An Approach Towards Design And Implementation Of Symmetric Encryption Techniques

Pawan Kumar Jha
Lecturer
Purbanchal University, Biratnagar, Nepal.

Supervisor
Dr.J.K.Mandal
Reader, Dept. of Computer Science&Application.

An overview of proposed techniques

This section provides an overall idea on all the techniques proposed in the thesis. Following are the proposed names of the set of independent techniques developed during the research work:

1. **Recursive Carry Addition (RCA)**
2. **Recursive Key Rotation (RKR)**
3. **Recursive Session Key Arithmetic (RSKA)**
4. **Cascaded Arithmetic Operation on Pair of Bits of Streams (CAOPB)**
5. **Recursive Modulo-2 Operation of Paired Bits of Streams (RMOPB)**
6. **Cascaded Recursive Key Rotation of a session key with Transposition and Addition of Blocks (CRKRTAB)**

Before pointing out the overview of each technique separately, a combined overview of all the techniques has been summarized in table 1.4. The table points out schematic characteristics of the techniques in terms of the following attributes:

- Whether bit-level implementation or not
- Whether public key system or private key system
- Whether techniques of encryption and decryption exactly the same (symmetric)
- Whether block cipher or stream cipher
- Whether a substitution technique or a transposition technique
- Whether the basic operation is Boolean or Non-Boolean
- Whether there is data compression/expansion or no alteration in size
- Whether there is any formation of cycle or not
- Whether the technique cascaded with more than one techniques

Table 1.4
Schematic characteristics of proposed techniques

	RCA	**RKR**	**RSKA**	**CAOPB**	**RMOPB**	**CRK RTAB**
Implementation in bit-level	√	√	√	√	√	√

Implementation not in bit-level						
Public key system						
Private key system	√	√	√	√	√	√
Symmetric	√	√	√	√	√	√
Block cipher		√	√	√	√	√
Stream cipher						
Substitution technique	√	√	√	√	√	√
Transposition technique						√
Boolean as basic operation		√	√	√	√	√
Chance of alteration in size				√		
No alteration in size	√	√	√		√	√
Formation of cycle	√	√	√	√	√	√
No formation of cycle						
Cascaded with more than one technique						√

From table 1.4, the following attributes that are common to all the proposed techniques:

- **Implementation in bit-level**
- **Private key system**
- **Symmetric**
- **Block cipher**
- **Formation of cycle**

For all the techniques there is no chance of any alteration (increase) in size while encrypting the source file except CAOPB technique.

Since all the techniques are block ciphers and have to be implemented in bit-level, the source file to be encrypted is to be converted into a stream of bits and the whole stream is to be decomposed into a finite number of blocks before the actual scheme is to be implemented for each block.

Section 1.5.1 to section 1.5.6 gives an overview for all the proposed techniques. Section 1.5.7 presents a brief proposal on the key structures of different proposed techniques. Section 1.5.8 points out the factors considered in the thesis for each of the proposed techniques for the purpose of evaluation.

An overview of the RCA technique [118]

This technique considers the plaintext as block of bits with different size like 4/ 8 /16 /32 /64 /128 /256. The rules to be followed for generating a cycle are as follows:

1 Consider any source stream of a finite number (where N=2^n, n =2 to 8).

1. Add 1(one) to the left most bit of the string with binary addition method. If there is carry, add this carry to the second bit of the source string and continue up to the last bit to get the first intermediate block.
2. Again add 1(one) to the second bit of the source stream with the binary addition method, and get the second intermediate block with the same carry addition to its respective places, continued at the last.
3. This addition process is continued till the last bit of the given string. If there is a carry at the last bit then simply add that carry to the first place of the respective iteration and continued up to the last.

This process is repeated until the source stream is generated. After a finite number of iteration the source stream is regenerated. Any intermediate block may be considered as the encrypted block for the technique.

It is a kind of block cipher and symmetric in nature hence, decoding is done following the same procedure. A comparison of the proposed technique with existing and industrially accepted RSA/TDES has also been done in terms of frequency distribution and homogeneity of source and encrypted files.

Chapter 2 of the thesis discusses the RCA technique in detail from different perspectives.

An overview of the RKR technique [119]

This technique considers the plaintext as block of bits with different size like 4/ 8 /16 /32 /64 /128 /256. The rules to be followed for generating a cycle are as follows:

1 Consider any source stream of a finite number (where N=2^n , n =2 to 8) and divide it into two equal parts.

2 Consider any key value (key= 2^n, where n=1 to 7) depends upon the source stream that is, key value is the half of the source stream).

3 Make the modulo-2 addition (X-OR) with the key value to the first half of the source stream, to get the first intermediate block.

4 Make the modulo-2 addition (X-OR) with the key value (but now the key value is reversed) to the last half of the source stream to get the second intermediate block.

This process is repeated until the source stream is generated. After a finite number of iteration the source stream is regenerated. Any intermediate block may be considered as the encrypted block for the technique.

It is a kind of block cipher and symmetric in nature hence, decoding is done following the same procedure. A comparison of the proposed technique with existing and industrially accepted RSA/TDES has also been done in terms of frequency distribution and homogeneity of source and encrypted files.

Chapter 3 of the thesis discusses the RKR technique in detail from different perspectives.

An overview of the RSKA technique [120]

The technique considers the plaintext as block of bits with different size like 4/ 8 /16 /32 /64 /128 /256. The rules to be followed for generating a cycle are as follows:

Step 1: Apply step 3 and step 4 exactly L number of times, for the values of the variable P ranging from 0 to (L-1) increasing by 1 after each execution of the loop.

Step 2: Apply arithmetic operation on the first two bit of the source stream with the two bit fixed key to get the first intermediate block.

Step 3: Apply Arithmetic operation again on the second two bit of the source stream with the two bit fixed key to get the second intermediate block.

When this process is continued, source block may be generated after a finite number of iteration. Any intermediate block may be considered as the encrypted block for the technique. It is a kind of block cipher and symmetric in nature hence, decoding is done following the same procedure.

A comparison of the proposed technique with existing and industrially accepted RSA/TDES has also been done in terms of frequency distribution and homogeneity of source and encrypted files.

Chapter 4 of the thesis discusses the RSKA technique in detail from different perspectives.

1.5.4 An overview of the CAOPB technique [121]

A stream of bits is considered as the plaintext. Like in other proposed techniques, the plaintext is divided into a finite number of blocks, each having a finite fixed number of bits like 8/ 16/ 32/ 64/ 128/ 256. The CAOPB technique is then applied for each of the blocks in the following way. The rules to be followed for generating cycles are as follows:

1. Consider any source stream of a finite number (where $N=2^n$, n =3 to 8) and divide it into two equal parts.
2. Make the source stream into paired form so that a pair can be used for the operation.
3. Make the modulo-2 addition (X-OR) between the first and second pair, second and third pair, third and fourth pair of the source stream, to get the first intermediate block.

This process is repeated until the source stream is generated. After a finite number of iteration the source stream is regenerated. Any intermediate block may be considered as the encrypted block for the technique. It is a kind of block cipher and symmetric in nature hence, decoding is done following the same procedure.

A comparison of the proposed technique with existing and industrially accepted RSA/TDES has also been done in terms of frequency distribution and homogeneity of source and encrypted files.

Chapter 5 of the thesis discusses the CAOPB technique in detail from different perspectives.

An overview of the RMOPB technique [122]

The technique, considers the plaintext as a stream of finite number of bits N, and is divided into a finite number of blocks, each also containing a finite number of bits n, where, $1<= n <= N$.

Let P = $s^0_0\ s^0_1\ s^0_2\ s^0_3\ s^0_4 \ldots s^0_{n-1}$ is a block of size n in the plaintext. Then the first intermediate block $I_1 = s^1_0\ s^1_1\ s^1_2\ s^1_3\ s^1_4 \ldots s^1_{n-1}$ can be generated from P in the following way:

$s^1_0\ s^1_1 = s^0_0 s^0_1 \oplus s^0_2 s^0_3$

$s^1_2\ s^1_3 = s^0_0 s^0_1 \oplus s^0_4\ s^0_5$

$s^1_i\ s^1_{j+1} = s^0_{i-j}\ s^0_{i-j+1} \oplus s^0_{i+j+2}\ s^0_{i+j+3}$, $0 <= i < (n-1)$, $0 <= j < (n-1)$; $\oplus$ stands for the exclusive-OR operation.

In the same way, the second intermediate block $I_2 = s^2_0\ s^2_1\ s^2_2\ s^2_3\ s^2_4 \ldots s^2_{n-1}$ of the same size (n) can be generated by:

$s^2_0 s^2_1 = s^1_0 s^1_1 \oplus s^1_2 s^1_3$

$s^2_2\ s^2_3 = s^1_0 s^1_1 \oplus s^1_4\ s^1_5$

$s^2_i\ s^2_{j+1} = s^1_{i-j}\ s^1_{i-j+1} \oplus s^1_{i+j+2}\ s^1_{i+j+3}$, $0 <= i < (n-1)$, $1 <= j < (n-1)$;$\oplus$ stands for the exclusive-OR operation.

If this process continues for a finite number of iterations, the source block P is regenerated forming a cycle, which depends on the value of block size n.

Any intermediate block in the recursive process may term as intermediate encrypted block for that source block. The operation is repeated for the whole stream in the source.

The same operation is performed for whole stream number of time with a varying block sizes. k such iteration is done and the final intermediate stream after k iterations generates the encrypted stream. All of the different block sizes and k constitute the key for the session. This key may be considered as session key for that particular session.

An overview of the Cascaded Recursive Key Rotation of a session key with Transposition and Addition of Blocks (CRKRTAB) [123]

This technique operates in three phases:

a. First phase encrypt the plaintext using Recursive Key Rotation

The technique considers the plaintext as a stream of finite number of bits in the form of blocks with different size like 8/ 16/ 32/ 64/ 128/ 256. The rules to be followed for generating a cycle are as follows:

1. Consider any source stream of a finite number (where $N=2^n$, n =3 to 8) and divide it into two equal parts.
2. Consider any key value (key= 2^n, where n=1 to 7) depends upon the source stream that is, key value is the half of the source stream).
3. Make the modulo-2 addition (X-OR) with the key value to the first half of the source stream, to get the first intermediate block.
4. Make the modulo-2 addition with the key value (but now the key value is reversed) to the last half of the source stream to get the second intermediate block.

b. Second phase encrypt the output of the first phase by Recursive Transposition of Blocks

The technique, considers the encrypted message from the first phase as blocks of bits with different size like 8/ 16/ 32/ 64/ 128/ 256. The bit swapping can be applied to each block separately. The principle of bit transposition is discussed in following for different block size.

a. Swapping on 8 bit:

Considering a block (X) with eight binary bits, we have

X = a b c d e f g h	
X 1 = c d a b g h e f	after 2 bit transposition on X
X 2 = g h e f c d a b	after 4 bit transposition on X_1
X 3 = e f g h a b c d	after 2 bit transposition on X_2
X 4 = a b c d e f g h	after 4 bit transposition on X_3

b. Swapping on 16 bit:

Considering a block (X) with eight binary bits, we have

X = a b c d e f g h i j k l m n o p	
X 1 = c d a b g h e f k l i j o p m n	after 2 bit transposition on X
X 2 = g h e f c d a b o p m n k l i j	after 4 bit transposition on X_1
X 3 = o p m n k l i j g h e f c d a b	after 8 bit transposition on X_2
X 4 = m n o p i j k l e f g h a b c d	after 2 bit transposition on X_3
X 5 = i j k l m n o p a b c d e f g h	after 4 bit transposition on X_4
X 6 = a b c d e f g h i j k l m n o p	after 8 bit transposition on X_5

8 bit, 16 bit, transpositions are applied on the block and the same process is followed to get back the original block. Swapping on 32, 64, 128 and 256 follows the same principle as described above.

c. Third phase encrypt the output of the second phase by Recursive Addition of Blocks

The technique, considers the encrypted message from the second phase as a stream of finite number of bits N, and is divided into a finite number of blocks, each also containing a finite number of bits n, where, 1<= n <= N.

Let P = $s^0_0\, s^0_1\, s^0_2\, s^0_3\, s^0_4 \ldots s^0_{n-1}$ is a block of size n in the plaintext. Then the first intermediate block $I_1 = s^1_0\, s^1_1\, s^1_2\, s^1_3\, s^1_4 \ldots s^1_{n-1}$ can be generated from P in the following way:

$s^1_0 = s^0_0 \oplus s^0_1$

$s^1_{n-1} = s^0_{n-1}$

$s^1_i = s^0_i \oplus s^0_{i+1}$, $1 < i < (n-1)$; $\oplus$ stands for the exclusive-OR operation.

In the same way, the second intermediate block $I_2 = s^2_0\, s^2_1\, s^2_2\, s^2_3\, s^2_4 \ldots s^2_{n-1}$ of the same size (n) can be generated by:

$s^2_0 = s^1_0 \oplus s^1_1$

$s^2_{n-1} = s^1_{n-1}$

$s^2_i = s^1_i \oplus s^1_{i+1}$, $1 < i < (n-1)$. $\oplus$ stands for the exclusive-OR operation.

Any intermediate block in the recursive process may term as intermediate encrypted block which can be used as cipher text for security for 256 bit block string.
The same operation is performed for whole stream number of time with a varying block sizes. K such iteration is done and the final intermediate stream after k iterations generates the cascaded form of the encrypted stream. All of the different block sizes and k constitute the key for the session. This key may be considered as session key for that particular session. This process is repeated until the source stream is generated.

Chapter 7 of the thesis discusses the CRKRTAB technique in detail from different perspectives.

A proposal on key structures for proposed techniques

Before proposing key structures of different proposed techniques, all the proposed techniques are categorized in following three ways [37, 38, 50, 51, 61, 90, and 96].

- **Block cipher with direct block-to-block conversion:** Technique(s) in which the task of encryption is done block wise and for a block of length N, all the N bits of the corresponding encrypted block are placed contiguously, so that the one-to-one relationship between the source block and the encrypted block can be established.
  Example: The RCA, and the RSKA techniques
- **Block cipher with non-contiguous bit-allocation:** Technique(s) in which the task of encryption is done block wise but for a block of length N, different resultant bits are not placed contiguously, so that no one-to-one or one-to-many relationship between the source block and the corresponding encrypted block can be established.
  Example: The RKR, and the CAOPB techniques
- **Block cipher with repeated block-to-block conversion:** Technique(s) in which the task of encryption is done block wise and for a block of length N, after a finite number of specific iterations, the source block is regenerated, so that a cycle is formed. The one-to-many relationship between the source block and the encrypted block can be established as any of the intermediate blocks can be considered as the corresponding encrypted block.
  Example: The RMOPB and the CRKRTAB techniques

Proposed key structure for block cipher with direct block-to-block conversion

To ensure a better security, varying lengths can be chosen for different blocks, and accordingly, the session key is to be structured, so that from the key, one gets the information regarding the lengths of different blocks. Once this information becomes available, the task of decryption can be performed easily to generate the exact source stream of bits. Naturally, this key is to be kept absolutely secret and hence the technique in this category is a **private key system**. Having different block lengths causes the key space to be reasonably large.

Proposed key structure for block cipher with non-contiguous bit-allocation

Here the scenario is the same as described in section 1.5.7.1. Although the techniques choosing varying block lengths will cause the process of encryption as well as decryption a bit more complicated, but, it can be handled properly, and this can produce the security of the highest level. Obviously, the key will consist of the information on the different blocks and, hence, it should be kept secret. As a result, these are **private key systems**.

Proposed key structure for block cipher with repeated block-to-block conversion

In this case, for each of the blocks its length as well as the identification value of the intermediate block considered as the encrypted block is to be placed in the key. The key is to be made secret, so that the techniques falling under this category are **private key systems**.

More about the key structures of these different proposed techniques are discussed and analyzed in the respective chapters and also in chapter 8, which is especially based on the key distribution.

Factors considered for evaluating proposed techniques

Several factors have been considered to evaluate the proposed techniques. These include the following:

- **Frequency distribution test**
- **Chi square test**
- **Analysis of the key space**
- **Computation of the encryption/decryption time**
- **Comparison of performance in terms of Chi square values with the RSA/TDSA technique**

Frequency distribution test

Frequency distribution test is considered to asses the degree of security of the proposed techniques against the cryptanalytic attack. Through this test, performed simultaneously on the original as well as the encrypted files, the frequencies of all 256 characters in two files are compared graphically. These are shown in two different graphs. It is the objective of the frequency distribution test to show that there exists no fixed relationship between the frequency of a character in the source file and that of the same character in the corresponding encrypted file, and that the characters are well distributed in the encrypted file [31, 32, 33, and 39].

Chi square test

Through the chi square test performed between the original and the encrypted files, the non-homogeneity of the two files is tested [116].

The "**Pearsonian Chi-square test**" or the "**Goodness-of-fit Chi-square test**" has been performed here to decide whether the observations onto encrypted files are in good agreement with a hypothetical distribution, which means whether the sample of encrypted files may be supposed to have arisen from a specified population. In this case, the chi-square distribution is being performed with (256-1)=255 degrees of freedom, 256 being the total number of classes of possible characters in the source as well as in the encrypted files. If the observed value of the statistic exceeds the tabulated value at a given level, the null hypothesis is rejected.

The "Pearsonian Chi-square" or the "Goodness-of-fit Chi-square" is defined as follows:

$$\mathbf{X^2 = \Sigma \{(f_0 - f_e)^2 / f_e\}}$$

Here f_e and f_0 respectively stand for the frequency of a character in the source file and that of the same character in the corresponding encrypted file.

On the basis of this formula, the Chi-square values have been calculated for sample pairs of source and encrypted files.

Analysis of the key space

The key space plays an important role in attempting to tackle the Brute-force attack successfully. The key space of each technique has been attempted to enlarge reasonably to make the techniques computationally secure.

1.5.8.4 Computation of encryption/decryption time

The result of the encryption/decryption time plays an important role in assessing the efficiencies of the algorithms from the execution point of view. Here it has been attempted to establish a relationship between the size of the file being encrypted and the encryption/decryption time.

Now, the time consumed in encrypting and decrypting files is related to the code written for that purpose and the architecture of the machine where the code is being executed. All the results in this regard shown in different chapters are taken after compiling and executing C codes in a machine of the following configuration:

Mother Board	**: 810T**
CPU	**: 1.2 GHz Celeron**
RAM	**: 256 MB**
HDD	**: 40GB**

Now, the same code, if is executed on a machine of a different configuration, will require different amounts of time to encrypt and decrypt. Also, if the code is written in a different approach, it may offer different execution time even if it is run on the same machine. But such changes do not produce any change in the relationship between execution time and the source file size. Still there will exist the “almost linear nature” of the relationship.

Comparison of performance with the RSA /TDES technique

By comparing the performance of the proposed techniques with the RSA and TDES technique, it is attempted to evaluate the proposed techniques with respect to the existing field of cryptography. The most popular public key system, the RSA technique and the private key system TDES, has been considered here as the model and the comparative analysis is done on the basis of frequency distribution and chi square distribution for the purpose of evaluation.

A note on merits of proposed techniques:

All the six independent techniques included in this thesis offer security of highly satisfactory level. The statistical results have been shown in the respective chapters using the frequency distribution test and the chi square test. For each technique, a reasonably large key space has been proposed, so that by the brute-force attack the chance of breaking the cipher by key estimation becomes computationally infeasible. Moreover, implementation of each of the algorithms is well tested with satisfactory performance. The execution time varies with the size of the file being encrypted. Only one out of the six proposed techniques causes alteration in size of file after being encrypted. But, in turn, this alteration in size helps in ensuring a better security and space efficiency also.

Following is the list of all the chapters in which all the proposed independent techniques have been presented:

- Chapter 2 entitled, ***Encryption Through Recursive Carry Addition (RCA)***, based on the RCA technique
- Chapter 3 entitled, ***Encryption Through Recursive Key Rotation (RKR) Technique*** , based on the RKR technique
- Chapter 4 entitled, ***Encryption Through Recursive Session Key Arithmetic (RSKA) Technique***, based on the RSKA technique
- Chapter 5 entitled, ***Encryption Through Cascaded Arithmetic Operation on Pair of Bits of (CAOPB)***, based on the CAOPB technique
- Chapter 6 entitled, ***Encryption Through Recursive Modulo-2 Operation of Paired Bits of Streams (RMOPB)***, based on the RMOPB technique
- Chapter 7 entitled, ***Cascaded Recursive Key Rotation of a session key with Transposition and Addition of Blocks (CRKRTAB,)*** based on CRKRTAB technique

The structure of the key plays the most important role, as all the proposed techniques are private key systems. The following chapter discusses this aspect:

- Chapter 8 entitled, ***Formation of Secret Key***, based on suitable key structure for the proposed techniques

Apart from introducing six independent cryptographic techniques, there is also one proposal to implement these techniques in cascaded manner. The following chapter shows how this cascading can be performed tactfully:

- Chapter 9 entitled, ***Encryption Through Cascaded Implementation of the Proposed Techniques***, based on the cascaded approach

Finally, a conclusive discussion on the entire development work, including the final assessment of all the proposed cryptographic systems, is done in the following chapter:

- Chapter 10 entitled, ***A Conclusive Discussion***, based on the conclusion on the entire work

The references are included in Appendix A, List of publications of the candidate are included in Appendix B, Source codes of proposed implemented techniques are included in Appendix C, List of tables of proposed implemented techniques are included in Appendix D, List of figures of proposed implemented techniques are included in Appendix E, List of graphs of proposed implemented techniques are included in Appendix F.