A Novel Approach for Secure Communication of Text Using Compression Mechanism

[1]N.Aditya Sundar, [2]G.Pandit Samuel, [3]Ch. D.Naidu, [4]M.V.Kishore
[1,2,3,4]Dept. of IT, ANITS, Visakhapatnam, Andhra Pradesh, INDIA

Abstract
The general encryption mechanism involves conversion of plain text into cipher text and decryption as vice versa, as well as multimedia(such as images, video, etc.,) the momentous aspect over here is that the converted cipher text always produces greater size than the original plain text. As there are innumerable number of internet users these days, more memory is accumulated in this process unnecessarily. To avoid such unnecessary accumulation of data we implement some data compression techniques over this encrypted data and also provide security mechanisms for the same and then send over the internet. But as we have many compression tools available in the open market, they fail to provide the security for the data at the same time. The same process can be implemented for the image encryption process. By encoding an image and compressing this encoded image along with security operations. Thus decryption provides the at most security for the data over the network. All the above discussed form of encoding is through Symmetric encryption where it can also be done through Asymmetric encryption. Using the Public Key Infrastructure, the encryption is done using the private key and then compressed and then it is decoded using public key and then decrypted using the secret key that has been shared between them.

Keywords
Compression Mechanism, Data Analysis, AES Cryptographic Algorithm, Symmetric Encryption

I. Introduction

In the present scenario, we have been the eye witness of the rapid growth of Internet users not only through personal computers but also through mobile users. As the chatting applications such as whatsapp and facebook messenger etc, came into limelight where the exchange of data in both text & multimedia causing more accumulation of data. The information through ebooks, multimedia, e-business, opensource applications,etc online makes the data more sensitive and vulnerable. The data needs to be confidential between the sender and the receiver. So we need to follow security mechanisms along with the compression techniques.

Discussing the concepts of network security there are two types of network attacks Viz. active attack and passive attack. If the data is modified during the transmission by the hacker or an unauthorized user then we call it an active attack, if the data is not modified but is captured by unauthorized user then we call it as a passive attack. Encryption process can be either symmetric or asymmetric. If we use same key for both encryption and decryption then we call it as symmetric key encryption , if we use different keys for encryption and decryption we call it as asymmetric key encryption. In our methodology we use symmetric key encryption algorithm.

To provide security to the data that is being transmitted we use cryptography and the various algorithms used can provide only security but at the cost of memory.
For Example: When we take the DES algorithm, if the plain text "Hello world!" is of x bytes we get the cipher text "#*T'O°%&" is of y bytes where y is always greater than x. Where the encrypted data accumulates more memory than the original data.

So we need to compress the data that is being encrypted and then provide security for this compressed data, so the data that is being transmitted is safe and precise with respect to the size. The compression techniques include lossless compression and lossy compression. The formats such as Jpeg, png,.. undergoes lossy compression whereas the format such as gif provides a lossless compression.

The above discussed idea of secure and compressed transmission of data is not confined to the plain text, but can be implemented for the multimedia data such as audio, video and even images. When it comes to images, encryption happens layer wise i.e., each layer is compressed and then encrypted separately. Many image formats such as GIF, PNG and BMP can regain their original size after compression but image formats such as JPEG cannot regain their originality after compression.

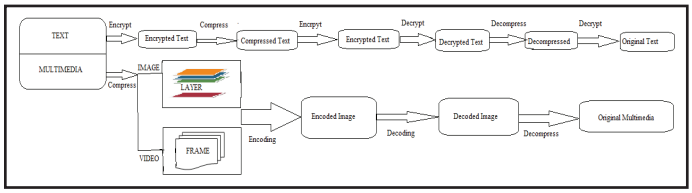

Fig. 1:

[1] Sharing the secret session key during the group key transfer protocols is still under research in the field of network security. In this paper we have studied a transfer protocol based on key authentication in which only authenticated users can recover the group key but unauthorized users cannot access the group key even though KGC can broadcast group key information to all the group members.And a new cryptographic algorithm based on remainder vector and quotient vector instead of a simple cipher text can elevate the security levels to another extent in case of authentication and even confidentiality. The proposed algorithm is lossless and the receiver can recover the original data without any loss of data. [1]
[2] In the present world increased traffic over the internet also increased the security attacks vulnerability which lead to safe guard the data over the internet to avoid such attacks. To safe guard the data data communications need to provide some protocols which provide integrity, confidentiality and privacy of the data .Along with safe transmission of data compression of data is also an important task to perform. A Mod-Encode algorithm which provides encryption followed by compression. In this technique server acts as agro key controller which generates shared group key like a Diffie-Hellmen key exchange mechanism.
A Losseless data compression technique that helps to transmit the data over the network without any loss of data over the receiver side. Many new alogorithms designed may be lossless or lossy it depends upon the application [2].

II. Compression Algorithm

The encrypted message M is a bi-tuple < Q, R> of quotient and remainder. So, the size of the encrypted message can be obtained by calculating the number of bits required to represent Q and R. Let X be the total number of bits required to represent R and is given by X=n*log2 Δ, where n is the length of the message and _log2Δ_ is the number of bits required to represent each remainder. The quotient Q is looked upon as a Base B number. Each qi needs log2 B bits for its representation. A number in Base 10 requires lesser number of bits than its equivalent in another Base B for representation, considering B<10. Therefore, to lessen the number of bits required to represent Q, we first convert it into a Base 10 number, say T. So, the number of bits required to represent Q is given as Y=log2 T. Hence, the total number of bits needed for representation of the encrypted message M can be given by X+Y.. Considering, a 7 -bit representation for every character as in ASCII (or 8-bit in Unicode), Z=n*7 is the total number of bits required for plain text message.

We can observe that Z >(X+Y). So this reduction in bits provides us with the desired compression and the Compression Ratio C.R. is given by C.R.=X+Y/2. [2]

[3] Semistatic word-based byte-oriented compression codes are known to be attractive alternatives to compress natural language texts. With compression ratios around 30%, they allow direct pattern searching on the compressed text up to 8 times faster than on its uncompressed version. In this paper we reveal that these compressors have even more benefits. We show that most of the state-of-the-art compressors such as the block-wise bzip2, those from the Ziv-Lempel family, and the predictive ppm-based ones, can benefit from compressing not the original text, but its compressed representation obtained by a word-based byte-oriented statistical compressor. In particular, our experimental results show that using Dense-Code-based compression as a preprocessing step to classical compressors like bzip2, gzip, or ppmdi, yields several important benefits. For example, the ppm family is known for achieving the best compression ratios. With a Dense coding preprocessing, ppmdi achieves even better compression ratios (the best we know of on natural language) and much faster compression/ decompression than ppmdi alone. Text indexing also profits from our preprocessing step. A compressed self-index achieves much better space and time performance when preceded by a semistatic word-based compression step. [3]

[4] The Internet has become one of the most important components of the world economy. It is a catalyst for business activities and effective governance, a major driver of scientific research and development efforts, an amazing source of growth and human progress. Spear Phishing, Target attacks and Data Breach Trends, man in the middle attack, Spoofing, Sniffing etc.

A sniffer program works at the Ethernet layer in combination with network interface cards (NIC) to capture all traffic traveling to and from internet host site. Further, if any of the Ethernet NIC cards are in promiscuous mode, the sniffer program will pick up all communication packets floating by anywhere near the internet host site. Most of packet sniffers are passive and they listen all data link layer frames passing by the device's network interface.

There are dozens of freely available packet sniffer programs on the internet.

Hijacking (man-in-the-middle attack) takes advantage of a weakness in the TCP/IP protocol stack, and the way headers are constructed. Hijacking occurs when someone between you and the person with whom you are communicating is actively monitoring, capturing, and controlling your communication transparently.

Man-in-middle attacks are like someone assuming your identity in order to read your message.

In hijacking and Man-in-middle attack we observe that the data can be easily tampered by the intruder who can harm the integrity of data or results in loss of data. For this case we would encrypt the data which is encoded where encoding is used for compressing data and adding security to data by changing the message format from plain text to encoded text. Encrypting this encoded data will help in providing double security to the data where the hacker is given complexity where he should be able to compromise both encryption and encoding which takes an increased period of time. [4]

III. Methodology

A. Implementation of ENCODING Algorithm For Secure Communication Of TEXT

The proposed ENCODING algorithm uses any standard encryption technique which incorporates lossless compression in order to regain the needs of less accumulation of data and secured communication.

Let L be a defined language over the ASCII value set Σ. Let the letters of Σ be indexed by a bijection function I that maps a letter to an integer i, where $1 \le i \le |\Sigma|$. Δ is a constant, called modulus constant.

Let the data string M be {m1,m2,m3,.....,mn} $\in \Sigma$ modulus operation on every I(mi) by Δ, sequentially yields remainder set R as {r1, r2 , r3 , . . . , rn} and the quotient set Q as {q1 ,q2 ,q3 , . . . ,qn}. The elements in R have the values between [0,Δ−1]. So, we consider the elements in R to be a vector of numbers in base R.

The quotient set is represented as Let B = [|Σ|/Δ]+1 be another parameter called Base-valuer. The elements of Q will have values within [0, B−1]. Consider Q as a base number in base B, i.e., (q 1 , q 2 , . . . , q n) B. If B is less then 10, then we convert Q B to a Q 10 number.

The ENCODING Algorithm for Encoder can be stated as :

1) Input: M ∈ Σ*
2) n = |M| , i.e. ASCII value of M
3) Z = n × bit size i.e. bit size is the number of bits require to represent each character.
4) for i = 1 to n
   4.1) Read mi, the i th character from M .
   4.2) Find R
      R[i] = I(m i)%Δ
   4.3) Find Q
      Q[i] = I(m i)/Δ
5) Representation of R
   for i = 1 to n
   5.1) Represent R[i] in Base Δ.
6) Representation of Q
   Interpret Q as Base B number and convert it to Base 10

The ENCODING Algorithm for Decoder as follows:

1) Input: Bi-tuple <R, Q>
2) Convert Q from Base 10 to Base B
   Let QB = (q1, q2, . . . , qn) be the representation in Base B
3) Interpret R as a vector of Base Δ number
4) for $1 \le i \le n$
      4.1) i = qi × Δ + ri

where qi the i th digit of QB ri the i th element of R.

4.2) mi = I^(−1)(i)

5) M = (m1 , m2 , . . . , mn)

B. Compression Mechanism

The encrypted message M is a bi-tuple < Q,R > of quotient and remainder. So, the size of the encrypted message can be obtained by calculating the number of bits required to represent Q and R. Let X be the total number of bits required to represent R and is given by X = n∗ log2 Δ where n is the length of the message and log2Δ is the number of bits required to represent each remainder. The quotient Q is looked upon as a Base B number. Each qi needs log2 B bits for its representation. As we know, a number in Base 10 requires lesser number of bits than its equivalent in another Base B for representation, considering B < 10. Therefore, to lessen the number of bits required to represent Q, we first convert it into a Base 10 number, say T. So, the number of bits required to represent Q is given as Y = log2 T. Hence, the total number of bits needed for representation of the encrypted message M can be given by X +Y. Considering, a 7-bit representation for every character as in ASCII (or 8-bit in Unicode), Z = n∗ 7 is the total number of bits required for plain text message. We can observe that Z >(X+Y). So this reduction in bits provides us with the desired compression and the Compression Ratio C.R. is given by C.R. = (X+Y) /Z

Let us explain the working of the above algorithm with an example.

Let us consider Σ to be the character with 256 characters. Let us take the sample message string M as ' Hello! '. The following illustrates the functioning of the above proposed algorithm.

It generates the corresponding ASCII values of the message M i.e., for H as 72, e as. l as, l as. o as, ! as.

Let us take the Δ value as 32 where it generates the base value as [(256/Δ)+1] = 9.

Table 1: Quotient and Remainder Vectors for Different Values of Δ.

Msg	Val	Q	R
H	72	2	8
e	101	3	5
l	108	3	12
l	108	3	12
o	111	3	15
!	33	1	1

The plain text is encoded into the Quotient Vector (Qb) as 2,3,3,3,3,1 and (Q10) as 2,3,3,3,3,1 and the Remainder Vector as 8,5,12,12,15,1.

With the help of ENCODING Algorithm, it converts the quotient vector and remainder vector into their corresponding ASCII values and displays the corresponding encoded data into plain text.

Thus the encoded data can be decoded as 'Hello!' using the ENCODING Algorithm.

C. Overview of AES Cryptographic Algorithm

In this paper, we proposed that the data that has been encoded is compressed using the ENCODING Algorithm and then encrypted using AES Algorithm which has been a strong symmetric key algorithm where the algorithm can be represented as follows:

1. Key Expansion

Round keys are derived from the cipher key using Rijndael's key schedule

2. Initial Round 1

AddRoundKey — each byte of the state is combined with the round key using bitwise xor

3. Sequence of Rounds

- **Sub Bytes**—a non-linear substitution step where each byte is replaced with another according to lookup table.
- **Shift Rows**—a transposition step where each row of the state is shifted cyclically a certain number of steps.
- **Mix Columns**—a mixing operation which operates on the columns of the state, combining the four bytes in each column.
- **AddRoundKey**

4. Final Round (no Mix Columns)

1. SubBytes
2. ShiftRows
3. AddRoundKey

IV. Data Analysis

The practical implementation of the topics, mentioned under methodology are explained in this section with detailed description. As per data analysis, we have done the following steps:

The Delta value need to be taken such as the value should be in between 1 and 256 as there are 256 ASCII values in a keyboard. The Delta value is taken so as to compress the value where the Delta value is always constant called Modulus constant.

As the process flow states, the sender enters the delta vale which helps to calculate the base value by using the formula (256/ delta)+1 and the receiver must also enter the same delta value as we are following the symmetric encryption technique.

The generation can be shown below:

Fig. 2:

Using the obtained base value, we generate the quotient vector(Q) and remainder vector(R) for ASCII set.

The quotient and remainder vector is unique for every ASCII value. The vectors R and Q are generated in such a way that the Q = (ASCII value)/delta and R = (ASCII value)% delta where the values are represented in base Delta.

The Q-R Table generation can be shown as follows:

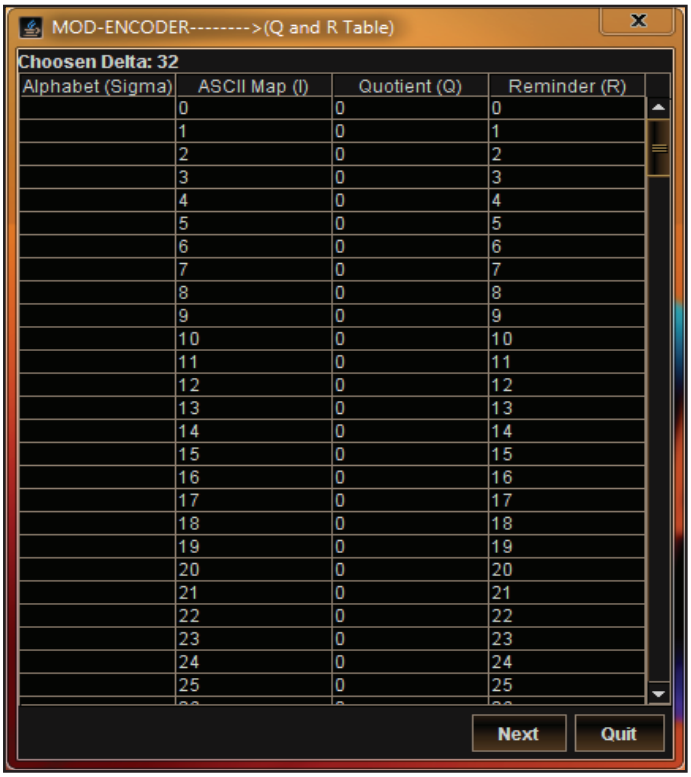

Fig. 3:

We need to establish the connection between sender and receiver and the text file i.e., plain text that we want to encrypt is uploaded. Using the <Q,R> vector we encode the plain text into encoded data which ensures or compression. In the encryption phase, we encrypt the obtained Quotient vector (Q) as well as the Remainder vector(R) compressed file using AES algorithm which was already discussed above.

The obtained value is also encrypted using the AES algorithm so as to provide more security.

Hence, the compressed (encoded), encrypted file is sent to the destination which ensures our security while passing the data through an public channel as shown in the figure.

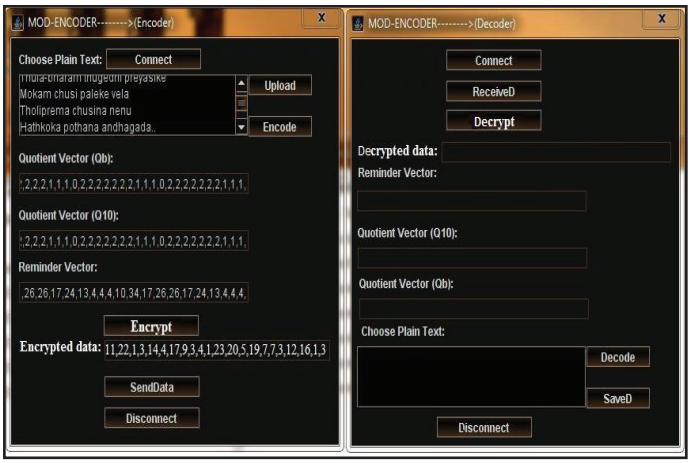

Fig. 4:

The data that is being received by the receiver now decrypts the value and then receives the Quotient vector and receives the Remainder vector after the connection has been established.

The generation can be shown as follows:

Fig. 5:

After the data that has been received at the destination is decrypted and decoded same as the above said components in the backward direction.

Thus the original plain text is obtained at the receiver side as shown in the below figure:

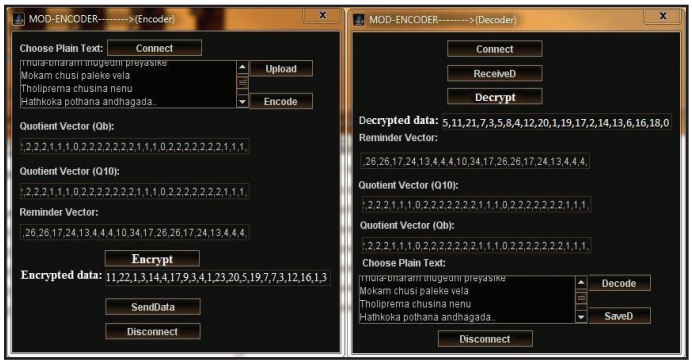

Fig. 6:

V. Conclusion and Future Enhancement

In this paper, we have proposed the secure way of transferring the data after compressing such that the accumulation of data can also be reduced in the Internet traffic. When the security issues of existing system with encoding [1] is compared with the proposed system, we identify the level of security increase in the proposed system which gives s integrity to the compressed data which adds us double security to the data which is passed through an unsecured channel.

The proposed architecture describes only about the text contents or a file, but this can be future enhanced to image, video, etc.

References

[1] G. Praveen Kumar, Biswas Parajuli, Arjun Kumar Murmu, Prasenjit Choudhury, Jaydeep Howlader,"A Lossless MOD-ENCODER Towards a Secure Communication", 2010 IEEE.

[2] R.L. Rivest,"The RC5 encryption algorithm", Proceedings of the 1994Leuven Workshop on Fast Software Encryption", pp. 86-96, Springer-Verlag, 1995.

[3] Amit Jain, Ravindra Patel,"An Efficient Compression Algorithm (ECA) for Text Data", ICSPS, pp.762-765, 2009 International Conference on Signal Processing Systems, 2009.

[4] Farina, A., Navarro, G., Parama, J.R.,"Word-Based Statistical Compressors as Natural Language Compression Boosters", Data Compression Conference 2008, pp. 162 - 171, Mar. 2008.

[5] Taher ElGamal,"A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms", IEEE Transactions on Information Theory, v. IT-31, n. 4, 1985, pp. 469472 or CRYPTO 84, pp. 1018, Springer-Verlag.