ISSN:0975-9646
Paras Negi et al, / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 13 (2) , 2022, 42-47

Review of Tree Traversal Techniques to Conceal the Data

Paras negi [#1] , Manoj kumar bisht [#2]

[1,2] *Department Of Computer Science, S.S.J. University, Almora, Uttarakhand, India*

#1 parasnet072@gmail.com

#2 manojssj2020@gmail.com

Abstract

Data is defined as information, facts, or any set of characters. Data concealment or encryption is the process of inscribing or encoding the data in such a way that only a licensed or authorized person can read it. As much as technology is going to increase, the number of security attacks is also increasing, and we also need to secure our data. There is no guarantee that whatever data we have sent may reach the decoder correctly or not. At present, there are so many algorithms for data encryption, but we still need a strong method for encrypting our data. This review article explains the previous research work done by the researcher and identifies the research gaps. This review article also explains a new way to encrypt data with the help of an almost complete binary tree. This article is all about: how we encrypt data? How Binary tree property works? Can we use *Full binary trees, almost complete binary trees, Perfect binary trees* for traversal?

Keywords: Almost complete binary tree, level order traversal, encryption, Pre/In/Post order.

Abbreviation: **FBT** Full binary tree, **ACBT** Almost complete binary tree, **PBT** perfect binary tree.

I INTRODUCTION

Data cryptography is a method of hiding data in a way to provide security. A cryptosystem is a set of secure keys to convert plain text to encrypted text and convert it back into plain text. At this time, there are so many encryption and decryption algorithms, but still, a new algorithm is needed for more strong security purposes. In this article, I propose a new idea for encryption with the help of an almost complete binary tree and also explain why we can't use full binary tree (FBT) and perfect binary tree (CBT) all the time. Our main technique for data hiding is tree traversal. Plain text is the original message, and cipher text is the coded message. We use the ***Almost complete binary tree*** property and then also use three main types of traversal techniques: in-order, pre-order, and post-order. A tree is a non-linear data structure where the nodes are connected by edges **(Fig 1)**. Now let us discuss the area that is required for our mechanism. In an almost complete binary tree (ACBT), we insert nodes level by level and from left to right. In ACBT, we can go to the next level only if the previous level is full. For example, our country's data structure is also a tree-type data structure. Where India is the root or parent node and states place the role of the child of India, further districts place the child of states and so on.

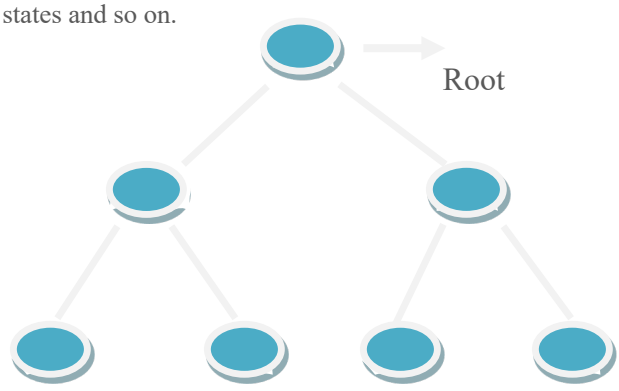


Fig 1: Almost complete binary tree.

II REVIEW OF LITERATURE

Several works in data encryption techniques have been completed with the assistance of various algorithms. Here we take a review of some research papers and discuss how they used algorithms to find encrypted code.

1. Sivakumar et al. [1] provide an encryption algorithm in two steps. The first step is substitution methods of cryptography, and the second step is transposition methods of cryptography. A binary tree is constructed at the second level. In which nodes are arranged from left to right. After two levels of encryption, the receiver receives the encrypted code.

2. Eswara Chandra Vidya Sagar [2] used a tree traversal technique to encrypt the data. First they decide the node size and then make a tree. After building the tree, they use either pre or post order and in order for encryption. With the help of these order receivers, they build a tree and then apply level-order traversal to read the data.

3. Natrajan et al. stated that the objective of this research is to develop multi-level encryption software that can be used to encrypt files, including text, images, and any other files on the secondary storage devices.

4. Mohammad Awad Al-Hazaimeh, In this paper, the researcher used the Shannon model of secret communication to define how plain text is converted into cipher text and then convert cipher text back to plain text. They compare the most popular encryption algorithms, Advanced Encryption Standard (AES), and Public Key Infrastructure (PKI), and analyse them.

5. Sukhraliya et al. [5] used ASCII values with a substitution array approach for encryption and decryption. They used the modulus and remainder approach to get a random number and, with the help of a program, encrypt and decrypt the message.

III BINARY TREE & TREE TRAVERSAL

Full Binary Tree (FBT): In Full Binary property, they have either 0 or 2 children, except for the leaf node.

Almost Complete Binary Tree: In ACBT, we arrange nodes level by level and from left to right. In ACBT, we can go to the next level only if the previous level is full.

In a Perfect Binary Tree (PBT), at each level, the internal node should always be full. Tree traversal is a technique to visit a node.

There are three main types of traversal techniques: In-order, Pre-order, Post-order **(Fig 2)**.

IN-order to visit nodes in a way of **"LEFT-ROOT-RIGHT"**

Pre-order visit nodes in a way of **"ROOT -LEFT -RIGHT"**

Post-order visit nodes in a way of **"LEFT- RIGHT- ROOT"**

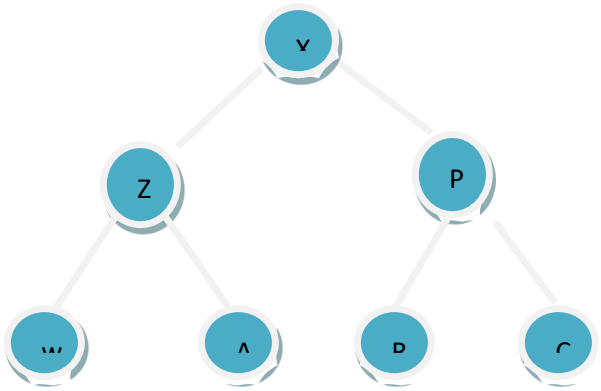


Fig 2: Tree traversal

Pre-order: X Z W A P R C

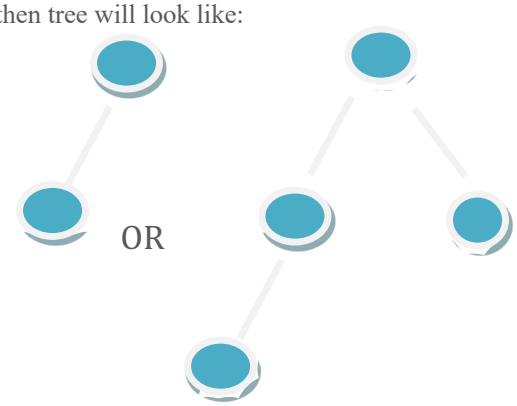
Post-order: W A Z R C P X

In-order: W Z A X R P C

IV NEW APPROACH

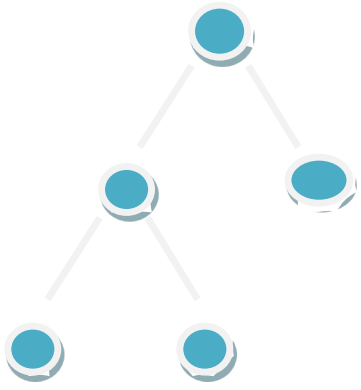
After reviewing the previous work, I found that the researcher used the tree traversal technique, but they did not specify which type of tree we should use. Some researchers used the term "binary tree" but they did not define which type of binary tree they used. Either it is a full binary tree or a perfect binary tree or something else. There are some cases when FBT and also PBT do not fully fulfill the condition.

- Full binary tree property define that they have either 0 or 2 children, but when the data has 2 or 4 or etc nodes then tree will look like:



For this type of above graph property that does not correspond to FBT property, So we can't use FBT.

- The perfect binary tree property states that internal nodes should always be completely filled, which means that each internal node has two children, but if the data contains like five or more nodes, the tree will look like this.



For this type of above graph property that does not correspond to PBT property, So we can't use PBT.

- Tree traversal for data encryption and defining the node in the correct way, we should use **an almost complete binary tree (ACBT)**, because in ACBT we derive the node in left-to-right order and move to the next level only if the previous level is filled completely. No matter what size of plain text it is, it always follows the ACBT property.

V WORKING METHODOLOGY

The proposed method uses two levels of encryption.

The first level It is based on substitution, using the numbers 0–25 to represent the English alphabet values. Suppose a plain text **M** and an increment of 1, then encryption is performed as follows: **MOD 26 E(M, 1) = (M+1) MOD 26**Decryption is performed as follows if the cipher value is C: D(C,1) = (C-1)MOD 26.

The second level uses binary tree traversal. An ACBT is constructed at the second level of encryption. Nodes are built from left to right using the in-order and either pre-order or post-order on the binary tree. Send the outcomes to the receiver, then the receiver applies the decryption mechanism.

A. Illustration of Idea

With the help of an example, we will illustrate our proposed method in this section.

Plain text: ***THIS IS NEW IDEA OF HIDING DATA***

First level encryption: E(M,1) =(M+1) MOD 26.

E(T,1) =(T+1) MOD 26 =(20+1) MOD 26=21= U

E(H,1) =(H+1)MOD 26 =(8+1) MOD 26=9=I

E(I,1) =(I+1) MOD 26 =(9+1) MOD 26=10= J

E(S,1) =(S+1)MOD 26 =(19+1) MOD 26=20=T

After applying this first level encryption, we get a converted text:

UIJT JT OFX JEFB PG IJEJOH EBUB

Second level encryption: construct an **Almost complete**

binary tree from the above-encrypted data **(Fig 3)**

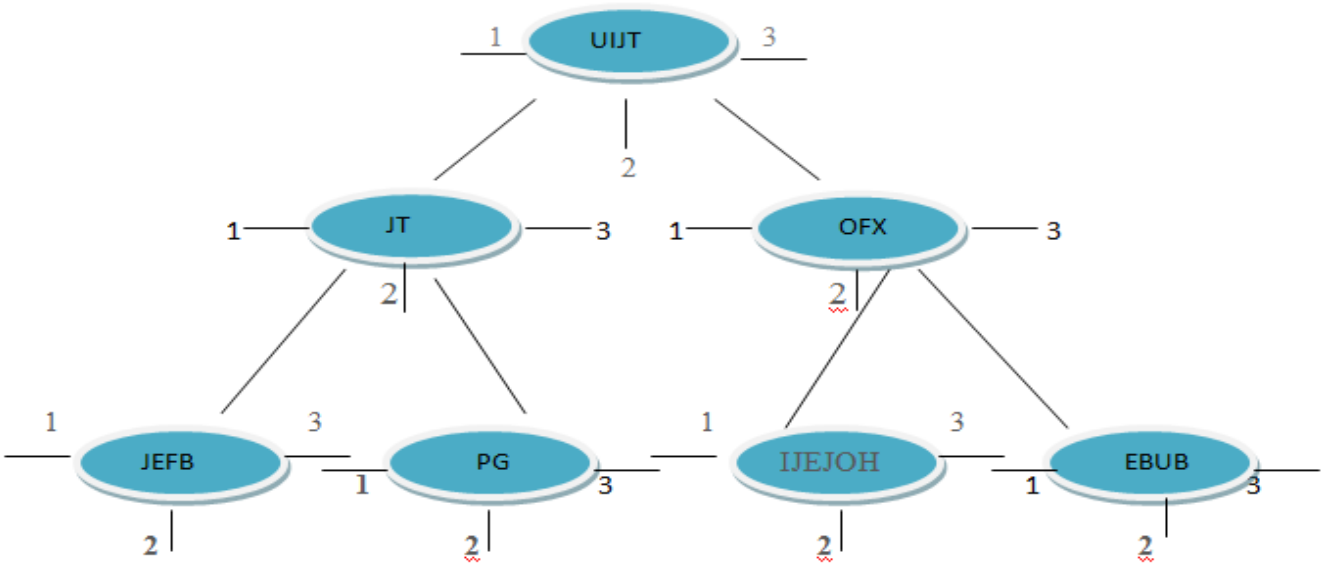


Fig 3: ACBT

Pre-order: UIJT JT JEFB PG OFX IJEJOH EBUB

In-order: JEFB JT PG UIJT IJEJOH OFX EBUB

Post-order: JEFB PG JT IJEJOH EBUB OFX UIJT

Second level decryption: After receiving the data, construct a binary tree with the help of in-order and post-order and then read data in Breadth first search (BFS) format.

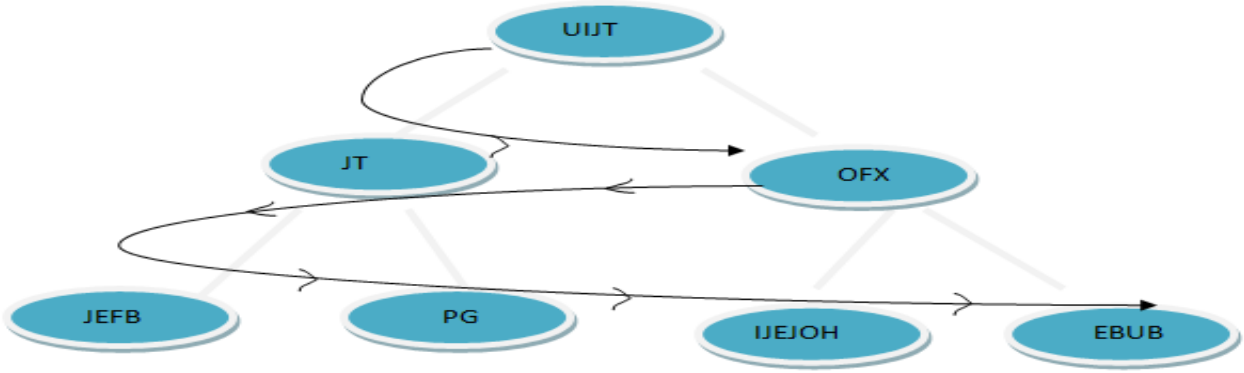


Fig: 4 Breadth First Search on the derived tree

After constructing BFS, we received data in the form of

UIJT JT OFX JEFB PG IJEJOH EBU

First level decryption: now decrypt the above data D(C,1)=(C-1)MOD 26

D(U,1)=(U-1)MOD 26=(21-1)MOD 26=20=T

D(I,1)=(I-1)MOD 26=(9-1)MOD 26=8=H

D(J,1)=(J-1)MOD 26=(10-1)MOD 26=9=I

D(T,1)=(T-1)MOD 26=(20-1)MOD 26=19=S

After completing first level decryption we get our actual data which is:

THIS IS NEW IDEA OF HIDING DATA

B. Encryption Algorithm

1) First Level Encryption:

Input: plain text Output: Intermediate ciphertext

Step 1: Start
Step 2: Convert each character of a word into its corresponding Alphabetic value (0-25)
Step 3: Adding value 1 and find out the mod E(M,1) =(M+1) MOD 26
Step 4: Convert the calculated Alphabetic values into the corresponding characters to get the intermediate cipher
Step 5: Stop

2) Second Level Encryption:

Input: Intermediate ciphertext Output: final ciphertext (order traversal)

Step 1: Start
Step2: Construct an Almost complete binary tree (level by level) from the intermediate ciphertext.
Step 3: Traverse the tree and find In-order and either Pre-order or Post-order.
Step 4: Store the final cipher text.
Step 5: Stop

Decryption Algorithm

3) Second Level Decryption:

Input: Final Ciphertext Output: Intermediate Ciphertext

Step 1: Start
Step 2: Construct a complete binary tree with the help of In-order and either Pre-order or Post-order
Step 3: Perform level order or BFS traversal of the final Binary tree to get the intermediate cipher.
Step 4: Store the intermediate ciphertext.
Step 5: Stop

4) First level Decryption:

Input: Intermediate Ciphertext Output: Plain text

Step 1: Start
Step 2: Convert each character of a word of intermediate text into its corresponding Alphabetic value (0-25)
Step 3: subtract value 1 and find out the mod D(C,1)=(C-1)MOD 26
Step 4: Convert the calculated Alphabetic values into the corresponding characters to get the decrypted cipher.
Step 5: Store the plain text.
Step 6: Stop

6 NOTABLE PROPERTIES

For tree traversal technique Almost complete binary tree (ACBT) full fill all the property. No matter how many nodes we have in our plain text, we can use ACBT while CBT and PBT cannot. The proposed mechanism uses two levels of encryption, which makes it vigorous and secure against attacks. There is no overhead in key generation, key sharing, and keeping the key securely. Also, this idea agrees with memory wastage because we are sending the data in two different traversals and also use first-level encryption. For future scope, with this mechanism, we can change the first-level encryption method and use some other encryption method. A method like the strong encryption algorithm provides next-level security and combats attacks.

7 Conclusion

Data concealment with the help of the first level encryption method and the main innovative mechanism Tree Traversal. This idea gives a new area of research to secure the data. Via the

almost complete binary tree (ACBT) property and tree traversal order, it is a new method to encrypt the data. The proposed method combats the frequency of attacks and provides the next level of security. This method is robust, secure, and reliable. According to us, this is one of the best uses of the tree traversal algorithm for securing data.

REFERENCES

[1] T. Sivakumar, K. Humshavarthini, M. Jayasree and M. Eswaran, "Data Encryption Using Binary Tree Treversal (DEBTT)", *International Journal of Advanced Technology in Engineering and Science*, vol. 5, no. 4, 2017.

[2] L. Eswara Chandra Vidya Sagar, "Data Security Using Tree Traversal", *Global Journal of Computer Science and Technology: C Software & Data Engineering*, vol. 15, no. 3, 2015.

[3] S. Natarajan, M. Ganesan and Krishnan Ganesan, "A Novel Approach for Data Security Enhancement Using Multi Level Encryption scheme", *International Journal of Computer Science and Information Technologies (IJCSIT)*, vol. 2, no. 1, 2011.

[4]. O. AwadAlHazaimeh, "A New Approach For Complex Encrypting And Decrypting Data", International *Journal of Computer Networks & Communications (IJCNC)*, vol. 5, no.2, 2013.

[5] V. Sukhraliya, S. Chaudhary and S. Solanki, "Encryption and Decryption Algorithm using ASCII values with substitution array Approach", *International Journal of Advanced Research in Computer and Communication Engineering,* vol. 2, no. 8, 2013.