



A Novel Approach for Data Security Inspired from Artificial Neural Network Architecture.

Rupsha Chakraborty

Independent Researcher

Kolkata, India

Abstract

The security of confidential information during data transmission is an important concern in communication networks. This paper presents an encryption technique based on a neural network architecture and a symmetric private key to improve the security of transmitted messages. The proposed method uses an encryption algorithm using neural network architecture that takes the input message and a private key to generate a ciphertext which has been transmitted throughout the networks. At the destination end, the decryption algorithm decrypts to reconstruct the original message. The proposed approach combines key encryption with neural network processing to provide a multi-level method for protecting confidential data during long-haul transmission.

Keywords: Neural Networks, Encryption, Cryptography, Cybersecurity, Ciphertext

Introduction

With the development of modern communications and the emergence of the Internet, a significant amount of information is transmitted through digital channels on a daily basis. The information can be of various types: personal, financial, business, or any other sensitive data, including confidential information. However, transmitting information through digital channels always carries the risk of this data being intercepted by unauthorized parties. This fact requires data to be protected from the moment of transfer to the point of receipt.

The most common way to ensure confidentiality is to apply encryption. The essence of this method is that plain text is encrypted using specific algorithms and a special key, which significantly complicates the interception and reading of information. Subsequently, the received ciphertext is decrypted back to the original message using a secret key. Symmetric encryption algorithms are characterized by the use of the same secret key for both encryption and decryption.

However, with the growing number of cyberattacks and threats, it becomes necessary to find additional ways to ensure confidentiality. One of the directions of research is neural networks, which can be applied to different areas of cybersecurity. These networks are capable of performing complex transformations, which can be used to implement additional levels of data protection. One of the ways to do this is to apply neural networks to encrypted information.

In this paper, a novel approach of data transmission using neural networks is considered. The method uses symmetric encryption heuristics and protects data with a private key. The data encryption process with the private key is shown, which is followed by the input of ciphertext into the neural network. After that, the data is broken down into fragments, which leaves the structure of the neural network. Subsequently, the fragments are collected and decrypted with the private key at the receiving port, after which they are converted back to the original message.

The purpose of this method is to improve the level of confidentiality, integrity by ensuring an additional level of data protection. Thus, the article explores a way to increase the security of the data transmission process using symmetric algorithms with neural networks.

Literature survey

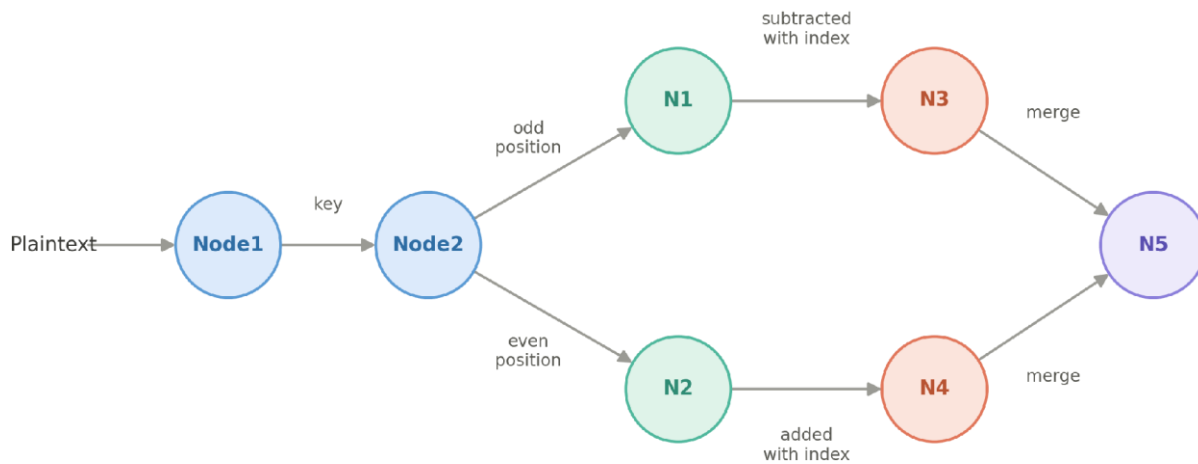
Mondal et al. [1] suggest an approach where plaintext and private key are converted into ASCII- and keyboard-position-based matrices, from which the cipher is generated by subtracting one from another and subjecting the result to a combination of genetic-algorithm-inspired crossover, mutation, and inversion operations, with an example plaintext given and shown to yield correct results after substitution.

Sadhukhan et al. [2] describe a "neural network" approach, where the plaintext is combined with a key array derived from it and passed through multiple stages of encryption using ASCII offsets and vowel/consonant reversal, resulting in a series of substituted cipher texts, one of which is demonstrated on a sample plaintext.

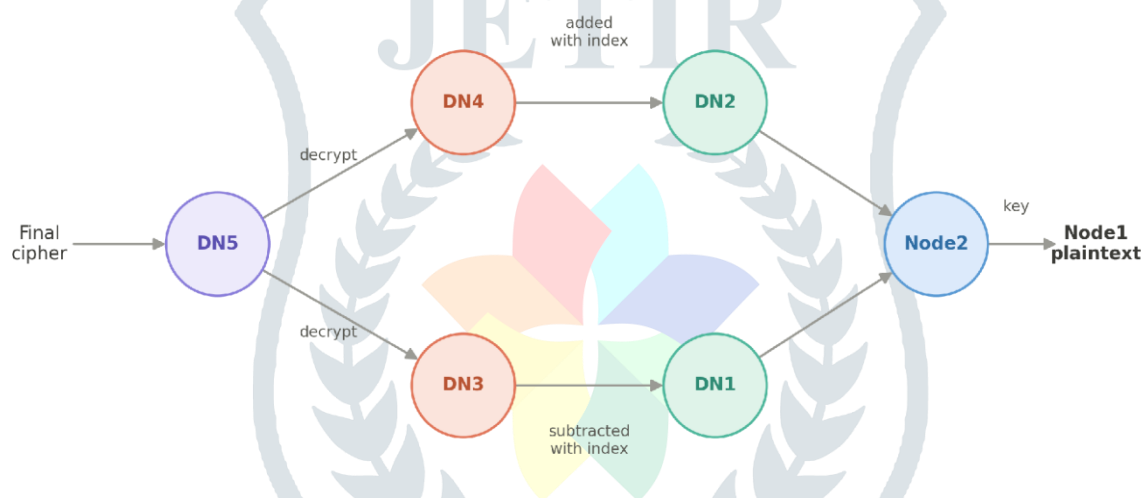
Paul and Bhattacharyya [3] explain a numeric symmetric-key algorithm in their paper, which employs a transposed matrix of the key, bitwise shifting, subtraction, and transformations using prime numbers in a multi-stage process arranged around a set of "hidden nodes" to form the cipher text, again demonstrated with a worked example.

System architecture with flow diagram

For encryption heuristics



For decryption heuristics



Algorithm

A: Proposed main Algorithm

Input: A plain text and a key is taken as input.

Process: Plain text is converted into cipher text using symmetric key and transmitted over the network.

Output: Encoded cipher text is decoded to get back original text at the receiving end of network.

Step 1: A plain text is taken as input for encryption.

Step 2: Call encryption algorithm using neural network architecture// Call C:proposed encryption algorithm

Step 3: Call secret key generation algorithm // Call B: proposed key generation algorithm

Step 4: Call decryption algorithm using neural network architecture // Call D: proposed decryption algorithm.

Step 5: Stop

B: Proposed key generation Algorithm

Input: Plain text.

Process: Key is generated by reversing the plain text.

Step 1: Take the plain text as input.

Step 2: Reverse the order of characters in the plain text.

Step 3: Store the reversed text as the secret key.

Step 4: Stop.

C: Proposed encryption algorithm

Input: Plain text and secret key (reversed plain text).

Output: Final cipher text.

Step 1: Take the plain text as input at Node1.

Step 2: Reverse the plain text to generate the key, and merge the plain text with the key alternately at Node2 to produce the intermediate cipher text

Step 3: Take the ASCII value of each character of the intermediate cipher text.

Step 4: Split the array by position — send the odd-positioned elements to N1 and the even-positioned elements to N2.

Step 5: Send N1 to N3, and subtract a positional index value from each element of N3.

Step 6: Send N2 to N4, and add a positional index value to each element of N4.

Step 7: Merge N3 and N4 alternately to obtain N5.

Step 8: Convert the values at N5 back into their corresponding characters to get the final cipher text.

Step 9: Stop.

D: Proposed decryption algorithm

Input: Final cipher text and secret key.

Output: Original plain text.

Step 1: Take the final cipher text as input at DN5.

Step 2: Take the ASCII value of each character and store it in an array.

Step 3: Split the array by position — odd positions to DN3, even positions to DN4.

Step 4: At DN3, add back the positional index to reverse the subtraction from encryption.

Step 5: At DN4, subtract back the positional index to reverse the addition from encryption.

Step 6: Send DN3 to DN1 and DN4 to DN2, and convert the values back into characters to recover the intermediate cipher text.

Step 7: Merge DN1 and DN2 alternately at Node1 to reconstruct the intermediate cipher text.

Step 8: Remove the interleaved key characters at Node1 to recover the original plain text.

Step 9: Stop.

Example Illustration**A. Example Illustrating Encryption Algorithm**

Plain text: HELLO WORLD

Step 1: Key Generation

The plain text "HELLO WORLD" is reversed to generate the key.

Key = DLROW OLLEH

Step 2: Intermediate Cipher Text Generation (Node 2)

Plain text array:

H E L L O _ W O R L D

Key array:

D L R O W _ O L L E H

Merged intermediate cipher text array:

H D E L L R L O O W _ _ W O O L R L L E D H

Step 3: ASCII Conversion

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
72	68	69	76	76	82	76	79	79	87	32	32	87	79	79	76	82	76	76	69	68	72

Step 4: Splitting by Position

N1 (odd positions):

68 76 82 79 87 32 79 76 76 69 72

N2 (even positions):

72 69 76 76 79 32 87 79 82 76 68

Step 5: N1 to N3 (Subtract Positional Index)

67 73 77 72 78 21 66 61 59 50 51

Step 6: N2 to N4 (Add Positional Index)

72 71 80 82 87 42 99 93 98 94 88

Step 7: Merging N3 and N4 (N5)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
72	67	71	73	80	77	82	72	87	78	42	21	99	66	93	61	98	59	94	50	88	51

Step 8: Character Conversion

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
H	C	G	I	P	M	R	H	W	N	*	NAK	c	B	J	=	b	;	^	2	X	3

Position 11 yields ASCII value 21, which corresponds to the non-printable ASCII control character **NAK (Negative Acknowledge)**, denoted by the Unicode control-picture symbol □.

Final cipher text: **HCGIPMRHWN□cB]=b;^2X3 is generated at source end .**

B. Example Illustrating Decryption Algorithm

The cipher text array: **HCGIPMRHWN□cB]=b;^2X3** is taken as input at destination end for decryption.

Step 1: ASCII Conversion

Gives back the array obtained in Step 7 of encryption.

Step 2: Splitting by Position

DN3 (odd positions):

67 73 77 72 78 21 66 61 59 50 51

DN4 (even positions):

72 71 80 82 87 42 99 93 98 94 88

Step 3: DN3 (Add Back Positional Index)

68 76 82 79 87 32 79 76 76 69 72

Step 4: DN4 (Subtract Back Positional Index)

72 69 76 76 79 32 87 79 82 76 68

Step 5: Merging at Node 2

DN3 values go to DN1 and DN4 values go to DN2. Both are converted back to characters and merged alternately, in original position order, to recover the intermediate cipher text array:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
H	D	E	L	L	R	L	O	O	W	_	_	W	O	O	L	R	L	L	E	D	H

This matches the intermediate cipher text array obtained in Step 2 of the encryption process, confirming the correctness of the decryption process.

Step 6: Removing the Key at Node 1

The plain text characters occupy the even positions of the array (0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20):

H E L L O _ W O R L D

Recovered plain text: HELLO WORLD at destination end.

Sample Output

```

Output

Enter the plain text to encrypt: hello world
Plain text      : hello world
Key (reversed)  : dlrow olleh
Final cipher text : 'hcgipmrhwn*\x15\x83b}]\x82[~RxS'
Recovered text   : hello world
Match original?  : True

=== Code Execution Successful ===

```

Conclusion

In this paper, we discussed a procedure for the secure transmission of plain text on a network through the process of encryption based on some concepts inspired from the Neural Network architecture. The plain text message is first processed together with a key to generate an intermediate cipher, which subsequently goes through several stages based on node processing, similar to the hidden layer in an Artificial Neural Network, wherein the data is divided, transformed through positional index operations, and recombined to generate the final cipher. On decryption, the same procedure is reversed to successfully decrypt the cipher into its original message. Through the combination of symmetric key cryptography and this node-based procedure, the proposed algorithm provides one additional layer of security to the conventional substitution cipher in order to ensure the integrity, availability, and confidentiality of the message in transmission.

Reference

- [1] S. Mondal, T. K. Mollah, A. Samanta, and S. Paul, "A Survey on Network Security Using Genetic Algorithm," *International Journal of Innovative Research in Science, Engineering and Technology (IJIRSET)*, vol. 5, no. 1, pp. 87–93, January 2016, doi: 10.15680/IJIRSET.2015.0501011.
- [2] P. Sadhukhan, A. Karmokar, A. Ganguly, and S. Paul, "Design And Implementation of Network Security Using Neural Network," *International Journal of Scientific Research in Science and Technology (IJSRST)*, vol. 11, no. 2, pp. 22–27, March-April 2024, doi: 10.32628/IJSRST52411186.
- [3] S. Paul and S. Bhattacharyya, "Implementation of Network Security Using Neural Network Architecture," *International Journal of Computer Science and Mobile Computing (IJCSMC)*, vol. 4, no. 6, pp. 815–821, June 2015.
- [4] C. Wang and Y. Zhang, "A novel image encryption algorithm with deep neural network," *Signal Processing*, vol. 196, p. 108536, 2022.
- [5] D. N. Mudaliar, N. Modi, and J. Dharwa, "A novel approach to solve network security, cryptography problems using genetic algorithm," in *Proc. Int. Conf. Computing Science, Communication and Security*, 2024, pp. 282–293.
- [6] N. Chaudhary, T. B. Shahi, and A. Neupane, "Secure image encryption using chaotic, hybrid chaotic and block cipher approach," *Journal of Imaging*, vol. 8, no. 6, p. 167, 2022.