	» Û	¼ Ü	½ Ý	¾ Þ	¿ ß	À à	Á á	
h	© É	ª Ê	« Ë	¬ Ì	- Í	® Î	¯ Ï	° Ð	± Ñ	² Ò	³ Ó	´ Ô	µ Õ	¶ Ö	· ×	¸ Ø	¹ Ù	º Ú	» Û	¼ Ü	½ Ý	¾ Þ	¿ ß	À à	Á á	Â â	
i	ª Ê	« Ë	¬ Ì	- Í	® Î	¯ Ï	° Ð	± Ñ	² Ò	³ Ó	´ Ô	µ Õ	¶ Ö	· ×	¸ Ø	¹ Ù	º Ú	» Û	¼ Ü	½ Ý	¾ Þ	¿ ß	À à	Á á	Â â	Ã ã	
j	« Ë	¬ Ì	- Í	® Î	¯ Ï	° Ð	± Ñ	² Ò	³ Ó	´ Ô	µ Õ	¶ Ö	· ×	¸ Ø	¹ Ù	º Ú	» Û	¼ Ü	½ Ý	¾ Þ	¿ ß	À à	Á á	Â â	Ã ã	Ä ä	
k	¬ Ì	- Í	® Î	¯ Ï	° Ð	± Ñ	² Ò	³ Ó	´ Ô	µ Õ	¶ Ö	· ×	¸ Ø	¹ Ù	º Ú	» Û	¼ Ü	½ Ý	¾ Þ	¿ ß	À à	Á á	Â â	Ã ã	Ä ä	Å å	
l	- Í	® Î	¯ Ï	° Ð	± Ñ	² Ò	³ Ó	´ Ô	µ Õ	¶ Ö	· ×	¸ Ø	¹ Ù	º Ú	» Û	¼ Ü	½ Ý	¾ Þ	¿ ß	À à	Á á	Â â	Ã ã	Ä ä	Å å	Æ æ	
m	® Î	¯ Ï	° Ð	± Ñ	² Ò	³ Ó	´ Ô	µ Õ	¶ Ö	· ×	¸ Ø	¹ Ù	º Ú	» Û	¼ Ü	½ Ý	¾ Þ	¿ ß	À à	Á á	Â â	Ã ã	Ä ä	Å å	Æ æ	Ç ç	
n	¯ Ï	° Ð	± Ñ	² Ò	³ Ó	´ Ô	µ Õ	¶ Ö	· ×	¸ Ø	¹ Ù	º Ú	» Û	¼ Ü	½ Ý	¾ Þ	¿ ß	À à	Á á	Â â	Ã ã	Ä ä	Å å	Æ æ	Ç ç	È è	
o	° Ð	± Ñ	² Ò	³ Ó	´ Ô	µ Õ	¶ Ö	· ×	¸ Ø	¹ Ù	º Ú	» Û	¼ Ü	½ Ý	¾ Þ	¿ ß	À à	Á á	Â â	Ã ã	Ä ä	Å å	Æ æ	Ç ç	È è	É é	
p	± Ñ	² Ò	³ Ó	´ Ô	µ Õ	¶ Ö	· ×	¸ Ø	¹ Ù	º Ú	» Û	¼ Ü	½ Ý	¾ Þ	¿ ß	À à	Á á	Â â	Ã ã	Ä ä	Å å	Æ æ	Ç ç	È è	É é	Ê ê	
q	² Ò	³ Ó	´ Ô	µ Õ	¶ Ö	· ×	¸ Ø	¹ Ù	º Ú	» Û	¼ Ü	½ Ý	¾ Þ	¿ ß	À à	Á á	Â â	Ã ã	Ä ä	Å å	Æ æ	Ç ç	È è	É é	Ê ê	Ë ë	
r	³ Ó	´ Ô	µ Õ	¶ Ö	· ×	¸ Ø	¹ Ù	º Ú	» Û	¼ Ü	½ Ý	¾ Þ	¿ ß	À à	Á á	Â â	Ã ã	Ä ä	Å å	Æ æ	Ç ç	È è	É é	Ê ê	Ë ë	Ì ì	
s	´ Ô	µ Õ	¶ Ö	· ×	¸ Ø	¹ Ù	º Ú	» Û	¼ Ü	½ Ý	¾ Þ	¿ ß	À à	Á á	Â â	Ã ã	Ä ä	Å å	Æ æ	Ç ç	È è	É é	Ê ê	Ë ë	Ì ì	Í í	
t	µ Õ	¶ Ö	· ×	¸ Ø	¹ Ù	º Ú	» Û	¼ Ü	½ Ý	¾ Þ	¿ ß	À à	Á á	Â â	Ã ã	Ä ä	Å å	Æ æ	Ç ç	È è	É é	Ê ê	Ë ë	Ì ì	Í í	Î î	
u	¶ Ö	· ×	¸ Ø	¹ Ù	º Ú	» Û	¼ Ü	½ Ý	¾ Þ	¿ ß	À à	Á á	Â â	Ã ã	Ä ä	Å å	Æ æ	Ç ç	È è	É é	Ê ê	Ë ë	Ì ì	Í í	Î î	Ï ï	
v	· ×	¸ Ø	¹ Ù	º Ú	» Û	¼ Ü	½ Ý	¾ Þ	¿ ß	À à	Á á	Â â	Ã ã	Ä ä	Å å	Æ æ	Ç ç	È è	É é	Ê ê	Ë ë	Ì ì	Í í	Î î	Ï ï	Ð ð	
w	¸ Ø	¹ Ù	º Ú	» Û	¼ Ü	½ Ý	¾ Þ	¿ ß	À à	Á á	Â â	Ã ã	Ä ä	Å å	Æ æ	Ç ç	È è	É é	Ê ê	Ë ë	Ì ì	Í í	Î î	Ï ï	Ð ð	Ñ ñ	
..																											

For encryption example,
The plaintext is (TO BE AT LOGGERHEAD WITH SOMEONE)
Keyword is “Apple-@()”
The encryption is iÄßŒÇ’`‰□aÜßÓÌ’²□□¢Ô□ãÎ¡¨Hœ°ÝÕÛÓ’i
The combination of Unicode characters in the key can lead to a significantly larger key space, thus enhancing the overall cryptographic strength due to a reduced likelihood of key collision.
The function *Vigenère_encrypt* in figure 3 is designed to take plaintext and key as inputs. It initializes an empty list called *encrypted_text* to hold the encrypted characters. The *ord()* function converts each character

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.17666872

of the key and plaintext into its corresponding Unicode integer value. A loop then iterates through each character of the plaintext, encrypting it by adding its Unicode value to that of the corresponding key character (cycling through the key using modulo arithmetic). The sum is wrapped within the full Unicode range using % 0x110000. This integer result is converted back to a character with *chr()* and appended to the *encrypted_text* list. After processing all characters, the list is joined into a single string, which is returned as the encrypted output. The encryption process completes in approximately 0.000002 seconds when measured with *perf_counter*. When the code is run, the interface is shown in figure 4 and the result of each input is shown in Table 1.

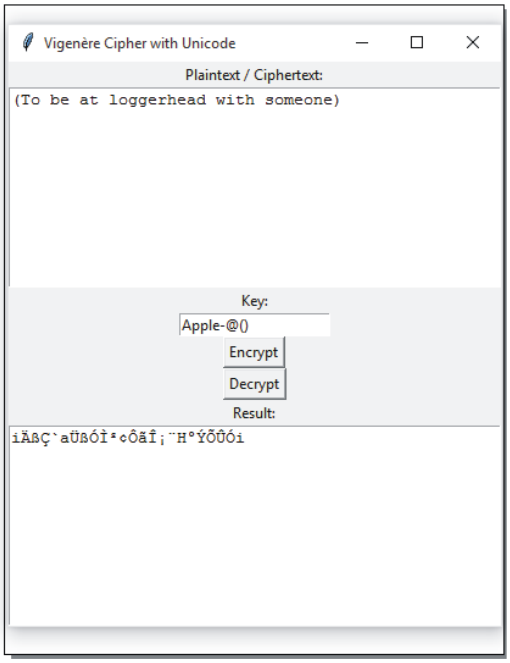

Figure 4: Encryption Screen Shot

B. Decryption

The *Vigenère_decryption* function offers a straightforward and effective approach for reversing text encrypted with the Vigenère cipher. Utilizing Unicode characteristics and modular arithmetic, the function accurately restores the original plaintext from the ciphertext.

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.17666872

```
def Vigenère_decrypt(ciphertext, key):
    """Decrypt ciphertext using the Vigenère cipher with the given key."""
    decrypted_text = []
    key_length = len(key)
    key_as_int = [ord(i) for i in key]  # Convert key characters to their Unicode integer
values
    ciphertext_int = [ord(i) for i in ciphertext]  # Convert ciphertext characters to their
Unicode integer values
    for i in range(len(ciphertext_int)):
        # Calculate the decrypted character's Unicode value
        value = (ciphertext_int[i] - key_as_int[i % key_length]) % 0x110000  # Wrap around
the full Unicode range
        decrypted_text.append(chr(value))    # Convert the Unicode value back to a
character
    return ''.join(decrypted_text)    # Join the list of characters into a single string
```

Figure 5: The Vigenere Cipher Decryption Code

For decryption,
Cipher text is iÄßŒÇ’`‰□aÜßÓÌ’²□□¢Ô□ãÎ¡¨Hœ°ÝÕÛÓ’i
Keyword is “Apple-@()”
The plaintext is (TO BE AT LOGGERHEAD WITH SOMEONE)
The function begins by creating an empty list called *decrypted_text* in figure 5 to hold the decrypted characters. It calculates the length of the key and converts both the key and the ciphertext into lists of their respective Unicode integer values. A loop then processes each character in the ciphertext; for each, it determines the decrypted value by subtracting the key’s corresponding character value (cycling through the key with modulo) from the ciphertext character’s value. The result is wrapped within the full Unicode range to ensure valid character codes. Each resulting integer is converted back to a character using *chr()* and added to the *decrypted_text* list. Finally, the list is joined into a single string and returned as the decrypted message. The process completes in approximately 0.000002 seconds, successfully decrypting all special characters. When the code is run, the interface is shown in Figure 6

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.17666872

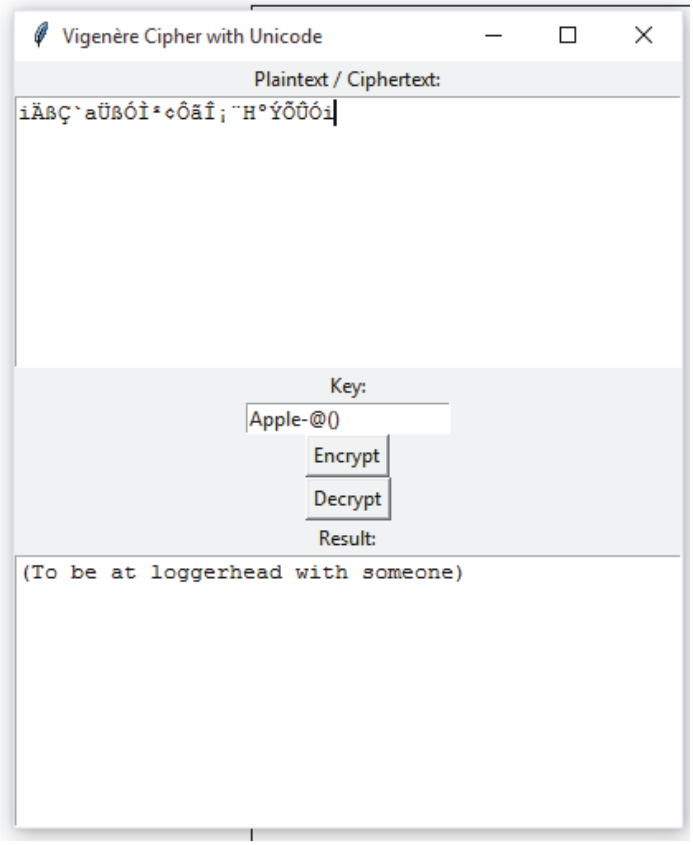

Figure 6: Decryption Screen Shot

C. Advantages

The enhancement of the Vigenère cipher with Unicode support provides a versatile combination of universality, security, flexibility, and compatibility, making it well-suited for modern encryption applications. The extensive range of Unicode allows for the seamless inclusion of diverse characters and scripts, supporting global communication and increasing the cipher's complexity against traditional cryptanalysis. Overall, this adaptation modernizes the classic cipher, aligning it with current technological standards. Unicode support enables the cipher to handle languages with non-ASCII characters, such as Chinese, Japanese, and Korean.

D. Comparison with Other Contemporary Vigenère Variations

Table 2 shows various other modifications suggested in cryptographic studies aim to tackle the primary weaknesses of the Vigenère cipher: its restricted character set and its vulnerability to cryptanalysis caused by the repetitive use of the key.

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.17666872

Table 2: Comparison of Alternative Modifications

Modification Type	Primary Goal	Methodology	Methodology
Randomized Keys/Hybrid Systems	To improve security against frequency analysis and brute-force attacks.	These approaches often combine the Vigenère cipher with other cryptographic techniques, such as stream ciphers or asymmetric encryption, employing complex algorithms to generate non-repeating or randomized keys.	While some hybrid methods expand the character set (e.g., to ASCII), their main focus is on making the key and encryption process more complex and unpredictable. The proposed project does not necessarily utilize a randomized key.
Extended Character Tables	To enhance both usability and security by enlarging the key space.	This approach is closely related to the proposed project, involving larger Vigenère tables—often 95x95—to include alphanumeric characters, symbols, and special characters.	The proposed project advances this idea further by aiming for full Unicode compatibility, which encompasses a much broader set of characters. While extended tables are a form of this, they are still limited compared to the entire Unicode standard.
Diffusion and Confusion	To make the link between plaintext, ciphertext, and key as intricate as possible.	These strategies add steps like transposing (rearranging) characters after substitution or layering multiple encryption stages. Some proposals also employ a "dynamic key" that changes with each character, avoiding repetition.	The proposed project, based on its title, emphasizes a single-layer substitution. These other methods introduce additional complexity to counteract attacks like Kasiski examination and the Index of Coincidence, which exploit repeated patterns.

The proposed project aims to adapt the Vigenère cipher for effective use within a Unicode environment, addressing a vital challenge to ensure the algorithm remains relevant. In contrast, other modifications primarily focus on enhancing security by incorporating advanced cryptographic principles such as randomness, improved key management, and hybrid techniques, which are essential for defending against contemporary attacks. Table 3 shows various mode for encryption.

E. Key Improvements

a) Unicode Support: The upgraded cipher now accommodates all Unicode characters, including symbols, emojis, and characters from non-English languages.
b) Character Encoding: It employs a standardized encoding scheme to ensure consistent and reliable encryption and decryption across different systems.
c) Key Generation: The process for generating keys has been adjusted to handle the expanded character set, ensuring keys are sufficiently long to maintain robust security.
d) Encryption and Decryption: These processes are revised to effectively manage the broader range of Unicode characters, guaranteeing accurate encryption and decryption of all such characters.

Volume 1, Issue 4, 2025
https://doi.org/10.5281/zenodo.17666872

Table 3: Comparison of classical, ASCII-based, and Unicode-based Vigenère ciphers

Feature	Classical Vigenère (mod-26)	ASCII-Extended Vigenère (mod-256)	Enhanced Unicode Vigenère
Alphabet Size (m)	26 (English alphabet A-Z)	256 (complete 8-bit byte range)	Very large (e.g., over 65,536 for UTF-16, approximately 1.1 million for full Unicode)
Formula	Ci = (Pi + Ki) mod 26	Ci = (Pi + Ki) mod 256	Ci = (Pi + Ki) mod m (where m is the code point range)
Supported Data	Only English plaintext (case-insensitive)	Basic text and any single-byte data (e.g., binary files)	Global language support, emojis, and special symbols
Security	Very weak; easily broken using frequency analysis and Kasiski/Friedman tests	Weak/obsolete; larger alphabet offers minimal security improvement, still vulnerable to known-plaintext and ciphertext-only attacks	Slightly better; larger modulus (m) complicates frequency analysis due to a broader key space, but it remains vulnerable if the key repeats
Practicality	No (mainly for historical or educational purposes)	Limited (suitable for legacy systems or simple byte-level tasks)	Limited (provides character compatibility but no real security)

F. Benefits

a) Enhanced Security: Supporting a larger character set makes the cipher more resistant to frequency analysis and other cryptanalytic attacks.
b) Greater Flexibility: It can now encrypt any Unicode character, making it highly applicable to modern computing environments.
c) Broader Compatibility: The improved cipher works seamlessly across various languages and character sets, increasing its versatility as an encryption tool.

By adapting the Vigenère cipher for Unicode compatibility, we have developed a more flexible and secure encryption method suitable for contemporary applications.

V. CONCLUSION

The traditional Vigenère cipher's encryption and decryption methods are limited to alphabetic characters, restricting its applicability in modern contexts where text often includes symbols, emojis, and non-English characters. Enhancing the cipher for Unicode support broadens its capacity to encrypt any Unicode character, making it more relevant for today's diverse digital communication needs.

REFERENCES

[1] Alanazi, N., Khan, E., & Gutub, A. (2022). Inclusion of Unicode Standard seamless characters to expand Arabic text steganography for secure individual uses. *Journal of King Saud University - Computer and Information Sciences*, *34*(4), 1343–1356. https://doi.org/10.1016/j.jksuci.2020.04.011
[2] Cahyanti, N. D., Turmudi, T., & Khudzaifah, M. (2023). Modifikasi Vigenere Cipher Menggunakan Grup Simetri untuk Mengamankan Pesan Teks. *Jurnal Riset Mahasiswa Matematika*, *2*(5), 173–185. https://doi.org/10.18860/jrmm.v2i5.16791
[3] Hammad, R., Latif, K. A., Amrullah, A. Z., Hairani, Subki, A., Irfan, P., Zulfikri, M., M, L. Z. A., Innuddin, M., & Marzuki, K. (2022). Implementation of combined steganography and cryptography

vigenere cipher, caesar cipher and converting periodic tables for securing secret message. *Journal of Physics: Conference Series*, *2279*(1), 012006. https://doi.org/10.1088/1742-6596/2279/1/012006

[4] Hassan, A. (2024). Analysis and Modification of Vigenère Cypher: *Journal of Mathematical Sciences & Computational Mathematics* · https://doi.org/10.15864/jmscm.5410.

[5] Keshta, I., Aoudni, Y., Sandhu, M., Singh, A., Xalikovich, P. A., Rizwan, A., Soni, M., & Lalar, S. (2023). Blockchain aware proxy re-encryption algorithm-based data sharing scheme. *Physical Communication*, *58*, 102048. https://doi.org/10.1016/j.phycom.2023.102048

[6] Malaghan, A., Aditya M.., Ajith B, Anjana S., & Karthik Gpn. (2022). Combined Steganography and Image Cryptography System for Secure Data Transfer. *ACS Journal for Science and Engineering*, *2*(1), 63–72. https://doi.org/10.34293/acsjse.v2i1.28

[7] Mir, U. H., Lone, P. N., Singh, D., & Mishra, D. C. (2023). A public and private key image encryption by modified approach of Vigener cipher and the chaotic maps. *The Imaging Science Journal*, *71*(1), 82–96. https://doi.org/10.1080/13682199.2023.2175436

[8] Mohammed, A.-A., & Olaniyan, A. (2016). Vigenere Cipher: Trends, Review and Possible Modifications. *International Journal of Computer Applications*, *135*(11), 46–50. https://doi.org/10.5120/ijca2016908549

[9] Nahar, K., Chakraborty, P.,(2020). A Modified Version of Vigenere Cipher using 95 95 Table. *International Journal of Engineering and Advanced Technology*, *9*(5), 1144–1148. https://doi.org/10.35940/ijeat.E9941.069520

[10] Purwanti, Nurcahya, S. D., & Nazelliana, D. (2024). Message Security in Classical Cryptography Using the Vigenere Cipher Method. *International Journal Software Engineering and Computer Science (IJSECS)*, *4*(1), 350–357. https://doi.org/10.35870/ijsecs.v4i1.2263

[11] Qowi, Z., & Hudallah, N. (2021). Combining caesar cipher and hill cipher in the generating encryption key on the vigenere cipher algorithm. *Journal of Physics: Conference Series*, *1918*(4), 042009. https://doi.org/10.1088/1742-6596/1918/4/042009

[12] Ramadhani, A.P., Tami, N.P., Lestari, A., Erkamim, M. (2024). Layered security model through integration of Vigenere and Hill Cipher in digital message encryption. Journal of Information System and Application Development Volume 2 Number 2, page 144-156 P-ISSN: 2988-5698 | E-ISSN: 2988-4721 https://jurnal.unmer.ac.id/index.php/jisad

[13] Raphael George A. Mendoza, Maurice Rubien C. Antonio, Raymund M. Dioses, Jonathan C. Morano, & Mark Christopher R. Blanco. (2024a). Enhancement of L. Voleti LSB technique applied to Image Steganography. *World Journal of Advanced Research and Reviews*, *22*(2), 1101–1106. https://doi.org/10.30574/wjarr.2024.22.2.1437

[14] Safitri, R., Prasetyo, P. W., Wijayanti, D. E., Arifin, S., Setyawan, F., & Repka, J. (2023). Text security by using a combination of the vigenere cipher and the rubik's cube method of size 4×4×4. *Al-Jabar : Jurnal Pendidikan Matematika*, *14*(2), 281–297. https://doi.org/10.24042/ajpm.v14i2.14276

[15] Singh, Y. (2021). Rectangular Generalized Vigenere Cipher", *International Journal of Research in Engineering and Science (IJRES) ISSN (Online): 2320-9364, ISSN (Print): 2320-9356 www.ijres.org Volume 9 Issue 9 ǁ PP. 75-81*

[16] Shoukat, N., Azam, M. (2022). An Improved Method of Vigenère Cipher to Securely Compress the Text by using Relative Frequency", International Journal of Innovative Science and Research Technology. Volume 7, Issue 10

[17] Subandi, A., Lydia, M. S., Sembiring, R. W., Zarlis, M., & Efendi, S. (2018). Vigenere cipher algorithm modification by adopting RC6 key expansion and double encryption process. *IOP Conference Series: Materials Science and Engineering*, *420*, 012119. https://doi.org/10.1088/1757-899X/420/1/012119