International Journal of Science and Research (IJSR)
ISSN: 2319-7064
SJIF (2020): 7.803

Arithmetic Addition Coding and Linear Invertible Equation using Prime Modulo with Multiplicative Inverse for Text Encryption

N. Subramanyan[1], Dr. K. Arunesh[2]

[1]Research Scholar, Department of Computer Science, Sri. S Ramasamy Naidu Memorial College, Sattur, TN, India
nsm2517[at]gmail.com

[2]Associate Professor, Department of Computer Science, Sri. S Ramasamy Naidu Memorial College, Sattur, TN, India
arunesh_naga[at]gmail.com

Abstract: ***One of the most important challenging tasks in transferring data over digital medium now a days is security. Confidentiality, Integrity, and Authenticity are the fundamental principles of information security. Every technique which deals security should be designed to succeed in achieving one or more of these fundamental principles. Coding based Cryptography is one of the research areas in the field of Cryptography. In this paper, a new method has been proposed which deals with a variant encryption scheme using arithmetic addition coding and linear inverse function. Arithmetic addition coding is a technique which divides the input data text into 4x4 blocks and then encodes using a series of simple addition operations. Linear inverse function encrypts the encoded data using a linear invertible equation using prime module with multiplicative inverse. Linear invertible function adds additional level of encryption to the encoded data. This combination giving best encrypted results than the existing method. The performance of proposed method is measured with encryption time and decryption times for various file sizes.***

Keywords: Arithmetic addition coding, Authenticity, Confidentiality, Cryptography, Encryption time, Decryption time, Integrity, Inverse function, Prime modulo

1. Introduction

Cryptography [1] is generally referred to as "the study of secret". Plain text contains data which can be directly read and understood easily without any additional conversions. Encryption is the process which masks the plaintext. Cipher text is the result of encryption which is unreadable and gibberish. Encryption is the only technique to hide data from others for whom it is not intended. Decryption is the process which reverts the encrypted text to the original plaintext.Fig.1 shows the steps involved in the encryption and decryption algorithms.

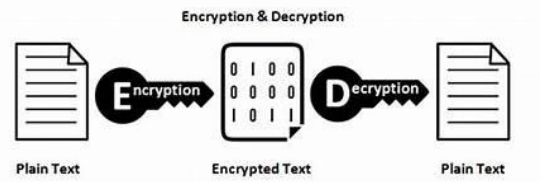

Figure 1: Encryption and Decryption

Authenticity, Confidentiality, Integrity, Non-Repudiation and Service Reliability and Availability are the five goals of cryptography [1], every security system should follow these goals to assure the secrecy of the system.

Authenticity: Ensures the identity of the sender and the receiver before sending and receiving the data using the system.

Confidentiality: Ensures that only authenticated sender and receiver are able to send and receive messages and nobody else others.

Integrity: Ensures that there no modification or change of data during the transmission.

Non-Repudiation: Gives assurance that neither the sender nor receiver can falsely deny that they have sent a certain message.

Service Reliability and Availability: Ensures the quality of service and availability of service to their users. Since secure systems usually get attacked by intruders, which may affect their availability.

Based on the input data they operate, encryption techniques are classified as block ciphers and stream ciphers.In block ciphers, the plain text is divided into blocks, which are given input to encryption system which produces cipher text blocks. Whereas stream ciphers operate on streams of data character by character to produce cipher text.

Symmetric and Asymmetric encryptions are the two main categories of encryption techniques based on the type of secrete keys used in encryption and decryption process. In symmetric encryption as shown in Fig.2 both the sender and the receiver uses a same shared key in encryption and decryption process. There is only one key in symmetric encryption.

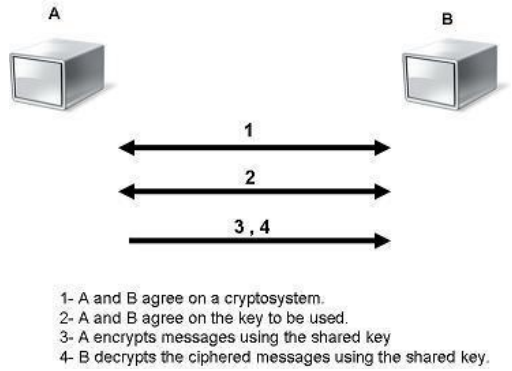

Figure 2: Symmetric Encryption

Asymmetric encryption is another kind of encryption where two key are used, one key is used in encryption and the other key used in decryption. It is also called as public key cryptography. Fig. 3 below illustrates the use of the two keys between node A and node B.

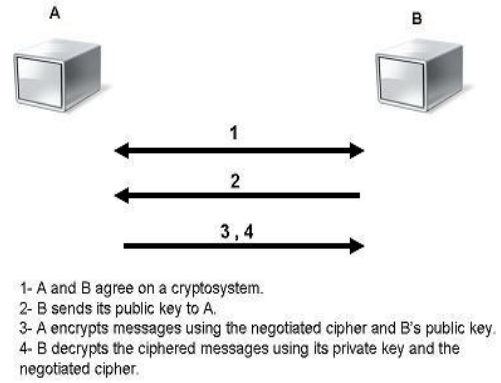

Figure 3: Asymmetric Encryption

Data Compression:

Data compression [2] also called as source coding or bit-rate reduction is the process of encoding information using fewer bits than the original representation. Any compression is one of two types either lossy or lossless compression. Statistical redundancy among bits are identified and eliminated to reduce the size of data in lossless compression[3], where as in lossy compression unnecessary less important bits are identified and removed. The process of reducing the size of a data file is referred to as data compression. Data compression is one of the challenging tasks when the things are in the form of storing, sending and receiving of data through network. Various compression algorithms are available to compress various types of data such as text, audio and video etc.

2. Existing Method

Nigam Sangwan [4] proposed a text encryption scheme by Combining Huffman coding [5] with two cryptographic techniques. The cryptographic techniques were implemented using XOR operations and random key. Huffman coding technique requires knowledge of encoding during decoding process; it requires additional bandwidth during encrypted data transmission. When the ASCII code system is considered, it will affect the compression size there by it impacts in the encryption process.

3. Proposed Method

The proposed work combines the arithmetic addition coding [6] and linear invertible equation using prime modulo with multiplicative inverse [7] for text encryption to generate the cipher text. Proposed method supports the ASCII code system.

Arithmetic Addition coding:

Nadeem Akhtar, Gufran Siddiqui and Salman Khan [6] proposed simple arithmetic addition, it is a coding technique. In this technique initially 16 numbers are considered and arrange them in a 4x4 block, and then the following steps are performed.

1) Distinct numbers among the 4x4 blockare identified and arrange them in ascending order.These are assumed as array indexes; initialize each cell of the array with the number of integers between itself andthe previous integer.
2) In this step, iterate through the block sequentially from start to end. At each step make the value stored in the cell is Highest, corresponding to the current number. It is done by adding [current highest value (excluding the value stored in the current cell) + 1] to the value stored in the current cell. Repeat the process for all 16 numbers. Finally array will contain the distinct numbers along with highest values.
3) The final encoded 4x4 block is stored in the following format.

←8 bits→ Starting Number	←4 bits→ (D_n) Distinct Numbers	←4 bits→ (B_m) Bits required by the maximum number	←D_n*B_ms bits→

Figure 4: Encoded data format

Let illustrate the process with an example consider a 4x4 block of numbers as shown below:

97	98	97	98
105	98	97	97
98	99	101	101
102	97	97	101

Figure 5: 4x4 block

The sixunique numbers 97, 98, 99,101,102&105.
After arrangingin ascending order, the values are

97	0
98	0
99	0
101	1
102	0
105	2

The first number in this example is 97. In the above array, the current highest value is 2. So add 1 to it and add the resulting value to the value initially stored in cellnumber 97. So the value of cell number 97 now becomes 3 (0 + [2+1]) which is now the highest number inthe updated array.

old	
97	0
98	0
99	0
101	1
102	0
105	2

updated	
97	3
98	0
99	0
101	1
102	0
105	2

Repeating the above process for all numbers in the block (i.e. 97, 98, 97, 98, 105, 98, 97, 97, 98, 99, 101, 101, 102, 97, 97, 101), the final array contains the values as follows.

97	488
98	101
99	102
101	696
102	208
105	16

The considered 4x4 block after encoding is stored in the following format.

Start	D_n	B_m	← D_n*B_m →					
←8→	←4→	←4→	←10→	←10→	←10→	←10→	←10→	←10→
97	6	10	488	101	102	696	208	16

Now, let us look the decoding process, first consider the previously encoded binary data

Start	D_n	B_m	← D_n*B_m →					
←8→	←4→	←4→	←10→	←10→	←10→	←10→	←10→	←10→
97	6	10	488	101	102	696	208	16

Following information is retrieved from the above binary data, the starting number is 97.

1	488
2	101
3	102
4	696
5	208
6	16

In decoding, the following steps are performed,

1) First find the highest value and write its corresponding indexed value in the 4x4 block in reverse order.
2) Next, subtract the (second highest value +1) from the first highest value and replace highest value with it.

In this example, 696 is the highest value which is stored in cell number 4. So, insert a value 4 in the 4x4 block as shown.

			4

Next, subtract the [second highest value + 1] (i.e. 488 + 1) from the highest value (i.e. 696), the value is 207. Replace 696 with 207. The updated array is as follows:

1	488
2	101
3	102
4	207
5	208
6	16

In the above updated array, next highest value is 488 which is in cell number 1. So insert 1 in the 4x4 block as shown below.

		1	4

Repeat the above process until the entire 4x4 block filled. The final 4x4 block will look as follows.

1	2	1	2
6	2	1	1
2	3	4	4
5	1	1	4

Reconstructed array from the 4x4 block as follows

1	0
2	0
3	0
4	1
5	0
6	2

The final recovered array gives the gapping information between adjacent numbers. We know that starting number is 97, with this information, retrieved original array as follows.

1	0	97
2	0	98
3	0	99
4	1	101
5	0	102
6	2	105

By replacing the numbers (i.e 1, 2, 3, 4,5& 6) in the 4x4 block with the above recovered array values the 4x4 block look as follows.

97	98	97	98
105	98	97	97
98	99	101	101
102	97	97	101

The above 4x4 block is same as that of 4x4 block we initially considered.

Cryptography using Inverse function:

Let consider the following theorem statement.

***Theorem*:** "If p is a prime number, then every non-zero integer modulo p has a multiplicative inverse " [8].

Using the above theorem for prime number *P*, the general form of linear inverse function with prime modulo multiplicative inverse is

$$f(x) = \frac{a}{b}x + \frac{c}{d} \quad (1)$$

Where a, b and c cannot be multiples of prime number *P*.

For example, if the prime number *P* is 53, then the linear inverse function with prime modulowith multiplicative inverse is

$$f(x) = 3x + 1 \quad (2)$$

The inverse function is

$$f(x)^{-1} = 18x - 18. \quad (3)$$

Steps involved in encryption process are as follows.

1) Read plain text.
2) Generate ASCII codes, then divide the plain text into chunks of 16 characters and arrange them into blocks of 4x4.
3) Apply arithmetic addition coding for each 4x4 block and generate binary vector.
4) Divide the binary vector into 8 bit chunks and convert the binary code to decimal value.
5) Using linear invertible function, generate the cipher codes for the result of above step.

Steps involved in decryption process are as follows.

1) Read the cipher codes.
2) Apply the linear invertible function to decrypt the cipher codes.
3) Generate a binary vector by finding the binary codes for theresult of step2.
4) Read the binary vector data and apply arithmetic addition decompression.
5) Convert the decimal values generated in above step to its equivalent ASCII Codes as plain text.

4. Experimental Results

Let execute the proposed method for the plaintext “Network Cryptography”.

Encryption process:

Step 1: Plaintext is“Network Cryptography”.

Step 2: ASCII codes for plaintext are
“78, 101, 116,119, 111, 114, 107, 32, 67, 114, 121, 112, 116, 111, 103, 114, 97, 112, 104, 121”

Dividing ASCII codes into chunks of 16 and padding 0s if not in size of 16.Arranging the chunks into 4x4 blocks.

78	101	116	119
111	114	107	32
67	114	121	112
116	111	103	114

97	112	104	121
0	0	0	0
0	0	0	0
0	0	0	0

ASCII codes arranged in 4x4 blocks

Step3:After applying the addition coding, the generated binary vector is
”00000000001010011000010000011001010000101010000011101110000101110100010001000101110000100010100110101011111001100101010001010010100010001000100100100110010010010101001011011100001000000000011111100001001110000010001110000101011 1”

Step4: After dividing the binary vector into 8 bit chunks the generated decimal values are0, 20, 194, 12, 161, 80, 119, 11, 68, 69, 132, 83, 87, 204, 168, 165, 17, 18, 76, 146, 165, 184, 64, 15, 194, 112, 71, 10, 255, 5

Step 5: After applying the inverse function, the generated cipher codes are1, 61, 69, 37, 227, 241, 101, 34, 205, 208, 140, 250, 5, 99, 248, 239, 52, 55, 229, 182, 239, 39, 193, 46, 69, 80, 214, 31, 252, 16.

Decryption process:

Step 1**:** Reading the cipher text.

“1, 61, 69, 37, 227, 241, 101, 34, 205, 208, 140, 250, 5, 99, 248, 239, 52, 55, 229, 182, 239, 39, 193, 46, 69, 80, 214, 31, 252, 16”

Step 2: Appling the linear inverse function for decrypting the cipher codes are “0, 20, 194, 12, 161, 80, 119, 11, 68, 69, 132, 83, 87, 204, 168, 165, 17, 18, 76, 146, 165, 184, 64, 15, 194, 112, 71, 10, 255, 5”

Step 3: Generating the binary vector data for the above step.

”00000000001010011000010000011001010000101010000011101110000101110100010001000101110000100010100110101011111001100101010001010010100010001000100100100110010010010101001011011100001000000000011111100001001110000010001110000101011 1”

Step 4: After applying the arithmetic addition decompression and after removing the padded 0’s

“78, 101, 116,119, 111, 114, 107, 32, 67, 114, 121, 112, 116, 111, 103, 114, 97, 112, 104, 121”

Step 5: The plain text is “Network Cryptography”.

5. Performance Evaluation

The basic performance parameter for compression method is compression ratio; it is the size of the compressed data to the size of the original data. Encryption time and decryption time are the two basic parameters to evaluate the performance of the cryptographic algorithm. Tables 1, 2 and 3 show the performance analysis between the existing method and proposed method. The compression ration of the proposed method is high when compared with the existing method. Encryption and decryption times are less for the proposed method when compared to the existing method. Table 3 shows the comparison of execution times of standard encryption techniques and the proposed method.

Table 1: Performance results of compression ratio, time taken for compression and encryption for various file sizes.

File size in (kB)	Compressed Size in (kB)		Compression Ratio (%)		Compression Time (seconds)		Time taken in encryption (seconds)		Total time (Compression+ Encryption) (seconds)	
	Existing Method	Proposed Method	Existing Method	Proposed Method	Existing Method	Proposed Method	Existing Method	Proposed Method	Existing Method	Proposed Method
0.0332	0.0175	0.0275	52.9	82.831	0.0467	0.0402	0.0156	0.0136	0.062	0.046
10.833	5.9141	9.0248	54.6	83.308	0.9531	0.8159	1.636	1.2355	2.59	2.097
21.337	11.833	18.335	55.45	85.929	2.1562	1.3989	3.483	3.1254	5.64	5.171
30.080	16.8046	28.543	55.87	94.891	3.3594	3.2408	5.6	5.3252	8.96	8.566
101.22	56.4785	97.035	55.79	95.857	32.687	33.729	30.734	30.987	63.42	64.716

Table 2: Performance results of time taken for decompression and decryption for various file sizes.

Compressed file Size(kB)	Decompression Time(seconds)		Time taken in decryption (seconds)		Total time (Decompression+ Decryption) (seconds)	
	Existing Method	Proposed Method	Existing Method	Proposed Method	Existing Method	Proposed Method
0.0275	0.0002	0.0283	0.0311	0.0044	0.0313	0.03282
9.0554	1.1875	1.9318	1.4063	0.0172	2.5938	1.94906
18.3355	2.9532	3.8752	3.0468	0.8013	6.0000	4.67653
28.5433	4.0313	4.6804	4.625	2.8639	8.6563	7.54434
97.0359	17.375	6.3852	14.8594	6.9876	32.2344	13.3728

Table 3: Execution time comparisons

Text file (kB)	DES (K4) (Seconds)	RC$ (K4,K5) (Seconds)	*DoubleEncryption (K1,K2,K3) (Seconds)*	Proposed Linear invertible equation (Seconds)
0.013	0.3	0.19	0.03125	0.030702
0.636	4.8	0.2	0.1565	0.062141
6.17	245	2.13	1.48437	0.380477
8.21	438	3.57	2.14062	0.586453

6. Conclusion

Transferring data over digital medium with security and limited bandwidth necessitates the use of compression and security. The combination of coding and cryptographic methods, first size of data is reduced then encryption is applied, which results good impact in time taken for encryption and decryption, saving memory and network bandwidth during transfer of data. Proposed method Arithmetic addition coding and linear invertible equation using prime modulo giving good results when compared with existing methods and also it is difficult for the third parities to break the system. The proposed method can be extended to UNICODE and can also be applied to other types of data such as image, audio and video.

References

[1] W. Stallings, Cryptography and Network Security, 4th ed., vol 8, 2006.
[2] Data Compression. [Online]. Available:http://en.wikipedia.org/wiki/Data_compression.
[3] L. Robert and R. Nadarajan, “Simple lossless preprocessing algorithms for text compression” IET Softw., 2009, Vol. 3, Iss. 1, pp. 37–45, August 2008.
[4] Nigam Sangwan, “Combining Huffman text compression with new double encryption algorithm” Published in: 2013 International Conference on Emerging Trends in Communication, Control, Signal Processing and Computing Applications (C2SPCA).
[5] Huffman Coding, 2021. Online Availablehttps://en.wikipedia.org/wiki/Huffman_coding
[6] Nadeem Akhtar, Gufran Siddiqui and Salman Khan, ”A Novel Image Compression Technique using Simple Arithmetic Addition”, Proc. of Int. Conf. on Recent Trends in Information, Telecommunication and Computing, ITC pp.319-327, 2014.
[7] J.B. Fraleigh, A First Course in Abstract Algebra, 7th ed., Addison Wesley, NY, USA, 2002, page 181.