computers

MDPI

Article

Enhancing Data Security: A Cutting-Edge Approach Utilizing Protein Chains in Cryptography and Steganography

Noura A. Mawla * and Hussein K. Khafaji

Department of Computer Science, AL-Rafidain University College, Baghdad 46036, Iraq; hussain.ketan.elc@ruc.edu.iq
* Correspondence: noora.ahmed.elc@ruc.edu.iq

Abstract: Nowadays, with the increase in cyber-attacks, hacking, and data theft, maintaining data security and confidentiality is of paramount importance. Several techniques are used in cryptography and steganography to ensure their safety during the transfer of information between the two parties without interference from an unauthorized third party. This paper proposes a modern approach to cryptography and steganography based on exploiting a new environment: bases and protein chains used to encrypt and hide sensitive data. The protein bases are used to form a cipher key whose length is twice the length of the data to be encrypted. During the encryption process, the plain data and the cipher key are represented in several forms, including hexadecimal and binary representation, and several arithmetic operations are performed on them, in addition to the use of logic gates in the encryption process to increase encrypted data randomness. As for the protein chains, they are used as a cover to hide the encrypted data. The process of hiding inside the protein bases will be performed in a sophisticated manner that is undetectable by statistical analysis methods, where each byte will be fragmented into three groups of bits in a special order, and each group will be included in one specific protein base that will be allocated to this group only, depending on the classifications of bits that have been previously stored in special databases. Each byte of the encrypted data will be hidden in three protein bases, and these protein bases will be distributed randomly over the protein chain, depending on an equation designed for this purpose. The advantages of these proposed algorithms are that they are fast in encrypting and hiding data, scalable, i.e., insensitive to the size of plain data, and lossless algorithms. The experiments showed that the proposed cryptography algorithm outperforms the most recent algorithms in terms of entropy and correlation values that reach −0.6778 and 7.99941, and the proposed steganography algorithm has the highest payload of 2.666 among five well-known hiding algorithms that used DNA sequences as the cover of the data.

Keywords: cyber-attacks; hacking; security; cryptography; steganography; hexadecimal system; protein motifs

check for updates

Citation: Mawla, N.A.; Khafaji, H.K. Enhancing Data Security: A Cutting-Edge Approach Utilizing Protein Chains in Cryptography and Steganography. *Computers* **2023**, *12*, 166. https://doi.org/10.3390/computers12080166

Academic Editors: Ömer Aslan and Refik Samet

Received: 10 July 2023
Revised: 8 August 2023
Accepted: 11 August 2023
Published: 19 August 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

In the realm of cyber security, the intertwining concepts of cryptography and steganography offer a robust approach to countering the escalating threats of computer and network attacks. The historical vulnerabilities that have plagued each of these fields independently are now being reimagined as strengths when combined.

The synergy between cryptography, the art of securing data through encryption techniques, and steganography, the science of concealing information within innocuous mediums, presents an innovative defense mechanism. This integrated strategy is particularly pertinent as the Internet continues its rapid expansion, necessitating novel solutions to safeguard sensitive information.

Amid this landscape, the imperative for heightened security measures is undeniable. Web servers and individual users' sites frequently fall victim to unauthorized compromises, often oblivious to the underlying vulnerabilities. These vulnerabilities, ranging

from perilous SQL injection to pervasive cross-site scripting, underscore the importance of fortified defenses. Institutional breaches, corporate infiltrations, website hijackings, and assaults on social media platforms pose grave risks to both confidentiality and stability. It is within this context that data confidentiality emerges as a paramount concern across diverse domains. OWASP's (Open Web Application Security Project) diligent efforts, encapsulated in its vigilant tracking of threats such as SQL injection and cross-site scripting, serve as crucial safeguards against such vulnerabilities. As technology's trajectory forges ahead, the unification of cryptography and steganography presents a potent solution to these contemporary challenges. This approach is underscored by the pressing need to uphold privacy and fortify systems against escalating threats, cementing the significance of data security in the digital age [1,2].

Cryptography, also known as the science of secret writing, involves techniques and mathematical principles to safeguard information. It focuses on data integrity, confidentiality, and authentication of the data source and entities involved. Encryption aims to make data unreadable, ensuring secure storage and transmission across insecure networks without compromising its content or exposing it to unauthorized parties [3–5].

Steganography and cryptography have a shared goal of safeguarding information against attacks, but their approaches to data protection differ. Cryptography transforms data to make it unreadable, while steganography conceals data within a cover, preserving its original structure to remain inconspicuous.

Various methods have been developed for hiding data, including audio, video, and images. Previous research has proposed numerous algorithms to hide sensitive data within images [6–8]. However, images have limited capacity for embedding data, leading to the storage of only small amounts of information within them [9].

To overcome the limited capacity of traditional methods, a groundbreaking approach called "DNA steganography" was introduced in 1999. DNA, composed of four bases (adenine, guanine, cytosine, and thymine), contains intricate information about organisms. Utilizing DNA sequences as carriers enables the secure transfer of sensitive data [10,11].

DNA steganography is an evolving scientific field that focuses on concealing data within a diverse range of DNA sequences to prevent unauthorized parties from reading and decrypting messages. The key to its success lies in safeguarding the patterns from potential adversaries [12–14]. This area of research was introduced in the previous years with novel and secure solutions for data hiding and transmission.

Animeshet et al. (2018) provided an overview of the approaches utilized in DNA cryptography and their real-time implementation [15]. The first approach is the DNA random one-time pad method, using non-repeating characters arranged randomly as ciphertext for input. While offering enhanced security, it is limited to short messages due to hardware constraints.

The second approach uses DNA chips, which represent a significant breakthrough in DNA cryptography. These chips enable encryption and decryption operations through biochemical processes and can encrypt images and plaintext messages. However, DNA chip characteristics may be susceptible to alteration by external factors, potentially affecting the ciphertext.

The third approach is DNA steganography, a recently developed technique for concealing substantial data using biological strands. However, environmental changes can lead to data contamination.

The fourth approach involves DNA fragmentation, dividing the DNA sequence into small fragments for added protection. Additionally, the encryption of the key itself is implemented. Hybrid methods combining conventional, elliptic curves, DNA, and quantum cryptography are also being explored.

Vantigaru (2019) [16] introduced a dual-layered approach to enhance algorithm performance. It combined DNA steganography in the first layer and AES and DES cryptography through parallel programming in the second layer. The combination resulted in 20–30% less time required compared to traditional encryption methods.

In Noura's groundbreaking study (2022) [17], the "stego-protein algorithm" was introduced, representing the first-ever exploration of data hiding using protein motifs. This algorithm stands out by providing the highest data-hiding capacity compared to other DNA and RNA-based algorithms. Utilizing the 20 protein bases, the stego-protein algorithm surpasses its DNA-based counterparts owing to the broader spectrum of protein bases available, which far exceeds the limited pool of four bases found in DNA and RNA.

In this paper, a method is proposed that employs protein sequences to accomplish the cryptography and steganography processes. Protein bases are utilized to generate the cipher key. The encrypted data is discreetly concealed and randomly distributed within the protein chains. This evades detection of the hidden data by statistical methods and analysis algorithms. This method offers a wide storage environment capable of accommodating a significant amount of data within the protein chains. In contrast to image-based hiding algorithms and the one-time pad algorithm, which have limited storage capacity, the proposed algorithm provides a greater capacity for data storage.

The amino acids used in previous concealment processes are biological components. They are susceptible to physical variations that may result in unpreserved biological sequence functionality. This vulnerability is eliminated in the proposed steganography technique, where the data are securely hidden within purposefully designed protein chains. Moreover, the proposed method, the steganography method using protein chains (SMUPC), outperforms the "stego-protein algorithm" and other algorithms in terms of correlation, entropy, payload capacity, and clear superiority in performance. Section 2 presents some concepts of the proteins related to the objectives of the paper before explaining the proposed method in Section 3.

2. Protein Chain

Computational biology, also known as bioinformatics, is a multidisciplinary field that revolves around the application of computer software to both biological and non-biological systems. Its primary focus lies in the analysis, interpretation, and manipulation of biological data, with the ultimate goal of addressing various biological challenges. The term "computational biology" stems from the convergence of information technology, biology, and mathematics.

At its core, computational biology is dedicated to modeling biological processes, particularly those involving genes, DNA, RNA, and protein chains. This approach proves especially valuable when comparing genes with other protein chains or exploring interactions within and between different organisms. Furthermore, computational biology delves into investigating the intricate relationship between evolution and living organisms, unraveling the underlying mechanisms that drive evolutionary changes.

A pivotal application of computational biology lies in the identification of patterns present in DNA and protein sequences. By employing sophisticated algorithms and analytical tools, researchers can discern functional regions within DNA and proteins, shedding light on their roles and contributions to various biological processes. This capability opens up new avenues for drug discovery, disease diagnosis, and the exploration of fundamental biological phenomena.

In essence, computational biology serves as a powerful ally to modern biology, harnessing the potential of cutting-edge computational techniques to gain deeper insights into the complexities of life, laying the foundation for groundbreaking discoveries, and advancing our understanding of the natural world [18].

In this research paper, the selection of the protein chain instead of the other well-known chains such as DNA and RNA sequences was driven by its broader capacity for data hiding. Unlike DNA and RNA, which are represented by only four bases, the protein chain boasts a more extensive repertoire of 20 bases, as indicated in Table 1. This larger base set offers increased flexibility and versatility in the steganography process.

Table 1. Protein bases' names, abbreviations, and their suggested binary representation and partitioning.

Amino Acid	Code of Amino Acid	Binary Code (010 XX XXX) Matching		Amino Acid	Code of Amino Acid	Binary Code (010 XX XXX) Matching	
		2-Bits	3-Bits			2-Bits	3-Bits
Proline	P	10	000	Aspartic acid	D	00	
Histidine	H	01		Leucine	L	01	100
Alanine	A	00		Threonine	T	10	
Isoleucine	I	01	001	Glutamic acid	E	00	101
Glutamine	Q	10		Methionine	M	01	
Tyrosine	Y	11		Phenylalanine	F	00	
Arginine	R	10	010	Asparagine	N	01	110
Cysteine	C	00		Valine	V	10	
Lysine	K	01	011	Glycine	G	00	111
Serine	S	10		Tryptophan	W	10	

This method capitalizes on the inherent patterns within the binary makeup of protein bases. The concept revolves around strategically selecting groups of bits, akin to assembling pieces of a puzzle, while meticulously preserving the overall structural coherence of the genetic code. This delicate artistry of integration ensures that the encoded text can be subtly enmeshed within the binary framework, effectively camouflaging the information without raising suspicions.

The beauty of this approach lies in its ability to harness the genetic blueprint as a carrier for hidden communication. As advancements in molecular biology and bioinformatics continue to unfold, the fusion of these disciplines paves the way for unobtrusive information embedding. This strategy holds potential for a wide array of applications, ranging from covert data transmission in genetic research to secure communication channels inspired by the very fabric of life itself. The choice of the protein chain as the medium for data hiding stems from its diverse and rich composition of amino acids. Each amino acid corresponds to a specific base within the protein sequence, and this diverse range of bases enhances the hiding capacity of the steganography algorithm. In contrast, DNA and RNA sequences, being limited to only four bases (adenine, cytosine, guanine, and thymine in DNA or uracil in RNA), offer a more constrained scope for concealing information.

By leveraging the 20 bases of the protein chain, researchers can embed data in a manner that effectively hides the information while preserving the integrity of the protein's primary function.

This method capitalizes on the inherent patterns within the binary makeup of protein bases. The concept revolves around strategically selecting groups of bits, akin to assembling pieces of a puzzle, while meticulously preserving the overall structural coherence of the protein base. This delicate artistry of integration ensures that the encoded text can be subtly enmeshed within the binary framework, effectively camouflaging the information without raising suspicions.

The beauty of this approach lies in its ability to harness the genetic blueprint as a carrier for hidden communication. As advancements in molecular biology and bioinformatics continue to unfold, the fusion of these disciplines paves the way for unobtrusive information embedding. This strategy holds potential for a wide array of applications, ranging from covert data transmission in genetic research to secure communication channels inspired by the very fabric of life itself. This approach not only ensures a higher level of security but also minimizes the chances of data loss or corruption during the hiding process [19–21].

In conclusion, the utilization of the protein chain as cover data in this steganography approach represents a strategic choice, capitalizing on its broader base and versatility to achieve more efficient and effective data concealment.

3. The Proposed Cryptography and Steganography Algorithms

In this section, we initially introduced the theoretical requirements that led to the design of the proposed algorithm. The protein chain will be used in both the encryption and ciphertext steganography processes. In the encryption process, a chain of proteins will be generated to be used as a key in the plaintext encryption and decryption processes. This key will be hidden in the main protein chain, and then it will be used by the recipient to decrypt the ciphertext.

The second use of the protein chain in this paper is in the field of steganography of ciphertext, where it is used as cover data, which plays the role of a container that hides the ciphertext, and then it will be sent to the recipient without raising suspicions about this message. The process of hiding occurs inside the protein bases. Initially, the protein bases are manipulated as binary ASCII codes. In this paper, the protein base in binary representation is divided into three groups; consider Figure 1.

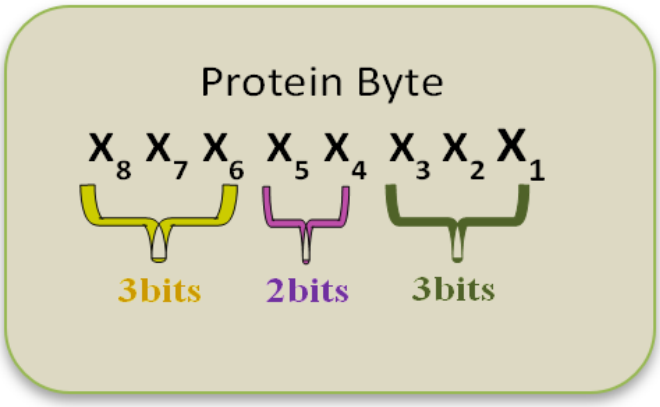

Figure 1. Suggested partitioning of protein base in binary representation/steganography.

The first group consists of three bits ($X_3X_2X_1$), the second group consists of two bits (X_5X_4), and the third group consists of three bits ($X_8X_7X_6$). All bases have the same pattern in the 3rd group, i.e., 010. Therefore, the last three bits from each protein base will not be mentioned in Table 1, but the first and second groups are mentioned, in which the operations of hiding the cipher texts will occur.

Table 1 is built based on the similarity of the first three bits of the protein bases. Figure 2 shows the groups of protein bases that share the same sequence of the first three bits, while Figure 3 shows the other groups of protein bases that have the same sequence of the fourth and fifth bits.

In this section, cryptography and steganography will be combined to protect data from penetration. In this paper, four algorithms are designed to perform the information protection process: Two of them are responsible for the encryption and decryption processes, and the other two are responsible for the cipher text hiding and discovery processes. These proposed algorithms will be presented with examples to explain their work.

3.1. Encryption Algorithm

This algorithm is responsible for converting the plaintext into cipher text using a key that is generated from a protein character set that will be used later in the processes of steganography, transmitting the cipher text, and delivering it to the second receiving party. Figure 4 illustrates the proposed algorithm and how it works, based on an illustrative example. It is written in an unprecedented manner that combines the algorithmic steps and the illustrative example steps to understand the algorithm.

In the first step, the key that will be used in the encryption process will be randomly generated from the protein chain, and its length will be twice the plaintext length. This key is converted to the hexadecimal system and divided equally into two parts, K1 and K2, after which the order of the second part of the key is reversed, as shown in the second step.

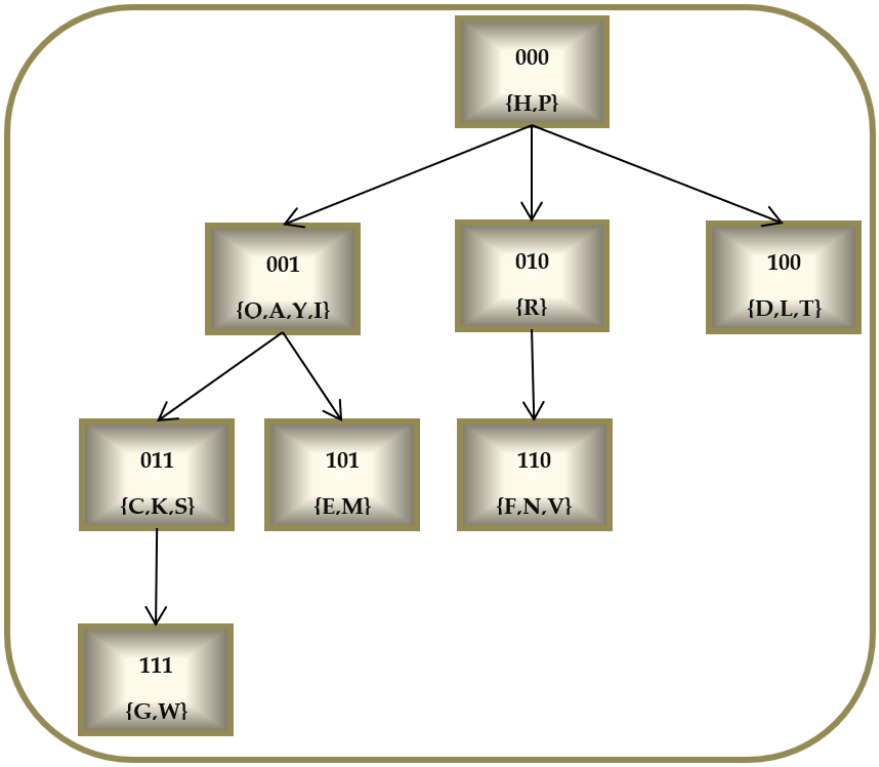

Figure 2. Similarity of protein bases based on binary representation (1st group).

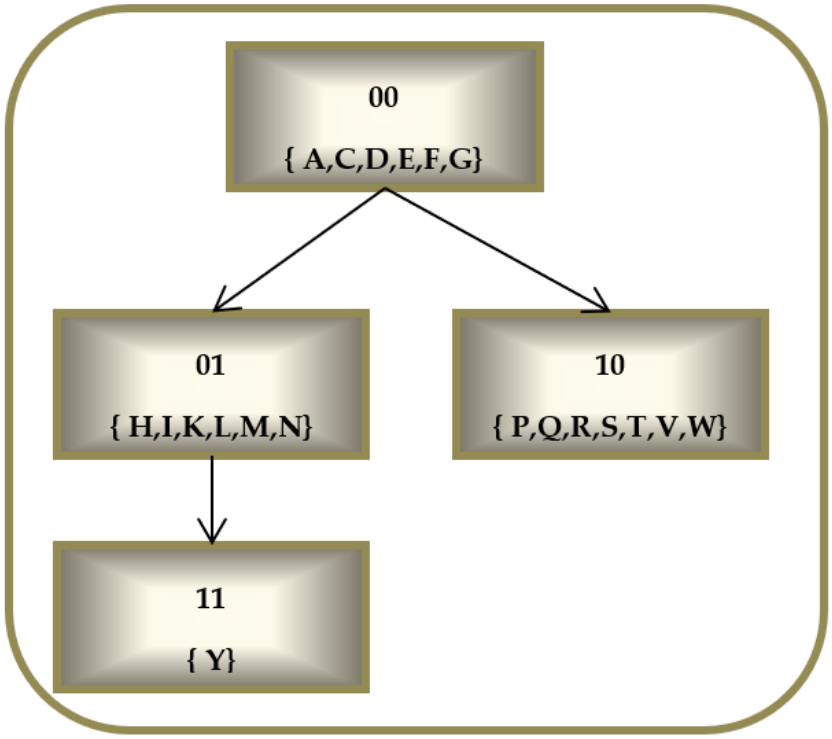

Figure 3. Similarity of protein bases based on binary representation (2nd group).

The first part of the key, K1, is passed to the third step and combined with the plain text after converting the plain text to the hexadecimal system and saving the result to the H_Sum matrix. The second part of the key, K2, is passed to the fourth step; each hexadecimal number of K2 is separated into two digits; multiply these two digits; the result is stored in the Mult_Dig matrix.

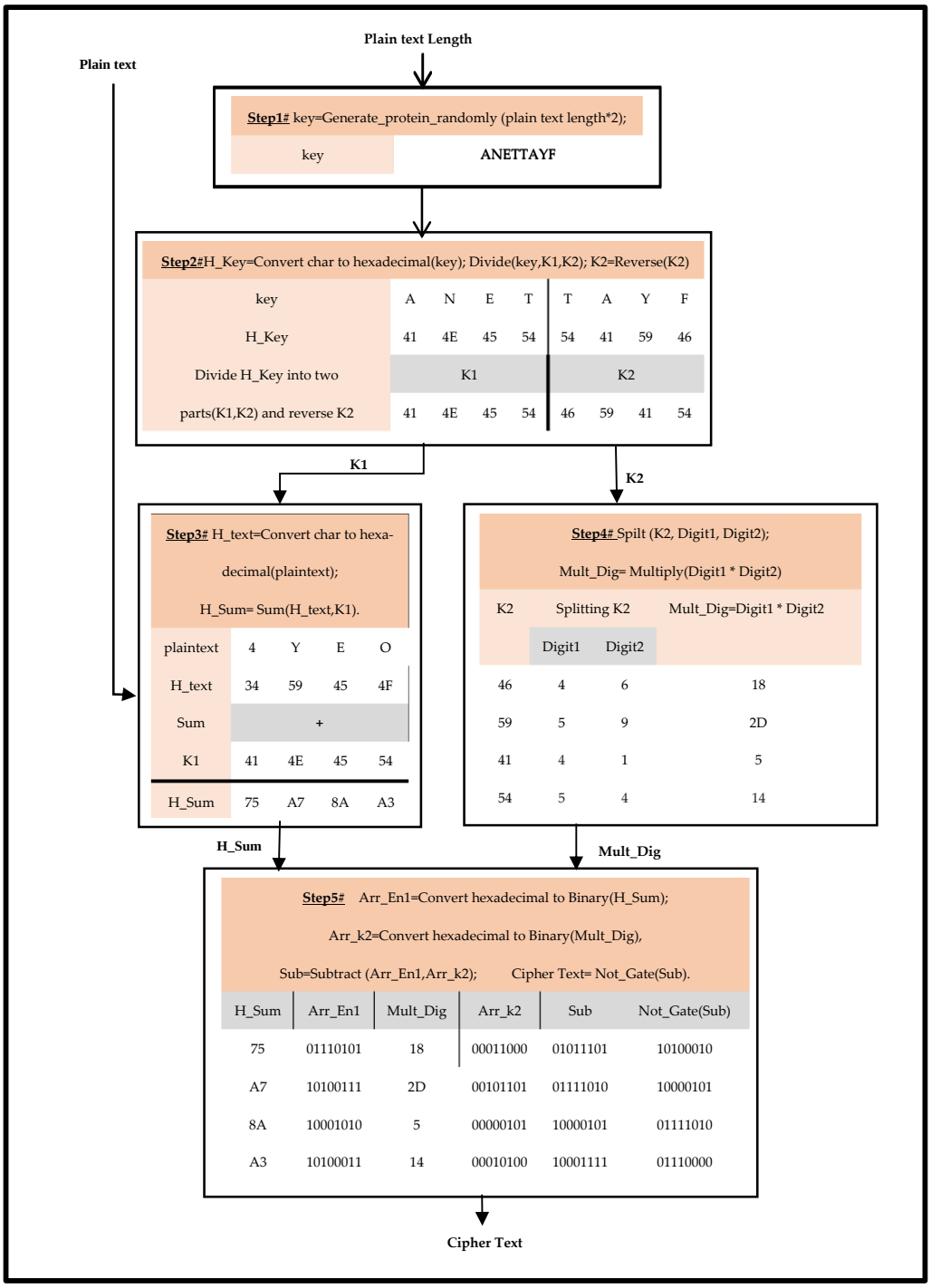

Figure 4. Block diagram and illustrative example for encryption algorithm.

H_Sum and Mult_Dig are converted to binary. Their results are saved in Arr_En1 and Arr_k2, respectively, as shown in the fifth step. Each byte of Arr_En1 and its corresponding byte of Arr_k2 are subtracted, and the result is passed to the NOT Gate to obtain the cipher text.

3.2. Steganography Method Using Protein Chains (SMUPC)

In this paper, a new algorithm is proposed for the process of hiding the cipher text in different random locations within the protein chain. The steps of this algorithm will be explained in detail in this section.

The hiding mechanism in this algorithm is based on dividing each byte of the ciphertext into three parts, two of which are three bits in size and one of which consists of two bits. Three bases of the protein chain, which were previously explained in Section 2, will be generated such that their bits in predetermined locations are the matching bits of these three bases.

This algorithm introduces a new technique in the hiding process that is based on a variable known as a "fixed variable." A fixed variable is a type of static variable that does not lose its final value when the algorithm is finished but instead keeps its last value, and this stored value is used when the algorithm is called again. This variable will have an effective role in determining the locations where the hiding operations will start in a protein chain, and these locations will differ each time the algorithm is called depending on the value of a fixed variable, as we will explain while displaying the algorithm steps. In the first use of the algorithm, this variable is devoid of any value. Figure 5 displays a block diagram that shows how the algorithm works with an illustrative example that explains how to hide the text that has been encrypted in Section 3.1.

The first step in this algorithm will be to initialize the protein chain, where several random protein bases are generated according to the fixed variable value. If the fixed variable value is zero, then the protein chain generated based on the fixed variable value is empty. This condition is executed only on the first use of the algorithm. This is because the value of the fixed variable will change with each use of the algorithm, where it will be increased by the plain text length, L, as shown in the last step of this algorithm, but if the fixed variable value is equal to a Z, then Z protein bases are randomly generated and added at the beginning of the protein chain.

In the second step, the length of the plaintext will be prepared for the hiding process within the protein chain. It will be converted from decimal to binary and divided into three groups of bits according to the following sequence: The first to the third bits are stored in the first group; the second group selects bits from the fourth bit to the sixth bit; and the last two bits of the seventh and eighth positions are stored in the third group, as shown in Figure 6. After that, three protein bases will be generated such that their bits are identical to the three groups mentioned previously.

For example, Figure 7 contains the byte of the plaintext length, i.e., 4, which will be divided into three sections (S2Bits = 00, S3Bits = 000, and S3Bits = 100). To hide the binary value of the plaintext length, a protein whose first three bits are identical to the first section (100) will be generated. Recall Figure 2, where there are three protein bases that meet these specifications; one of them will be chosen randomly in the process of hiding. Suppose that the generated base is D. For the second section, i.e., (000), the same process will be performed to generate a suitable protein base. Indeed, there are two identical options (H and P), and one of them will be chosen to hide this section. For illustration, P is selected as shown in Figure 7.

The third section, i.e., (00), is only two bits long. A protein base in which the fourth and fifth bits are identical to the third section must be selected. Recalling Figure 3, it is noticed that there are six protein bases matching this section: A, C, D, E, F, and G. Suppose that A is chosen for this purpose.

The three protein bases that contain the length of the plaintext are added to the protein chain at positions Z + 1, Z + 2, and Z + 3, as shown in the third step. In the fourth step, L protein bases are randomly generated and added along with the key to the protein chain to complete the process of hiding the cipher key. The fifth step is responsible for setting the locations where the ciphertext will be hidden, based on Equation (1):

$$\text{New location} = \text{Old location} + ((\text{Old location}/3L) + (\text{Old location}\%\ 3\ L))/2 \quad (1)$$

Note: If this was the first step, i.e., i = 1, in the process of hiding the ciphertext, the value of the old location, (x [1]), would be set equal to the current protein chain length.

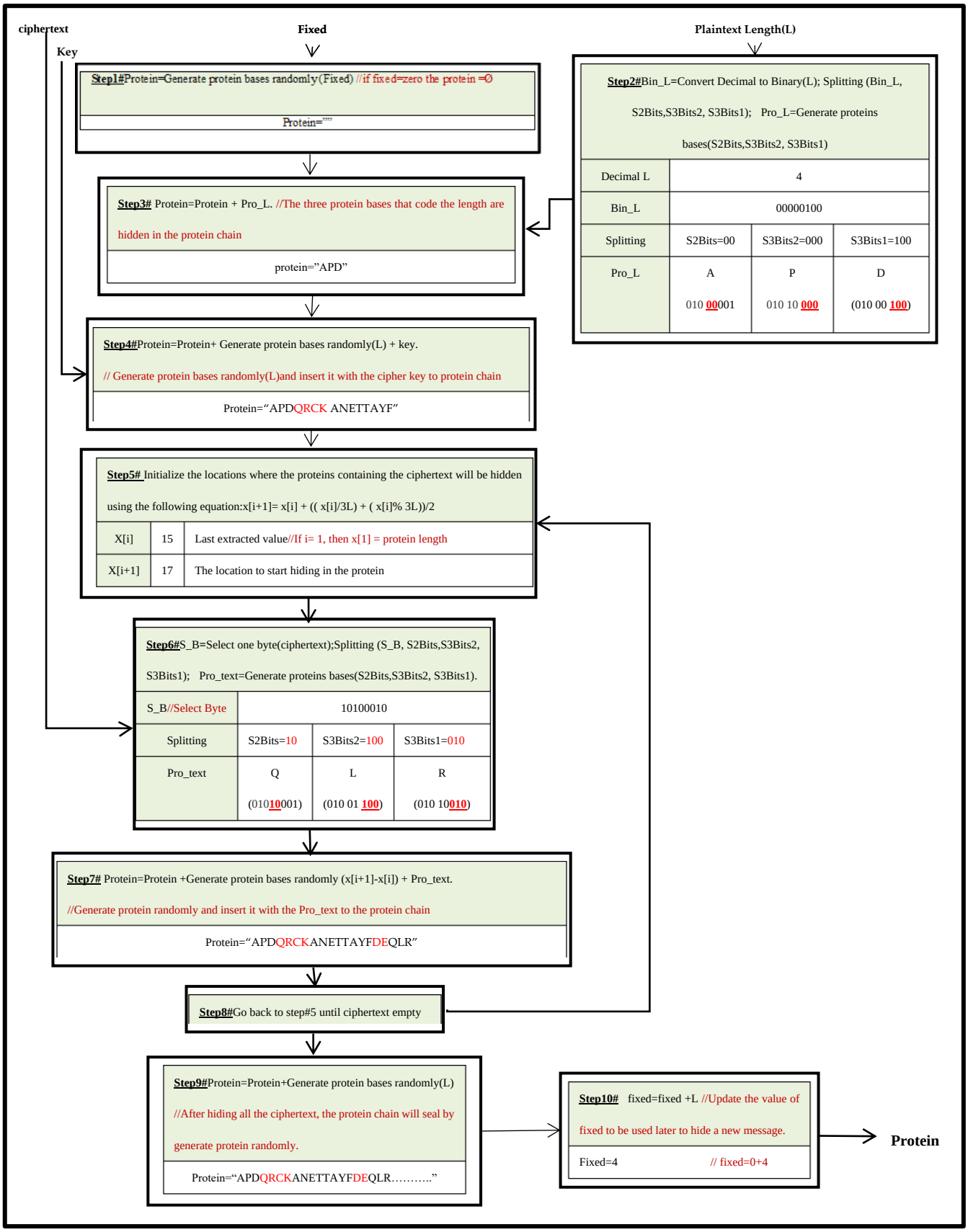

Figure 5. Block diagram and illustrative example for SMUPC algorithm.

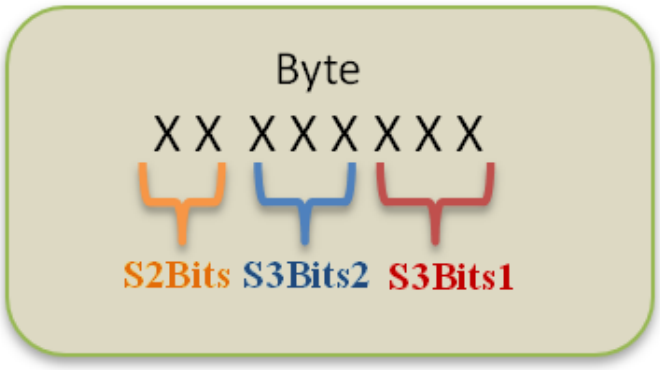

Figure 6. Method of dividing a base byte.

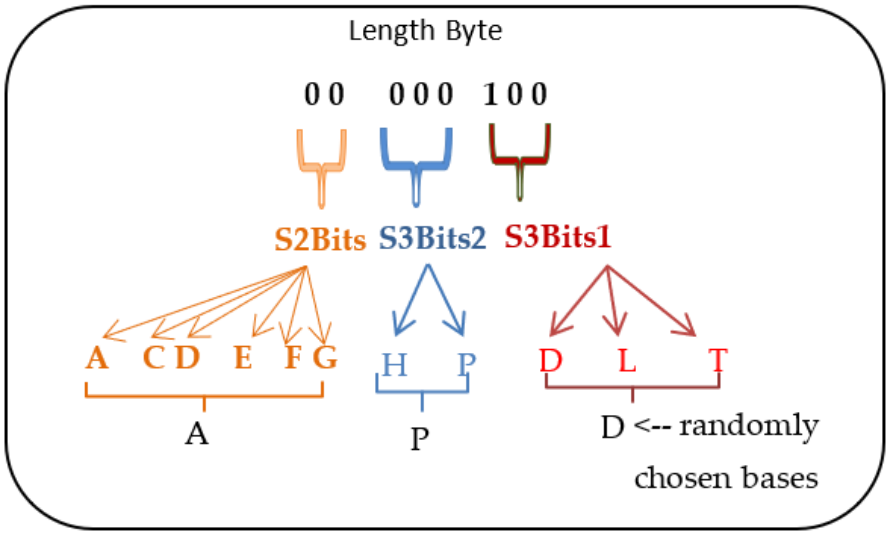

Figure 7. Generating method of protein bases for text hiding.

In the sixth step, the ciphertext is initialized for the hiding process. The process begins with selecting a byte from the ciphertext, and the same operations that were previously explained in the second step will be repeated on the encrypted byte as a hash and by creating three new protein bases called Pro_text. As shown in the seventh step, random protein bases are generated based on the difference between the new and old locations (x[i + 1] − x[i]) and added to the protein chain along with the Pro_text. Steps 5–7 are repeated until all ciphertext bytes have been concealed. In the ninth step, a random protein sequence of length equal to the length of the plain text is generated and added to the protein chain for stamping.

Finally, the value of the fixed variable will be increased by the value of L. This new value will be kept in the fixed variable and will be used later when the algorithm is reused again to encrypt and hide a new text.

3.3. Discovery Algorithm

Figure 8 presents the function of the discovery algorithm, which extracts the ciphertext from the protein chain in addition to the plaintext length and the key that was previously used in the encryption process and will be used later in the decryption process. The extraction of hidden elements from the protein chain is affected by the value of the fixed variable. In the first use of this algorithm, the value of the fixed variable is zero; after that, its value changes every time the discovery algorithm is called.

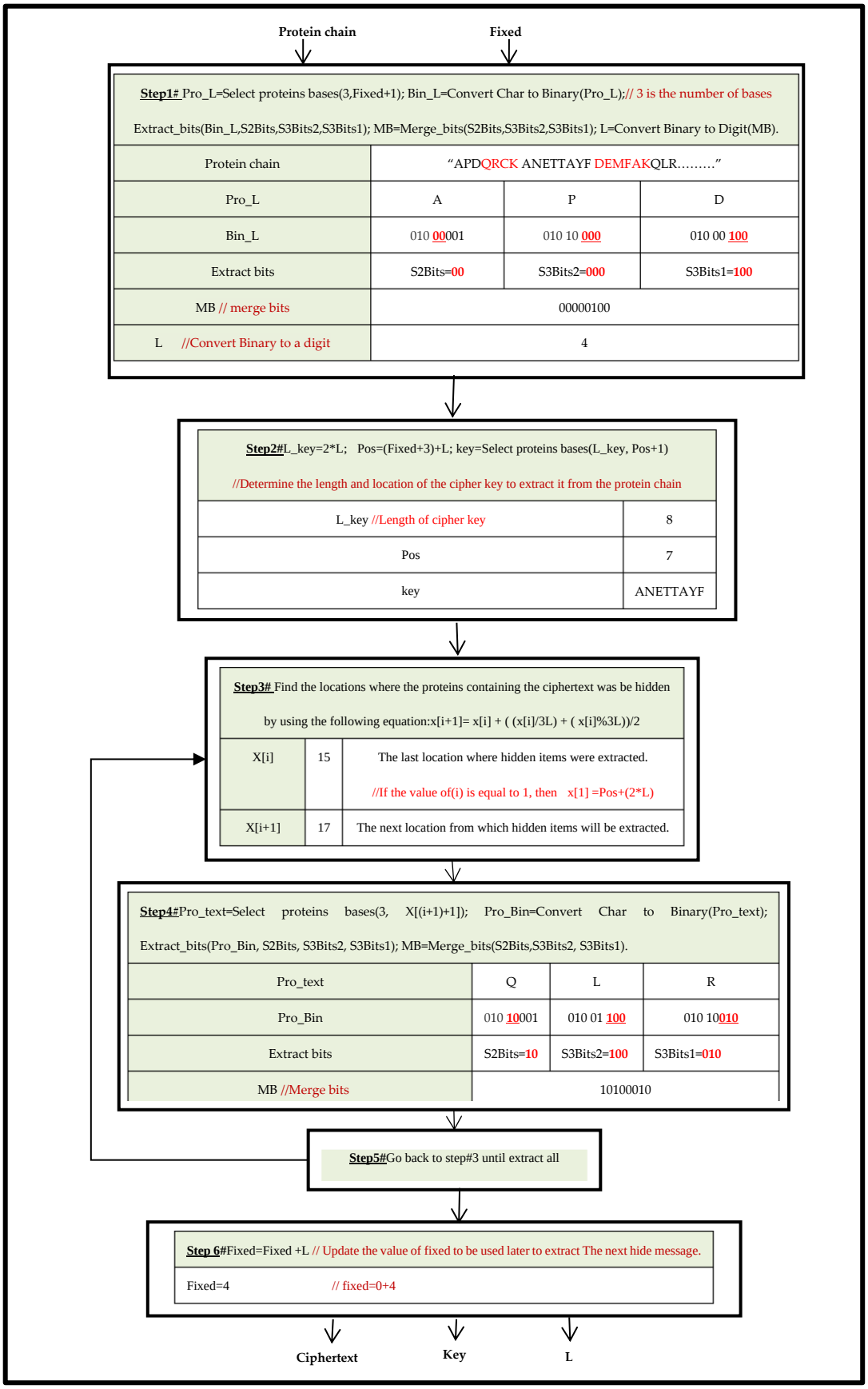

Figure 8. Block diagram and illustrative example for discovery algorithm.

In the first step, the length of the plaintext is extracted from the positions (Fixed_variable + 1, Fixed_variable + 2, and Fixed_variable + 3) of the protein chain. The position Fixed_variable+1 contains two bits that are hidden in the fourth and fifth bits of the protein character, and each of the remaining two characters of the protein contains three bits that are hidden in the first three positions of each byte, as shown in Figure 9. These bits are extracted from these three locations and combined to form a single byte. The collected byte is converted from binary to decimal to obtain the plaintext length.

In the second step, the key is located and extracted from the protein chain by using the length of the plaintext. The plaintext length was extracted in the previous step. The third step is responsible for identifying the locations where the ciphertext is hidden using the same equation that has been used in Section 3.2. After that, the ciphertext is extracted from the protein chain, as shown in the fourth step. Steps 3 and 4 are repeated until all of the ciphertext from the protein chain has been extracted.

In the last step of the algorithm, the value of the fixed variable is updated. Its value increases with the plaintext length that has been extracted from the protein chain in the first step. This new value will be stored and used later when this algorithm is used again.

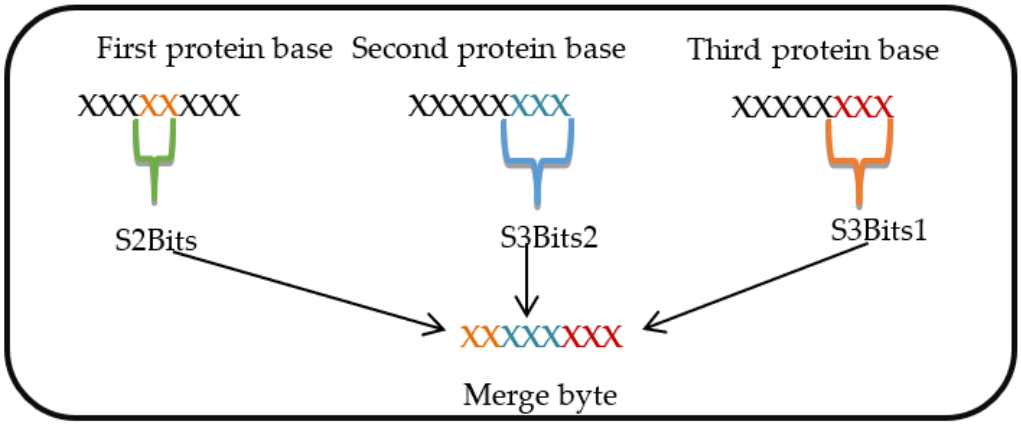

Figure 9. Byte partitioning.

3.4. Decryption Algorithm

After extracting the ciphertext, plaintext length, and cipher key from the protein chain in the previous section, the ciphertext will be decrypted and converted into an understandable plaintext using the following decryption algorithm, shown with an example in Figure 10. The steps of this algorithm are similar to those of the encryption algorithm described earlier in Section 3.1, but the order of these steps is different.

In the first step, the cipher key will be converted to hexadecimal, and then it will be divided into two equal parts, K1 and K2. The K2 matrix will undergo a reverse order of its contents to facilitate its use later in the next stage of the algorithm. Each hexadecimal number of K2 will be split into two digits. Then, these two digits will be multiplied by each other, and the resulting values will be converted into binary code and stored in Arr_k2, as shown in the second step.

Arr_k2 and the cipher text are passed to the third step. The cipher text will be passed to the Not gate, and the result will be saved in the Bin_Cipher matrix. Each byte of Arr_k2 will be combined with its counterpart in the Bin_Cipher, and the result will be converted to a hexadecimal number and stored in the H_En1 matrix. The final step in the decryption algorithm is to extract the plain text by subtracting the numbers in H_En1 from the first part of the key, K1, and converting the result to characters to obtain the plain text.

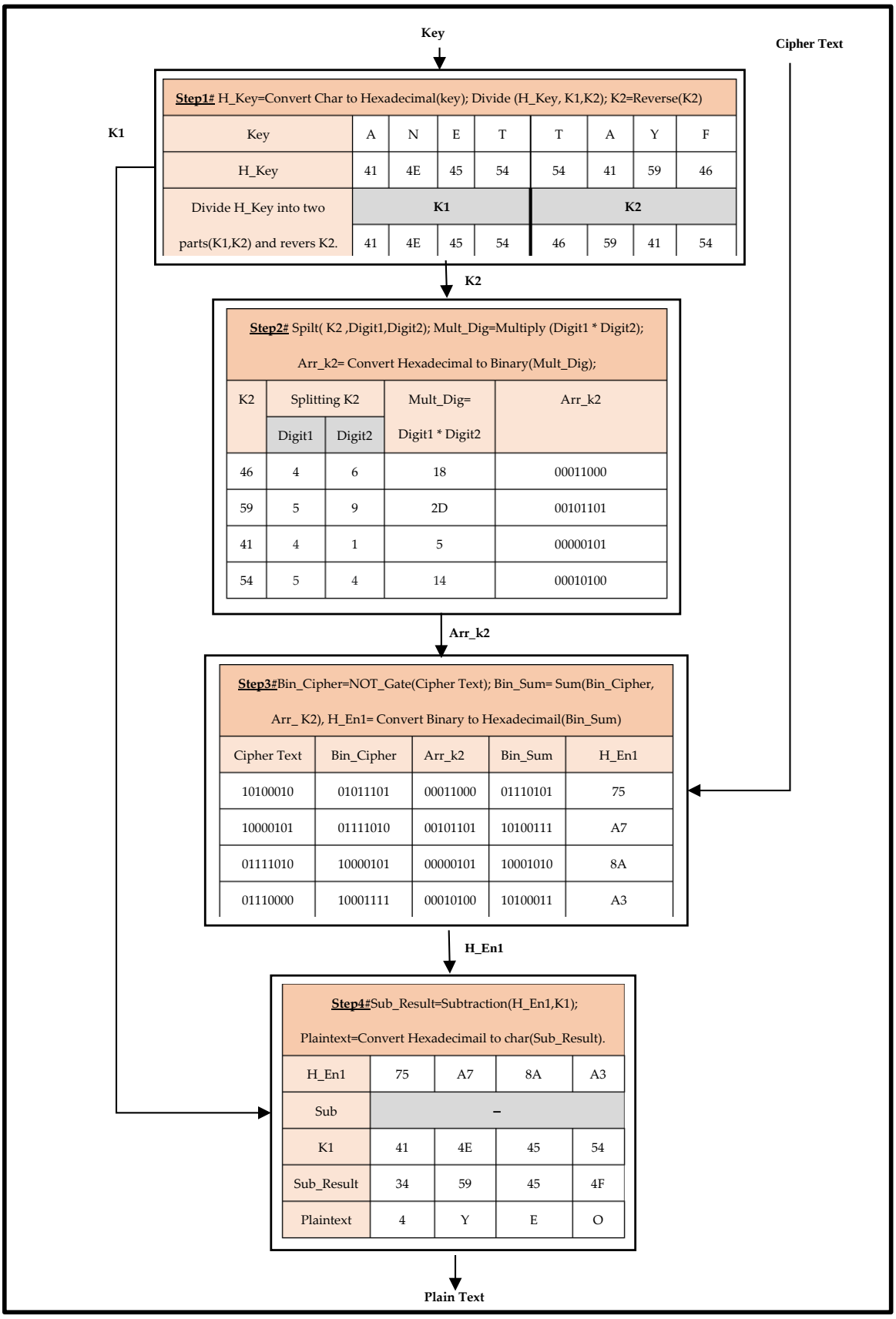

Figure 10. Block diagram and illustrative example for the decryption algorithm.

4. Experimental Result

In this section, we will evaluate the security of the algorithms discussed earlier and examine whether they meet the security criteria outlined by their respective factors. We will analyze the robustness and effectiveness of each algorithm in meeting the desired security objectives.

4.1. Comparison Criteria for the Proposed Cryptography Algorithm

We compared the proposed algorithms with algorithms presented in [5,17,22–24] according to correlation coefficient and information entropy.

4.1.1. Correlation Coefficient

Calculating the correlation coefficient and assessing how well the attacker can access the plaintext data allow the researchers to assess the encryption's strength. The more the result of the correlation coefficient moves away from the value of 1 and approaches the value of −1, the cipher text's resistance increases [25]. The correlation coefficient is calculated using Equation (2) as follows:

$$\begin{gathered} \mathrm{CorrCoef}(n, m) = \frac{\sum_{i=1}^{k}(n_i - E(n))(m_i - E(m))}{D(n)\ D(m)} \\ E(n) = \frac{1}{k}\sum_{i=1}^{k} n_i,\ \ D(n) = \sqrt{\sum_{i=1}^{k}(n_i - E(n))^2} \end{gathered} \tag{2}$$

where n_i represents plain text, m_i represents cipher text, k denotes total characters of plaintext, and E(n) and E(m) represent average values of n_i and m_i, respectively.

The suggested algorithm achieves excellent results compared to earlier encryption algorithms for the average correlation coefficients obtained in Table 2 for data files whose sizes range from 1256 to 10,277 bytes.

Table 2. The averages of the correlation coefficient comparison.

Algorithms	Ref [25]	Ref [23]	Ref [17]	Ref [24]	Ref [5]	Proposed Algorithm
Avg. Correlation Coefficient	0.0091	0.0021	−0.0000091	−0.002945	−0.4735	−0.6778

4.1.2. Information Entropy Analysis

This metric, known as the Shannon entropy, is used to calculate how difficult it is to guess or crack a particular encryption algorithm. Shannon's entropy, Equation (3), states that the amount of information contained in a message t is proportional to the logarithm of its size (k) [26,27]:

$$H(t) = -\sum\{0 \le i \le k-1\}\ p(t_i)\log_2 p(t_i) \tag{3}$$

H(t) = 8 is the best entropy that can be obtained for a message t, where $P(t_i)$ denotes the likelihood of m_i and $t = t_0, t_1, t_2, \ldots t_{255}$. In reality, it is not possible to reach the entropy of a perfect case, and hence a data encryption method must be able to create encryption with as much entropy as possible in order to make it more secure. Secure cryptography must be able to operate on the information in the entropy, and as such, the encrypted message should not divulge any information about the original message. The results show that the entropy value of the suggested cryptosystem method is approximately 7.99941, as shown in Table 3.

Table 3. Entropy comparison.

Algorithms	Ref [25]	Ref [24]	Ref [17]	Ref [23]	Ref [5]	Proposed Algorithm
Entropy	7.9978	7.7403	7.8199	7.90190	7.9983	7.99941

4.2. Comparison Criteria of SMUPC Algorithm

The efficacy of a protein steganography technique is dependent on several factors, which include but are not limited to:

- Data payload

The payload in the steganography literature, which is related to the DNA sequences, has two definitions; it is the largest amount of bits that can be concealed by the cover media, or it is the length of a sequence after and before steganography. This concept can be generalized to protein sequences. Accordingly, the best value for the payload is obtained when the payload equals zero, as determined by Equation (4).

$$\text{Payload} = |\text{PS}_{\text{after\_stego}}| - |\text{PS}_{\text{before\_stego}}| \tag{4}$$

where $|\llbracket \text{PS} \rrbracket \_(\text{after\_stego})|$ is the length of the protein sequence to be the cover and $|\llbracket \text{PS} \rrbracket \_(\text{before\_stego})|$ is the length of the protein sequence after the hiding process.

According to the first definition, the payload unit for biosequences such as DNA, RNA, and proteins is the number of bits per nucleotide (bpn). The suggested steganography approach takes three bytes to cover each byte of the encrypted data, S. Thus, if S is N bytes long, then (3 × N) protein bases really contain the bytes of S. So, Equation (5) will be used to determine the payload:

$$\text{Payload} = \frac{(\text{N} \times 8 \text{ bit})}{3 \times \text{N nucleotides}} = \frac{8}{3} = 2.666 \text{ bpn} \tag{5}$$

Table 4 shows the payload of some previous algorithms and the SMUPC algorithm; all algorithms are DNA dependent except the state-of-the-art algorithm presented in [17], which used protein motifs to hide encrypted data.

- The cracking probability (CP)

Assessing the probability to ensure a minimum cracking probability is crucial for establishing a secure steganography technique. This analysis involves evaluating the likelihood of an attacker successfully breaking the hidden information within the carrier. The steganography technique can be considered more secure by ensuring a minimum cracking probability, as it becomes increasingly challenging for unauthorized individuals to detect or extract hidden data.

The SMUPC algorithm generates a cover protein sequence for each session; therefore, there are unlimited possible sequences. But if we suppose that the range of generated sequences is included in the known proteins in the NCBI archive and that there are currently 1,098,741,385 sequences of proteins accessible online in the NCBI archive, the likelihood of correctly predicting a reference protein chain as shown in Equation (6) is:

$$\text{CP} = \frac{1}{\text{Number of Protein Sequences in NCB I}} = \frac{1}{1,098,741,385} \tag{6}$$

Additionally, and according to Table 1, the likelihood that each part of the binary secret message will be guessed as shown in Equation (7):

$$\frac{1}{2^{2 \text{ bits}}} \times \frac{1}{2^{3 \text{ bits}}} \times \frac{1}{2^{3 \text{ bits}}} = \frac{1}{256} \tag{7}$$

since a secret message's binary byte is segmented into three parts: one 2-bit part of the first segment and two 3-bit parts of the second and third segments.

- Blindness

The SMUPC algorithm is a blindness technique that has the ability to operate without transmitting the original protein chain to the recipient. The level of security is increased by decreasing the amount of data required to be sent to the recipient.

- Enhancing data security (encrypting)

Encrypting the confidential data before embedding it within a protein sequence offers greater protection compared to directly embedding the original data.

- Functionality conservation

The SMUPC algorithm generates a unique cover protein sequence for every session, meaning there is no specific functionality to preserve in the generated sequences.

- The key

The keys play a crucial role in the data-hiding process as they enhance the steganography system's security against attacks. In this algorithm, the information is encrypted before being hidden within the protein chain. The encryption process relies on a key randomly selected from the protein bases. Moreover, two keys are used to specify the random addresses of the bases where the hiding will take place.

The first key determines the starting address for the hiding process in the protein chain. It hides both the cipher key and the length of the cipher key in random locations in the chain. The second key generates additional random addresses where the ciphertext will be hidden within the protein chain. These hiding keys remain anonymous to both the sender and the recipient, changing each time the steganography algorithm is employed, thereby enhancing the algorithm's strength.

- Capacity

The capacity of data-hiding techniques refers to the data amount that can be included in the carrier. It is crucial for a data-hiding technique to maintain a significant hiding capacity.

Table 4 shows the high hiding capacity and predominance of the SMUPC algorithm, with a difference of 1.436 bpn over the nearest algorithm. This difference is noteworthy even if the algorithm does not support preserving the function of the protein used as a cover for the hidden data because the algorithm generates this protein and no specific protein is adopted for this task.

Moreover, this technology incorporates the encryption of the plaintext using a cipher key, along with two additional keys dedicated to concealing the ciphertext within the protein chain. This approach significantly enhances the robustness and security of the technique. Additionally, the support for blindness, where the original protein sequence is not transmitted to the recipient, further elevates the level of security. Consequently, the probability of cracking the hidden information becomes exceptionally low, if not virtually non-existent.

Table 4. Comparison between the proposed algorithm and five DNA-based and one protein-based algorithm.

Approach	Type of Chain	Payload	Cracking Probability	Functionality Conservation	Blindness	Enhancing Data Security	Using Keys	Capacity
LSBase [28]	DNA	0.333	Middle	Yes	Yes	No	One key	Low
Abdullah [29]	DNA	0.327	Low	Yes	Yes	Yes	X	Low
Highly improved DNA based steganography techniques [30]	DNA	1.46	Low	No	No	Yes	Two keys	Middle
Data hiding using double DNA sequences techniques [30]	DNA	0.574	Middle	No	Yes	Yes	X	Middle
Insertion method [31]	DNA	0.58	Low	No	No	No	X	Low
Complementary method [31]	DNA	0.07	Middle	No	No	No	X	Low

Table 4. *Cont.*

Approach	Type of Chain	Payload	Cracking Probability	Functionality Conservation	Blindness	Enhancing Data Security	Using Keys	Capacity
Substitution method [31]	DNA	0.82	High	No	No	No	X	Middle
Stego-protein algorithm [17]	Protein	1.23 in average case	Very low	Yes	Yes	Yes	Two keys	High
Proposed Algorithm (SMUPC)	Protein	2.666	Very low	No	Yes	Yes	Three keys	High

5. Conclusions

In this paper, new cryptography and steganography algorithms have been proposed to create a secure environment for the data. This environment includes an algorithm that uses encryption techniques to convert data from plaintext to incomprehensible, distorted text. This was conducted by generating a cipher key from a protein chain whose length is twice of the plaintext length to be encrypted. Both the key and the plaintext are converted to the hexadecimal system. This key will be divided into two parts. The first part of the key is combined with the plaintext through the hexadecimal addition process, and the result is converted to a binary system. As for the second part of the key, segmentation operations will be performed on it. The single number will be divided into two numbers, between which a multiplication operation will take place, and the result will be converted to the binary system. Later, it will be subtracted from the previous operation result by combining the first part of the key with the explicit text and then passing the final result via the Not gate to obtain the cipher text. This will result in a cipher text that will be hidden within a protein chain that is generated in a manner appropriate to the steganography process. The principle of steganography in this research depends on several directions that influence the security of the proposed algorithm:

1. The environment in which the information will be hidden:

In the realm of information concealment, careful selection of the environment is paramount. This research embraces the protein chain as the optimal canvas for this purpose. The rich diversity of protein bases, constituting an expansive array, plays a pivotal role in minimizing the recurrence of discernible patterns. This strategic maneuver, in turn, substantially diminishes the chances of an adversary successfully deciphering the cipher text, bolstering the security of the hidden information.

2. The hiding mechanism:

Central to this endeavor is the method employed to facilitate the covert operations. Guiding these intricate maneuvers is an algorithm tailored to engage with binary data, a format known for its inherent simplicity. The algorithm's design partitions a single byte of data into three discrete segments of bits. These individual segments find their covert haven within the confines of a protein base. Remarkably, a mere trio of protein bases collaboratively conceals one byte of data. This calculated approach, harmonizing data and cover bytes, inherently ascribes a distinctive payload ratio that further underscores the ingenuity of this methodology.

3. Distribution of hiding locations within the protein chain:

An integral facet of the steganography algorithm employed in this paper revolves around identifying inconspicuous locations for information embedding. A pinnacle achievement in this regard is the introduction of unpredictable hiding sites, a feature of paramount significance. At the heart of this accomplishment lies the "fixed variable", a constant that holds the key to initiating the hidden process within the protein chain. Maintaining a static nature, this variable ingeniously adapts its value based on the cumulative length of prior concealed texts.

This elusive value remains shrouded in mystery for both sender and receiver, fostering an environment of unparalleled security.

This intrinsic property fortifies the bedrock of cryptography and steganography. Augmenting this approach, a mathematical formula comes into play, generating a series of random numbers. These numbers serve as a guide for concealing cipher text discreetly across the expanse of the protein chain. This innovative amalgamation of fixed variables and mathematical formulas renders the locations indiscernible, elevating the entire process's robustness and effectiveness.

The proposed cryptography algorithm sets itself apart by exhibiting exceptional performance in comparison to the latest cryptographic techniques. It excels in terms of entropy and correlation, which are essential measures of randomness and unpredictability. This heightened level of entropy ensures that the encrypted data remains highly secure and resistant to decryption attempts by unauthorized individuals.

Similarly, the proposed steganography algorithm surpasses its predecessors by achieving the highest payload capacity. This means that it can effectively embed a larger volume of data within the protein sequence. This enhanced data-hiding capacity is invaluable for concealing substantial amounts of information while maintaining the cover medium's integrity and appearance.

A standout feature of the steganography algorithm is its support for blindness. By operating without the need to transmit the original cover medium, it enhances security further. The secret keys used for hiding the information remain anonymous to both the sender and recipient, providing an additional layer of protection against potential attacks. This anonymity ensures that even if the steganography process is intercepted, the hidden data remains well guarded and difficult to decipher.

In conclusion, the combined strength of the proposed cryptography and steganography algorithms creates a formidable data security framework. The high entropy, superior performance, and extensive data-hiding capacity make these algorithms highly effective in safeguarding sensitive information against various threats, providing a robust solution for ensuring data confidentiality and integrity.

The sender and the recipient can agree on the encryption key through any key management mechanism, and they can use the encapsulation method, as the protein capsule contains the hidden ciphertext, the key, and the length of the hidden data because it is important for performing the discovery process calculations and updating the value of the "fixed variable".

Author Contributions: Conceptualization, N.A.M.; Methodology, N.A.M.; Software, N.A.M.; Formal analysis, H.K.K.; Investigation, H.K.K.; Resources, H.K.K.; Writing—original draft, N.A.M.; Writing—review & editing, N.A.M.; Supervision, H.K.K. Both authors contributed equally. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The protein sequences that can be used as cover data can be obtained from the NCBI GenBank website. (https://www.ncbi.nlm.nih.gov) (accessed on 11 February 2023). Additionally, all the proposed algorithms are presented in the manuscript for future research.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Naser, S.M. Cryptography: From ancient history to now, its applications and a new complete numerical model. *Int. J. Math. Stat. Stud.* **2021**, *9*, 11–30.
2. Kiruba, B.; Saravanan, V.; Vasanth, T.; Yogeshwar, B.K. OWASP Attack Prevention. In Proceedings of the 2022 3rd International Conference on Electronics and Sustainable Communication Systems (ICESC), Coimbatore, India, 17–19 August 2022; pp. 1671–1675. [CrossRef]
3. Ramya, T.; Ramya, G.; Raju, K.; Ravi, J.; Verma, D. An Efficient AES Algorithm for Cryptography Using VLSI. *ECS Trans.* **2022**, *107*, 5605.

4. Giuseppe, A.; Danilo, F.; David, N.; Daniele, V. Match Me if You Can: Matchmaking Encryption and Its Applications. *J. Cryptol.* **2021**, *34*, 16.
5. Mawla, N.A.; Khafaji, H.K. An Ultra Lightweight Cipher Algorithm For IoT Devices and Unmanned Aerial Vehicles. In Proceedings of the 2023 International Conference On Cyber Management And Engineering (CyMaEn), Bangkok, Thailand, 26–27 January 2023; pp. 240–244. [CrossRef]
6. Chi, H.-X.; Chang, C.-C.; Wang, X.; Lin, C.-C. Hiding and Extracting Important Information in Encrypted Images by Using the Transformation of All Possible Permutations and VQ Codebook. *Electronics* **2022**, *11*, 3475. [CrossRef]
7. Ali, A.; Abdelmotalib, A. A secure image steganography using LSB and double XOR operations. *IJCSNS Int. J. Comput. Sci. Netw. Secur.* **2020**, *20*, 139–144.
8. Manish, C.; Gaurav, K.S. Enhanced Image Steganography Technique for Hiding Multiple Images in an Image Using LSB Technique. *TEST Eng. Manag.* **2020**, *83*, 30561–30565.
9. Malathi, P.; Gireeshkumar, T. Relating the Embedding Efficiency of LSB Steganography Techniques in Spatial and Transform Domains. *Procedia Comput. Sci.* **2016**, *93*, 878–885. [CrossRef]
10. Na, D. DNA steganography: Hiding undetectable secret messages within the single nucleotide polymorphisms of a genome and detecting mutation-induced errors. *Microb. Cell Factories* **2020**, *19*, 128. [CrossRef] [PubMed]
11. Khalifa, A. A Blind DNA-Steganography Approach using Ciphering and Random Sequence Splicing. In Proceedings of the 2020 10th International Conference on Information Science and Technology (ICIST), Bath, London, Plymouth, UK, 9–15 September 2020; pp. 86–90. [CrossRef]
12. Dimopoulou, M.; Antonini, M.; Barbry, P.; Appuswamy, R. Image storage onto synthetic DNA. *Signal Process. Image Commun.* **2021**, *97*, 116331. [CrossRef]
13. Dejian, F.; Shuliang, S. A New Scheme for Image Steganography based on Hyperchaotic Map and DNA Sequence. *J. Inf. Hiding Multimed. Signal Process.* **2018**, *9*, 392–399.
14. Farahat, M.A.; Abdo, A.; Kassim, S.K. A Systematic Literature Review of DNA-Based Steganography Techniques: Research Trends, Data Sets, Methods, and Frameworks. In *Digital Transformation Technology, Lecture Notes in Networks and Systems*; Magdi, D.A., Helmy, Y.K., Mamdouh, M., Joshi, A., Eds.; Springer: Singapore, 2022; Volume 224. [CrossRef]
15. Animesh, H.; Soumya, G.; Sampad, J. A review on DNA-based cryptographic techniques. *Int. J. Netw. Secur.* **2018**, *20*, 1093–1104. [CrossRef]
16. Vantigaru, S.; Basavaraj, R. Two layer encryption schemes for symmetric algorithms using DNA Sequences. *Int. J. Math. Comput. Sci.* **2019**, *14*, 953–964.
17. Mawla, N.A.; Khafaji, H.K. Protein Motifs to Hide GA-Based Encrypted Data. *Sci. Program.* **2022**, *2022*, 1846788. [CrossRef]
18. Raghuvanshi, D.; Solanki, V.; Arora, N.; Hashmi, F. Computational of Bioinformatics. *Int. J. Trend Sci. Res. Dev.* **2022**, *4*, 128–131.
19. Abdullah, A.A.; Ali, S.H.; Mstafa, R.J.; Haji, V.M. Image steganography based on DNA sequence translation properties. *UKH J. Sci. Eng.* **2020**, *4*, 15–26. [CrossRef]
20. Khalifa, A. A Secure Steganographic Channel Using DNA Sequence Data and a Bio-Inspired XOR Cipher. *Information* **2021**, *12*, 253. [CrossRef]
21. Ho, B.; Seonwoo, M.; Hyun-Soo, C.; Sungroh, Y. DNA Privacy: Analyzing Malicious DNA Sequences Using Deep Neural Networks. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **2022**, *19*, 888–898.
22. Muhammad, M.H.; Sagheer, A.; Muhammad, A.K.; Ehab, M.M. A Novel and Efficient Multiple RGB Images Cipher Based on Chaotic System and Circular Shift Operations. *IEEE Access* **2020**, *8*, 146408–146427.
23. Banik, S.; Pandey, S.K.; Peyrin, T.; Sasaki, Y.; Sim, S.M.; Todo, Y. GIFT: A small present-Towards reaching the limit of lightweight encryption. In Proceedings of the Cryptographic Hardware and Embedded Systems (CHES), Taipei, Taiwan, 25–28 September 2017.
24. Shen, H.; Shan, X.; Xu, M.; Tian, Z. A New Chaotic Image Encryption Algorithm Based on Transversals in a Latin Square. *Entropy* **2022**, *24*, 1574. [CrossRef]
25. Xu, M.; Tian, Z. A novel image encryption algorithm based on self-orthogonal Latin squares. *Optik* **2018**, *171*, 891–903. [CrossRef]
26. Zhang, X.; Zhang, L. Multiple-image encryption algorithm based on chaos and gene fusion. *Multimed. Tools Appl.* **2022**, *81*, 20021–20042. [CrossRef]
27. Wang, T.; Ge, B.; Xia, C.; Dai, G. Multi-Image Encryption Algorithm Based on Cascaded Modulation Chaotic System and Block-Scrambling-Diffusion. *Entropy* **2022**, *24*, 1053. [CrossRef]
28. Khalifa, A.; Helmy, H.S. Hiding secret Information in DNA sequences using silent mutations. *Br. J. Math. Comput. Sci.* **2015**, *11*, 1. [CrossRef]
29. Abdullah, A.; Eesa, A.S.; Abdo, A.M. New Data Hiding Approach Based on Biological Functionality of DNA Sequence. *Sci. J. Univ. Zakho* **2019**, *7*, 184–189. [CrossRef]
30. Al-Harbi, O.A.; Alahmadi, W.E.; Aljahdali, A.O. Security analysis of DNA based steganography techniques. *SN Appl. Sci.* **2020**, *2*, 172. [CrossRef]
31. Shiu, H.; Ng, K.; Fang, J.; Lee, R.; Huang, C. Data hiding methods based upon DNA sequences. *Inf. Sci.* **2010**, *180*, 2196–2208. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.