Pipeline Data Compression and Encryption Techniques in E- Learning environment

[1] A.V.N.Krishna, [2] Dr. A.Vinaya Babu, [3] B.Vishnu Vardhan

[1] Associate Professor, CSE Dept. Indur Institute Of Engg. & Tech. Siddipet, Medak Dist. Andhra Pradesh. India.
E mail: hariavn@yahoo.com, Mobile: 9849520995.
[2] Director, School for Continues and Distance Education, J.N.T.U. Hyderabad.
[3] Associate Professor, CSE Dept.. Indur Institute Of Engg. & Tech. Siddipet, Medak Dist. Andhra Pradesh, India .

Abstract:

One of the recent developments in telecommunication industry is the introduction of communications through Internet. Some of the applications of internet are easy communications, vast library of information at one place, electronic business through internet, e learning modules and so on. E- learning is an important application of communication through Internet. E Learning is an integrated and continues approach to build knowledge skills and competencies through web enabled technologies. Effective e learning is having advantage in delivering the right content to the right person at the right time.

E learning uses the new or existing network connections connected to internet. The wide area network maintains a high speed connection to an internet service provider that local centers can use to connect to the central LAN from a geographical distance away. Though they offer great accessibility, every one using the internet can see the traffic that passes between a local center and central office over these insecure internet LAN connections.

Considering the fact that different encryption approaches target different types of computational complexity, it may be interesting to see if any further improvement can be achieved when different data encryptions are arranged with different compressions into a pipeline operation.

***Keywords**: Compression Algorithms, RSA Encryption, Arithmetic coding algorithm, Comparative study, A new algorithm.*

1. INTRODUCTION.

1.**RSA Encryption**

RSA algorithm is established on the basis of:

$$M^{\alpha(n)} \equiv 1 \pmod{(n)} \qquad (1)$$

where n is a product of two large prime numbers P and Q. M is any integer which satisfies (0≤*M*<n) and ***α***(*n*) is the Euler totient function which is set to be: ***α***(*n*)=*n* – 1, and relatively prime to n. According to the properties of totient functions, we have:

$$\begin{aligned}\alpha(n) &= \alpha(P\times Q) = \alpha(P)\times\alpha(Q)\\ &= (P-1)\times(Q-1) = n-(P+Q)+1\end{aligned} \qquad (2)$$

To obtain the encryption key, a random integer, d, is selected to be greater than both P and Q. The integer, d, is also relatively prime to ***α***(*n*), namely gcd(d, ***α***(*n*))=1 (gcd=greatest common divisor). After that, another integer, e, can be computed by the following equation:

$$\begin{aligned}e\times d &\equiv 1(\mathrm{mod}(\alpha(n)))\ or\\ e\times d &= 1+k\times\alpha(n) \qquad (k=0,1,2...)\end{aligned} \qquad (3)$$

Therefore, if we choose the pair (e, n) as an encryption key, and (d, n) a decryption key, we will obtain the cyphertext for any integer M (0≤*M*<n) as:

$$C = E(M) \equiv M^{e}(\mathrm{mod}(n)) \qquad (4)$$

For decryption, we have:

$$D(C) = D(M^{e}) = (M^{e})^{d}(\mathrm{mod}(n)) = M^{e\times d}(\mathrm{mod}(n)) \quad (5)$$

From equations (1) and (3), it becomes:

$$D(C) = M^{e\times d} = M^{k\times z(n)+1} = M^{k\times z(n)} \times M(\mathrm{mod}(n)) = M(\mathrm{mod}(n)) \quad (6)$$

When the input message is represented by a series of integers, which are less than n (message = M), we can get the encrypted message correctly decrypted. Regarding the security of this encrypted message, as long as the prime numbers P and Q are selected large enough, it will be very difficult to find out the two numbers only from n by any existing factoring algorithm.

In the operating procedure, the encryption key (e, n) is normally made public, but the decryption key (d, n) is always kept private. Whenever a message is encrypted by anyone in public, only the person with the decryption key can get the right message.

The encryption algorithm can also be used in the public-key cryptosystem. Assuming person A needs to send a signed cheque to person B, he can use his private key d_A to sign the cheque (message M). The signed file becomes $S=d_A(M)$. For the purpose of privacy, he can use person B's public encryption key e_B to encrypt S, and send the encrypted file $e_B(S)$ to person B. Person B can use his private key d_B to get S. Knowing that the cheque is signed by person A, he then use person A's public encryption key e_A to decrypt S, and finally get the decrypted cheque.

2. A NEW ENCRYPTION ALGORITHM.

The new algorithm has the following features…

1. A set of mono alphabetic substitution rule is used.
2. A matrix is used which is used as a key.
3. The matrix key generates a sequence.
4. This sequence is used to the character in the plain text by a particular chosen rule.

The new algorithm is combination of

a. Substitution cipher
b. Matrix key which generates sequential pattern.
c. Modified ceaser algorithm.
d. Coding method.

The steps that are involved in the proposed algorithm.

1. The letters of alphabet were given numerical values starting from 0
2. A random matrix used as a key. Let it be **A.**
3. Generate a "**ternary vector**" for 2^3 values i.e from 0 to 8
4. Let this be "**B**".
5. Multiply **A * B**$^{|}$ **;**
6. consider remainder of the multiple with 3.
7 A sequence is generated.
8. This sequence used as key.
9. Each numerical value of the plain text is added to the key to generate cipher text.
10. The algorithm is reversed to get plain text from cipher text.

It can be seen that to extract the original information from the coded text is highly impossible for the third person who is not aware of encryption keys and the method of coding.

Even if the algorithm is known it is very difficult to break the code and generate key, given the strength of the algorithm. Thus given a short response time through internet communication, the algorithm is supposed to be safe.

Example:

```
                    n=0 to 7
                    n1=floor(n/2);r1=n-n1*2;
                    n2=floor(n1/2);r2=n1-n2*2;
                    n2=r3;
                    r=[r3 r2 r1];

                    r=r'=  |      r3    |
                          |      r2     |
                          |      r1     |
                            |__      ___|
                                        __            __
                              A=key=   |   4      2    -2  |
                                      |   2      4    -5  |
                                      |   -3     2     3 !
                                        |__              _|
                    r=mod(A*r  , 3);

                    r=r(3,1)+r(2,1)*2+r(1,1)*4
          End;
                        OUTPUT
          KEY  r =   21   8    11    17     10     16    19    7;
```

Encryption mechanism:

Plain Text

	a	v	n	k	r	i	s	h	n
	10	31	23	20	27	18	28	17	23
Key:	21	08	11	17	10	16	19	07	21
Total :	31	39	34	37	37	34	47	24	44
%36	31	3	34	01	01	34	11	24	08
Cipher	v	03	y	01	01	y	b	o	08	I

Decryption mechanism:

Cipher	31	3	34	01	01	34	11	24	08	
+36	0	36	0	36	36	0	36	0	36	
total	31	39	34	37	37	34	47	24	44	
Key	21	08	11	17	10	16	19	07	21	
Difference	10	31	23	20	27	18	28	17	23	
Plain text	a	v	n	k	r	I	s	h	n	.

3. ARITHMETIC CODING

It would appear that Huffman or Shannon-Fano coding is the perfect means of compressing data. However, this is *not* the case. As mentioned above, these coding methods are optimal when and only when the symbol probabilities are integral powers of 1/2, which is usually not the case.

The technique of *arithmetic coding* does not have this restriction: It achieves the same effect as treating the message as one single unit (a technique which would, for Huffman coding, require enumeration of every single possible message), and thus attains the theoretical entropy bound to compression efficiency for any source.

Arithmetic coding works by representing a number by an interval of real numbers between 0 and 1. As the message becomes longer, the interval needed to represent it becomes smaller and smaller, and the number of bits needed to specify that interval increases. Successive symbols in the message reduce this interval in accordance with the probability of that symbol. The more likely symbols reduce the range by less, and thus add fewer bits to the message.

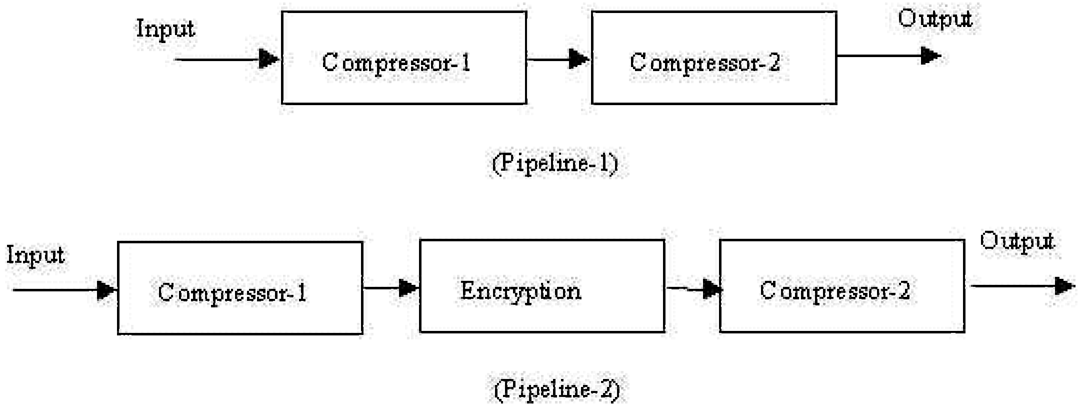
```
 1                                         Codewords
+-----------+-----------+-----------+          /-----\
|           |8/9 YY     |  Detail   |<- 31/32   .11111
|           +-----------+-----------+<- 15/16    .1111
|    Y      |           | too small |<- 14/16   .1110
|2/3        |    YX     | for text  |<- 6/8     .110
+-----------+-----------+-----------+
|           |           |16/27 XYY  |<- 10/16   .1010
|           |           +-----------+
|           |    XY     |           |
|           |           |   XYX     |<- 4/8     .100
|           |4/9        |           |
|           +-----------+-----------+
|           |           |           |
|    X      |           |   XXY     |<- 3/8     .011
|           |           |8/27       |
|           |           +-----------+
|           |    XX     |           |
|           |           |           |<- 1/4     .01
|           |           |   XXX     |
|           |           |           |
|0          |           |           |
+-----------+-----------+-----------+
```

As an example of arithmetic coding, lets consider the example of two symbols X and Y, of probabilities 0.66 and 0.33. To encode this message, we examine the first symbol: If it is a X, we choose the lower partition; if it is a Y, we choose the upper partition. Continuing in this manner for three symbols, we get the code words shown to the right of the diagram above - they can be found by simply taking an appropriate location in the interval for that particular set of symbols and turning it into a binary fraction. In practice, it is also necessary to add a special end-of-data symbol, which is not represented in this simple example.

In this case the arithmetic code is not completely efficient, which is due to the shortness of the message - with longer messages the coding efficiency does indeed approach 100%.

4. PIPELINE DATA COMPRESSION AND ENCRYPTION

The two pipelines of data transmission can be designed as shown in Figure 1:



(Pipeline-1)

(Pipeline-2)

Figure 1: *Pipeline data compression*

The pipeline is designed to analyze the impact of encryption upon the compressed output and to see if the encryption is the favorable transform we are looking for. After compression, the compressed output is a complete random data file (noise) with its redundancy being partly removed. When the encryption algorithm is operated on the random file, the output can be expected to be another random data file. But the encryption process is completely lossless. Therefore, its insertion will not incur any loss of information, which is exactly what we desired. Hence, the experiment is carried out to see the performance of different encryption algorithms on the transformed random data file. The results are illustrated in Table

Table : Second Pipeline Experimental Results with different encryption techniques.

Parameters for comparative study	Encryption: New Algorithm	Encryption: RSA algorithm
1. Response time	Less than 1 sec.	More than 1 sec.
2. Length of the key: 27 decimal digits.	5 MIPS yrs	5 MIPS yrs
3. Computational overhead. – per character.	6 computations	11 computations(p=11 &q=13 depending on prime numbers.
4. Computational complexity	Exponential	Exponential.

CONCLUSION

The increasing popularity of internet in e learning makes it highly desirable to use encryption and compression techniques. The study provides some insight into computational complexity, response time and code breaking time of some encryption algorithms. Thus the study provides some insight into overall improvement in data transmission , better bandwidth utilization, reduction in computational complexity and overall improvement in e learning technologies. Thus it can be verified the purpose of encryption and compression algorithms in virtual private networks which forms an integral component of e learning technologies.

REFERENCES:

1. 1.Rivest R.L. et.al. ‘A method for obtaining digital signatures and public-key cryptosystems’, Commun. Of the ACM, Vol 21, No 2, February 1978.
2. 2.Jiang, J. ‘Pipeline algorithms of RSA data encryption and data compression’ *Proceedings of ICCT’96: International Conference on Communciation Technology*, IEEE Press, Vol 1, pp 1088-1091, 1996.
3. 3.Jiang, J. and Jones S. ‘Parallel design of arithmetic coding’ *IEE Proceedings-E: Computer and Digital Techniques*, Vol. 141, No. 6, November, 1994, pp 327-323, ISSN: 1350-2387.
4. 4.Welch T.A. ‘A technique for high-performance data compression’, IEEE Computer, 17(6), June 1984, pp8-19.
5. 5.Fiala, E.R. and Greene, D.H. ‘Data compression with finite windows’, Commun. Of the ACM, 32(4), April 1989, pp490-505.

Article received: 2006-03-28