Mahmud Yunus; Mohamad Ilham Dwi Firmansyah; Kistosil Fahim Subiono

A cryptography using lifting scheme integer wavelet transform over min-max-plus algebra

Kybernetika, Vol. 60 (2024), No. 5, 576–602

Persistent URL: http://dml.cz/dmlcz/152716

KYBERNETIKA — VOLUME *60* (2024), NUMBER 5, PAGES 576–602

A CRYPTOGRAPHY USING LIFTING SCHEME INTEGER WAVELET TRANSFORM OVER MIN-MAX-PLUS ALGEBRA

Mahmud Yunus, Mohamad Ilham Dwi Firmansyah, and Subiono

We propose a cryptographic algorithm utilizing integer wavelet transform via a lifting scheme. In this research, we construct some predict and update operators within the lifting scheme of wavelet transforms employing operations in min-max-plus algebra, termed as lifting scheme integer wavelet transform over min-max-plus algebra (MMPLS-IWavelet). The analysis and synthesis process on MMPLS-IWavelet is implemented for both encryption and decryption processes. The encryption key comprises a sequence of positive integers, where the first element specifies MMPLS-IWavelet type and subsequent elements indicate the levels of each executed transformation. The decryption key involves three components: the original encryption key, a binary encoding of the analyzed signal, and a sequence of non-negative integer representing the length of coefficient signals from the approximation and detail signals. We present a rigorous analysis confirming the correctness of the proposed cryptographic scheme, and evaluate its performance based on various metrics such as correlation value between plaintext and ciphertext, encryption quality, computation time, key sensitivity, entropy analysis, and key space analysis. We also analyze the computational costs of the encryption and decryption processes. The experimental results demonstrate that the proposed algorithms empirically yield satisfactory performance, exhibiting a near zero correlation between plaintext and ciphertext for most of test data, high encryption quality (over 80%), substantial key sensitivity, the large key space, and greater randomness in ciphertext compare to plaintext. The algorithm is efficient in terms of computational time and has linear complexity with respect to the number of input characters. The vast key space makes it highly impractical for brute-force approaches to find the decryption key directly.

Keywords: cryptography, lifting scheme, min-max-plus algebra, wavelet

Classification: 15A80, 94A60, 42C40

1. INTRODUCTION

The advancement of long-distance communication system has facilitated the exchange of information between individuals over extensive distances, necessitating secure and confidential communication channels. Cryptography provides a robust solution to ensure the security and privacy of such communication. Over the years, researchers

DOI: 10.14736/kyb-2024-5-0576

have developed various cryptographic algorithms. Among the pioneering works, the Diffie–Hellman protocol [11] stands out as the first public key cryptographic algorithm, utilizing the discrete logarithm within a large finite field. Following this, Rivest, Shamir, and Adleman introduce published the RSA protocol [19], based on the modulo property and factoring problem of the product of two large prime numbers. Cryptographic methods have also been applied to image-based messages. In [31], Zarkar et al. created an anti-phishing structure utilizing visual cryptography for image-based authentication with the RSA algorithm in the encryption process.

Wavelet transform is a mathematical technique that facilitates the analyze signals or data in the frequency and time domains simultaneously. The proposed method integrates the concepts of resolution analysis and Fourier transform. Wavelet transform enables the analysis of signals at various resolutions, providing more detailed information about the signal characteristics [14]. The basic concepts of wavelets have a long-standing presence in various fields, such as abstract analysis, signal processing, image processing, and theoretical physics [13]. The wavelet transforms are categorized into continuous and discrete type, with discrete wavelet transforms being widely utilized in cryptographic algorithm development. For example, in [8] a cryptographic algorithm using discrete wavelet transformation demonstrated short encryption times and tend to be constant for each number of input characters. This is proving useful for securing large commercial applications databases. Wavelet-based cryptographic algorithms have also enhanced image-based message security. For example, in [18, 28] introduce a wavelet transform to increase the security in image transmission.

A lifting scheme constructs a wavelet transform for signal decomposition and reconstruction, introduced by Sweldens [22, 23, 24].This scheme involves two main processes: prediction and updating, and is widely applied in wavelet transform, especially in transformations involving integer domain and codomain (see [5, 6, 7, 16, 20, 26, 29]). Some examples of wavelet transform involving integer domain and codomain for image compression can be seen in [4, 10, 17], wherein the mathematical operations are predicated on max-min-plus algebra. In addition, max-min-plus algebra has been applied in cryptographic algorithms. The investigations in [3] and [9] utilized the max-plus algebra semiring to develop encryption and decryption keys. Building upon these works, Cahyono et al. [2] formulated a cryptographic algorithm employing wavelet transform based on max-plus algebra (MP-Wavelet). This research, uses the MP-wavelet in [4], demonstrated a low correlation value between plaintext and ciphertext in empirical studies, signifying a weak statistical relationship. Consequently, it becomes exceedingly challenging to infer the structure of the plaintext based solely on the corresponding ciphertext. The proposed algorithm demonstrates good encryption quality and a relatively short running time, and the complexity of the algorithm is linear relative to the number of input characters.

In this research, we construct an integer wavelet transform employing lifting scheme over min-max-plus algebra, referred to as (MMPLS-IWevelet). The test data comprises text files in .txt format, each containing 65,536 characters from the Basic Multilingual Plane (BMP), represented in 16-bit encoding. We use maximum, minimum, and sum operations to construct predict and update operators, which enhance the algorithm's

execution efficiency. We construct some types of predict and update operators to provide variability in the encryption key sequence. The encryption key is formulated as an array of positive integers, where the first element specifies the type of transformation used, and the subsequent elements denote the level of the transformation to be executed. The decryption key consists of three components. The encryption key, a sequence of numbers derived from encoding the signal from the transformation, and an array of positive integers representing the lengths of signal coefficient approximations and details after executing some levels of transformations. Furthermore, we analyze the performance of the algorithm. Some aspects we analyze consist of computation time, correlation values between plaintext and ciphertext, encryption quality, entropy analysis, key sensitivity, and key space analysis.

The structure of this paper is organized as follows. Section 2 introduces min-max-plus algebra. In Section 3, we explain the mechanism of wavelet decomposition and reconstruction schemes. Section 4 describes the decomposition and reconstruction signals using a lifting scheme. Section 5 describes the integer wavelet transform using a min-max-plus lifting scheme (MMPLS-IWavelet). Section 6 discusses the main parts of this paper, i. e., we introduce cryptographic algorithms using MMPLS-IWavelet. Section 7 analyzes the performance of the proposed cryptographic algorithms. Finally, Section 8 provides the conclusion of this paper and open problems.

2. MIN-MAX-PLUS ALGEBRA

Max-plus algebra is a semi-idempoten algebraic structure. Suppose we have $\mathbb{R}_\varepsilon = \mathbb{R} \cup \{\varepsilon\}$ with $\varepsilon = -\infty$. For each $m, n \in \mathbb{R}_\varepsilon$ define

$$m \oplus n = \max\{m, n\} \quad \text{and} \quad m \otimes n = m + n.$$

The structure $(\mathbb{R}_\varepsilon, \oplus, \otimes)$ is called max-plus algebra where ε is neutral elements for $\oplus$ and $e = 0$ is the identity element for $\otimes$. Suppose $a \in \mathbb{R}_\varepsilon$ and $n \in \mathbb{N}$, exponents in max-plus algebra are defined by

$$a^{\otimes n} = \underbrace{a \otimes a \otimes \cdots \otimes a \otimes a}_{n \text{ times}} = n \times a, \tag{1}$$

and in general for $r \in \mathbb{R}$ we get $a^{\otimes r} = r \times a$. Suppose $-b$ denotes the usual additive inverse of b. The difference operator $\oslash$ is defined as $a \oslash b = a \otimes -b$ for all $a, b \in \mathbb{R}_\varepsilon$. Given $\mathbb{R}_{\varepsilon'} = \mathbb{R} \cup \{\varepsilon'\}$ with $\varepsilon' = +\infty$ and for every $m, n \in \mathbb{R}_{\varepsilon'}$ define

$$m \oplus' n = \min\{m, n\} \quad \text{and} \quad m \otimes n = m + n. \tag{2}$$

The structure $(\mathbb{R}_{\varepsilon'}, \oplus', \otimes)$ is called min-plus algebra. Given a mapping $T : \mathbb{R}_\varepsilon \to \mathbb{R}_{\varepsilon'}$ with $T(m) = am$ for each $m \in \mathbb{R}_\varepsilon$ and $a < 0$, it can be shown that T is an isomorphism that implies $\mathbb{R}_\varepsilon \cong \mathbb{R}_{\varepsilon'}$. Max-plus algebra or min-plus algebra have applications in various fields, including discrete-time systems, cryptography [2], image processing [4], control theory [32], transportation scheduling systems [12], queuing theory [15], and optimization problems [27].

3. WAVELET DECOMPOSITION AND RECONSTRUCTION SCHEME

Discrete wavelet transform decompose the initial signal into low-resolution (approximation) and high-resolution (detail). Wavelet decomposition and reconstruction can follow: coupled and uncoupled scheme decomposition [10]. In the coupled scheme, let V_i and W_i be two signal spaces. Analysis operator exist for approximation signals $\psi_i^{\uparrow} : V_i \to V_{i+1}$ and for detail signal $\omega_i^{\uparrow} : V_i \to W_{i+1}$. The synthesis operator $\Phi_i^{\downarrow} : V_{i+1} \times W_{i+1} \to V_i$. Each of these operators satisfies the following conditions:

$$\Phi_i^{\downarrow}\left(\psi_i^{\uparrow}(s_i), \omega_i^{\uparrow}(s_i)\right) = s_i, \tag{3}$$

$$\psi_i^{\uparrow}\left(\Phi_i^{\downarrow}\left(s_{i+1}, d_{i+1}\right)\right) = s_{i+1}, \tag{4}$$

$$\omega_i^{\uparrow}\left(\Phi_i^{\downarrow}\left(s_{i+1}, d_{i+1}\right)\right) = d_{i+1}, \tag{5}$$

for $s_i \in V_i$ and $d_i \in W_i$ where i is a non-negative integer. See Figure 1 for the illustration.

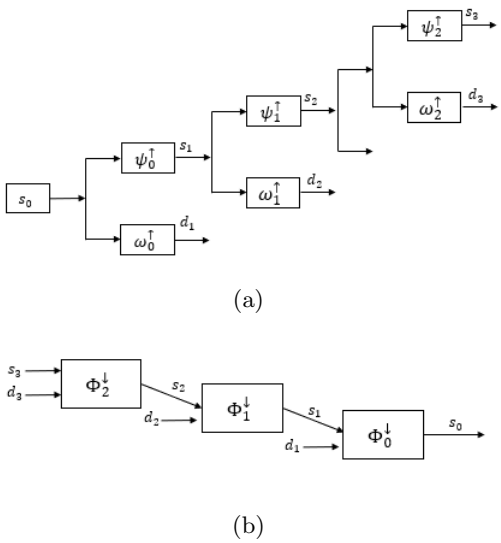

Fig. 1: (a) Decomposition of the approximation signal to the third level, (b) Reconstruction of the signal to the initial signal.

4. LIFTING SCHEME

The lifting scheme is a technique for constructing a signal decomposition and reconstruction, introduced by Sweldens (see [22, 23, 24]). Suppose that signal space S_i is decomposed into approximation signal S_i and detail signal D_i. The lifting scheme consists of four steps: split, predict, update, and merge.

1. **Split**: This step divides the signal S_i into two disjoint subsets based on even and odd indices, denotes as even_i and odd_i, respectively. The process of splitting the signal based on even and odd indices is called the lazy wavelet transform, defined as:
$$\text{Split}(S_i) := (\text{even}_i, \text{odd}_i). \tag{6}$$

2. **Predict**: We observe that the elements of even_i and odd_i alternate.If the elements of initial signal exhibit strong correlation, then even_i and odd_i are strongly correlated as well. This implies that on subset can predict the other with a reasonable accuracy. In this case, the even subset is used to predict the odd subset as the detail signal. In this step, the operator $\mathcal{P}$ is defined as follows
$$d_{i+1}(n) = \text{odd}_i(n) \oslash \mathcal{P}(\text{even}_i(n)), \tag{7}$$
where $d_{i+1}(n) \in D_{i+1}$.

3. **Update**: In this step, the detail signals obtained from the prediction step are utilized to update even_i transforming it into the approximate signal S_{i+1}. The operator $\mathcal{U}$ is defined as follows
$$s_{i+1}(n) = \text{even}_i(n) \otimes \mathcal{U}(d_{i+1}(n)). \tag{8}$$

4. **Merge:** For the inverse scheme (signal reconstruction), the $\text{odd}_i(n)$ signal can be quickly recovered using the following equation:
$$\text{odd}_i(n) = d_{i+1}(n) \otimes \mathcal{P}(\text{even}_i(n)), \tag{9}$$
where $\text{even}_i(n)$ is obtained by
$$\text{even}_i(n) = s_{i+1}(n) \oslash \mathcal{U}(d_{i+1}(n)). \tag{10}$$
Once the odd_i and even_i signal have been successfully reconstructed, the next step is to merge them to obtain the original signal. This step is referred to as the lazy inverse wavelet transform and is defined as:
$$S_i := \text{Merge}(\text{even}_i, \text{odd}_i). \tag{11}$$

An illustration of the wavelet transform decomposition and reconstruction using a lifting scheme can be seen in Figure 2. In this research, we construct various types of operators predict ($\mathcal{P}$) and update ($\mathcal{U}$) operators, which serve as different types of encryption keys.

5. WAVELET DISCRETE TRANSFORM BASED ON MAX-MIN-PLUS LIFTING SCHEME

In this research, we construct a wavelet transform with both domain and codomain in $\mathbb{Z}$. Let S_0 be the input signal. For $i \geq 1$, $S_i : \mathbb{Z} \to \mathbb{Z}$ represents the signal space at level i obtained from the analysis process via the operator $\mathcal{U}$. Similarly, let $D_i : \mathbb{Z} \to \mathbb{Z}$

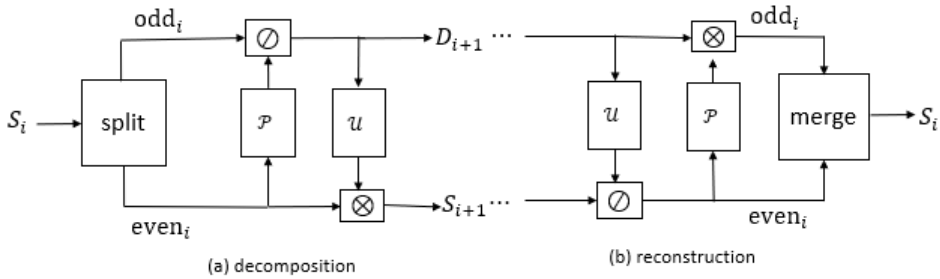

Fig. 2: Signal decomposition and reconstruction using lifting scheme.

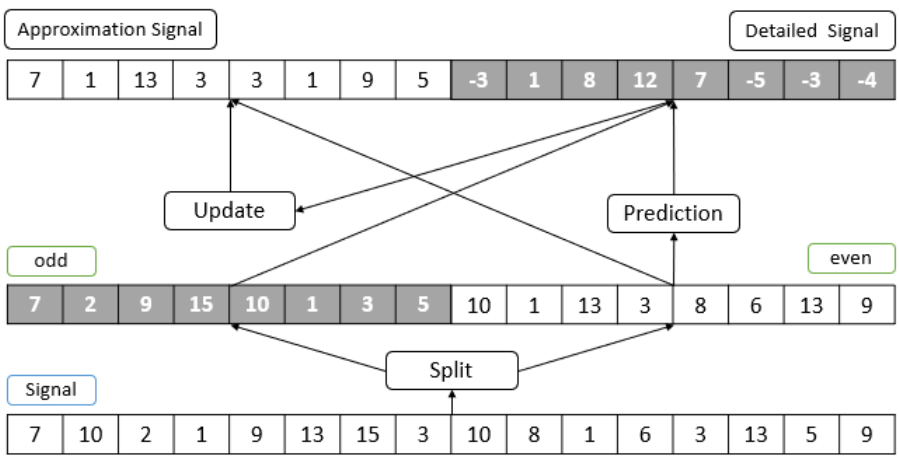

Fig. 3: Signal decomposition illustration with MinLS.

be a detail signal at level i obtained from the synthesis process via the operator $\mathcal{P}$. We construct wavelet decomposition and reconstruction schemes using a lifting scheme. This involves constructing several $\mathcal{P}$ and $\mathcal{U}$ operators. For a given signal space S_i, the signal is split into even and odd indices, denotes as, even$_i$ and odd$_i$. Based on (7)and (8), let $x(n) \in \text{even}_i$ and $y(n) \in D_i$, we provide several types of predict and update operators as follows:

1. **type 1:** Min lifting scheme (MinLS)

$$\begin{array}{rcl} \mathcal{P}(x(n)) & = & x(n-1) \oplus' x(n) \\ \mathcal{U}(y(n)) & = & y(n) \oplus' y(n+1) \oplus' e \end{array} \tag{12}$$

2. **type 2:** Max lifting scheme (MaxLS)

$$\begin{array}{rcl} \mathcal{P}(x(n)) & = & x(n-1) \oplus x(n) \\ \mathcal{U}(y(n)) & = & y(n) \oplus y(n+1) \oplus e \end{array} \tag{13}$$

3. **type 3:** Max-min lifting scheme (MaxMinLS)

$$\begin{array}{rcl} \mathcal{P}(x(n)) & = & x(n-1) \oplus \lfloor x(n-1) \otimes x(n) \rfloor^{\otimes \frac{1}{2}} \\ \mathcal{U}(y(n)) & = & y(n) \oplus' \lfloor y(n) \otimes y(n+1) \rfloor^{\otimes \frac{1}{2}} \oplus' e \end{array} \tag{14}$$

4. **type 4:** Average-min lifting scheme (AveMinLS)

$$\begin{array}{rcl} \mathcal{P}(x(n)) & = & \left\lfloor (x(n-1) \otimes x(n))^{\otimes \frac{1}{2}} \right\rfloor \\ \mathcal{U}(y(n)) & = & y(n) \oplus' y(n+1) \oplus' e \end{array} \tag{15}$$

where $e = 0$ is the identity element of the operation $\otimes$ in min-max-plus algebra. Every type is very crucial in generating encryption keys, constructing the cryptographic algorithms, and determining the lifting scheme used for encryption and decryption. Figure 3 illustrate the process of determining approximation and detail signal in wavelet decomposition via lifting scheme over min-max-plus algebra. For instance, we select the type 1 MinLS as the lifting scheme. Suppose the initial signal is $S_0 = [7\ 10\ 2\ 1\ 9\ 13\ 15\ 3\ 10\ 8\ 1\ 6\ 3\ 13\ 5\ 9]$. The next step involves dividing S_0 into two parts based on even and odd indices. We obtain $\text{odd}_0 = [7\ 2\ 9\ 15\ 10\ 1\ 3\ 5]$ and $\text{even}_0 = [10\ 1\ 13\ 3\ 8\ 6\ 13\ 9]$. Subsequently, an analysis is conducted to generates the approximation and detail signals as follows:

- Based on equations (7) and (12), the elements of the detail signal d_1 are computed as follows:

$$\begin{array}{l} d_1[1] = \text{odd}_0[1] \oslash \text{even}_0[1] = 7 \oslash 10 = -3, \\ d_1[2] = \text{odd}_0[2] \oslash (\text{even}_0[1] \oplus' \text{even}_0[2]) = 2 \oslash (10 \oplus' 1) = 1, \\ d_1[3] = \text{odd}_0[3] \oslash (\text{even}_0[2] \oplus' \text{even}_0[3]) = 9 \oslash (1 \oplus' 13) = 8, \\ d_1[4] = \text{odd}_0[4] \oslash (\text{even}_0[3] \oplus' \text{even}_0[4]) = 15 \oslash (13 \oplus' 3) = 12, \\ d_1[5] = \text{odd}_0[5] \oslash (\text{even}_0[4] \oplus' \text{even}_0[5]) = 10 \oslash (3 \oplus' 8) = 7, \\ d_1[6] = \text{odd}_0[6] \oslash (\text{even}_0[5] \oplus' \text{even}_0[6]) = 1 \oslash (8 \oplus' 6) = -5, \\ d_1[7] = \text{odd}_0[7] \oslash (\text{even}_0[6] \oplus' \text{even}_0[7]) = 3 \oslash (6 \oplus' 13) = -3, \\ d_1[8] = \text{odd}_0[8] \oslash (\text{even}_0[7] \oplus' \text{even}_0[8]) = 5 \oslash (13 \oplus' 9) = -4. \end{array}$$

  As a result, the detail signal d_1 is $d_1 = [-3\ 1\ 8\ 12\ 7\ -5\ -3\ -4]$

- Based on equations (8) and (12), the elements of the approximation signal s_1 are computed as follows:

$$\begin{array}{l} s_1[1] = \text{even}_0[1] \otimes (d_1[1] \oplus' d_1[2] \oplus' e) = 10 \otimes (-3 \oplus' 1 \oplus' 0) = 10 \otimes -3 = 7, \\ s_1[2] = \text{even}_0[2] \otimes (d_1[2] \oplus' d_1[3] \oplus' e) = 1 \otimes (1 \oplus' 8 \oplus' 0) = 1 \otimes 0 = 1, \\ s_1[3] = \text{even}_0[3] \otimes (d_1[3] \oplus' d_1[4] \oplus' e) = 13 \otimes (8 \oplus' 12 \oplus' 0) = 13 \otimes 0 = 13, \\ s_1[4] = \text{even}_0[4] \otimes (d_1[4] \oplus' d_1[5] \oplus' e) = 3 \otimes (12 \oplus' 7 \oplus' 0) = 3 \otimes 0 = 3, \\ s_1[5] = \text{even}_0[5] \otimes (d_1[5] \oplus' d_1[6] \oplus' e) = 8 \otimes (7 \oplus' -5 \oplus' 0) = 8 \otimes -5 = 3, \\ s_1[6] = \text{even}_0[6] \otimes (d_1[6] \oplus' d_1[7] \oplus' e) = 6 \otimes (-5 \oplus' -3 \oplus' 0) = 6 \otimes -5 = 1, \\ s_1[7] = \text{even}_0[7] \otimes (d_1[7] \oplus' d_1[8] \oplus' e) = 13 \otimes (-3 \oplus' -4 \oplus' 0) = 13 \otimes -4 = 9, \\ s_1[8] = \text{even}_0[8] \otimes (d_1[8] \oplus' e) = 9 \otimes (-4 \oplus' e) = 9 \otimes -4 = 5. \end{array}$$

  As a result, the approximation signal is s_1 where $s_1 = [7\ 1\ 13\ 3\ 3\ 1\ 9\ 5]$.

Consider a signal at level $k-1$ that transforms to level k and has an odd cardinality. In this case, an additional element is appended to the signal at the last index duplicating the previous last element. Following this adjustment, analysis process is carried out. During the synthesis process, the additional elements added previously are removed to restore the initial signal. Therefore, Consequently, in the proposed cryptographic algorithm, the length of the ciphertext exceeds that of the plaintext.

6. CONSTRUCTION OF CRYPTOGRAPHIC ALGORITHM

In this section, we develop a cryptographic algorithm using MMPLS-IWavelet. The algorithm consists of the encryption process, the decryption key generation, and decryption process.

6.1. Encryption process

The encryption process transforms the plaintext into its corresponding ciphertext. The encryption process consists of the following steps:

1. The plaintext input,consisting of N characters is transformed into an array P_{BMP} where
$$P_{\text{BMP}} = [P[1], P[2], \ldots, P[N]].$$
The array P_{BMP} is a sequence of non-negative integer where $P[i]$ represents the the BPM code for the ith character in the plaintext for $i = 1, 2, \ldots, N$.

2. The encryption key, denoted by Key_e, is a sequence of finite positive integers of length m. Here, we have
$$\text{Key}_e = [k_e[1], k_e[2], \ldots, k_e[m]]$$
where $m \in \mathbb{N}$ and $k_e[1]$ is the type MMPLS-IWavelet used. Here, $k_e[1] \in \{1, 2, 3, 4\}$ and $k_e[i] \in \mathbb{N}$ where $k_e[i] \leq \log_2 N$ for $2 \leq i \leq m$. Each $k_e[i]$ for $2 \leq i \leq m$ represents the levels used in the transformation.

3. The approximation signals and detail signals of each transformation process for each level in Key_e are calculated using the type of MMPLS-IWavelet described in Section 5.

4. A binary code is generated from the approximation and detail signals. For example, if $|\text{Key}_e| = m$, the binary code is generated from the approximation and detail signal resulting from the last transformation at level $k_e(m)$. We define $S_{k,m}$ as the resulting signal consisting of approximation and detail signals with transformations of type k with $m-1$ transformations steps. The following formula generates the binary signal code
$$\mathcal{BC}(S_{k,m}[i]) = \begin{cases} 1, & S_{k,m}[i] < 0 \\ 0, & S_{k,m}[i] \geq 0, \end{cases} \tag{16}$$
where $S_{k,m}[i]$ is ith element in $S_{k,m}$ for $i = 1, 2, 3, \ldots, |S_{k,m}|$. This binary code $\mathcal{BC}(S_{k,m}[i])$ will be used in constructing the decryption key during the decryption process.

5. There is a possibility that the values of $S_{k,m}$ exceed 65503, and we define an array ρ, to handle such a case where

$$\rho[i] = \left\lfloor \frac{|S_{k,m}[i]|}{65503} \right\rfloor, \tag{17}$$

for $i = 1, 2, 3, \ldots, |S_{k,m}|$, where $\lfloor x \rfloor$ denotes the nearest integer less than or equal to x.

6. The BMP code for ciphertext denoted by C_{BMP} is obtained under the following condition condition:

(a) If $\sum_{i=1}^{|S_{k,m}|} \rho[i] = 0$ then

$$C_{\text{BMP}}[i] = |S_{k,m}[i]| + 32. \tag{18}$$

(b) If $\sum_{i=1}^{|S_{k,m}|} \rho[i] \neq 0$ then

$$C_{\text{BMP}}[i] = |S_{k,m}[i]| \mod 65503 + 32, \tag{19}$$

and

$$C_{\text{BMP}}[i + |S_{k,m}|] = \rho[i] + 32, \tag{20}$$

for $i = 1, 2, 3, \ldots, |S_{k,m}|$. The addition of 32 ensure that all BMP codes from the ciphertext can be represented as a character. The value 65503 is derived by subtracting 32 from the total number of characters in the BMP.

7. The result of the encryption is the ciphertext obtained by converting C_{BMP} into text.

6.2. Generation of the decryption key

The decryption key consists of three components: Key_{d_1}, Key_{d_2}, and Key_{d_3}. Here, Key_{d_1} is identical to Key_e as described in the encryption process. The second key, Key_{d_2}, is obtained through the following procedure. First, a sequence of finite non-negative integer is obtained by partitioning $\mathcal{BC}(S_{k,m}[i])$ into several parts, each consisting of 16 elements, resulting a new array of 16-bit code. The value Key_{d_2} is obtained by converting these value into their corresponding decimal value. The third key, Key_{d_3} is a sequence of positive integers representing the length of the coefficients of approximation and detail signals for each level in every executed transformation.

For example, suppose a signal of length 10 is encrypted using the key level [2 3 3]. The first transformation of level 2 resulting in the decomposition of signal length into [3 3 5 10]. The second transformation of level 3 decompose the signal length into [2 2 3 6 11], and the third transformation at level 3 further decomposes the signal length into [2 2 4 7 13]. As a result, the third decryption key, Key_{d_3} satisfies the following condition

$$\text{Key}_{d_3} = [2\ 2\ 4\ 7\ 13\ 2\ 2\ 3\ 6\ 11\ 3\ 3\ 5\ 10].$$

6.3. Decryption process

An encryption process $\mathcal{E}$ is correct with respect to the decryption process $\mathcal{D}$ if for every message $\mathcal{M}$ we have $\mathcal{D}(\mathcal{E}(\mathcal{M})) = \mathcal{M}$ [21]. Based on equations (18), (19), and (20) every message $\mathcal{M}$ is encrypted to produce $\mathcal{E}(\mathcal{M})$ using C_{BMP} and three decryption keys $(\text{Key}_{d_1}, \text{Key}_{d_2}, \text{Key}_{d_3})$. It can be inferred from the encryption process that $\mathcal{E}(\mathcal{M})$ is generated from $S_{k,m}$. If $\sum_{i=1}^{|S_{k,m}|} \rho[i] = 0$, then the condition (18) holds, which is

$$\mathcal{E}(M) = |S_{k,m}| + 32 \Rightarrow |S_{k,m}| = \mathcal{E}(M) - 32. \tag{21}$$

Based on equation (16), $S_{k,m}$ is non-negative if only if $\mathcal{BC}[i] = 0$ and negative if only if $\mathcal{BC}[i] = 1$. Therefore, equation (21) becomes

$$S_{k,m} = (\mathcal{E}(M) - 32) \cdot (-1)^{\mathcal{BC}}.$$

If $\sum_{i=1}^{|S_{k,m}|} \rho[i] \neq 0$, then equations (19) and (20) are obtained. This means that $|\mathcal{E}(\mathcal{M})| \neq |\mathcal{BC}|$, or more precisely, $|\mathcal{E}(\mathcal{M})| = 2|\mathcal{BC}|$. Consider the following

$$\mathcal{E}(M)[i] = |S_{k,m}[i]| \mod 65503 + 32 \Rightarrow |S_{k,m}[i]| \mod 65503 = \mathcal{E}(M)[i] - 32, \tag{22}$$

for $i = 1, 2, 3, \ldots, |\mathcal{BC}|$. From equation (22) we obtain

$$|S_{k,m}[i]| = k[i] \cdot 65503 + \mathcal{E}(M)[i] - 32,$$

for $k[i] \in \mathbb{Z}$. In this case, the following equation holds:

$$k[i] = \left\lfloor \frac{|S_{k,m}|[i]}{65503} \right\rfloor = \rho[i],$$

for $i = 1, 2, \ldots, |\mathcal{BC}|$, where $\lfloor x \rfloor$ denotes the nearest integer less than or equal to x. From equation (20) we have $k[i] = \rho[i] = \mathcal{E}(\mathcal{M})(i + |S_{k,m}|) - 32$. Based on equation (16), $S_{k,m}$ is non-negative if only if $\mathcal{BC}[i] = 0$ and is negative if only if $\mathcal{BC}[i] = 1$. Therefore, equation (22) becomes

$$S_{k,m}[i] = ((\mathcal{E}(\mathcal{M})[i + |S_{k,m}|] - 32) \cdot 65503 + \mathcal{E}(M)[i] - 32) \cdot (-1)^{\mathcal{BC}[i]},$$

for $i = 1, 2, 3, \ldots, |\mathcal{BC}|$. Next, $S_{k,m}$ is inversely transformed using Key_{d_1} in reverse order using MMPLS-IWavelet to obtain the original message $\mathcal{M}$.

Here, we provide a detail description of the decryption process using the MMPLS-IWavelet and the decryption key generated from the encryption results. This step involves converting the ciphertext back into plaintext using the synthesis process of the specified MMPLS-IWavelet type. The decryption process is outline as follows:

1. The ciphertext is converted to BMP code. We denote C_{BMP} as a finite non-negative integers where the value of each element is greater than or equal to 32.

2. Input the first decryption key Key_{d_1} which has the same structure as the encryption key Key_e.

3. Convert the second decryption key, Key_{d_2}, into a binary code $\mathcal{BC}(S_{k,m})$ where each element in Key_{d_2} is represented as a 16-bit binary code. The results of each conversion of decimal numbers to binary code are merged to form the array $\mathcal{BC}(S_{k,m})$.

4. Obtain the $S_{k,m}$ signal (4th stage of encryption) with the following condition :

   (a) if $|\mathcal{BC}(S_{k,m})| = |C_{\text{BMP}}|$ then

   $$S_{k,m}[i] = (C_{\text{BMP}}[i] - 32) \cdot (-1)^{\mathcal{BC}(S_{k,m}[i])} \tag{23}$$

   (b) if $|\mathcal{BC}(S_{k,m})| \neq |C_{\text{BMP}}|$ then

   $$S_{k,m}[i] = ((C_{\text{BMP}}[i + |S_{k,m}|] - 32) \cdot 65503 + C_{\text{BMP}}[i] - 32) \cdot (-1)^{\mathcal{BC}[i]} \tag{24}$$

   for all $i = 1, 2, \ldots, |\mathcal{BC}(S_{k,m})|$ and $k = 65503$.

5. Input the third decryption key Key_{d_3} correctly. Then execute the synthesis process signal using $S_{k,m}$ convert it to signal P_{BMP} using synthesis process type selected from MMPLS-IWavelet.

6. The plaintext is obtained by converting the BMP ciphertext code P_{BMP} back into text.

6.4. A simple illustrative example

This section provides a simple example of the encryption process, key generation, and decryption using the MMPLS-IWavelet cryptographic algorithm. As an illustration, consider the plaintext described in Figure 4 as an input, which contains 34.

Thank you Gracias 謝謝 Спасибо شكرًا

Fig. 4: Plaintext containing the word “thank you” in five languages from around the world.

1. The plaintext is converted into BMP code, denoted as P_{BMP}, where $P_{\text{BMP}}[1] = 84$, $P_{\text{BMP}}[2] = 104$, $P_{\text{BMP}}[3] = 97$, …, $P_{\text{BMP}}[34] = 1575$. Let s_0 be the original signal, with $s_0[i] = P_{\text{BMP}}[i]$ for $i = 1, 2, \ldots, 34$.

2. In this example, we choose the encryption key $\text{Key}_e = [2, 2, 3]$, indicating the use of MMPLS-IWavelet type 2 (MaxLS) with two transformations at levels 2 and 3.

3. The analysis process of MMLPLS-IWavelet type MaxLS is carried out using equations (6), (7), (8), and (13). This process aims to obtain the signal result after two transformations, denoted as $S_{2,3}$.

(a) The analysis process for the first transformation at level 2 is as follows. For level 1, split the signal s_0 into signals e_1 (even) and o_1 (odd), where $e_1[i] = s_0[2i]$ and $o_1[i] = s_0[2i-1]$ for $i = 1, 2, \ldots, 17$. We obtain $e_0[1] = 104$, $e_0[2] = 110, \ldots, e_0[17] = 1575$, and $o_0[1] = 84, o_0[2] = 97, \ldots, o_0[17] = 1611$. Thus, the approximation signal (s_1) and detail signal (d_1) are as follows. For the detail signal, we have:

$$\begin{aligned}
&d_1[1] = o_0[1] \oslash e_0[1] = 84 \oslash 104 = -20,\\
&d_1[2] = o_0[2] \otimes (e_1[1] \oplus e_0[2]) = 97 \oslash (104 \oplus 110) = -13,\\
&\vdots\\
&d_1[17] = o_0[17] \oslash (e_1[16] \oplus e_0[17]) = 1611 \oslash (1585 \oplus 1575) = 26.
\end{aligned}$$

For the approximation signal, we have:

$$\begin{aligned}
&s_1[1] = e_0[1] \otimes (d_1[1] \oplus d_1[2] \oplus e) = 104 \otimes (-20 \oplus -13 \oplus e) = 104,\\
&s_1[2] = e_0[2] \otimes (d_1[2] \oplus d_1[3] \oplus e) = 110 \otimes (-13 \oplus -3 \oplus e) = 110,\\
&\vdots\\
&s_1[17] = e_0[17] \otimes (d_1[17] \oplus e) = 1575 \otimes (26 \oplus e) = 1601.
\end{aligned}$$

For level 2, since $|s_1| = 17$, is of odd cardinality, an additional element is added such that $s_1[18] = s_1[17] = 1601$. Split the signal s_1 into two signals $e_1[i] = s_1[2i]$ and $o_1[i] = s_1[2i-1]$ for $i = 1, 2, \ldots, 9$. We obtain $e_1[1] = 104, e_1[2] = 42, \ldots, e_1[9] = 1601$ and $o_1[1] = 110, o_1[2] = 121, \ldots, o_1[9] = 1601$. Applying equations (8), (7), and (13) as in the previous process, we obtain the approximation signal (s_2) and detail signal (d_2) as follows: $s_2[1] = 110, s_2[2] = 121, \ldots, s_2[9] = 1601$ and $d_2[1] = -6, d_2[2] = -79, \ldots, d_2[9] = -10$. Thus, the result of the first transformation is $S_{2,2} = [s_2\ d_2\ d_1]$, where $S_{2,2}[1] = 110, S_{2,2}[2] = 121, \ldots, S_{2,2}[35] = 26$.

(b) The analysis process for the second transformation of level 3 is as follows. For level 1, since $|S_{2,2}| = 35$, is of odd cardinality, an additional element is added such that $S_{2,2}[36] = S_{2,2}[35] = 26$. Redefine the initial signal s_0 as $s_0[i] = S_{2,1}[i]$ for $i = 1, 2, \ldots, 36$. Split the signal s_0 into e_0 and o_0, where $e_0[1] = 121, e_0[2] = 115, \ldots, e_0[18] = 26$, and $o_0[1] = 110, o_0[2] = 114, \ldots, o_0[18] = 26$. Thus, the approximation signal (s_1) and detail signal (d_1) are as follows. For the detail signal, we have

$$\begin{aligned}
&d_1[1] = o_0[1] \oslash e_0[1] = 110 \oslash 121 = -11\\
&d_1[2] = o_0[2] \oslash (e_0[1] \oplus e_0[2]) = 114 \oslash (121 \oplus 115) = -7\\
&\vdots\\
&d_1[18] = o_0[18] \oslash (e_0[17] \oplus e_0[18]) = 26 \oslash (15 \oplus 26) = 0.
\end{aligned}$$

For the approximation signal, we have:

$$\begin{aligned}
&s_1[1] = e_0[1] \otimes (d_1[1] \oplus d_1[2] \oplus e) = 121 \otimes (-11 \oplus -7 \oplus e) = 121\\
&s_1[2] = e_0[2] \otimes (d_1[2] \oplus d_1[3] \oplus e) = 115 \otimes (-7 \oplus 34524 \oplus e) = 34639\\
&\vdots\\
&s_1[18] = e_0[18] \otimes (d_1[18] \oplus e) = 26 \otimes (0 \oplus e) = 26 \otimes e = 26.
\end{aligned}$$

For level 2, following the same process as in the previous level by decomposing the signal s_1, we obtain the signals s_2 and d_2 as follows: $s_2[1] = 35613, s_2[2] = 2585, \ldots, s_2[9] = 26$, and $d_2[1] = -34518, d_2[2] = 974, \ldots, d_2[9] = -11$. For level 3, following the same process, the decomposition of the signal s_2 into s_3 and d_3 is obtained, where $s_3[1] = 35613, s_3[2] = 10, \ldots, s_3[5] = 26$, and $d_3[1] = 33028, d_3[2] = -2595, \ldots, d_3[5] = 0$. Thus, we obtain the result of the transformation of level 2: $S_{2,2} = [s_3\ d_3\ d_2\ d_1]$, where $S_{2,3}[1] = 35613, S_{2,3}[2] = 10, \ldots, S_{2,3}[37] = 0$.

(c) From $S_{2,3}$, the binary encoding $\mathcal{BC}(S_{2,3})$ is obtained based on (16), resulting in $\mathcal{BC}(S_{2,3})[1] = 0$, $\mathcal{BC}(S_{2,3})[2] = 0$, $\mathcal{BC}(S_{2,3})[3] = 0$, $\ldots, \mathcal{BC}(S_{2,3})[11] = 1, \ldots, \mathcal{BC}(S_{2,3})[37] = 0$.

(d) Based on the output signal $S_{2,3}$ and equation (17), we find that $\rho[i] = 0$ for $i = 1, 2, 3, \ldots, 37$.

(e) Based on equation (18), C_{BMP} is obtained, where $C_{\text{BMP}}[1] = |S_{2,3}[1]| + 32 = |35613| + 32 = 35645$, $C_{\text{BMP}}[2] = |S_{2,3}[2]| + 32 = |10| + 32 = 42$, ..., $C_{\text{BMP}}[37] = |S_{2,3}[37]| + 32 = |0| + 32 = 32$.

(f) Convert C_{BMP} to text, the ciphertext as depicted in Figure 5

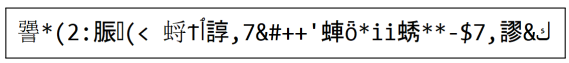

Fig. 5: Ciphertext of plaintext in Figure 4.

Now, We discuss the generation of the second decryption key Key_{d_2} obtained from $\mathcal{BC}(S_{2,3})$, which is represented as 00000010001011100011101110111111101110. This binary code is divided into several groups, each consisting of 16 bits. As a result, we get the following three groups: 0000001000101110, 0011101110111111, and 01110. Ihe last group, contains only five digits, so we add eleven of leading zeros to form a complete 16-bit groups: 0000001000101110, 0011101110111111, and 0000000000001110. Converting these binary groups into decimal numbers yields, 558, 15295, and 14. Therefore, the second decryption key is Key_{d_2} = [558 15295 14]. For the third key Key_{d_3}, since the plaintext has a length 34 characters, the first transformation at level 2 produces a signal length decomposition of [9 9 17 34]. Subsequently, for the second transformation at level 3, the signal length decomposition of [5 5 9 18 35]. Thus, the third decryption key is Key_{d_3} = [5 5 9 18 35 9 9 17 34].

Next, we describe the decryption process of ciphertext shown in Figure 5 with three decryption keys i. e., Key_{d_1}= [2 2 3], Key_{d_2}= [558 15295 14], and Key_{d_3} = [5 5 9 18 35 9 9 17 34].

1. Convert the ciphertext to BMP code, denoted as C_{BMP}, where $C_{\text{BMP}}[1] = 35645, C_{\text{BMP}}[2] = 42, \ldots, C_{\text{BMP}}[37] = 32$.

2. Each element in Keyd_2, when converted to 16-bits binary code, is obtained sequentially as 0000001000101110, 0011101110111111, and 0000000000001110. Since the length of the plaintext is 37, the last binary code has the leading 11

zeros removed, resulting in 01110. Merging all the binary codes we have 000000100010111000111011011111101110. Thus, $\mathcal{BC}(S_{2,3})$ is obtained, where $\mathcal{BC}(S_{2,3})[1] = 0$, $\mathcal{BC}(S_{2,3})[2] = 0$, $\mathcal{BC}(S_{2,3})[3] = 0, \ldots, \mathcal{BC}(S_{2,3})[11] = 1, \ldots,$ $\mathcal{BC}(S_{2,3})[37] = 0$.

3. We can see that $|C_{\text{BMP}}| = |\mathcal{BC}|$, we calculate $S_{2,3}$ based on equation (23). We obtain $S_{2,3}[1] = 35613, S_{2,3}[2] = 10, \ldots, S_{2,3}[37] = 0$.

4. The synthesis process of MMPLS-IWavelet type 2 (MaxLS) is performed based on equations (9), (10), and (13). The synthesis process starts with the second transformation at level 3 and then continues with the synthesis process from the first transformation at level 2 (the reverse of the encryption process). The third decryption key Key_{d_3} is partitioned into several parts based on the number of transformations performed or the length of the key Key_{d_1} minus one. If $|\text{Key}_{d_1}| = m$, then the first partition Key_{d_3} corresponds to $\text{Key}_{d_1}[m]$ with a partition of length $\text{Key}_{d_1}[m] + 2$, the second partition corresponds to $\text{Key}_{d_1}[m-1]$ with a partition of length $\text{Key}_{d_1}[m-1] + 2$, and so on. In this example, the partition of Key_{d_3} is [5 5 9 18 35 | 9 9 17 34]. These partitions for determining whether it is necessary to remove the last element for each transformation result in the analysis process, noting that in the analysis process, the last element of the signal is added if the signal has an odd cardinality. The original signal s_0 is obtained, where $s_0[1] = 84, s_0[2] = 104, s_0[3] = 97, \ldots, s_0[34] = 1575$. This result yields P_{BMP}, the BMP code of the plaintext, where $P_{\text{BMP}}[i] = s_0[i]$ for $i = 1, 2, \ldots, 34$.

5. Convert P_{BMP} to text, resulting in the plaintext shown in Figure 4.

7. ANALYSIS AND EMPIRICAL RESULTS

We evaluate the performance of the constructed cryptographic algorithm using various encryption keys. Several evaluation metrics are utilized to asses the algorithm's performance. We employ nine test data sets comprising various characters and fonts from several countries worldwide. Summary of research results relevant MATLAB source codes, and test data sets are available at

`https://github.com/sebelumSyah/CryptographyMMPLS-IWavelet.git`

MATLAB is chosen for its exceptional performance, especially in handling large matrix and vector operations, due to its optimized algorithms and multi-threading support.

7.1. The correlation test between plaintext and ciphertext

In this section, we calculate the correlation value, which quantifies the relationship between plaintext and ciphertext. The correlation value is computed using the following

Plaintext Files	Number of Character	encryption key			
		Type 1	Type 2	Type 3	Type 4
Test Data 1.txt	32	0.11172	0.08729	0.05113	0.10476
Test Data 2.txt	107	0.01721	0.05867	0.07281	0.00386
Test Data 3.txt	345	0.00589	0.0777	0.01413	0.00784
Test Data 4.txt	714	0.01285	0.05324	0.02861	0.02788
Test Data 5.txt	1592	0.00273	0.04431	0.05509	0.01085
Test Data 6.txt	4164	0.03951	0.00942	0.01192	0.00084
Test Data 7.txt	12265	0.00075	0.00349	0.00750	0.00335
Test Data 8.txt	28800	0.00283	0.00017	0.01128	0.00932
Test Data 9.txt	54147	0.04218	0.01984	0.06627	0.04146

Tab. 1: Correlation value for key level [2 3 2 3].

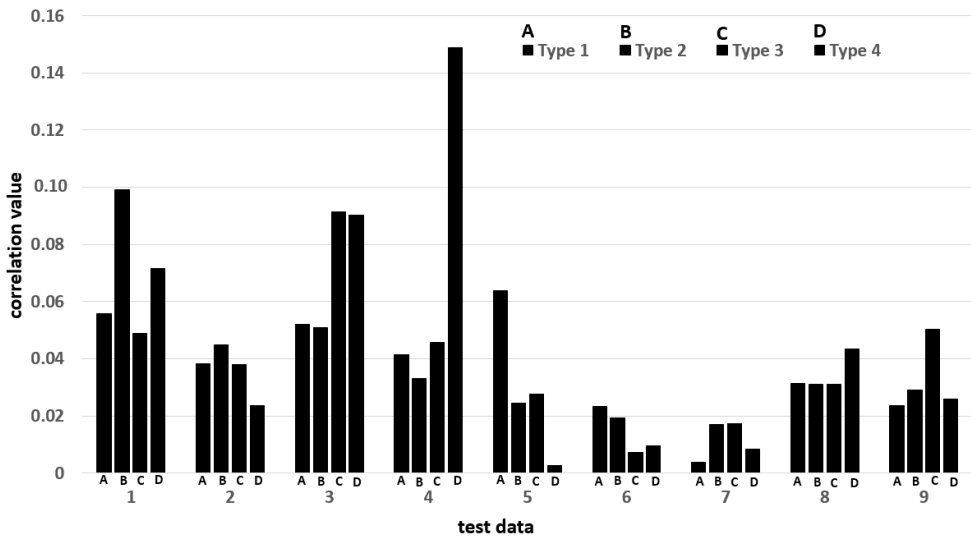

Fig. 6: Correlation value between plaintext and ciphertext for key level [3 4 2 5 5].

formula [1]

$$\mathrm{Cor} = \left| \frac{\left(N\sum_{k=1}^{N} P(k)C(k)\right) - \left(\sum_{k=1}^{N} P(k)\right)\left(\sum_{k=1}^{N} C(k)\right)}{\sqrt{\left(N\sum_{k=1}^{N}(P(k)^2) - \left(\sum_{k=1}^{N} P(k)\right)^2\right)\left(N\sum_{k=1}^{N}(C(k)^2) - \left(\sum_{k=1}^{N} C(k)\right)^2\right)}} \right| \quad (25)$$

where N is the number of characters in the plaintext. The notation $P(k)$ and $C(k)$ represent the BMP codes of plaintext and ciphertext, respectively. For example, the plaintext in Figure 4 and the ciphertext in Figure 5 have a correlation coefficient of 0.12892. If the correlation value is close to zero, it indicates the plaintext and ciphertext

have a weak relationship and can be considered to have no significant relationship [30].

We compute the correlation value of each test data set, which contains different characters and test all MMPLS-IWavelet types. The setting of the level key in encryption is arbitrary, including the length of the level key (number of transformations) and the decomposition level for each transformation. For instance, we set the key level to [2 3 2 3] for all types. In Table 1, the MinLS type shows a correlation interval value between 0.0008 and 0.1117, the MaxLS type hows a correlation interval value between 0.0002 and 0.0873, MaxMinLS type shows a correlation interval value between 0.0075 and 0.0728, and the AveMinLS type shows a correlation interval value between 0.0034 and 0.1052.

We also apply the key level [3 4 2 5 5] as a comparison with the previous example, where the settings for these key level are also arbitrary. From the experiment, we obtain that the correlation value between the plaintext and ciphertext at key level [3 4 2 5 5] is not significantly different to that of key level [2 3 2 3]. Figure 6 illustrates that the correlation values for each test data and MMLS-IWavelet type lie within the interval $0 \leq \text{Cor} \leq 0.2$. Based on the experiments, the correlation between plaintext and ciphertext tends to be very small, very close to zero, for a large number of plaintext characters. We conclude that the plaintext and ciphertext have a weak relationship for all types. More detailed information about other key level experiments to calculate correlation values is available at the Github link provided at the beginning of Section 7.

7.2. Encryption quality

We evaluate the quality of the encryption process using the encryption quality (EQ) metric. Encryption quality is calculated by comparing the number of occurrences of a character in plaintext and ciphertext. The value of EQ is the average value of the absolute difference between the number of occurrences of a character in plaintext and ciphertext [1]. Mathematically, this is expressed as

$$EQ = \frac{\sum_{i=32}^{n} |H_i(C_{\text{BMP}}) - H_i(P_{\text{BMP}})|}{n - 32}, \tag{26}$$

where $H_i(C_{\text{BMP}})$ and $H_i(P_{\text{BMP}})$ represent the number of occurrences of the ith character of the BMP code and n represents the number of characters employed in this study, with $n = 65535$, corresponding to the number of characters in the BMP. Additionally, we determine calculate the maximum EQ value, which occurs when all characters of the plaintext and ciphertext are different. For example, using the plaintext shown in Figure 4 and the ciphertext in Figure 5 with the key level [2 2 3], the encryption quality percentage of 85.13%. In our experiment, we also consider a key level [2 3 2 3] to compute the encryption quality percentage. Based on Table 2, the MinLS type achieves an average of EQ percentage of 81.13%, the MaxLS type achieves an average percentage of EQ of 89.79%, the MaxMinLS type achieves an average percentage of EQ of 93.77%, and the AveMinLS type achieves an average percentage of EQ of 92.68%.

The encryption quality percentage can be enhanced by increasing the number of transformations and decomposition levels for each transformation. For this analysis, we set the key level to [3 3 4 4 5] to calculate the encryption quality percentage and compare it with encryption quality percentage for previous key level. Based on Table 3, the MinLS type achieves an average EQ percentage of 87.26%, the MaxLS achieves an average EQ

plaintext Files	Encryption Quality(EQ)				Max (EQ)	Percentage of (EQ) %			
	Type 1	Type 2	Type 3	Type 4		Type 1	Type 2	Type 3	Type 4
Test Data 1.txt	0.0009	0.0009	0.0009	0.0009	0.0010	93.75	96.87	96.87	96.87
Test Data 2.txt	0.0027	0.0029	0.0033	0.0033	0.0034	79.01	84.38	97.76	96.87
Test Data 3.txt	0.0103	0.0105	0.0106	0.0106	0.0107	95.88	98.15	99.00	98.72
Test Data 4.txt	0.0161	0.0188	0.0214	0.0214	0.0220	73.05	85.41	97.50	97.36
Test Data 5.txt	0.0473	0.0481	0.0485	0.0485	0.0486	97.36	98.93	99.74	99.68
Test Data 6.txt	0.0939	0.1110	0.1141	0.1085	0.1273	73.82	87.18	90.23	85.29
Test Data 7.txt	0.2847	0.3248	0.3599	0.483	0.3745	76.03	89.41	96.11	93.02
Test Data 8.txt	0.6402	0.7378	0.7350	0.7314	0.8793	72.80	83.91	83.58	83.17
Test Data 9.txt	1.1329	1.3865	1.3741	1.3749	1.6534	68.52	83.86	83.11	83.15

Tab. 2: Encryption quality test for key level [2 3 2 3].

Plaintext Files	Encryption Quality(EQ)				Max (EQ)	Percentage of (EQ) %			
	Type 1	Type 2	Type 3	Type 4		Type 1	Type 2	Type 3	Type 4
Test Data 1.txt	0.0008	0.0009	0.0009	0.0009	0.0010	84.37	93.75	93.75	96.87
Test Data 2.txt	0.0031	0.0031	0.0034	0.0034	0.0035	88.49	90.26	96.46	97.34
Test Data 3.txt	0.0106	0.0106	0.0106	0.0106	0.0107	98.43	98.72	99.01	99.00
Test Data 4.txt	0.0188	0.0191	0.0217	0.0214	0.0220	85.51	86.89	98.96	97.43
Test Data 5.txt	0.0483	0.0485	0.0485	0.0486	0.0488	98.96	99.47	99.53	99.65
Test Data 6.txt	0.1064	0.1119	0.1176	0.1081	0.1274	83.52	87.81	92.23	84.81
Test Data 7.txt	0.3250	0.3480	0.3637	0.3513	0.3746	86.74	92.88	97.07	93.77
Test Data 8.txt	0.7098	0.7456	0.7518	0.7353	0.8793	80.71	84.79	85.49	83.61
Test Data 9.txt	1.3010	1.4009	1.4172	1.3660	1.6536	78.67	84.72	85.70	82.61

Tab. 3: Encryption quality test for key level [3 3 4 4 5].

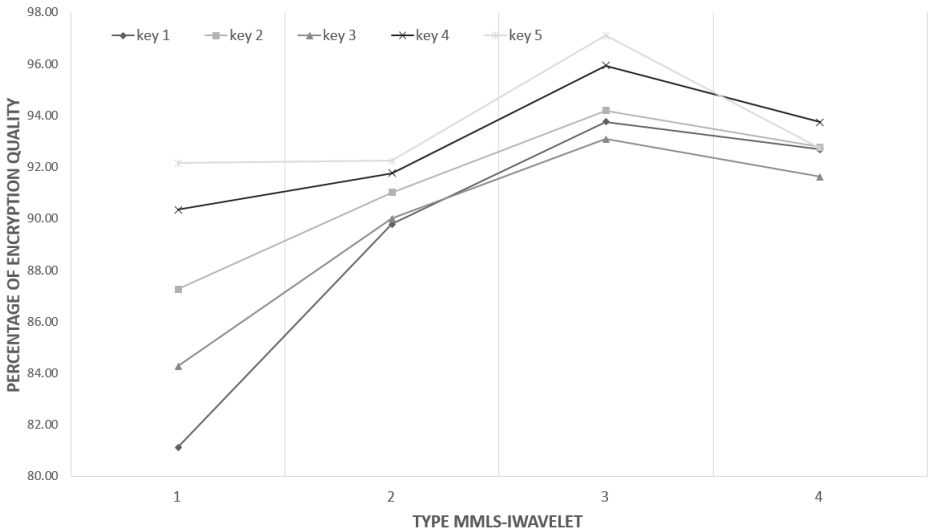

Fig. 7: Average encryption quality percentage for each test data for several level keys.

percentage of 91.0%, the MaxMinLS type achieves an average EQ percentage of 94.20%, and for the AveMinLS type achieves an average EQ percentage of 92.79%. The results show indicate that the encryption quality for all types of MMPLS-IWavelet has an

Plaintext Files	Encryption time (miliseconds)				Decryption time (miliseconds)			
	Type	Type 2	Type 3	Type 4	Type 1	Type 2	Type 3	Type 4
Test Data 1.txt	0.0081	0.0082	0.0081	0.0071	0.0119	0.0105	0.0106	0.0104
Test Data 2.txt	0.0447	0.0392	0.0514	0.0223	0.0416	0.0909	0.0714	0.0782
Test Data 3.txt	0.0523	0.0532	0.0538	0.0531	0.0767	0.0794	0.0841	0.1074
Test Data 4.txt	0.0931	0.0959	0.0891	0.1020	0.1412	0.1406	0.1480	0.1801
Test Data 5.txt	0.1930	0.1934	0.1914	0.1925	0.2994	0.3157	0.3185	0.3111
Test Data 6.txt	0.6872	0.7275	0.4844	0.4861	0.8793	0.8008	0.7755	0.7794
Test Data 7.txt	1.5322	1.3491	1.3356	1.5513	2.4421	2.5419	2.4712	2.4303
Test Data 8.txt	3.2031	3.3703	3.3654	3.3365	5.9021	5.8713	5.9153	5.9999
Test Data 9.txt	5.7285	5.9611	6.0031	6.0324	11.6881	11.6885	12.0223	11.6959

Tab. 4: the encryption and decryption time for MinLS, MaxLS, MaxMinLS, and AveMinLS at key level [2 3 2 3].

average EQ percentage above 80%. We have conducted several experiments for various types of level keys and the results are consistent with those presented in Tables 2 and 3, as summarized in Figure 7.

In Figure 7, we illustrate the average encryption quality percentage for each test data using several key levels, including; key 1 = [2 3 2 3], key 2 = [3 3 4 4 5], key 3 = [3 4 3 4], key 4 = [3 4 3 4 4 4], and key 5 = [4 5 3 5 4 4 5 5]. More detailed information from the experiments of these keys is available at the Github link provided at the beginning of Section 7. Figure 7 demonstrate that the encryption quality percentage increases with number of transformation levels and the number of transformations executed. In addition, it was concluded that the type MaxMinLS consistently achieved the highest encryption quality percentage, while the MinLS type achieved the lowest.

7.3. Algorithm complexity and running time

In this section, we the algorithm's complexity based on computational cost. For computational cost, we estimate the number of operations required for both the encryption and decryption processes. For the analysis process in MMPLS-Iwavelet type MinLS, based on equations (7), (8), and (12), assuming that N is the number of characters in the plaintext, we observe that there is one comparison operator C, one subtraction operator S, and one addition operator A at the kth level. Since the number of channels in the lifting scheme is two, assuming that W_i is the number of computations at each level i where $1 \leq i \leq k$, we have

$$W_i = \frac{N}{2^i}(C + S + A).$$

Therefore, the total number of computations in the analysis process for all levels i where $1 \leq i \leq k$ is

$$\begin{aligned} W &= W_1 + W_2 + \ldots + W_k \\ &= \frac{N}{2}(C+S+A) + \frac{N}{2^2}(C+S+A) + \ldots + \frac{N}{2^k}(C+S+A) \\ &= \left(\sum_{i=1}^{k} \frac{N}{2^i}\right)(C+S+A) \\ &= \left(\frac{2^k-1}{2^k}\right) N(C+S+A) < (C+S+A)N. \end{aligned}$$

It can be observed that the number of computations in the MinLs analysis process is $W < (C+S+A)N$. Therefore, the algorithmic complexity of the MinLs analysis process is $O(N)$. Since each transformation in the analysis process has an algorithmic complexity of $O(N)$, the encryption algorithm consequently also has a complexity of $O(N)$. For the synthesis process based on equations (9), (10), and (12), let N' be the number of characters in ciphertext. We observe there is one comparison operator C', one subtraction operator S', and one addition operator A' at the kth level. Since the number of channels in the lifting scheme is two, and assuming W_i' represents the number of computations at each level i where $1 \leq i \leq k$, we have

$$W_i' = \frac{N'}{2^{k-i+1}}(C'+S'+A').$$

Therefore, the total number of computations in the synthesis process for all levels i where $1 \leq i \leq k$ is

$$\begin{aligned} W' &= W_1' + W_2' + \ldots + W_k' \\ &= \frac{N'}{2^k}(C'+S'+A') + \frac{N'}{2^{k-1}}(C'+S'+A') + \ldots + \frac{N'}{2}(C'+S'+A') \\ &= \left(\sum_{i=1}^{k} \frac{N'}{2^i}\right)(C'+S'+A') \\ &= \left(\frac{2^k-1}{2^k}\right) N'(C'+S'+A') < (C'+S'+A')N'. \end{aligned}$$

It can be observed that the number of computations in the MinLS synthesis process is $W < (C'+S'+A')N'$. Therefore, the algorithmic complexity of the MinLS synthesis process is $O(N')$. Since each transformation in the synthesis process has an algorithmic complexity of $O(N')$, the decryption algorithm consequently also has a complexity of $O(N')$. Thus, we conclude that the cryptographic algorithm using MMPLS-IWavelet type MinLS has a linear complexity with respect to the number of characters. Similarly, based on equations (7), (8), (9), and (10), for each type of MMPLS-IWavelet, we deduce that the cryptographic algorithm using other MMPLS-IWavelet types also exhibit linear complexity with respect to the number of characters.

Regarding running time, in this research we use MATLAB 2022b software to implement cryptographic algorithms with MMPLS-IWavelet. We recorded the computational time of the algorithm we construct for each plaintext dataset and for all

types of MMPLS-IWavelet to determine the empirical running time of the algorithm in practice. We ran the program on Intel(R) Core(TM) i5-4200M CPU 2.50GHZ (4CPUs) with 8GB RAM.

Table 4 shows that the running times of the proposed cryptographic algorithms for all types of the rementioned MMPLS-IWavelet are very short, even when the plaintext contains more than 50,000 characters. Based on the various data test with lengths ranging from 31 to 54,147 characters, we observe that the encryption time for MinLS type ranges from 0.0081 ms to 5.7285 ms, for MaxLS type from 0.0081 ms to 5.9611 ms, for MaxMinLS from 0.0081 ms to 6.0031 ms, and for AveMinLS from 0.0071 ms to 6.0324 ms. Meanwhile, the decryption time for MinLS scheme ranges from 0.0119 ms to 11.6881 ms, for MaxLS scheme from 0.0105 ms to 11.6885 ms, for MaxMinLS scheme from 0.0106 ms to 12.0223 ms, and for AveMinLS scheme from 0.0104 ms to 11.6959 ms. Thus, we infer that the running time of the proposed MMPLS-IWavelet-based cryptographic algorithms is practically efficient.

7.4. Key sensitivity analysis

We investigate the sensitivity of the encryption key. The sensitivity of the encryption key measured by calculating the percentage of change in ciphertext if the encryption key is modified. In this paper, we use the Hamming metric to quantify changes in ciphertext due to alterations. The Hamming metric or Hamming distance between two sequences of the same length is defined as the number of positions at which the corresponding symbols differ. If the sizes of the two ciphertexts are different, then we choose the size of the smaller ciphertext as the reference length for measuring the Hamming metric.

For example, the plaintext in Figure 4 is encrypted into ciphertext in Figure 5 using key level of [2 3] with the MaxLS type. Modifiying the key level [3 2], results in a 91.66% change in the ciphertext, and while changing the key level to [2 3 3],results in a 97.29% change. We modify the encryption key by swapping the order of levels in the encryption key or by adding transformations. In this research, we present results from two experiments with different initial keys. In the first experiment, the initial encryption key level is [4 4 3 3]. The key is modified by changing the sequence to become Key_1 = [4 3 4 3]. For the second experiment, we define Key_2 by adding level 4 to Key_1, resulting in Key_2 = [4 4 3 3 4]. The results of experiment are shown in in Table 5. For the second experiment, the initial encryption key at level [2 3 2 3]. The modification keys for this experiment are Key_1 = [2 3 3 2] and Key_2 = [2 3 2 3 2]. The results of the this experiment are shown in Table 6.

Based on Tables 5 and 6, we find that the most significant percentage of ciphertext change is achieved by increasing the number of transformations in the key level. Regarding the sensitivity of the decryption key, the key sensitivity value is identical to that of encryption key because the decryption process requires the same key used in encryption. This indicates that the decryption key has very high sensitivity.

Plaintext File	MMPLS-IWavelet							
	MinLS		MaxLS		MaxMinLS		AveMinLS	
	Key 1	Key 2	Key 1	Key 2	Key 1	Key 2	Key 1	Key 2
Test Data 1.txt	56.25	100.0	43.75	100.0	65.62	100.0	59.37	100.0
Test Data 2.txt	34.82	100.0	33.03	99.10	38.39	100.0	34.82	100.0
Test Data 3.txt	26.42	100.0	25.85	99.71	28.69	100.0	27.84	100.0
Test Data 4.txt	24.44	99.86	24.17	99.44	25.69	100.0	25.00	100.0
Test Data 5.txt	23.32	100.0	23.32	99.94	24.26	100.0	24.06	100.0
Test Data 6.txt	98.41	99.43	99.25	99.64	99.76	99.85	99.52	99.54
Test Data 7.txt	97.22	99.42	99.38	99.80	99.83	99.91	99.73	99.83
Test Data 8.txt	22.11	99.05	22.19	98.54	22.61	99.60	22.47	99.30
Test Data 9.txt	95.13	98.72	97.44	98.30	99.28	99.60	98.89	99.16

Tab. 5: Sensitivity of encryption key, with initial key level is [4 4 3 3], modified key levels are $\text{Key}_1 = [4\ 3\ 4\ 3]$ and $\text{Key}_2 = [4\ 4\ 3\ 3\ 4]$.

Plaintext File	MMPLS-IWavelet							
	MinLS		MaxLS		MaxMinLS		AveMinLS	
	Key 1	Key 2	Key 1	Key 2	Key 1	Key 2	Key 1	Key 2
Test Data 1.txt	46.87	96.87	50.00	93.75	50.00	100.0	50.00	96.87
Test Data 2.txt	42.85	98.21	42.85	99.10	42.85	100.0	42.85	100.0
Test Data 3.txt	41.19	100.0	41.19	99.43	41.76	100.0	41.76	100.0
Test Data 4.txt	40.27	99.44	40.27	99.16	41.11	99.86	40.97	99.72
Test Data 5.txt	40.32	99.81	40.57	99.81	40.89	100.0	40.89	100.0
Test Data 6.txt	39.70	98.68	40.28	99.18	40.54	99.76	40.54	99.59
Test Data 7.txt	39.45	97.90	40.26	99.25	40.63	99.92	40.60	99.83
Test Data 8.txt	39.30	97.94	39.52	97.99	40.47	99.58	40.32	99.24
Test Data 9.txt	38.74	96.97	39.42	97.64	40.40	99.52	40.19	99.17

Tab. 6: Sensitivity of encryption key, with initial key level is [2 3 2 3], modified key levels are $\text{Key}_1 = [2\ 3\ 3\ 2]$ and $\text{Key}_2 = [2\ 3\ 2\ 3\ 2]$.

7.5. Entropy analysis for plaintext and ciphertext

We measure the randomness of plaintext and ciphertext based on entropy analysis [30]. Suppose we are given the message T, the entropy value of T is defined by $E(T)$ as follows

$$E(T) = \sum_{t \in T} \text{Prob}(t) \log_2 (\text{Prob}(t)), \tag{27}$$

where $\text{Prob}(t)$ denotes the probability that symbol t appears in message T. The higher the value of $E(T)$ means the higher the level of randomness of symbols in the message T. That means the message has a greater variety and is more difficult to predict. For example, the plaintext in Figure 4 has an entropy value of 4.5473, and the ciphertext in Figure 5 has an entropy value of 5.0473. Table 7 shows the entropy values of plaintext and ciphertext using the key level in the encryption process [2 3 2 3]. It can be seen in the table that the entropy values of the ciphertext are higher than the plaintext entropy values for all MMLS-Wavalet types. That indicates that the ciphertext is much

Plaintext File	Entropy of Plaintext	Entropy of Ciphertext			
		Type 1	Type 2	Type 3	Type 4
Test Data 1.txt	3.6678	4.6639	4.8125	4.9375	4.5000
Test Data 2.txt	4.1946	6.2593	6.0494	6.7895	6.7359
Test Data 3.txt	5.7188	8.3060	8.4346	5.5471	8.4537
Test Data 4.txt	4.6072	7.5446	6.9024	9.3259	9.2390
Test Data 5.txt	7.1051	6.2453	6.3234	6.7196	6.3747
Test Data 6.txt	4.3976	7.3639	8.2694	9.0225	8.0062
Test Data 7.txt	4.0763	8.0965	8.9708	10.4401	9.4410
Test Data 8.txt	4.6838	7.3219	6.3880	5.2550	4.8573
Test Data 8.txt	4.1918	6.8279	6.0877	7.9484	7.1463

Tab. 7: Entropy analysis for plaintext and ciphertext.

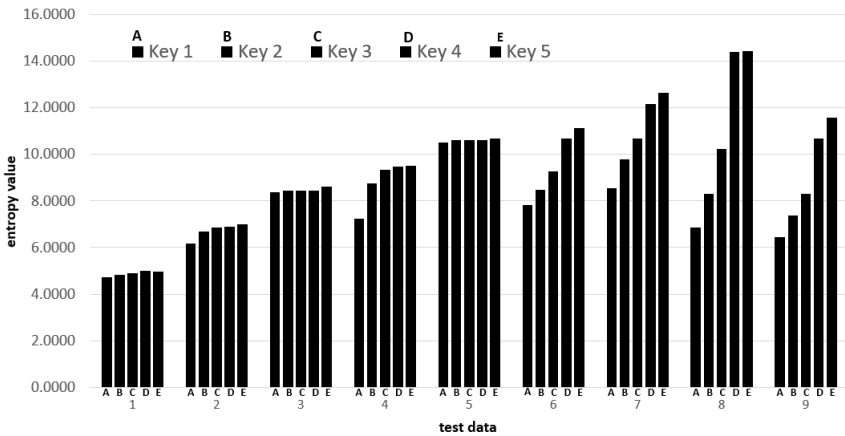

Fig. 8: Average entropy values for each MMPLS-IWevelet type for each ciphertext test data for several key variances.

more random or uniformly distributed than the plaintext. Table 7 also shows that the MaxMinLS type has the highest entropy value among the others.

In Figure 7, we summarize the experimental results for determining the average for each MMPLS-Wavelet types from each ciphertext test data by considering several example key levels, including; $\text{key}_1 = [2\ 3\ 2\ 3]$, $\text{key}_2 = [3\ 2\ 4\ 5\ 5]$, $\text{key}_3 = [4\ 5\ 4\ 3\ 3\ 2\ 1\ 4\ 3\ 2\ 3\ 4\ 3\ 4\ 5\ 5\ 5]$, $\text{key}_4 = [4\ 5\ 4\ 3\ 3\ 2\ 1\ 4\ 3\ 2\ 3\ 4\ 3\ 4\ 5\ 5\ 5]$, and $\text{key}_5 = [5\ 4\ 5\ 3\ 1\ 2\ 4\ 5\ 3\ 5\ 2\ 3\ 4\ 1\ 2\ 2\ 3\ 4\ 3\ 5\ 2\ 4]$. More detailed information from the experiments of these keys is available at the Github link provided at the beginning of Section 7. Based on Figure 7, entropy analysis for different types of keys shows that the longer and larger the values in Key_e, the more random and uniformly distributed the ciphertext will be compared to the plaintext. This confirms that increased key complexity enhances the randomness and security of the encrypted message.

7.6. Key space analysis and cryptanalysis

In this section, we analyze the possible constructs of the decryption key. Assuming the length of the plaintext is N and the length of the ciphertext is N', as discussed in Section 6.2, the decryption key is divided into three parts as follows:

(a) The first decryption key, denoted by Key_{d_1} is the encryption key which is a sequence of finite positive integers. It can be represented as

$$\text{Key}_{d_1} = [\text{Key}_{d_1}[1], \text{Key}_{d1}[2], \ldots, \text{Key}_{d_1}(m)],$$

where $\text{Key}_{d_1}[1]$ denotes the type of MMPLS-IWavelet, so there are four possibilities. Here, $\text{Key}_{d_1}[i]$ for $i = 2, 3, \ldots, m$ denotes the levels that are executed in each transformation. Given the plaintext length N, the number of possible values for $\text{Key}_{d_1}[i]$ for $i = 2, 3, \ldots, m$ is $\lfloor \log_2 N \rfloor$. The length of Key_{d_1} does not have a maximum limit, as it depends on the sender's choice. Assuming Key_{d_1} has length m, the number of ways to arrange Key_{d_1} is $4 \times (\lfloor \log_2 N \rfloor)^m$.

(b) The second decryption key Key_{d_2} is a sequence of finite non-negative integer where each element is the binary code encoding of $\mathcal{BC}(S_{k,m}[i])$ from the resulting signal transformation $S_{k,m}$ which consisting of approximation and detail signals. The length of $S_{k,m}$ is the same as the length of the ciphertext and $\mathcal{BC}(S_{k,m}[i])$ contains elements 0 and 1. Hence the number of possibilities of constructing the second decryption key is $2^{N'}$.

(c) The third decryption key, Key_{d_3}, is a sequence of positive integers representing the length of the signal coefficient approximation and signal detail for each level in each transformation. Note that Key_{d_3} is closely related to Key_{d_1}. If $|\text{Key}_{d_1}| = m$ then Key_{d_3} is partitioned into $m-1$ parts. Let $\text{Key}_{d_3(k)}$ be the kth part of Key_{d_3} with $|\text{Key}_{d_3(k)}| = \text{Key}_{d_1}[k+1] + 2$ for $k = 1, 2, \ldots, m-1$. Based on Section 6.2 we obtain that $N \leq \max(\text{Key}_{d_3(k)}) < N'$ for $k = 1, 2, \ldots, m-1$ and the other elements of $\text{Key}_{d_3(k)}$ depend on $\max(\text{Key}_{d_3(k)})$. Therefore, the number of possibilities for $\max(\text{Key}_{d_3(k)})$ is $N' - N$, so the number of ways to construct Key_{d_3} is $(N' - N)^{m-1} \times (\lfloor \log_2 N \rfloor)^{m-1}$.

Based on the above calculations, assuming that the lengths of the plaintext and ciphertext are N and N' respectively, and the length of Key_{d_1} is m, there are $4 \times (\lfloor \log_2 N \rfloor)^{2m-1} \times 2^{N'} \times (N' - N)^{m-1}$ possibilities for decryptions key from cryptographic algorithms using MMPLS-IWavelet. The key space of the decryption key grows exponentially in terms of N, N', and m (the length of Key_{d_1}). This result implies that obtaining the decryption key using exhaustive search attack is computationally infeasible.

In cryptanalysis, it is assumed that the attacker knows how to encrypt and decrypt data using a cryptographic algorithm in cryptanalysis. A ciphertext-only attack is a cryptanalysis model where the attacker only has access to the encrypted data. The attack is successful if the attacker can obtain the plaintext or, even better, the key [25].

In MMPLS-IWavelet, an attacker must determine the type of scheme used, the number of transformations used, and the levels executed for each transformation.

When using brute-force attack, the attacker must be able to guess the key length and the arrangement of levels used for each transformation based on the estimated plaintext length.

8. CONCLUSIONS AND OPEN PROBLEMS

In this research, we have developed a cryptogaphic algorithm using an integer wavelet transform based on the min-max-plus lifting scheme (MMPLS-IWavelet). The encryption and decryption process in this cryptography are executed through the decomposition and reconstruction of MMPLS-IWavelet. The key space analysis of the decryption key shows that it is computationally infeasible to obtain the decryption key exhaustively. We conducted experiments for some test data set in .txt format containing various characters to evaluate the performance of the constructed cryptographic algorithms across multiple evaluation metrics. In Table 1, the correlation value between plaintext and ciphertext for types MMPLS-IWavelet scheme types shows fluctuation but generally remain sufficiently close to zero as the number of characters in the plaintext increase. The encryption quality percentage for each MMPLS-IWavelet type average above 80%, indicating significant difference in the distribution of characters between plaintext and ciphertext. According to Tables 3 and 2, the MaxMinLS type exhibits the highest percentage for encryption quality among the four types, while the MinLS type has the lowest percentage in this category. The key level significantly impacts on ciphertext changes, as evident in Table 5. Entropy analysis indicates that the randomness of characters in the ciphertext is consistently greater than in plaintext. Tests with various level keys reveal that the MaxMinLs type consistently has the highest entropy value among all types. The decryption key exhibits identical sensitivity due to the requirement of using the same key for both encryption and decryption, suggesting its high sensitivity. Based on the computational cost, the asymptotic complexity of the proposed algorithm is linear with respect the number of input characters. The running time of the proposed algorithm is relatively fast,taking no more than 13 ms to encrypt a text document containing around 50,000 characters. Empirical tests indicates that the MaxMinLS scheme outperforms the four other proposed schemes. Overall, the constructed cryptographic algorithm demonstrates satisfactory and efficient performance.

One drawback of the proposed cryptosystem is that the generated ciphertext can be longer than the original plaintext. This is not the case for many currently used cryptographic systems, such as RSA. Therefore, a future challenge is to develop alternative encryption schemes that maintain equal ciphertext and plaintext lengths while still achieving good performance based on the empirical tests presented in this paper. In this paper, we have demonstrate that brute-force attacks are computationally infeasible for retrieving the key based on the key space analysis in Section 7.6. However, the cryptosystem may be vulnerable to other attack methods such as Known-plaintext attack, Chosen-plaintext attack, Differential cryptanalysis, and others. Further research is required to address these potential vulnerabilities.

(Received August 21, 2023)

REFERENCES

[1] J. Arul and M. Venkatesulu: Encryption quality and performance analysis of GKSBC algorithm. J. Inform. Engrg. Appl. *2(10)* (2012).

[2] J. Cahyono, Subiono, D. Adzkiya. and B. Davvas: A cryptographic algorithm using wavelet transforms over max-plus algebra. J. King Saud University-Computer Inform. Sci. *34* (2020), 2, 627–635. DOI:10.1016/j.jksuci.2020.02.004

[3] M. Durcheva: Some applications of idempotent semirings in public key cryptography. ACM Commun. Comput. Algebra *49* (2015), 1, 9.DOI:10.1145/2768577.2768600

[4] K. Fahim and M. Yunus: Max-plus algebra-based wavelet transforms and their applications in compressed image. Int. J. Tomography Simul. *30* (2017), 1, 118–126.

[5] K. Fujinoki and K. Ashizawa: Directional Lifting Wavelet Transform for Image Edge Analysis. Signal Process. J. Pre-proof (2023), 118–126.DOI:10.1016/j.sigpro.2023.109188

[6] M. Gafsi, N. Abbassi and Amdouni, A. Rim, M. A. Hajjaji, A. Mohamed, and A. Mtibaa: Hardware implementation of the Haar 2D discrete wavelet transform with an application to image watermarking. In: 2022 5th International Conference on Advanced Systems and Emergent Technologies (IC_ASET), IEEE 2022, pp. 324–329.DOI:10.1109/IC_ASET53395.2022.9765864

[7] A. Gon and A. Mukherjee: FPGA-Based Low-Cost Architecture for R-Peak Detection and Heart-Rate Calculation Using Lifting-Based Discrete Wavelet Transform. Circuits Systems Signal Process. *42* (2022), 1, 580–600.DOI:10.1007/s00034-022-02148-7

[8] D. Goswami, N. Rahman, J. Biswas, A. Koul, L. R. Tamang, and A. K. Bhattacharjee: A discrete wavelet transform based cryptographic algorithm. Int. J. Computer Sci. Network Security *11* (2011), 4, 178–182.

[9] D. Grigoriev and V. Shpilrain: Tropical cryptography. Commun. Algebra *42* (2014), 6, 2624–2632.DOI:10.1080/00927872.2013.766827

[10] H. J. Heijmans and J. Goutsias: Nonlinear multiresolution signal decomposition schemes. II. Morphological wavelets. IEEE Trans. Image Process. *9* (2000), 11, 1897–1913.DOI:10.1109/83.877211

[11] M. Hellman: New directions in cryptography. IEEE Trans. Inform. Theory *22* (1976), 6, 644–654.DOI:10.1109/TIT.1976.1055638

[12] F. Kistosil, D. Adzkiya, and Subiono: Generalized public transportation scheduling using max-plus algebra. Kybernetika *54* (2018), 2, 243–267.DOI:10.14736/kyb-2018-2-0243

[13] R. Klees and R. Haagmans: Wavelets in the Geosciences. Springer, Berlin 1996.DOI:10.1007/BFb0011091

[14] S. Mallat: A Wavelet Tour of Signal Processing. Elsevier, New York 2009. DOI:10.1016/B978-0-12-374370-1.X0001-8

[15] D. A. Merdekawati and Subiono: Closed Shop Scheduling Optimisation using Max-Plus Automata. J.f Physics: Conference Series *1341* (2019), 4, 042015.DOI:10.1088/1742-6596/1341/4/042015

[16] R. Naseer, M. Nasim, M. Sohaib, J. Younis, A. Mehmood, M. Alam, and Y. Massoud: VLSI architecture design and implementation of 5/3 and 9/7 lifting Discrete Wavelet Transform. Integration *87* (2022), 253–259. DOI:10.1016/j.vlsi.2022.07.009

[17] H. Nobuhara, D. B. K. Trieu, T. Maruyama, and B. Bede: Max-plus algebra-based wavelet transforms and their FPGA implementation for image coding. Inform. Sci. *180* (2010), 12, 3232–3247.DOI:10.1016/j.ins.2010.05.003

[18] M. B. Parthasarathy and B. Srinivasan: Increased security in image cryptography using wavelet transforms. Indian Jo.Sci. Technol. *269* (2014), 21–34.DOI:10.17485/ijst/2015/v8i12/62433

[19] R. L. Rivest, A. Shamir, and L. Adleman: A method for obtaining digital signatures and public key cryptosystems. Commun. ACM *21* (1978), 2, 120–126.DOI:10.1145/359340.359342

[20] S. A. Salehi and D. D. Dhruba: Efficient hardware implementation of discrete wavelet transform based on stochastic computing. In: IEEE Computer Society Annual Symposium on VLSI (ISVLSI) 2020, pp. 422–427.DOI:10.1109/ISVLSI49217.2020.00083

[21] J. H. Silverman, J. Pipher, and J. Hoffstein: An Introduction to Mathematical Cryptography. Kybernetika, Springer, New York 2008. DOI:10.1007/978-0-387-77993-5

[22] W. Sweldens: The lifting scheme: A construction of second generation wavelets. SIAM J. Math. Anal. *29* (1998), 2, 511–546.DOI:10.1137/S0036141095289051

[23] W. Sweldens: ZAMM-Zeitschrift fur Angewandte Mathematik und Mechanik. SIAM J. Math. Anal. *76* (1996), 2, 41–44.

[24] W. Sweldens: Lifting scheme: a new philosophy in biorthogonal wavelet constructions. Wavelet Appl. Signal Image Process. III *2569* (1995), 68–79.DOI:10.1117/12.217619

[25] C. Swenson: Modern Cryptanalysis: Techniques for Advanced Code Breaking. John Wiley and Sons, Indianapolis 2008.

[26] A. Szczesna, A. Switonski, J. Slupik, J. H. Zghidi, H. Josinski, and K. Wojciechowski: Quaternion lifting scheme applied to the classification of motion data. Inform. Sci. *575* (2021), 732–746.DOI:10.1016/j.ins.2018.09.006

[27] Y. Tao and C. Wang: Global optimization for max-plus linear systems and applications in distributed systems. Automatica *119* (2020), 109104. DOI:10.1016/j.automatica.2020.109104

[28] S. Tedmori, and N. Al-Najdawi: Image cryptographic algorithm based on the Haar wavelet transform. Inform. Sci. *269* (2014), 21–34.DOI:10.1016/j.ins.2014.02.004

[29] A. Ukasha: Double compression efficiency for image data hiding using integer wavelet transform. In: International Conference on Engineering and MIS (ICEMIS), 2022, pp. 1–7.DOI:10.1109/ICEMIS56295.2022.9914045

[30] R. Walpole: Introduction to Statistics. New York 1974.

[31] S. Zarkar, S. Vaidya, A. Bharambe, A. Tadvi, and T. Chavan: Secure server verification by using encryption algorithm and visual cryptography. Int. J. Sci. Res. (IJSR) (2014), 310–313.

[32] H. Zhang, Y. Tao, Yuegang. and Z. Zhang: Strong solvability of interval max-plus systems and applications to optimal control. Systems Control Lett. *96* (2016), 88–94.DOI:10.1016/j.sysconle.2016.07.005

Mahmud Yunus, Corresponding author. Department of Mathematics, Faculty of Science and Data Analytics, Institut Teknologi Sepuluh Nopember, Kampus ITS Sukolilo-Surabaya 60111. Indonesia.
e-mail: yunusm@matematika.its.ac.id

Mohamad Ilham Dwi Firmansyah, Department of Mathematics, Faculty of Science and Data Analytics, Institut Teknologi Sepuluh Nopember, Kampus ITS Sukolilo-Surabaya 60111. Indonesia.
e-mail: 7002222004@student.its.ac.id

Subiono, Department of Mathematics, Faculty of Science and Data Analytics, Institut Teknologi Sepuluh Nopember, Kampus ITS Sukolilo-Surabaya 60111. Indonesia.
e-mail: subiono2008@matematika.its.ac.id