International Journal of Progressive Sciences and Technologies (IJPSAT)
Vol. 2 No. 2 April. 2016, pp.76 -82

http://ijpsat.ijsht-journals.org

# A New Approach to Encryption using Huffman Coding

***Sanjali Gupta[1], Nikhil Shanker Mathur[2], and, Priyank Chauhan[3]***

[1,2,3]Computer Science Department,
VIT University,
Vellore, Tamil Nadu, India



**ABSTRACT**: In today's world, where Digital Communication is becoming an integral part of everyone's life, cyber security has become a challenge. There are various threats to the data being sent digitally. Changing the data into an unidentifiable form can protect it from attackers. Cryptography is about constructing protocols to prevent third party interventions. Various cryptographic techniques have been proposed to make the data transfer secure. Symmetric key algorithms are the widely used for encryption. The message to be transferred should be as small as possible so as to reduce space complexity that is why compression techniques find their applications in encryption. In this paper, a new approach to encrypt text files using a mix of Huffman coding, combination of symmetric key and public key cryptography and the binary operator XOR is proposed.

**KEYWORDS:** Symmetric keys, Asymmetric Keys, Cryptography, Decryption, XOR.

## 1. INTRODUCTION

Humans like to socialize and for this, they communicate with each other and want their communications to be secure. Secure communication is when two parties are exchanging information and they do not want any third party to get to know about the content. For this data needs to be sent in a form which is not susceptible to interception.

Compression can further help the communication by compressing the data being sent and thus making it easier to communicate as the space needed to store and transmit data decreases.

Cryptography is the science of using mathematics to encrypt and decrypt data. Cryptography enables you to store sensitive information or transmit it across insecure networks (like the Internet) so that it cannot be read by anyone except the intended recipient. [1]

## 2. CRYPTOGRAPHIC TECHNIQUES

### 2.1 Encryption and Decryption

Data that can be read and understood by anyone without any special knowledge about it is called **plaintext** or **clear text**. The method of disguising the plaintext in such a way as to hide the information is called **encryption**. Encrypting plaintext results in unreadable gibberish called **cipher text**. You use encryption to ensure that information is hidden from anyone for whom it is not intended, even those who can see the encrypted data. The process of reverting cipher text to its original plaintext is called **decryption**.

**Corresponding Author:** Priyank Chauhan (priyankchauhan014@gmail.com)

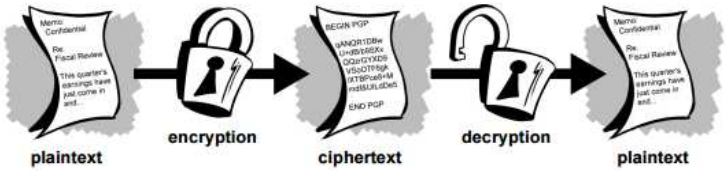



*Fig. 1. Encryption and Decryption [2]*

## 2.2 Conventional Cryptography

In conventional cryptography, also called secret-key or symmetric-key encryption, one key is used to both encrypt and decrypt the data. The famous Caesar's Cipher is an example of this technique. Only the person who knew the "shift by 3" rule [2] could understand the message. While sending the encrypted data, the key is shared through another secure channel so as to make it possible for the receiver to decrypt the cypher text into original plaintext. Which makes it a bit insecure.

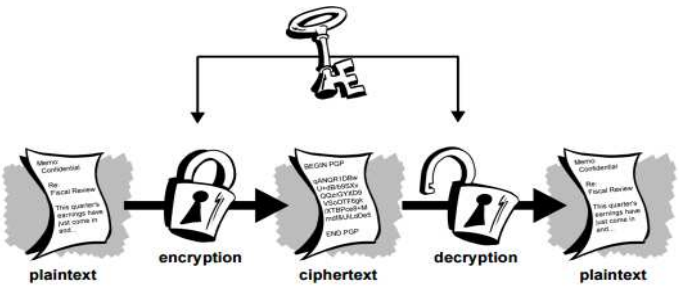



*Fig. 2. Conventional Cryptography [2]*

## 2.3 Public Key Cryptography

In this technique, there are two keys involved, one for encryption and other for decryption. The receiver already has a private key which is never used in any communications. Only the public key is sent along with the cipher text. No one without having the private key can decrypt the code, thus making the communication safer.

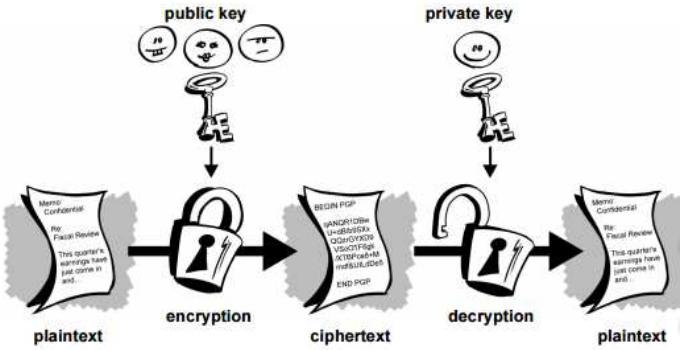



*Fig. 3. Public Key Cryptography [2]*

## 3. HUFFMAN CODING

Huffman coding is a lossless data compression technique used widely to compress images. [3] The main idea is to assign variable length codes to alphabets/symbols based upon there occurrence in the text. In this technique, the code generated is a prefix code[4], that is, if a symbol, say 'A', is assigned a code 0 then no other symbol can have a code starting with 0.

It is the most efficient and widely accepted compression technique. It makes use of trees to assign codes to various symbols depending upon there frequency.

### Algorithm:

1. Calculate the frequency of each symbol.
2. Take the two least frequent symbols and assign them to two leaf nodes. And assign the sum of the corresponding frequencies to the parent node.
3. Now select next two least frequency nodes from the rest of the nodes along with the newly created node and form another parent node,
4. Repeat step 3 till a complete binary tree is formed.
5. Starting from the root, assign '0' to the left child and '1' to the right child of every node till you reach the leaf nodes.
6. To assign the code to the symbol, trace the path from root to the corresponding node.

The run time complexity of Huffman for n characters is O (n log n). [3]

## 4. THE PROPOSED APPROACH

### 4.1 Encryption Algorithm

**Step 1.** Compress the given text using Huffman coding algorithm as explained above.

Here is an example to do that:
Let's say the data being sent is '**HELLO'.**

1. Calculating the frequencies of all the symbols:
   **H: 1; E: 1; L: 2; O: 1.**
2. Constructing the tree:

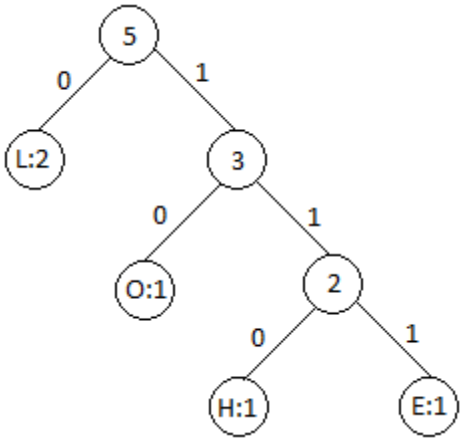



***Fig. 4. Huffman Tree, T***

**Code:**
H: 110
E: 111
L: 0
O: 10

Therefore, the compressed code becomes, say **H = 1101110010.**

**Step 2.** Take two keys 'A' and 'B'. 'A' is used as public key, not known to the receiver beforehand and 'B' (should be small) is used as the private key previously known by the receiver.

For our example, let's say A=1001 and B= 1000.

**Step 3.** Take the first 'n' bits of the compressed code where 'n' is the number of bits in 'A'.
Perform XOR operation on these bits with A and shift A by one bit. Repeat this till you reach the end of the code. The new code formed is E.

For our example, the operation will be as follows:

**1101**110010
**^1001**
0**1001**10010
**^1001**
00**0001**0010
**^1001**
001**0000**010
**^1001**
0011**0010**10
**^1001**
00111**0111**0
**^1001**
001111**1100**
**^1001**
0011110101

The encrypted code, E now becomes 0011110101.

**Step 4.** Multiply 'E' by 'B' to change E to E*B.

For our example:

0011110101
*1000
0011110101000

Thus the final encrypted data to be sent, E= 0011110101000.

While sending the data 'E', 'A' and the Huffman tree 'T' is sent to the receiver so as to enable him to decrypt the message. B is known to the receiver previously.

## 4.2 Decryption Algorithm

Receiver will now have E, A, the Huffman tree T sent by the sender along with B.

**Step 1.** Divide E by B.

For our example:
E' = 0011110101000 / 1000 = 0011110101.

**Step 2.** Starting from the last 'n' bits of the obtained code from last step, Apply XOR operation on E' and A; where 'n' is the number of bits in A. And keep repeating till you reach the first bit of E'.

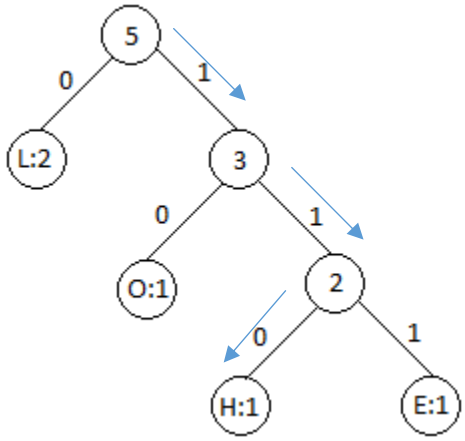
For our example:
001111**0101**
**^1001**
00111**1110**
**^1001**
0011**1011**10
**^1001**
0011**1001**010
**^1001**

001**1000**0010
^**1001**
0**0000**10010
^**1001**
**0100**110010
^**1001**
1101110010

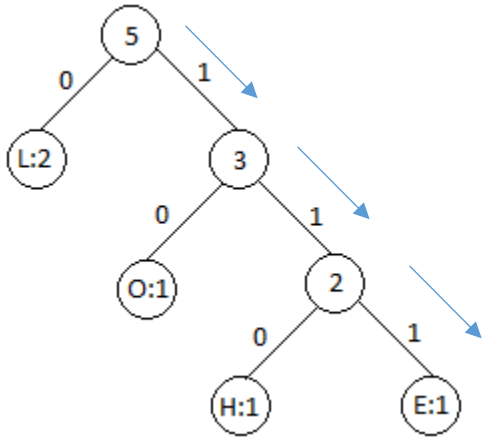
Thus, E" = 1101110010.

**Step 3.** Now uncompress the data obtained, for which trace the Huffman tree T, according to the bits of E'. As you reach a leaf node, note the symbol associated with it and restart from the root node again.

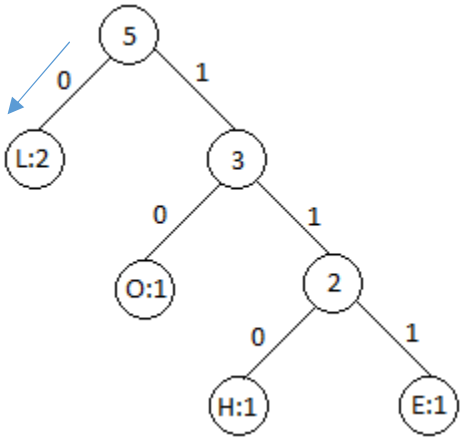
For our example:
Tracing from root node;



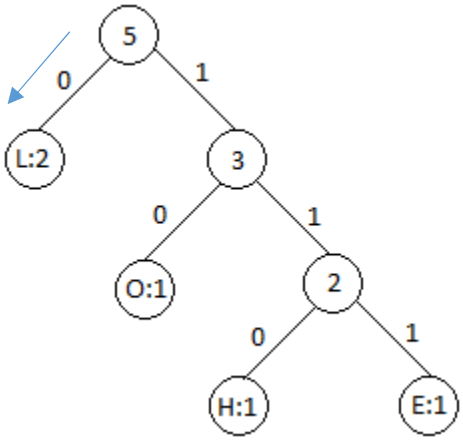

**<u>110</u>**1110010
**H**



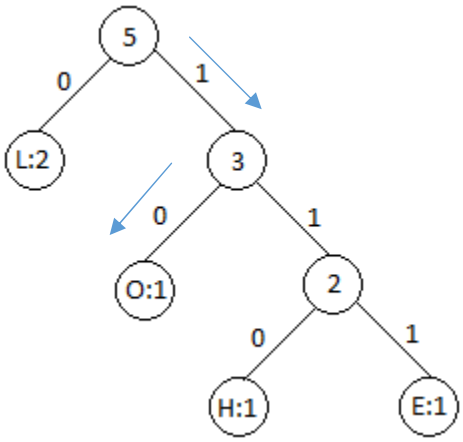

110**<u>111</u>**0010
**H E**




1101110010
**H E L**




1101110010
**H E LL**




1101110010
**H E LLO**

Thus, we decrypted the encrypted message.

## 5. BENEFITS OF PROPOSED ALGORITHM

- It is easy to understand and implement as simple binary operations are being used.
- Huffman compression makes encryption easy by reducing the length of data to be encrypted.
- It can be used by the people who have no preexisting security arrangements for exchanging data.
- Due to the use of a random private key it is computationally infeasible to decrypt the data.

## 6. CONCLUSION

Secure transfer [4] of data has become an issue due to large amount of data being exchanged through internet. The purpose of this paper was to develop a way to reduce the size of the data and encrypt it in such a way so that it can be transferred with utmost security. To achieve this, we used the famous Huffman coding technique for compression. Two keys are being used, one public and one private to encrypt the data so that it can't be interpreted by the outer world. The algorithm proposed here is quite simple and can be used for encryption of any kind of text data. Decryption can't be done computationally as the key selected is from a large domain.

## ACKNOWLEDGMENT

Prof Yamuna M, VIT University

## REFERENCES

[1] Nigam Sangwan, "Text Encryption with Huffman Compression", *International Journal of Computer Applications,* vol. 54, no.6, September 2012.

[2] *An Introduction to Cryptography,* 1999. [Online] Available: ftp://ftp.pgpi.org/pub/pgp/6.5/docs/english/IntroToCrypto.pdf

[3] *Huffman Coding*, 2011. [Online] Available: http://en.wikipedia.org/wiki/Huffman_coding

[4] *Huffman Coding*, 2014. [Online] Available: http://www.columbia.edu/~cs2035/courses/csor4231.F11/huff.pdf

[5] Charles P. Pfleeger, Shari Lawrence Pfleeger, *Security in Computing,* 3rd Ed. Prentice Hall Professional, 2003.