IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

CLOUDE MIRRORING A TECHNIQUE OF DATA RECOVERY USING ETSFS ALGORITHM

[1]Soniya S. Agrawal, Dr. vijay S. Gulhane, Mr. Harshal N. Datir

[1]Student, [2]Professor,

[1]Computer Science & Engineering,

[1]Sipna college of Engineering & Technology, Amravati, India

Abstract: - Database encryption is a security system includes different encryption algorithms for database security. Most of the Organization stores their information in enormous databases that empowers uncomplicated recovery, controls, and furthermore helps in a proficient method for sharing. Database security has now turn into a more dynamic issue as information is the best advantage for any association. Because of the quick increment in the database issue as data is the great asset to any organization. To beat these issues, various security strategy have developed to ensure the information in databases. For Example Enhanced Transposition, Substitution, Folding & Shifting (ETSFS). Each of them has its particular benefits and faults. Using the ETSFS algorithm to secure sensitive data (information) in a databases .It has imperative on information size and number of exceptional characters, the proposed technique change concentrates on the encryption of expansive information considering a wide range of unique characters and an arbitrary generator is utilized for producing keys in substitution stage. The proposed strategy of the paper concentrated on the future work of the ETSFS algorithm for secure sensitive Email system.

Index Terms - Transposition, Substitution, Shifting, Folding, Encryption, Decryption.

I. INTRODUCTION

Information security has reliably been a noteworthy issue in web applications. Database security has vital significance in organization, military, personnel and government areas. Associations are putting away huge measure of information in database for information mining and different sorts of investigation. Some of this information is viewed as delicate and must be shielded from exposure. Challenges for security in database are expanded because of the colossal prevalence of e-business. As of late, insider assault assemble more consideration than occasional episodes of malware. Database frameworks are normally conveyed somewhere inside the organization system and subsequently insiders has the most effortless chance to assault and trade off them, and after that take the information. So information must be shielded from inside aggressors too. Numerous traditional database security frameworks are proposed for giving security to database, yet at the same time the delicate information in database are helpless against assault on the grounds that the information are put away as plaintext only. Within the sight of security dangers, database security is getting to be noticeably a standout amongst the most earnest difficulties since much harm to information can happen in the event that it experiences assaults and unapproved get to. With databases in complex, multi-layered applications, assailants may achieve the data inside the database. Harm and abuse of delicate information that is put away in database does influence solitary client; as well as conceivably a whole association.

Database encryption would be one of the possible solutions where be one of the access to sensitive information is dependent on the key available, which promises a minimal damage and high performance when it is effective.

The ETSFS algorithm provide by avoid the constraint on the data size and special characters by proposing the usage of all special characters on the data size. This improvement allow handling all special characters and different sizes of input data for processing

performed dynamically depending on the input data given by the user; and based on the input size, keys are generated that shows a variation from existing DES algorithm. It showed a successful implementation by accepting almost all special characters.

II. LITERATURE REVIEW

Data plays a very important role and is stored in database system which should be organized such that it safeguards the data. Most of the organizations sensitive data is housed in database and a backup is maintained for future use. Unauthorized access is one of the serious threat it should be addressed to enhance database security. Encryption, which plays a important role in safeguarding the information, is defined as the process of transforming information into no readable form except by those holding a key to decrypt. The database security mechanism, algorithms like TSFS, DES, and AES came into focus, which are different from other and had a few advantages and disadvantages based on their optimization ways. The DES algorithm is one of the well known symmetric key algorithms considered as insecure for many applications and presents AES as a replacement. D. Manivannan, R. Sttjarani proposed well-organized encryption technique using the symmetric - key. popularly known as TSFS algorithm, which includes transpositions and substitutions as features in the techniques that limits encryption and decryption operation times. Later an enhanced TSFS is proposed which is an extended work on TSFS which can encrypt the data that contains alphanumeric and few special characters ensuring high level of security to encrypted data but imposes few constraints on the data size and the special characters used. The cloud computing is the top threat identified by the Cloud Security Alliance. Attacker can infiltrate a public cloud. For providing effective and more security for the database these three keys are expanded in to 12 sub keys by using the key expansion technique. The principle quality of calculation is in the substitution change in light of the fact that choosing the key for discovering figure gave greater security to the encode picture. Pictures are in database are encode and afterward the encode pictures are taken as information. In this calculation the numeric plaintext have numeric figure, character plaintext have character figure if the information is alphanumeric sort then the yield figure message additionally alphanumeric, so there is no requirement for change the information field sort the encoded information are put away in the database. Lightweight cryptography is a relatively the new field aimed to develop more efficient cryptographic implementation in response to typical constraints in the hardware used in Internet of Thing (IoT).

III. PROPOSED WORK

3.1 Architecture Design

The Primary Purpose of Encryption is to ensure the certainly of advance information put away on PC frameworks or transmitte through the Internet or other PC systems. Present day encryption calculation assume a fundamental part in the security confirmation of IT framework and correspondence as they can give classification, as well as the accompanying key components of security. Information frequently alluded to as plaintext is encoded utilizing an encryption calculation and an encryption key. This is procedure creates figure message that must be seen in the first frame if unscrambled with the right key. Unscrambling is essentially the backward of encryption, taking after similar strides however turning around the request in which the keys are connected.

In general the encryption process for securing data that other user can not access or read that information may be images, audio or plain text data by selecting the images, audio, or text file that contents are converted into the binary character or ASCII code

Main objective of this proposed work is to protection of information that is text, Special character or numbers.

Proposed system has two main phases.

A. Encryption Method

B. Decryption Method

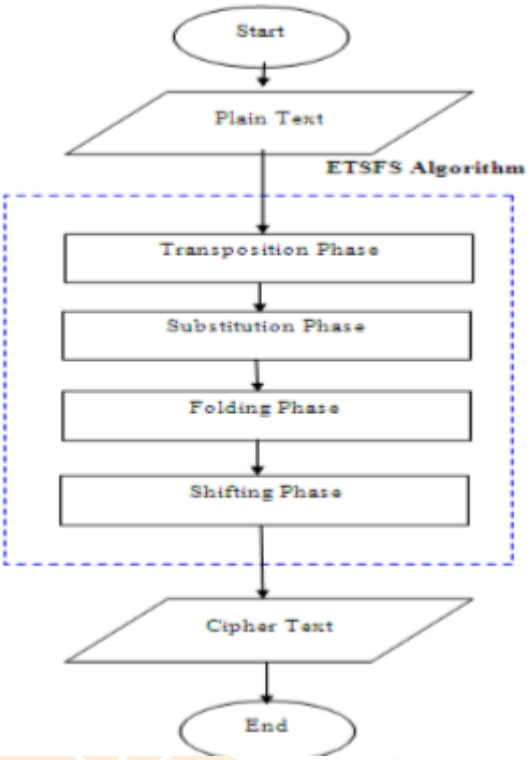

Fig.3.1.1 Encryption Method

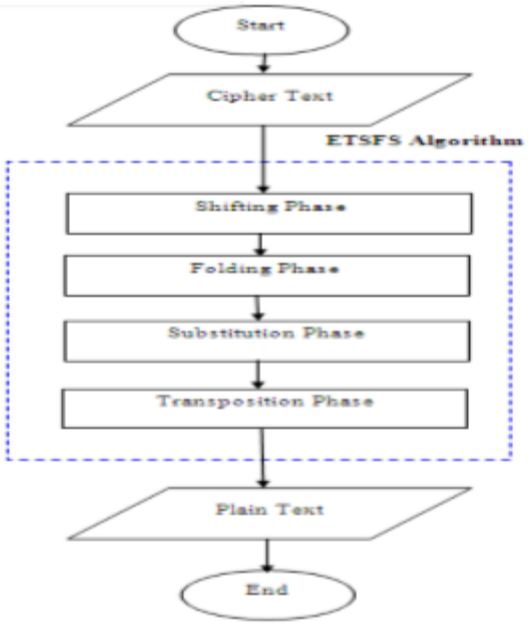

Fig.3.1.2 Decryption Method

3.2 METHODOLOGY

The proposed algorithm involves the collection of new key values to consider randomly. The ETSFS algorithm which have four phases.

1) Transposition Phase 2) Substitution Phase 3) Folding Phase 4) Shifting Phase.

The ETSFS algorithm includes the following alphanumeric characters and a few of special symbols (*, -, /, :, @ and _) only used to encrypt the data that is taken as input. But in the proposed methodology it included almost all the special characters. It is a symmetric encryption algorithm which can be inversed that cancels the encryption.

The constrained restricted on the number of characters is successfully imposed by accepting different data sizes dynamically depending on the user input length and if the length of data is less than the near square matrix size then the characters are replaced by *'s. Let say, the input string

is 14 in length, then the nearby square matrix is 4x4 and the one character that is left is replaced by *'s. The four techniques of ETSFS are described as:

Algorithm encryption (String data, Array[12] keys)

Pre: data is plain text.

keys is array that contains 12 4x4-key matrices.

Post: encryptedData is data after encrypting. Matrix[4,4] dataMatrix;

String encryptedData; if (data length < 16)

padd data by adding *'s; else if (data length > 16) cut the data

after 16; end if

dataMatrix = data;

key = expandKeys (keys); for (int i=0; i<12; i++)

dataMatrix = transposition (dataMatrix);

dataMatrix = substitution (dataMatrix, keys(i), keys((i+1)mod 12)); dataMatrix = folding (dataMatrix);

dataMatrix = shifting (dataMatrix); end for

encryptedData = dataMatrix; return encryptedData

End encryption

3.2.1 Transposition Phase :

Transposition transformation changes the location of the data matrix elements by using diagonal transposition that reads the data matrix in the route of zigzag diagonal starting from the upper left corner after getting the data and pads it with *s if it is less than 16 digits. Fig. 4.2.1 shows the transposition process when they entered data was: 6923@domain.Sa, the transposition algorithm in encryption side

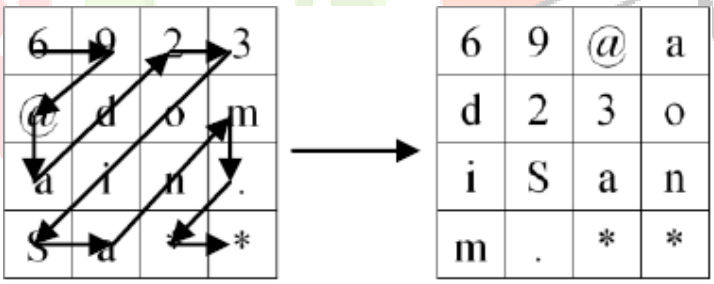

Fig 3.1 Transposition Phase Example

Algorithm transposition (Matrix data)

Pre: data is 4x4 matrix that contains the data should be encrypted.
Post: data is data after changing symbols location.
Matrix temp;

temp[0,0] = data[0,0];
temp[0,1] = data[0,1];
temp[0,2] = data[1,0];
temp[0,3] = data[2,0];
temp[1,0] = data[1,1];
temp[1,1] = data[0,2];
temp[1,2] = data[0,3];
temp[1,3] = data[1,2];
temp[2,0] = data[2,1];
temp[2,1] = data[3,0];
temp[2,2] = data[3,1];
temp[2,3] = data[2,2];
temp[3,0] = data[1,3];
temp[3,1] = data[2,3];

```
temp[3,2] = data[3,2];
temp[3,3] = data[3,3];

data = temp; return data;
End transposition
```

3.2.2: Substitution Phase:

The second algorithm is substitution transformation. It replaces one data matrix element with another by applying certain function. If the element represents an alphabetic character, it then will be replaced with another character. If the element represents a number, it will be replaced with a number,. Confusion happens if the data is composed of alphabetic and numeric digits, and the modulus size (M) will be 26 for any digit, as illustrated in the next example. If one element in the data was 4, k1=5, k2=5, M = 26, then the result of substitution process is 14 as the paper presents.

This result causes two problems. The first problem, is that the length of the data will be changed and increased; for example, when the plan text size is 16 digits, the cipher text size will be 17 digits if one element only changes, and that contradicts the TSFS algorithm's feature. The second problem, since the inverse operation decrypts the data digit by digit also, is that then it will deal with each element in the cipher text individually (1 then 4). As a result, the decrypted data will be different from the data that have been encrypted. Therefore, the ETSFS algorithm gives M the following values: 26 if p is aliphatic, 10 if p is numerical and 7 if p is symbolic.

The decryption function [1] D is:

$$D(E(x)) = (((E(x) - k2) \bmod M) - k1) \bmod M \quad (2)$$

Since most of the programming languages such as Java and C++ deal with the modulus as the remainder of an integer division, some of the results may have minus sign, and this will create a problem because there is no data that have minus sign representation. So, one more step has been added to the ETSFS algorithm implementation to check if the result includes the minus sign, and then apply:

$$D(E(x)) = M - |D(E(x))| \quad (3)$$

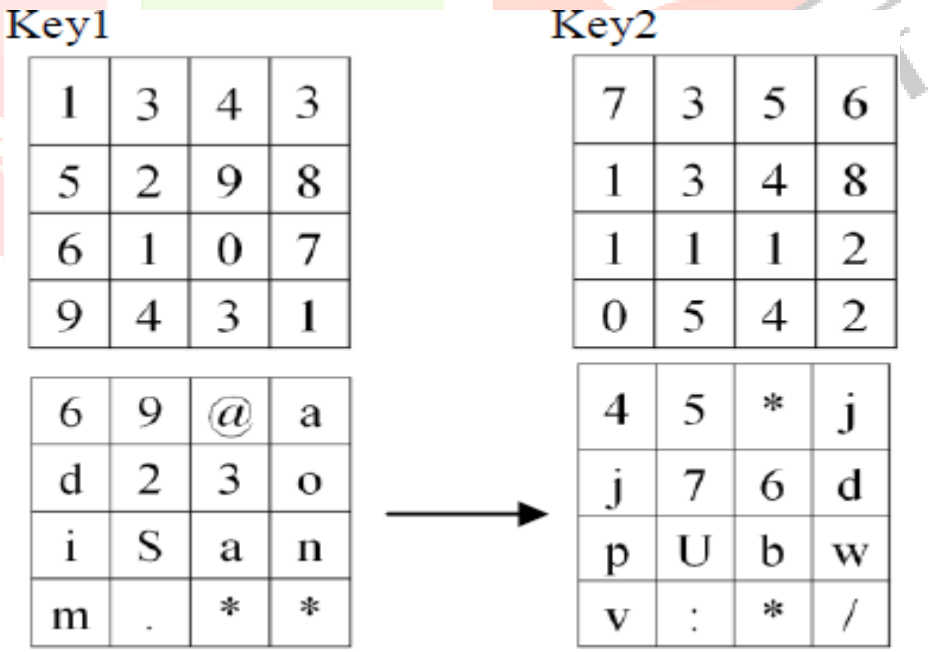

Fig 3.2: Substitution Example

Algorithm substitution (Matrix data, Matrix key1, Matrix key2)

Pre: data is 4x4 matrix.

Key1 and key2 are 4x4 matrix used to encrypt data.

Post: data is data after applying substitution encryption method. Matrix temp;

int M;

```
for (int i=0; i<4; i++) for (int j=0; j<4; j++)
        if (data[i,j] is alphabet) M=26;
```

4	5	*	j
j	7	6	d
p	U	b	w
v	:	*	/

→

/	:	*	v
d	b	U	j
w	6	7	p
j	5	*	4

else if (data[i,j] is number) M=10;

else if (data[i,j] is symbol) M=7;

end if

temp[i,j]= (((k1[i,j]+ numric(data[i,j]) mod M)+k2[i,j]) mod M; end for

end for

data = temp; return data;

End substitution

3.2.3: Folding Phase :

The third algorithm is folding transformation. It shuffles one of the data matrix elements with another in the same entered data, like a paper fold. The data matrix is folded horizontally, vertically and diagonally [1]. The horizontal folding is done by exchanging the first row with the last row. The vertical one is done by exchanging the first column with the last column. The diagonal fold is done by exchanging the inner cells, the upper-left cell with the down-right cell and the upper-right cell with the down- left cell.

4	5	*	j
j	7	6	d
p	U	b	w
v	:	*	/

→

/	:	*	v
d	b	U	j
w	6	7	p
j	5	*	4

Fig 3.3: Folding Example 1

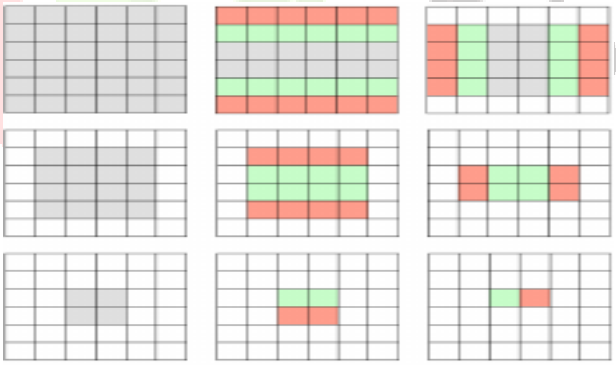

Fig 3.4: Folding Example 2

Algorithm folding (Matrix data)
Pre: data is 4x4 matrix of data get from substitution technique.
Post: data is data matrix after applying folding technique.

```
Matrix temp;
temp[0,0] = data[3,3];
temp[0,1] = data[3,1];
temp[0,2] = data[3,2];
temp[0,3] = data[3,0];
temp[1,0] = data[1,3];
temp[1,1] = data[2,2];
temp[1,2] = data[2,1];
temp[1,3] = data[1,0];
temp[2,0] = data[2,3];
temp[2,1] = data[1,2];
temp[2,2] = data[1,1];
temp[2,3] = data[2,0];
temp[3,0] = data[0,3];
temp[3,1] = data[0,1];
temp[3,2] = data[0,2];
temp[3,3] = data[0,0];
data = temp;
return data;
```

End folding

3.3.4: Shifting :

The last part of the algorithm is the shifting transformation, which provides a simple way to encrypt using a 16-array element of numeric digits to exchange a letter with another. Each element of the array must contain the numeric representation of the data. Each digit must appear only once in each element of the array. The digits can appear in any order. In shifting process, the algorithm replaces each element in the data matrix by its position within its array element.

The ETSFS algorithm uses four 16-arrays instead of one array as the TSFS algorithm uses, because the described shifting process in has confusion. For example, if an element in the plain text is 4 and its position within the array is 15, then the shifting process in returns 15, which is causing the same two problems that were described in substitution transformation. So, the ETSFS algorithm separates each type from other.

The ETSFS algorithm uses four 16-arrays, one for numeric, one for symbols, but because it is difficult to enumerate all symbols in this project; the suggested ETSFS algorithm considers only two types of symbols. Symbols that are used in emails (-, ., @, _) and symbols that are used in IP addresses (/, :). The last two 16-arrays are used for alphabetic, where one for capital letters and the other for small letters. We used that to enhance TSFS algorithm and make it is sensitive for the type of letter.

I/P	Array Element	O/P
/	0 1 2 3 4 5 6	/
:	1 2 3 4 5 6 0	/
*	2 3 4 5 6 0 1	@
v	3 4 5 6 7 8 9 10 11 12 13 14 15 ... 23 24 25 0 1 2	s
d	4 5 6 7 8 9 10 11 12 13 14 15 16 ... 24 25 0 1 2 3	z
b	5 6 7 8 9 10 11 12 13 14 15 16 ... 24 25 0 1 2 3 4	w
U	6 7 8 9 10 11 12 13 14 15 16 ... 24 25 0 1 2 3 4 5	O
.	.	.
.	.	.
4	1 5 4 6 0 7 2 8 3 9	2

Fig 3.5: Shifting Example

IV. RESULTS AND DISCUSSION

In this paper we have Implemented ETSFS algorithm technique that prevents users from inferring sensitive information from database. By using ETSFS algorithm (Enhanced Transposition , Substitution, Folding, Shifting), we protected the sensitive information into the cipher text. But data will decrypt only the reverse process of (Shifting, Folding, substitution, Transposition) ETSFS algorithm and get the original text or messages. When the both semantic inference of encryption side and decryption side 100 % collaborate or matched with each other. We concluded that the best way to this approach of securing sensitive data is by using encryption techniques and ensuring database security from attackers.The Enhanced TSFS algorithm methodology is explained in which security is ensured in databases by simultaneously increasing the performance of encryption and decryption process.

V. ACKNOWLEDGMENT

WE WOULD LIKE TO EXPRESS OUR DEEPEST APPRECIATION TO THOSE WHO PROVIDED US THE POSSIBILITY TO COMPLETE THIS PAPER. A SPECIAL GRATITUDE WE GIVE TO OUR GUIDE DR. V. S. GULHANE, WHOSE CONTRIBUTION IN SIMULATING SUGGESTIONS AND ENCOURAGEMENT HELPED US TO COORDINATE OUR PROJECT ESPECIALLY IN WRITING THIS PAPER.

REFERENCES

[1] Prathyusha Uduthalapally,Bing Zhou “Improvement of ETSFS Algorithm for secure Database” 4th international Symposium on Digital Forensics & Security (ISDFFS16)25-27 April 2016 Little Rock,AR

[2] Hanan A, Abeer, Heba, “Lightweight Symmetric Encryption Algorithm for Secure Database.” IJACSA International Journal of Advanced Computer Science and Applications, Saudi Arabia.

[3] D. Manivannan, R. Sttjarani, Light weight and secure database encryption using TSFS algorithm. Proceedings of the International Conference on Computing Communication and Networking Technologies, 2010, pp. l-7.

[4] H. Alanazi, B. Zaidan, A. Zaidan, H. Jalab, M. Shabbir, Y. Al-Nabhani. New comparative study between , 3DES and AES within nine factors, Journal of Computing 2 (2010) 152-157.

[5] Pooja, Kanchan Narula “Enhancing Data Security in Cloud Computing with WebOS Using TSFS Algorithm ”ISSN : 2319-7064.

[6] Amandeep kaur1, Mrs. Shailja Kumari “Secure Database Encryption in Web Applications” IJARCCE ,vol. 3 issue 7, july 2014.