Improving data security with the utilization of matrix columnar transposition techniques

Tulus1,*, Syafraz Sy2, Kiki A. Sugeng3, Rinovia Simanjuntak4, and J.L. Marpaung5

1Mathematics Department, Universitas Sumatera Utara, Medan, Indonesia
2Mathematics Department, Universitas Negeri Andalas, Padang, Indonesia
3Mathematics Department, Universitas Indonesia, Depok, Indonesia
4Mathematics Department, Institut Teknologi Bandung, Bandung, Indonesia
5Student of Mathematics, Universitas Sumatera Utara, Medan, Indonesia

Abstract. The Graph Neural Network (GNN) is an advanced use of graph theory that is used to address complex network problems. The application of Graph Neural Networks allows the development of a network by the modification of weights associated with the vertices or edges of a graph G (V, E). Data encryption is a technique used to improve data security by encoding plain text into complex numerical configurations, hence minimizing the probability of data leaking. This study seeks to explain the potential of improving data security through the application of graph neural networks and transposition techniques for information manipulation. This study involves an algorithm and simulation that discusses the use of the transposition approach in manipulating information. This is accomplished by the implementation of a graph neural network, which develops the interaction between vertices and edges. The main result of this research shows empirical evidence supporting the notion that the length of the secret key and the number of characters utilized in data encryption have a direct impact on the complexity of the encryption process, hence influencing the overall security of the created data.

1 Introduction

The utilization of Graph Neural Network (GNN) in the realm of data security has emerged as a subject of growing significance due to the escalating intricacy of cyber dangers and the imperative to safeguard confidential information. The Graph Neural Network (GNN) is a machine learning methodology that is highly appropriate for the representation and analysis of data that possesses a network or relational structure, such as social graphs, communication networks, or IT architecture. The utilization of Graph Neural Network (GNN) has emerged as a potent methodology in tackling the progressively intricate data security issues. In the realm of cybersecurity, data frequently encompasses not only discrete data elements, but also exhibits intricate interconnections and organizational frameworks. The Graph Neural Network (GNN) is a machine learning methodology that facilitates the comprehension of data presented in a graphical format by comprehending the interrelationships among the various entities inside the graph [1]. The detection of cyber risks is a prominent use of Graph Neural Networks (GNN) in the field of data security. In the context of a communication network graph, Graph Neural Networks (GNN) possess the capability to discern potentially malicious traffic patterns and categorize nodes or connections that could potentially partake in a cyber-attack.

Furthermore, Graph Neural Networks (GNNs) can be effectively employed for the purpose of monitoring abnormalities in network structure and identifying atypical alterations in entity behaviour. The utilization of GNN is also observed in the domain of data security, namely for the purpose of permits and access verification [2]. Within the realm of access management systems, Graph Neural Networks (GNN) can be utilized to facilitate the modelling of role-based permissions and inter-entity interactions. The system enables the dynamic allocation of permissions based on the network structure, ensuring that only authorized entities are granted access. The financial transaction chart enables GNN to detect intricate fraudulent patterns by examining the interrelationships among various entities, including bank accounts and transactions. The use of security measures aids banks and financial organizations in safeguarding client data and mitigating financial losses resulting from fraudulent activities. The utilization of Graph Neural Networks (GNN) in the field of data security is experiencing significant and rapid expansion. This enables enterprises to enhance their ability to detect and safeguard their sensitive data by gaining a comprehensive comprehension of the network architecture and the interplay of entities inside their data. The utilization of Graph Neural Networks (GNN) in the realm of data security is expected to expand alongside technological advancements, in response to the escalating complexity of cyber threats [3–5].

* Corresponding author: tulus@usu.ac.id

© The Authors, published by EDP Sciences. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).
The transposition technique is a well-established cryptographic method employed to transform plaintext into ciphertext through the substitution of character or symbol sequences inside a message. The integration of transposition techniques with the Graph Neural Network (GNN) represents a novel and inventive strategy for enhancing data security. This approach involves the utilization of Graph Neural Networks (GNN) to reorganize nodes or letters within textual graphs, resulting in specific permutations that pose challenges for unauthorized entities to comprehend. The utilization of transposition techniques in conjunction with Graph Neural Networks (GNN) in the field of cryptography exhibits considerable promise in enhancing data security. Nevertheless, it is crucial to bear in mind that the integrity of data protection is contingent upon not alone the encryption methodology employed, but also the implementation of robust key management protocols and supplementary security measures. This technique signifies a significant advancement in safeguarding data in the current era of complex digital environments. One notable benefit of employing Graph Neural Networks (GNNs) in the domain of transposition is their inherent capability to comprehend the contextual information included within the dataset. The GNN model has the ability to consider the interconnections of characters and identify specific patterns that can go unnoticed by traditional transposition techniques. The final outcome is a robust ciphertext that presents a formidable challenge for an unauthorized entity lacking the requisite knowledge of the encryption key to decipher

The fundamental principle underlying the utilization of cryptography in data encryption is the identification of associations among sets of vertices that are interconnected by edges inside each subset of G(v,e) [7,8]. Given a plaintext denoted as \( P \) and ciphertext denoted as \( C \), the encryption function can be expressed as follows:

\[
E(P) = C
\]

(1)

The decryption process involves the mapping of the ciphertext to its corresponding plaintext in order to retrieve the encrypted information.

\[
D(C) = P
\]

(2)

In this scenario, the encryption-decryption function is executed collaboratively to restore the safeguarded data, thereby assuming the role of the process.

\[
D(E(P)) = P
\]

(3)

2 Methods

2.1 Hill Cipher Technique

The Hill Cipher is a cryptographic encryption method that uses a key matrix to substitute plaintext with ciphertext. The Hill Cipher, developed by Lester S. Hill in 1929, is renowned for its capacity to address the vulnerabilities inherent in basic substitution techniques, such as the Caesar Cipher. The Hill cipher algorithm utilizes a collection of characters, often comprising letters, extracted from a plaintext, and transforms them into a cluster of ciphertext characters through the utilization of a key matrix [9]. One notable characteristic of the Hill Cipher is the utilization of a key matrix, which is required to be a square matrix. The dimensions of the matrix are contingent upon the magnitude of the character block intended for encryption. During the encryption procedure, the plaintext is divided into segments consisting of \( n \) characters. Subsequently, each segment is transformed into a numerical vector by substituting letters with corresponding values based on their respective positions within the alphabet. One notable characteristic of the Hill Cipher is its capacity to encrypt messages using larger character blocks, hence enhancing its security. Nevertheless, the Hill Cipher algorithm is not without its vulnerabilities. In order to ensure successful decryption, it is imperative that the key matrix employed possesses an inverse modulo \( n \). Failure to identify the inverse matrix can result in a complex decryption process. Moreover, given that the key matrix is publicly accessible, an adversary with knowledge of both the plaintext and the corresponding ciphertext can attempt to deduce the inverse of the key matrix. While the Hill Cipher algorithm may not be extensively employed in contemporary security applications, its conceptual significance remains crucial for a fundamental comprehension of cryptography. The aforementioned technique serves as an exemplification of the substitution concept that underlies numerous advanced encryption algorithms, including the Advanced Encryption Standard (AES).

In its application, Hill Cipher applied modulo arithmetic to enhance the security of communications and encryption. When given \( a \in \mathbb{Z} \) and \( m \in \mathbb{Z} > 0 \). Operation \( a \mod m = r \) such that \( a = mq + r \) with \( 0 \leq r \leq m \). On the Hill Cipher technique, the encryption process is mathematically a matrix conversion in order \( n \times n \).

\[
\begin{bmatrix}
c_1 \\
c_2 \\
\vdots \\
c_n
\end{bmatrix} = A \begin{bmatrix}
c_1 \\
c_2 \\
\vdots \\
c_n
\end{bmatrix} \mod A
\]

(4)

In the course of his study, Ravinder Kaur employed a lengthy square matrix in the encryption procedure, whilst the decryption method entailed the utilization of the left inverse of said square matrix. The encryption key matrix is utilized in the process of decrypting the ciphertext by employing the inverse generalized matrix with the inverse left. The concept of the Generalized Inverse of a Matrix represents a formal elucidation of the interpretation of the inverse matrix. The inverse of a \( m \times n \) matrix can be obtained by using the inverse of the generalized matrix. The length of the ciphertext resulting from the Hill cipher modification, utilizing the inverse of a square matrix with a longer dimension than the plaintext, serves to enhance the security of the message. Nevertheless, it is important to note that the method employed in this particular study was a variant of the Hill cipher, limiting the encryption of messages to a single iteration. The next phase of the study entails conducting research with advanced encryption techniques to investigate the practical application of compilation algorithms in the fields of cryptography, specifically focusing on transposition columnar and RSA methods. The primary objective is to enhance the
security of secret messages. The columnar transposition cipher is a cryptographic method that falls within the broader category of transposition techniques. In transposition techniques, the encryption process involves the use of permutations. The encryption procedure employed in the investigation involves the application of columnar first transposition, followed by re-encryption utilizing the RSA algorithm. To enhance the efficacy of message security, it is proposed to employ a composite cryptographic approach by integrating two distinct algorithms, hence referred to as the super encryption method.

### 2.2 Columnar Transposition

The Columnar Transposition technique is a well-established encryption method in the field of cryptography that is employed to obfuscate the order of characters in plaintext, resulting in the generation of ciphertext [10,11]. The technique employed in this process involves the reconfiguration of the content's character depending on a predetermined rule, typically utilizing a key or password that is possessed by the one seeking to perform the encryption and decryption. The encryption process of Columnar Transposition commences with the insertion of plaintext messages into a table or grid, often of square dimensions. The characters are subsequently inserted into the cells of the table, proceeding from left to right and row by row, following the sequence determined by the key. The outcome is a tabular representation wherein the characters have been reorganized. In order to produce ciphertext, the characters within this table will be selected sequentially from the top to the bottom, proceeding column by column. The aforementioned procedure will generate an altered sequence of characters, known as ciphertext, distinct from the initial plaintext. This procedure may serve as one of the phases within the super encryption methodology, wherein the Columnar Transposition approach can be amalgamated with other encryption methods to augment the degree of communication confidentiality. The Columnar Transposition technique is characterized by its simplicity and its ease of implementation in programming. While this method exhibits a certain level of resilience against basic forms of assaults, such as simple substitution or transposition, it is considered less robust in comparison to contemporary encryption algorithms employed in present-day cryptographic systems. Columnar Transposition is commonly employed as an introductory method in the field of cryptography, serving instructural and illustrative objectives.

The columnar transposition technique is a symmetric encryption system that employs an identical key for both the encryption and decryption processes. The utilization of the columnar transposition algorithm enhances the level of complexity in the encryption process inside this advanced encryption approach.

- The objective is to determine the columnar key transposition.
- Create a tabular representation consisting of two rows, where the first row corresponds to the plaintext and the second row has a number of columns equal to the length of the key.
- Arrange the column in ascending order based on the numerical sequence.
- The process of accessing encrypted text messages, often known as ciphertext.

While the process of decrypting messages using columnar transposition is:

- Determines the columnar transposition key used in the encryption process.
- Write ciphertext in a table of the first column and the next column with the number of rows corresponding to the length of cipher text divided by the key length.
- Sort columns in key sequence.
- Get original text (plaintext).

### 2.3 Encryption-Decryption Technique

The Hill cipher algorithm can be modified by applying the inverse operation on the left side of the encryption key. The key utilized in the conventional Hill cipher technique consists of a square matrix with dimensions \( n \times n \). The Hill cipher modification utilizes a square matrix of dimensions \( m \times n \) in the field \( \mathbb{Z}_p \) as the key. In this discourse, the author employs a methodology wherein the material, initially presented as letters, is subsequently transformed into numerical values within a \( \mathbb{Z}_p \) set. Consequently, calculations are performed within the confines of mod \( 53 \). If any plaintext encrypted using this approach can be successfully decrypted back to its original plaintext.

Let consider a scenario where \( x_1, x_2, ..., x_n \) represents a plaintext, and \( y_1, y_2, ..., y_n \) represents a corresponding ciphertext. Let \( x_1 \) and \( x_2 \) be two plaintexts belonging to the set of integers mod \( 53 \). Next, consider a key in the form of a \( 3 \times 2 \) square matrix.

\[
A = \begin{bmatrix} a & b \\ c & d \\ e & f \end{bmatrix} \in M_{3 \times 2}(\mathbb{Z}_{53})
\]

The implementation of alternatives against the Hill cipher encryption algorithm is currently considered.

\[
C = KP \mod n
\]

Then,

\[
P = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}
\]

\[
C = \begin{bmatrix} a & b \\ c & d \\ e & f \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \mod 53
\]

\[
C = \begin{bmatrix} ax_1 + bx_2 \\ cx_1 + dx_2 \\ ex_1 + fx_2 \end{bmatrix} \mod 53
\]

So, we will get the modulo equation

\[
ax_1 + bx_2 \equiv y_1 \mod 53
\]

\[
cx_1 + dx_2 \equiv y_2 \mod 53
\]

\[
ex_1 + fx_2 \equiv y_3 \mod 53
\]

Let \( a, b, c, d, e, f, x_1, x_2, x_3, y_1, y_2, y_3 \) be integers where \( (\Delta, n) = 1 \), so...
We eliminate the two equation above, so we get
\[ \begin{align*}
\Delta_1 &= dy_1 - by_2 \pmod{53} \\
\Delta_2 &= cy_1 - dy_2 \pmod{53}
\end{align*} \]
where,
\[ \begin{align*}
adx_1 + bdx_2 &= dy_1 \pmod{53} \\
bcx_1 + bdx_2 &= by_2 \pmod{53}
\end{align*} \] (11)
We eliminate the two equation above, so we get
\[ (ad - bc)x_1 = dy_1 - by_2 \pmod{53} \] (12)
Where \( \Delta = ad - bc \), Then multiply the two squares by \( \Delta \) which is the inverse of \( \Delta \), so that obtained so \( x_2 = \Delta (dy_1 - by_2) \pmod{53} \). The same phase we will get \( x_1 = \Delta (cy_1 - dy_2) \pmod{53} \).

On the modification of the Hill cipher algorithm with this square-long matrix with \( n = 53 \).

Table 1. Hill ciphertext modification \( Z_{53} \)

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>
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: f: g: h: i:
j: k: l: m: n:
o: p: q: r: s:
t: u: v: w: x:
y: z: Space: 52

2.4 Encryption Algorithm

The principle of modulo and its reading process are used to implement the algorithm for data encryption by transforming plaintext into ciphertext.

Encryption Algorithm

1. Determining columnar transposition key
2. The given text is transformed into a tabular format consisting of two rows
3. Sort columns by number sequence
4. Get the ciphertext
5. Determines the key matrix that has invers
6. Determine the plaintext \( p = x_1, x_2, x_3, \ldots, x_l \) and convert to matrix \( n \times 1 \pmod{53} \)

\[ P = P_1, P_2, P_3, \ldots, P_l \]

where
\[ P_1 = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \pmod{53}, P_2 = \begin{bmatrix} x_{n+1} \\ x_{n+2} \\ \vdots \\ x_{2n} \end{bmatrix} \pmod{53}, \]
\[ P_l = \begin{bmatrix} x_{(l-1)n+1} \\ x_{(l-1)n+2} \\ \vdots \\ x_{ln} \end{bmatrix} \pmod{53} \]

7. Second encryption with source key \( K_e \)

\[ C_i = (K_e P_i) \pmod{53} = (A P_i) \pmod{53} \]

8. Covert to ciphertext

\[ C = y_1, y_2, \ldots, y_n \]

2.5 Decryption Algorithm

The application of matrix transposition can be expressed as follows:
\[ A_{ij} = A_{ji}^T \]
so, \( (A_{ji}^T)^T = A_{ij} \) (13)
the decryption algorithm can be written as:

Decryption Algorithm

1. Find the ciphertext
2. Set the length of key
3. Substituted that \( P \equiv (C - K) \pmod{n} \)
4. Transpose the matrix \( (A_{ji}^T)^T = A_{ij} \)
5. Convert the plaintext to \( Z \)
6. Plaintext decryption

3 Result and Discussions

3.1 Experimental Simulation

In the present scenario, we will undertake a python-based simulation to perform data encryption. The encrypted data consists of the phrase “The Great Professor” and is secured using the “maestro” and “top” encryption key.

1. Set the length of secret key \( K_1 = 7, K_2 = 3 \)
   Note that, \( K_1 \) used to encrypt the data and \( K_2 \) for transpose the matrix
2. Arrange the plaintext into the matrix with 7 columns

Table 2. Data security matrix

<table>
<thead>
<tr>
<th>Char</th>
<th>T</th>
<th>h</th>
<th>e</th>
<th>X</th>
<th>G</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>code</td>
<td>19</td>
<td>33</td>
<td>30</td>
<td>52</td>
<td>6</td>
<td>43</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Char</th>
<th>e</th>
<th>a</th>
<th>t</th>
<th>X</th>
<th>P</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>code</td>
<td>30</td>
<td>26</td>
<td>45</td>
<td>52</td>
<td>15</td>
<td>43</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Char</th>
<th>o</th>
<th>f</th>
<th>e</th>
<th>s</th>
<th>s</th>
<th>o</th>
</tr>
</thead>
<tbody>
<tr>
<td>code</td>
<td>40</td>
<td>31</td>
<td>30</td>
<td>44</td>
<td>44</td>
<td>40</td>
</tr>
</tbody>
</table>

Table 3. Manipulation the plaintext

<table>
<thead>
<tr>
<th>Char</th>
<th>T</th>
<th>h</th>
<th>e</th>
<th>X</th>
<th>G</th>
<th>r</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>code</td>
<td>19</td>
<td>33</td>
<td>30</td>
<td>52</td>
<td>6</td>
<td>43</td>
<td>30</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C</th>
<th>26</th>
<th>40</th>
<th>37</th>
<th>6</th>
<th>13</th>
<th>50</th>
<th>37</th>
</tr>
</thead>
<tbody>
<tr>
<td>Char</td>
<td>a</td>
<td>t</td>
<td>X</td>
<td>P</td>
<td>r</td>
<td>o</td>
<td>f</td>
</tr>
<tr>
<td>code</td>
<td>26</td>
<td>45</td>
<td>52</td>
<td>15</td>
<td>43</td>
<td>40</td>
<td>31</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C</th>
<th>33</th>
<th>52</th>
<th>6</th>
<th>22</th>
<th>50</th>
<th>47</th>
<th>38</th>
</tr>
</thead>
<tbody>
<tr>
<td>Char</td>
<td>e</td>
<td>s</td>
<td>s</td>
<td>o</td>
<td>r</td>
<td></td>
<td></td>
</tr>
<tr>
<td>code</td>
<td>30</td>
<td>44</td>
<td>44</td>
<td>40</td>
<td>43</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C</th>
<th>37</th>
<th>51</th>
<th>51</th>
<th>47</th>
<th>50</th>
</tr>
</thead>
</table>
| a. Convert the plaintext \( C \equiv (P + K) \pmod{n} \)

b. Transposition the matrix using the secret key \( K_2 = 3 \)
Throughout the encryption process, a modification of the plaintext resulted in the ciphertext “aGlGvzeoMzWmzxlyxylvx”. The subsequent step involves the decryption of ciphertext into plaintext. Th is study shows empirical evidence supporting the notion that the length of the secret key and the number of characters utilized in data encryption and decryption, allowing it a suitable foundation for simulating noise in the transformation of data. The technique has been empirically demonstrated to exhibit the same effectiveness in both data encryption and decryption, hence influencing the overall security of the created data.

## 4 Conclusions

This study developed an algorithm that could act as a valuable resource to increase data security through the application of Modulo calculations on matrix transposition. The technique has been empirically demonstrated to exhibit the same effectiveness in both data encryption and decryption, allowing it a suitable foundation for simulating noise in the transformation of plaintext into ciphertext. This study shows empirical evidence supporting the notion that the length of the secret key and the number of characters utilized in data encryption have a direct impact on the complexity of the encryption process, hence influencing the overall security of the created data.

## Acknowledgement

We thankful to collaboration research named Riset Kolaborasi Indonesia 2023 and research institution Universitas Sumatera Utara with contract No: 22/UN5.2.3.1/PPM/KP-RKI/2023.

## References

4. B. Swami, R. Singh, and S. Choudhary, Procedia
6. A. Meenakshi, A. Kannan, and R. Cep, 1 (2023)
7. A. Razaq, G. Alhamzi, S. Abbas, and M. Ahmad, Heliyon 9, e15902 (2023)