Text Encryption with Huffman Compression

Nigam Sangwan
Institute of Technology and
Management University,
Gurgaon, Haryana, India

ABSTRACT
Communication between a sender and receiver needs security. It can be done in any form, like plain text or binary data. Changing the information to some unidentifiable form, can save it from assailants. For example, plain text can be coded using schemes so that a stranger cannot apprehend it.

Cryptography is a subject or field which deals with the secret transmission of messages/ data between two parties. Cryptography is the practice and study of hiding information. It enables you to store delicate information and transmit it across insecure networks so that it cannot be read by anyone except the authorized recipient. Applications of cryptography include ATM cards, computer passwords, and electronic commerce etc.

Messages, which are being sent between two parties, should be of small size so that they occupy less space. Data compression involves encoding of information using fewer bits than the original representation. Compression algorithms reduce the redundancy in data representation to decrease the storage required for that data.

Symmetric key cryptography algorithms [6] are fast and mostly used type of encryption. Many types of data compression techniques do exist. Huffman Compression [5] is a lossless data compression technique and can be considered as the most efficient algorithm. In this paper, a combination of new Symmetric key algorithm and existing Huffman compression algorithm has been proposed. Proposed method works on text data. Algorithms have been provided in the paper itself.

General Terms
Algorithms, Compression, Security.

Keywords
Huffman Compression, Cryptography, Symmetric key.

1. INTRODUCTION
Communication is the exchange of thoughts, messages, or information, as by speech, visuals, signals, writing or behavior. When two entities are communicating and do not want a third party to listen in or know, they need to communicate in such a way that it doesn't get intercepted. Secure communication is needed and there exists many tools for this. With secure communication, if we use compressed form of the messages which are being sent, this will make an effective and powerful system. Compressed message will be smaller in size than the original and less bits will be needed to make it confidential.

Cryptography is the art of achieving security by encoding messages and Data Compression is reduction of the storage space required for data by changing its format. Main objective is to make data so secure that no one can decrypt it and with reduced size as communication between people is increasing day by day in one or other form.

2. CRYPTOGRAPHY AND HUFFMAN COMPRESSION

2.1 Cryptography
Cryptography has been derived from the Greek word kruptos, which means **hidden** [5]**.** Cryptography is the art of achieving security by encoding or transforming messages to make them non-readable. Cryptography is considered as a branch of both mathematics and computer science, and is affiliated with information theory, computer security and engineering also. Cryptography is the strongest tool for controlling against many kinds of security threats. It's one form or another has been practiced ever since man has communicated his thoughts in speech or writing.

2.1.1 Goals of Cryptography
Cryptography has three main goals [6]:

Authentication:
Authentication is the act of confirming the truth of an attribute of an entity i.e. the assurance that the communication entity is the one that it claims to be [8].
Trademark, digital signatures are examples which can be used for authentication purpose.

Data Confidentiality:
It is about the protection of data from unauthorized access. To protect the data, encryption of messages is being used.

Data Integrity:
The assurance that data received is exactly as sent by an authorized entity i.e. contains no modification, insertion, deletion. It refers to trustworthiness of information over its entire transmission. Data integrity can be achieved by following some rules, by placing check values.

2.1.2 Cryptosystem
- In cryptography, **encryption** is the process of transforming information using an algorithm to make it unreadable to anyone except those possessing secret information. The result of the process is encrypted information.
- The reverse process, i.e., to make the encrypted information readable, is known to as **Decryption**.
- The information which is transformed to meaningless form is known as **Plain Text**.
- The unreadable or meaningless form is known as **Cipher Text**.
- Algorithm which is used to transform plaintext into cipher text uses a device, called as, **Key [6]**.

- A system for Encryption and Decryption is called a cryptosystem. Shown in Fig 1.

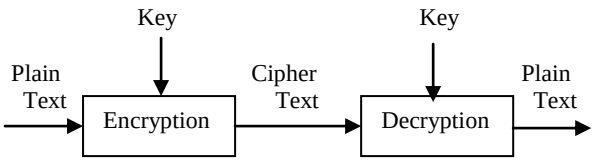

Fig 1: Cryptosystem

2.1.3 Types of Cryptography

There are two types of Cryptography: Symmetric key Cryptography and Asymmetric key Cryptography.

- **Symmetric Key Cryptography**:

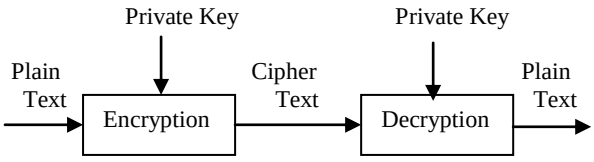

Fig 2: Symmetric Key Cryptography

It is also known as **Secret key cryptography (private key)**. **Symmetric-key algorithms** are the algorithms under cryptography that use the same cryptographic keys for both encryption of plaintext and decryption of cipher text. The keys represent a shared secret between two or more parties that can be used to maintain a private information link. For example: AES, DES.

- **Asymmetric Key Cryptography**:

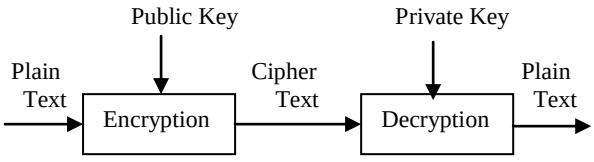

Fig 3: Asymmetric Key Cryptography

Also known as Public key Cryptography. **Public-key cryptography** refers to a cryptographic system requiring two separate keys, one to encrypt the plaintext, and one to decrypt the cipher text. One of these keys is published or public and the other is kept private [6]. For example: RSA.

2.2 Huffman Compression

Huffman coding is an entropy encoding algorithm used for lossless data compression. It was developed by David A. Huffman [5]. The Huffman encoding algorithm is an optimal compression algorithm when the frequency of individual letters is used to compress the data. The codes generated using this technique or procedures are called Huffman codes [4]. Huffman coding uses the method of prefix code or property. The idea is that the encoding for any one character isn't a prefix for any other character. For instance, if A is encoded with 0, then no other character will be encoded with a zero at the front.

Huffman has come out as the most efficient compression technique. A comparison study shows [2],

Table 1. Huffman Compression for a file of size 1272Bits

Compressed Size	Compression Ratio
852	66.98

Huffman can compress a file to a great extent. For example, we have a text 'Me'. Its present size is 16bits. On using Huffman, it can be compressed to a size of 2bits, which is approximately 8 times less than the original.

Two families of Huffman Encoding have been proposed [9]:

- **Static Huffman Algorithms:**

Static Huffman Algorithms calculate the frequencies first and then generate a common tree for both the compression and decompression processes.

- **Adaptive Huffman Algorithms:**

The Adaptive Huffman algorithms develop the tree while calculating the frequencies and there will be two trees in both the processes.

3. NEW SYMMETRIC KEY ALGORITHM WITH HUFFMAN COMPRESSION

Both techniques will be applied on Text based data. In first part, data is compressed using existing Huffman coding or compression algorithm and in second part, new proposed symmetric key cryptographic algorithm has been applied to make it secure.

3.1 Part 1: Huffman Text Compression

Step 1: Take input in form of Text based data.

Step 2: Compress the data using Huffman Compression Technique or algorithm.

Huffman (C)

```
1   n ←|C|
2   Q ← C
3   for i ← 1 to n-1
4       do allocate a new node z
5           left[z] ← x ← EXTRACT-MIN(Q)
6           right[z] ← y ← EXTRACT-MIN(Q)
7           f[z] ← f[x] + f[y]
8           INSERT(Q,z)
9   return EXTRACT-MIN(Q)
```

Where,

- C is a set of n characters.

- Each character c ϵ C (belongs to) is an object with a defined frequency f[c].
- Q is a min-priority queue which is used to identify the two least frequent objects to merge together. The result of the merger of two objects is a new object whose frequency is the sum of the frequencies of the two objects that were merged.
- Run time complexity of Huffman for n characters is O (n log n).

Step 3: Huffman Coded text, say 'H', is generated after compression.

3.2 Part 2: New Symmetric Key Algorithm

3.2.1 Encryption Algorithm

Step 1: Add 1 to H (Huffman Coded Text). Perform Binary Addition.

Step 2: Reverse the digits obtained from the addition (H+1), denote as $R_{(H+1)}$.

Step 3: Take First Secret Key, say A.
'A' should not be more than the length of $R_{(H+1)}$.

Step 4: Binary Addition of A to $R_{(H+1)}$, i.e. ($R_{(H+1)}$ + A).

Step 5: Obtain Two's Complement of ($R_{(H+1)}$ + A). Denote as $T_{(R\,(H+1)\,+\,A)}$.

Step 6: Take Second Secret Key, say B.
'B' should be small and not more than the length of $T_{(R\,(H+1)\,+\,A)}$.

Step 7: Divide $T_{(R\,(H+1)\,+\,A)}$ by B. Denote the Quotient as 'E', Encryption.

Step 8: If remainder is a non zero number then, receiver will receive remainder Rem, two Secret Keys A, B and Encrypted Text E. If Rem is a zero, then he/she will receive E, A, B.

Encryption E = $(T_{(R\,(H+1)\,+\,A)})$ / B

3.2.2 Example (Encryption)

Let's take an example. Text: "Algo".
ASCII Size of text: 32bits.

Part 1: Applying Huffman Compression on text.

Table 2: Huffman Compression on text.

Chars	ASCII	Frequency	Huffman	ASCII
'A'	65	1	00	01000001
'g'	103	1	01	01100111
'l'	108	1	11	01101100
'o'	111	1	10	01101111

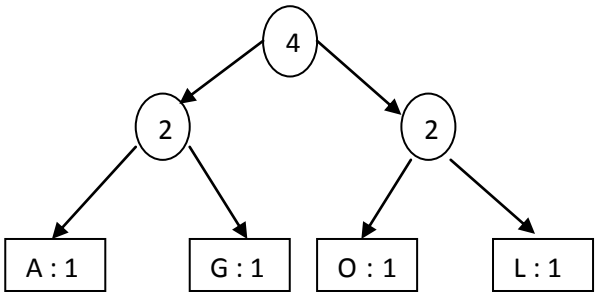

Fig 4. Binary tree formed from Compression

Huffman Coded text: 00110110.

Size after compression: 8bits.

Part 2: New Symmetric Key Cryptography (Encryption)

Step 1: Add 1 to H = 00110110

0	0	1	1	0	1	1	0
						+	1
0	0	1	1	0	1	1	1

Step2: Reversing it $R_{(H+1)}$.

$R_{(H+1)}$:

1	1	1	0	1	1	0	0

Step 3: Let's assume First Secret Key A: 1101101.

Step 4: Add A to $R_{(H+1)}$.

	1	1	1	0	1	1	0	0
+		1	1	0	1	1	0	1
1	0	1	0	1	1	0	0	1

Step 5: Find Two's complement of ($R_{(H+1)}$ + A).

T:

0	1	0	1	0	0	1	1	1

Step 6: Take Second Secret Key B. It should be small.

Let's assume B : 10010

Step 7: Divide T by B.

0	1	0	1	0	0	1	1	1
/				1	0	0	1	0

Now Quotient or Encrypted Text E:

1	0	0	1

and Remainder Rem:

1	0	1

Encrypted Text and two secret keys will be with Receiver for decryption.

3.2.3 Decryption Algorithm

Receiver will have the Encrypted text and secret keys with them.

Step 1: Take the encrypted text E and two Secret Keys A and B and Rem (if available).

Step 2: Now Multiply E with Secret Key B. Binary Multiplication should be done. Add Rem to (E*B) if available.

Step 3: Find Two's complement of ((E*B) +Rem), denote as $T_{((E*B)+Rem)}$.

Step 4: Subtract Secret Key A from $T_{(E*B)}$, i.e. ($T_{((E*B))+Rem)} - A$).

Step 5: Reverse the digits obtained from ($T_{((E*B)+Rem)} - A$), denote as $R_{(T((E*B)+Rem) - A)}$.

Step 6: Subtract 1 from $R_{(T(E*B) - A)}$, denote as D = ($R_{(T((E*B)+Rem) - A)}$) – 1.

Decryption D = ($R_{(T ((E*B)) +Rem)-A)}$) -

3.2.4 Example (Decryption)

Step 1: Take E =

1	0	0	1

A =

1	1	0	1	1	0	1

B =

1	0	0	1	0

Rem =

1	0	1

Step 2: Multiply E*B and Add Rem. (E*B) +Rem.

0	1	0	1	0	0	1	1	1

Step 4: Find Two's Complement of (E*B) +Rem, i.e. T

1	0	1	0	1	1	0	0	1

Step 5: Subtract Secret Key A from T.

1	0	1	0	1	1	0	0	1
-		1	1	0	1	1	0	1
	1	1	1	0	1	1	0	0

Step 6: Reverse it, i.e. (T – A)

0	0	1	1	0	1	1	1

Step 7: Subtract 1 from (T-A)

0	0	1	1	0	1	1	1
-							1
0	0	1	1	0	1	1	0

D =

0	0	1	1	0	1	1	0

It is same as that of H, Huffman Coded Text.

This can be uncompressed to get the original file now [3].

4. ADVANTAGES OF PROPOSED STRATEGY

- The new proposed Symmetric Key Cryptographic Algorithm has been built using simple binary operations.
- It is easy to understand.
- There is a limit on Secret Keys which are being chosen.
- If there are n bits in Huffman Coded Text then, an assailant will have to try 2^n different options to find First Secret Key and another 2^n options to find Second Secret Key.
- Using Huffman compression Technique before encryption, compresses the original file to a great extent, making it useful as less data will need to be encrypted.

5. CONCLUSION

Security has become an important issue over time for large sized data [7]. The main goal of this paper was to find out a way of making data or messages highly secured and smaller in size than the original. To achieve this aim, it uses an existing most effective Huffman Compression Technique on data (so that it can be turned into a small sized file), with a newly developed Block type Symmetric Key Algorithm (to make it secure). Two Private keys have been used which are known to both sender and receiver but are secret from the outside world, that is why known as Secret Key Cryptography. Whole system fulfils the goals of Cryptography and is simple but doesn't leave behind the security issues. It is not vulnerable to Brute-force attack due to large key domain. Currently, this algorithm can be applied on text data only. Next task in future would be to improve the system such that it can be applied on other forms of data, like image, audio, video.

6. REFERENCES

[1] Ayushi, a Symmetric Key Cryptographic Algorithm, International Journal of Computer Applications (0975 - 8887), Volume 1 – No. 15.

[2] S.R. Kodituwakku, U.S. Amarasinghe, Comparison of Lossless Data Compression Algorithms for Text Data, Indian Journal of Computer Science and Engineering, Vol 1 No 4 416-425.

[3] Mohamed F. Mansour, EFFICIENT HUFFMAN DECODING WITH TABLE LOOKUP, IEEE 2007.

[4] An article on "Huffman Codes", available at http://www.columbia.edu/~cs2035/courses/csor4231.F11/huff.pdf

[5] Articles on "Huffman Coding", available at http://en.wikipedia.org/wiki/Huffman_coding, http://en.wikipedia.org/wiki/Data_compression

[6] An article on "Symmetric Algorithm", available at http://www.encryptionanddecryption.com/algorithms/symmetric_algorithms.html

[7] Cryptography and Network Security, Principles and Practices by William Stallings.

[8] Security in Computing by Charles P. Pfleeger, Shari Lawrence Pfleeger.

[9] Pushpa R. Suri1 and Madhu Goel, "Ternary Tree and Memory-Efficient Huffman Decoding Algorithm", IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 1, January 2011.