Enhanced Selective Encryption for Encryption of Plain Text using Multiple Indexing
Shubham Srivastava
Student of B.Tech
Department of ECE
I.E.R.T Institute Sitapur, Uttar Pradesh, India

Abstract:
Selective encryption is one of the most commonly used encryption technique for the multimedia files now-a-days. The major advantage of the algorithm is that it only encrypt selected data which is critical and needed to be secured which take less time and make the multimedia files more usable over public network. The main problem with the selective encryption is that it produces weak cipher which make it unsecure for the plain text. This paper proposes new modifications in selective encryption which targeted to overcome the problem of weak cipher so that it can be used on plain text too, the major change is to use the multiple indexing instead of single but by maintaining the less time.

Keywords: Advanced Encryption Standard; Cipher Complexity; Linear Congruential Theorem; Prime Number indexing; Pythagorean Theorem; Selective Encryption; multiple indexing

I. INTRODUCTION

The evolution of technologies over the centuries have made many common people bewildered and excited. They have experienced so many changes over the years that they have no idea what new technology will evolve tomorrow. People’s imagination has helped to bring out new ideas in technology. The smaller and smarter technologies are all in rage in the market. They have easy access of the internet to make cash transaction, do shopping, to communicate and to store personal data. They have also made people vulnerable to spamming and violation of the privacy. I proposed an encryption algorithm based on Selective Encryption (SE) that will encrypt the plaintexts with less processing time as compared to Advanced Encryption Standard (AES) while maintaining the desired cipher complexity. The algorithm is formulated by using multiple indexing on SE. Indexing is done based on three methods: Prime Number Indexing, any mathematical equation and any random generator. The paper covers the comparative time analysis of AES with the proposed algorithm. The algorithms was implemented and successfully executed in java. The result shows that there are vast computation time differences between AES and proposed algorithm for small file sizes and for large file sizes there is small computation time differences but still they have better cipher complexity than the AES.

II. SELECTIVE ENCRYPTION OF MULTIMEDIA

Selective encryption is a new vogue in image and video content protection in multimedia cryptography. It consists of encrypting only the subset of data thereby reducing the time complexity of computation. The aim of selective encryption is to reduce the amount of data to encrypt while preserving a certain level of security. Moreover, selective encryption also preserves some codec functionalities such as scalability. The general selective encryption works as follows. The image is first compressed (if needed). Afterwards the algorithm only encrypts part of the bitstream with a well-proven ciphering technique; incidentally a message can be added during this process. In order to guarantee a full compatibility with any decoder, the bitstream is only altered at places where it does not compromise the compliance to the original format. This theory is sometimes referred to as format compliance [2]. The receiver decrypts the bitstream with the decryption key and then decompresses the image. The receiver will be able to decompress the image even if the key is unknown, but this image will significantly differ from the original.

There are many methods available for selection process where crucial data are selected and encrypted before sending them through insecure channel.

A. Classification of Selective Encryption

Selective Encryption Algorithms are classified based on following approaches:

1) Precompression
2) Incompression
3) Postcompression
In Precompression method, the compression is done after the encryption. They are inherently format compliant and irrelevant for lossy compression. But drawback of using this method is that it causes bandwidth expansion and adversely impact compression efficiency. Hence this method is not compression friendly.

**Figure 2. Precompression Approach** [9]

In Incompression based algorithms, both encryption and compression are jointly performed. But in this approach the format compliance and compression friendliness are adversely impacted.

**Figure 3. Incompression Approach** [9]

In Post compression based algorithms, encryption is done after the compression. This approach is compression friendly. Encryption and decryption does not need any modification at encoder or decoder sides. Moreover, it is nonformat compliant.

**Figure 4. Postcompression Approach** [9]

The best approach for my proposed selective encryption of plaintext is precompression approach.

### III. SELECTIVE ENCRYPTION OF PLAINTEXT

Selective encryption is computationally faster and secure with moderate cipher complexity. It is a technique to save computational power, overhead, speed, time and provide quick security by only encrypting a selected portion of a bitstream. Selective Encryption was first proposed by Spanos and Mapies in 1995 using post-compression method. Since then many new selective encryption was proposed with various different methods with the focus of improving its time complexity and security and its robustness with plain-text or chosen-text attack. I have studied various techniques that have been proposed in selective encryption till 2013. From the extensive research and study, it is understandable that cipher complexity for encrypting plaintext by using existing selective encryption is very weak even though its time complexity is less as compared to AES. I formulated an algorithm that overcomes the limitation exhibited by existing SE on plaintext. I used three different methods to select the data from plaintext on which I applied AES. This process of selection is called indexing. I used multiple rounds of indexing for maximum selection of data from the plaintext thereby increasing the cipher complexity without compromising my goal of achieving lesser time complexity as compared to AES. When I used existing SE on plaintext by only taking half of the elements of plaintext, I found that computation time is much lesser as compared to AES. I used 128-bits AES for encryption purpose with 128-bits key.

**Table 1. Relation Between Cipher_Complexity and No. Of Rounds**

<table>
<thead>
<tr>
<th>Cipher_Complexity</th>
<th>No. Of Rounds (N)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low</td>
<td>3</td>
</tr>
<tr>
<td>Medium</td>
<td>6</td>
</tr>
<tr>
<td>High</td>
<td>10</td>
</tr>
</tbody>
</table>

Indexing is done using the following steps.

1) X be the number selected for indexing
2) All the columns that are multiples of X are selected from the plaintext array.
3) All the elements of the selected columns are extracted and assembled together to form Selective Table.

I applied AES algorithm to encrypt the elements of the Selective Table which is the size of 128 bits or 16 bytes. Then Cipher Table is formed. The corresponding cipher is replaced to its original location at plaintext array. Selection of the ‘X’ number for indexing can be done using any of following methods:
**A. Prime Number Indexing**

In this method, I selected first 10 prime numbers for maximum N = 10. Prime Number is special class of numbers that influence many things in the world of cryptography. Prime Number is an old technique normally used for this kind of purposes. It is proven that primes are unique numbers. They are unique in a manner that the product of prime number with any number has the best chance of being unique. Generally, any popular encryption scheme uses prime number to create a very good level of security. For 10 rounds for indexing, then first ten prime numbers are selected i.e. 2, 3, 5, 7, 11, 13, 17, 19, 23 and 29. Then columns that are multiple of all these prime numbers are selected to encrypt. E.g. for first round, all the elements at position of multiples of 2 i.e. 2, 4, 6, 8, etc. in plaintext array are selected and encrypted. For second round, multiples of 3 elements are selected and so on.

**B. Mathematical function or equation**

In this method, any mathematical functions or equations such as Fourier series or Pythagoras theorem equation for second method of indexing.

Fourier series equation is
\[
f(x) = a_0 + \sum_{n=1}^{\infty} \left( a_n \cos \frac{2\pi nx}{l} + b_n \sin \frac{2\pi nx}{l} \right) (1)
\]

Pythagoras equation is
\[
H^2 = B^2 + P^2 \ 	ext{where} \ H: \text{Hypotenuse}, B: \text{Base}, P: \text{Perpendicular}
\]

User can decide which mathematical equation or function he wants to use. Here the user can make alteration on indexing based on his requirement. Thereby I overcame the constraint on algorithm being static.

In my algorithm I used Pythagoras equation
\[
X_i = \sqrt{\text{cipher complexity}}^2 + (\text{prime}[i])^2 (2)
\]

where prime[i] = 2,3,5,7,11,13,17,19,23 and ‘i’ is the \(i^{th}\) round

**C. Random Generator**

Random Generator is a device that generates a sequence of random number that has no significant pattern. Here also, it depends on the user which random generator he wants to use for indexing. Thereby, overcoming the constraint on algorithm being static. In my algorithm I used Linear Congruential Theorem of Pseudo-Random Generator. It uses four parameters.

\[
m \quad \text{the modulus} \quad m>0 \\
a \quad \text{the multiplier} \quad 0<\alpha<m \\
c \quad \text{the increment} \quad 0<\beta<m \\
X_0 \quad \text{the starting value} \quad 0\leq X_0 < m
\]

The sequence of random numbers \([X_i]\) is obtained via the following iterative equation:
\[
X_{i+1} = (aX_i+c) \mod m (3)
\]

I used \(a = \text{cipher complexity}, c = \text{cipher complexity}, m = 50, X_0 = 1\) for our implementation of algorithm.

**IV. PERFORMANCE RESULT AND ANALYSIS**

In this section, the result obtained by executing the proposed algorithm in java are shown.

**A. Time Analysis**

I analyzed the average time obtained on various methods of proposed algorithm with the existing SE and AES by using varying filesizes under different file size range. Here, I considered the filesize as the number of characters read from plaintext.

**B. Cipher Analysis**

For cipher complexity, I can explain with the help of an example. When Cipher Complexity = Low while using Prime Number Indexing SE method. The maximum number of encryption done on some elements of plaintext is twice. Fig.6. explains this in detail.

**C. Filesize Analysis**

While studying the time analysis, I also observed that the filesize of cipher while using AES is greater than the all the

From the above table, you can see that there is a huge average time difference between AES and the proposed algorithms. S_AES is existing SE, S_PI is SE using prime indexing method, S_PT is Pythagoras Theorem method and S_RG is Random Generator method.

**TABLE.IV. AVERAGE TIME OF DIFFERENT ALGORITHMS FOR FILESIZE GREATER THAN 100K**

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Low</th>
<th>Medium</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td>AES</td>
<td>4288.6ms</td>
<td>5127.4ms</td>
<td>5834.2ms</td>
</tr>
<tr>
<td>S_AES</td>
<td>3859.8ms</td>
<td>4756.6ms</td>
<td>4947.4ms</td>
</tr>
<tr>
<td>S_PI</td>
<td>3878.4ms</td>
<td>4110ms</td>
<td>5582.4ms</td>
</tr>
<tr>
<td>S_PT</td>
<td>4205.4ms</td>
<td>4529ms</td>
<td>5498.8ms</td>
</tr>
<tr>
<td>S_RG</td>
<td>5080.4ms</td>
<td>5591ms</td>
<td>5992.8ms</td>
</tr>
</tbody>
</table>

From studying Tables II, III, IV you can see that average time of all the proposed algorithms is comparatively less than that of AES.

**TABLE.V. BEST PROPOSED METHOD**

<table>
<thead>
<tr>
<th>Filesize Range</th>
<th>Low</th>
<th>Medium</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-10K</td>
<td>S_PT</td>
<td>S_PT</td>
<td>S_PT</td>
</tr>
<tr>
<td>10K-100K</td>
<td>S_PT</td>
<td>S_PT</td>
<td>S_RG</td>
</tr>
<tr>
<td>&gt;100K</td>
<td>S_PI</td>
<td>S_PI</td>
<td>S_PT</td>
</tr>
</tbody>
</table>

From the Table V, you can see that the best method is Pythagoras Theorem equation and Prime Number Indexing.
proposed method. This we achieved by compressing the cipher obtained from the proposed algorithms.

**TABLE VI. AVERAGE FILE SIZE DIFFERENCE BETWEEN AES AND PROPOSED ALGORITHMS**

<table>
<thead>
<tr>
<th>Filesize Range</th>
<th>Average Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-10K</td>
<td>1084.6</td>
</tr>
<tr>
<td>10K-100K</td>
<td>12371.8</td>
</tr>
<tr>
<td>&gt;100K</td>
<td>190850.2</td>
</tr>
</tbody>
</table>

![Figure 6. Elements encrypted in Plaintext while using Prime Number Indexing Selective Encryption method when Cipher_Complexity is Low](image)

Figure 6. Elements encrypted in Plaintext while using Prime Number Indexing Selective Encryption method when Cipher_Complexity is Low

V. CONCLUSION

From the various observations of varying filenames with different keys, the following merits were observed in my algorithm:

- Time complexity is lower as compared to AES for all proposed methods.
- File sizes of the encrypted file for the three proposed methods is lower than the existing AES algorithm.
- As the number of rounds increases the number of encryption done on some elements are equivalent to rounds thereby increasing the cipher complexity.

This application of using this algorithm is in mobile nodes or sensor nodes in ad-hoc network where there are limitations on power consumption, resources, fast processing, etc.

My future work on this will focus on making algorithm more dynamic where the indexing has some relation to the key I used for encryption and crypt analysis of the generated cipher.

VI. ACKNOWLEDGMENT

I would like to thank Er. Manuj Verma, Assistant Professor of I.E.R.T for his kind co-operation and technical support rendered by him on making this dissertation a success. I extend my sincere thanks to my advisor Er. Ankit Sharma, for her valuable guidance and assistance throughout my dissertation.

VII. REFERENCES


