



# SIMPLEST AND ROBUST CIPHER TEXT GENERATION BY USING SPLITTING AND MERGING TECHNIQUES

<sup>1</sup>Dr. U. Thirupalu, <sup>2</sup>Mavalluru. Swathi, <sup>3</sup>A. Hemantha Kumar, <sup>4</sup>N. Anil Kumar Chowdari,

<sup>5</sup>Kondisetty Kavitha

<sup>1</sup>Associate Professor, <sup>2</sup>Assistant Professor, <sup>3</sup>Associate Professor, <sup>4</sup>Assistant Professor, <sup>5</sup>Assistant Professor

<sup>1,2,3,4,5</sup> Department of Computer Science and Engineering

<sup>1,2,3,4,5</sup> Audisankara College of Engineering & Technology, Gudur, India

**Abstract:** Ciphertext generation is a process of transforming legible text into illegible text. In this scenario, we develop new techniques, i.e. splitting and merging the plaintext and using various kinds of mathematical operation on the plaintext for generating the cipher text. In this paper, we generate decimal (ASCII) equivalent for the given plaintext, then split the decimal into two parts by performing one of the simplest mathematical operation i.e. divisional operation on derived decimals. After splitting the decimal into two parts, then store them into two different matrices: one is act as Key and another is act as source to generate Cipher text. To generate the cipher text, we may implement simple character generation as cipher or performing XOR operation on the text that is depending on the text which we given as input plaintext. Performing the relevant operation as reverse by merging the two matrix decimals and finding the character equivalent for getting Plaintext from the given Cipher text.

**Keywords:** Plaintext, Split Decimals, Merge Decimals, Like Elements, Unlike Elements, XOR, Cipher text

## I. INTRODUCTION:

Today we are in a technological era. All our activities are done today with the help of various types of technologies which are available. Mostly our data or information are stored or transferred between the devices, we should secure our sensitive data or information by using various kinds of techniques. Cryptography [1] is one of mathematical technique, which is used to shield our data or information. The main purpose of cryptography technique [2] is to protect the confidential information by hiding large binary sequences. To protect our sensitive data or information, there are several types of existing cryptosystems [3]. They are: DES, AES, Blowfish, RSA etc.

Data Encryption Standard (DES) is a block cipher which is used to encrypt data in a block of size 64 bits each as input and produce 64 bits of cipher text as output by using 56 bits key. Now a day it is insecure for modern applications.

Advanced Encryption Standard (AES) [4] algorithm (also known as the Rijndael Algorithm) is a symmetric block cipher algorithm with a block size of 128 bits. It converts these individual blocks using keys of 128, 192, and 256 bits. Once it encrypts these blocks, it joins them together to form the cipher text.

Blowfish is a variable-length, symmetric, 64-bit block cipher. The key size of Blowfish is a variable length of up to 448 bits, making it more secure compared to other encryption algorithms.

RSA algorithm is an asymmetric key [5] cryptosystem. It uses logarithmic functions to keep the working complex enough to withstand brute force and streamlined enough to be fast post-deployment.

Apart from the above existing algorithms, we design and develop a newest and simplest cryptosystem as symmetric that is “Simplest and Robust Cipher Text Generation by Using Splitting and Merging Techniques”. This is the technique to secure the confidential information in different ways by finding the elements as Like-Elements or Unlike-Elements. The new technique which we design and developed is one of the robust technique for the hackers.

## II. METHODOLOGY

In this paper, we generate a simplest and robust cipher text for the given plaintext. In this work, first we find out the dividable elements are like-elements or unlike-elements. If the elements are like elements we done one kind of approach otherwise we done another kind of approach these are shown in the following.

**2.1 Like Elements:** Like elements are commonly referred to elements which belong to the same set or category and share similar properties or characteristics. This concept is often used in discussions about sets, vectors, matrices, or other mathematical structures where elements can be compared or grouped based on their attributes.

For example, in set theory, “like elements” are elements that belong to the same set. In linear algebra, like elements could refer to elements of a vector or matrix that have the same value or properties, such as all the elements in a row or column.

In general, the notion of “like elements” helps us to identify patterns, make comparisons, and perform operations on collections of objects that exhibit similar behavior or properties.

**2.2 Unlike Elements:** Unlike elements are commonly refers to elements that do not share the same properties or characteristics, or do not belong to the same set or category.

The concept of unlike elements is important in mathematics for distinguishing between different objects, structures, or entities, and for understanding how they interact with each other within mathematical systems.

For example, in set theory, unlike elements are elements from different set. In algebra, unlike elements could refer to elements or vectors or matrices that have different values or properties, such as elements from different rows or columns. In group theory, unlike elements are elements of a group that do not have the same identify or inverse.

In out paper, like elements and unlike elements are plays an important role to generate a strongest and robust cipher text.

### 2.3 Operational Procedure: To generate a Simplest and Robust Cipher we follow the following steps:

Step – 1: First we create a matrix which is suitable for storing the given plaintext into the matrix. For this we create N x N matrix by finding the N in the following formula.

$$N = \sqrt{L}$$

In the above formula, L is length of the given input Plaintext. For instance, the string “abcdefghi” is the plaintext. Here the plaintext length is 9. The value of L is 9, so the square root value of 9 is 3. Then the N value is 3 in such a way we create a suitable matrix for storing exact elements which we given the plaintext as input. If the plaintext size lesser than the matrix size then remain cells leave as empty. The following matrix is the example of filling characters of given input plaintext.

|   |   |   |
|---|---|---|
| a | b | c |
| d | e | f |
| g | h | i |

Step – 2: Here, we get the decimal values of the given plaintext. Then store them in to a separate matrix which is equivalent size of the input matrix. The following way shows the process of generating decimals of

|   |   |   |
|---|---|---|
| a | b | c |
| d | e | f |
| g | h | i |

Plain text

|     |     |     |
|-----|-----|-----|
| 97  | 98  | 99  |
| 100 | 101 | 102 |
| 103 | 104 | 105 |

Decimal Equivalent

the given input Plaintext.

Step – 3: Perform modulo operation on the decimal by 10 and store the remainder value into a separate matrix. The matrix which is created through this procedure is act as Key. The following process shows the

|     |     |     |
|-----|-----|-----|
| 97  | 98  | 99  |
| 100 | 101 | 102 |
| 103 | 104 | 105 |

Plaintext Decimals

*MOD 10 =*

|   |   |   |
|---|---|---|
| 7 | 8 | 9 |
| 0 | 1 | 2 |
| 3 | 4 | 5 |

Modulo from Decimals

process of modulo on decimal and generated as Key elements of this Cryptosystem.

Step – 4: Perform divisional operation on the decimal by 10 and store the quotient value into a separate matrix. The matrix which is created through this procedure is act as intermediate code for generating cipher text. The following process shows the process of modulo on decimal and generated as Key elements of this Cryptosystem.

|     |     |     |
|-----|-----|-----|
| 97  | 98  | 99  |
| 100 | 101 | 102 |
| 103 | 104 | 105 |

Plaintext Decimals

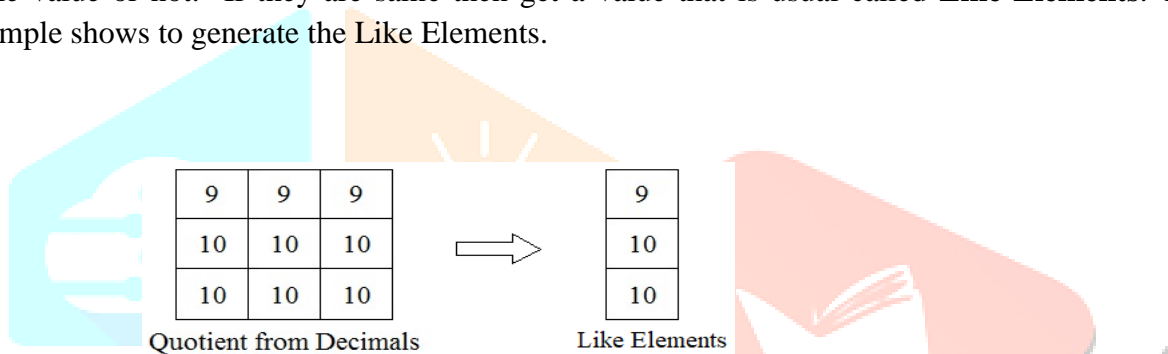
$$DIV 10 =$$

|    |    |    |
|----|----|----|
| 9  | 9  | 9  |
| 10 | 10 | 10 |
| 10 | 10 | 10 |

Quotient from Decimals

Cryptosystem.

Step – 5: Find mean of every column of a row intentionally to decide the contents of the row are having same value or not. If they are same then get a value that is usual called **Like Elements**. The following example shows to generate the Like Elements.



Quotient from Decimals

Like Elements

If we get *Like Elements* successfully then check the length of the plaintext is equivalent to the cells in N x N matrix or not. If they are equal, then just append tow zeros (00) to the last cell of the LikeElements. Otherwise append tow zeros (00) along with actual length of the input plaintext. This feature helps us to recognize the exact length of the plaintext at the time of generating plaintext from the ciphertext at destination end. After appending the tow zeros to the last cell in Like Elements then generate character (ASCII) for the Like Elements value that are act as Cipher text. The following process shows the generation

|    |
|----|
| 9  |
| 10 |
| 10 |

Like Elements

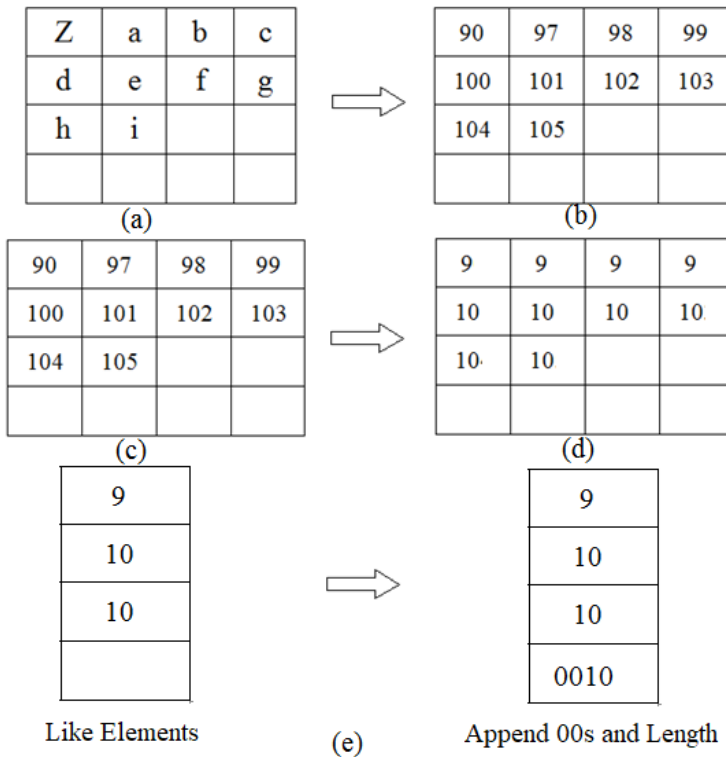


|      |
|------|
| 9    |
| 10   |
| 0010 |

Like Elements with append 00s

of ciphertext for the Like Elements.

If the matrix size is greater than the actual size of the input plaintext then append two zeros and actual length of the input plaintext. It is shown in the following way.



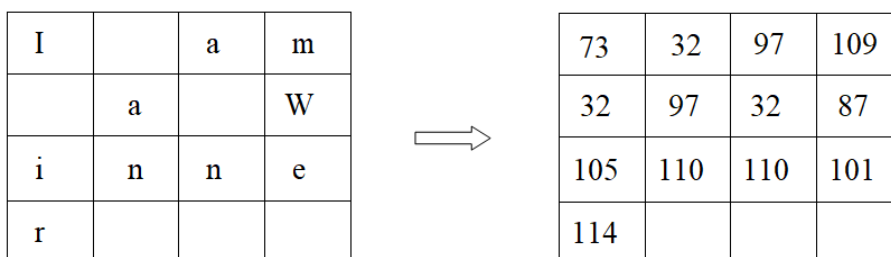
The points which are covered in the above procedure are:

- a) Plaintext with the length 10 character.
- b) Decimals equivalent of the plaintext.
- c) Decimals for generating source of cipher text.
- d) Derived values from the plaintext decimals (performing divisional operation on source)
- e) Final set for generating cipher text.

### 2.4 Substitution Technique:

It is a type of encryption technique where a set of characters or units of text are replaced by others in order to encrypt a text sequence. In this paper we use XOR operation to generate cipher. This substitution technique is applied when the derived set of the matrix is not singleton set. In this situation performing XOR operation on cipher source matrix value with key matrix values for generating cipher text valued matrix as an output matrix, from this output matrix, we generate ASCII characters directly as a ciphertext.

For example, if we take the plaintext as “**I am a Winner**” to generate the ciphertext using splitting and merging technique we place the plain text into the suitable matrix and generate decimal equivalent of the



plaintext then store them into another matrix. It is shown in the following way.

From the decimal values, performing modulo operation, we get the remainder values, and then store them to a separate matrix. It is shown in the following way.

|     |     |     |     |                 |   |   |   |   |
|-----|-----|-----|-----|-----------------|---|---|---|---|
| 73  | 32  | 97  | 109 | <b>MOD 10 =</b> | 3 | 2 | 7 | 9 |
| 32  | 97  | 32  | 87  |                 | 2 | 7 | 2 | 7 |
| 105 | 110 | 110 | 101 |                 | 5 | 0 | 0 | 1 |
| 114 |     |     |     |                 | 4 |   |   |   |

From the decimal values, performing divisional operation, we get the quotient values then store them to a separate matrix. It is shown in the following way:

|     |     |     |     |                 |    |    |    |    |
|-----|-----|-----|-----|-----------------|----|----|----|----|
| 73  | 32  | 97  | 109 | <b>DIV 10 =</b> | 7  | 3  | 9  | 10 |
| 32  | 97  | 32  | 87  |                 | 3  | 9  | 3  | 8  |
| 105 | 110 | 110 | 101 |                 | 10 | 11 | 11 | 10 |
| 114 |     |     |     |                 | 11 |    |    |    |

From the quotient matrix, we perform various mathematical operations to decide the elements are unlike elements unlike or not. Here these are unlike elements. It is shown in the following way.

|    |    |    |    |          |      |
|----|----|----|----|----------|------|
| 7  | 3  | 9  | 10 | <b>⇒</b> | 7.25 |
| 3  | 9  | 3  | 8  |          | 5.75 |
| 10 | 11 | 11 | 10 |          | 10.5 |
| 11 |    |    |    |          | 11   |

Unlike Elements

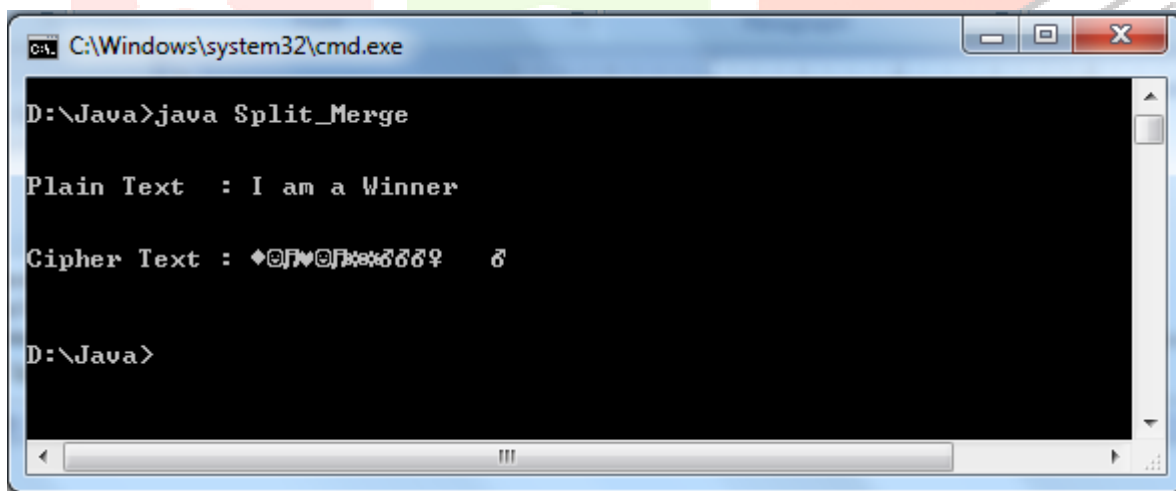
As per our splitting and merging technique, the selected elements are unlike elements. So that we perform an XOR operation on both modulo matrix elements and quotient matrix elements for getting new matrix elements that are source to generate the cipher text. It is shown in the following way.

|   |   |   |   |   |    |    |    |    |   |    |    |    |    |
|---|---|---|---|---|----|----|----|----|---|----|----|----|----|
| 3 | 2 | 7 | 9 | ⊕ | 7  | 3  | 9  | 10 | = | 4  | 1  | 14 | 3  |
| 2 | 7 | 2 | 7 |   | 3  | 9  | 3  | 8  |   | 1  | 14 | 1  | 15 |
| 5 | 0 | 0 | 1 |   | 10 | 11 | 11 | 10 |   | 15 | 11 | 11 | 11 |
| 4 |   |   |   |   | 11 |    |    |    |   | 15 |    |    |    |

The following matrix which is generated by performing XOR operation is the source for generating Ciphertext. From this matrix, we can generate the characters that are act as cipher text.

|    |    |    |      |
|----|----|----|------|
| 4  | 1  | 14 | 3    |
| 1  | 14 | 1  | 15   |
| 15 | 11 | 11 | 11   |
| 15 |    |    | 0013 |

**2.5 Input / Output:** The above task is performed through Java Programming we got a Ciphertext from the given plaintext. The following figure is shown the input and output of the given plaintext that is “I am a Winner”.



```

C:\Windows\system32\cmd.exe
D:\Java>java Split_Merge
Plain Text : I am a Winner
Cipher Text : ◆@J▼@J▼x8889 8
D:\Java>

```

Figure 1: Generation of cipher from the given plaintext

### III. CONCLUSION

In this paper, we generate a simplest and robust cipher text. Which is more notorious for the hackers because our newly created methodology follows two kinds of procedure one is like elements another is unlike elements. Through this new approach, the proposed system may use either of the above. So that the hacker may not be get the actual plaintext from the generated cipher text. Another strongest thing which we introduced in this paper is a more secure against the brute force attack.

### REFERENCES

- [1] Pushendra Tiwari and Dr MD. Vaseem Naiyer, The Brief Review of Cryptography Techniques For The Cloud Data Security, IJCRT, Volume 11, Issue 8, August 2023.
- [2] T. Sangeetha, Analysis of Tools And Technologies In Cryptography, IJCRT, Volume 11, Issue 12, December 2023.
- [3] Monika Khetarpal, Overview of Cryptography in Network Security, IJCRT, Volume 11, Issue 2, February 2023.
- [4] Smruthi Ranjan Pradhan at all, A Review on Cryptography, IJCRT, Volume 10, Issue 3, March 2022.
- [5] Tushar, Anikat Sharma and Ankit Mishra, Cryptographic Algorithm for Enhancing Data Security: A theoretical Approach, IJERT, Vol. 10, Issue 03, March - 2021

