**Manual for**

**Triple DES**

|  |
| --- |
| Triple DES |
| Document Version/Details: Ver 1.0/ 21-April-2018 |

|  |
| --- |
| Key |

Triple DES

Contents

[1. Introduction 3](#_Toc512369349)

[1.1 Need 3](#_Toc512369350)

[1.2 Use 3](#_Toc512369351)

[1.3 Benefits 3](#_Toc512369352)

[2. Basic system requirements 3](#_Toc512369353)

[3. Interface 4](#_Toc512369354)

[3.1 Encrypt field 4](#_Toc512369355)

[3.2 Encrypt Button 4](#_Toc512369356)

[3.3 Encrypted Field 4](#_Toc512369357)

[3.4 Decrypt Button 4](#_Toc512369358)

[3.5 Decrypted Field 4](#_Toc512369359)

[3.6 Import File 5](#_Toc512369360)

[4. Working of Triple DES 5](#_Toc512369361)

[4.1 Generation of Key 6](#_Toc512369362)

[4.2 Padding zeros at the end of the string 7](#_Toc512369363)

[4.3 Initial permutation of string 7](#_Toc512369364)

[4.4 Converting the each 8-bit block of string into 64-bit binary code 7](#_Toc512369365)

[4.5 Sixteen Rounds 8](#_Toc512369366)

[4.5.1 Expansion 8](#_Toc512369367)

[4.5.2 Key and Right 48 Bit XOR 8](#_Toc512369368)

[4.5.3 SBoxing 9](#_Toc512369369)

[4.5.4 XOR of the Right 32-bit string and left 32-bit string 9](#_Toc512369370)

[4.6 Exchange of blocks 10](#_Toc512369371)

[4.7 Final permutation of string 10](#_Toc512369372)

[4.8 Repeat the whole above steps 2 more times 10](#_Toc512369373)

[5. Run time 11](#_Toc512369374)

[5.1 Encryption & Decryption for Triple DES 11](#_Toc512369375)

[5.2 Encryption & Decryption for DES 11](#_Toc512369376)

[5.3 Conclusion 11](#_Toc512369377)

[6. References 11](#_Toc512369378)

# Introduction

## Need

Encryption is important especially during these days when the data that we are using should be protected and even if gets in wrong hands then those people should not be able to decrypt the data. Multinational organizations use various algorithms and encryption ways to protect the data from getting stolen. Sometimes the data exists in the form of text and it could be in human readable format. Sometimes people store their password in a document, or sometimes there might be a confidential data which the companies might not want to share. This data which is stored into files, can be hacked and the hacker could easily read the important data.

## Use

It is possible to encrypt data. This encrypted data is not in a human readable format. Thus, even if the data gets into wrong hands, then it is almost impossible for the hacker to decrypt it. For protecting the data, we are using a data encryption algorithm – Triple Data Encryption Standard Algorithm. This encryption algorithm will convert the data into a binary format (not human readable format) which is difficult to decrypt back to its own original text.

## Benefits

There are many benefits of using this algorithm:

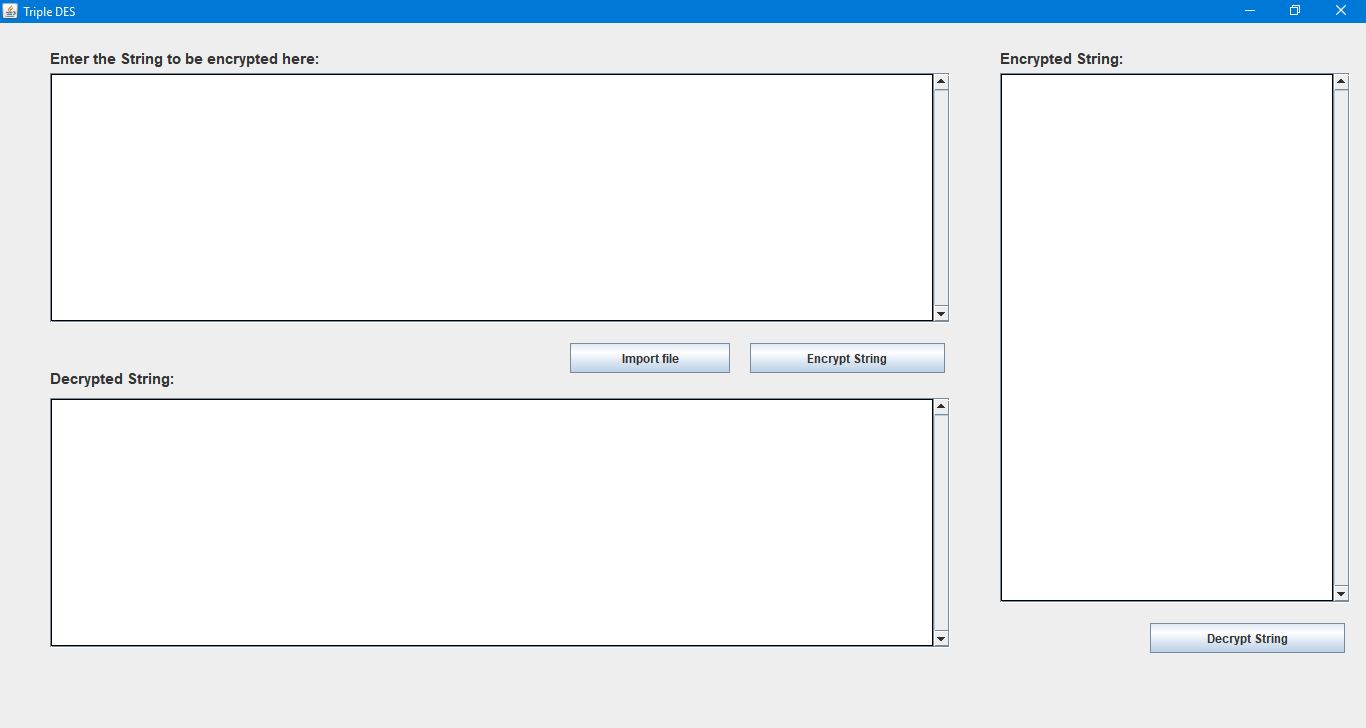
1. Key is almost unidentifiable to find even for the creator
2. Encrypted string is in binary format which is hard to decrypt
3. Can encrypt large amounts of texts without any loss of data

# Basic system requirements

* Windows OS
* JRE 1.6
* 2GB RAM

# Interface

This encryption algorithm is made in Java using Java Swing and has a very simple GUI for the end user to understand and use it. The picture below is how the application looks like.



## Encrypt field

This is the place where the text could be added which we need to encrypt. This program allows to enter any amount of data inside this field.

## Encrypt Button

The encrypt button will work only when there is something to encrypt. If the user adds text inside the encrypt field and presses the encrypt button, the encrypt button would be triggered and the encryption algorithm would be triggered.

## Encrypted Field

The encrypted field would display the encrypted text. This encrypted text would be in binary form.

## Decrypt Button

The decrypt button would decrypt the binary formatted text which we get from the encrypted field and will trigger the decryption algorithm. This decryption algorithm will convert the binary to the original text format.

## Decrypted Field

The decrypted field will show the decrypted code that we get from binary format text. This decrypted field will be same as the original text.

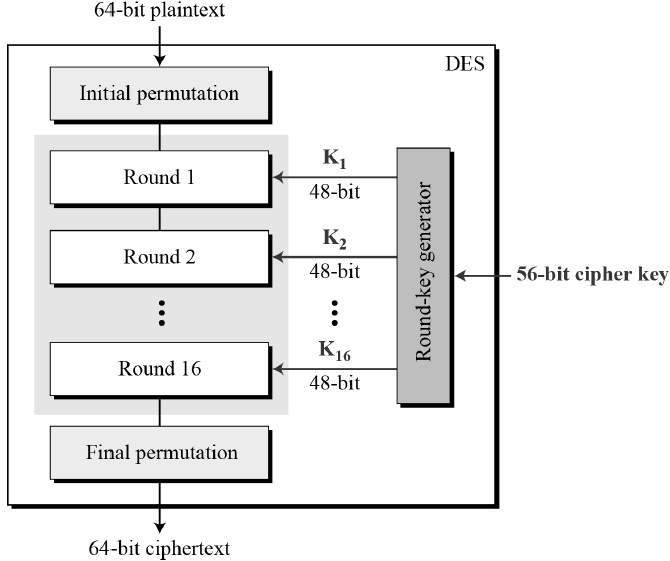
## Import File

This button will allow the user to import text files so that It would be beneficial for user to encrypt large files

# Working of Triple DES

Encryption

The diagram below displays the way the encryption works:



## Generation of Key

One of the most important steps of this algorithm is the generation of the key. They key would be different each time we would be using the application. It is also important that the key should be unique and hard to find. Thus, to generate a key, I am using timestamp function which will get the time in milliseconds. This millisecond would serve as a unique key and would be hard to find.

Next step is Converting the timestamp into binary format of 56-bit. Then Divide the timestamp into 48-bit left key and 8-bit right key. During the encryption process, we would be using only the 48-bit binary key and ignore the end 8-bit key.

**Result: 48-Bit Binary Key**

7 Bit Timestamp

56 Bit Binary Key

## Padding zeros at the end of the string

One thing that this algorithm need is that the string needs to be in 64 bits before it is encrypted. Thus, before converting it into 64 bits, the string needs to be made into an 8-bit string so that the binary (8-bit) could be formed. For example, if the string is hello, then,

**Result: hello000**

## Initial permutation of string

In this permutation, we reverse the string. If the string is hello000 then,

**Result: 000olleh**

## Converting the each 8-bit block of string into 64-bit binary code

In this step, the string of 8-bit is converted into a 64-bit block of binary code

Therefore, a string like 000elleh will look like

8 blocks: 00010111 00000100 11111011 01101011 01101111 00101100 10010100 10101100

This string is then divided into two halves

Result:

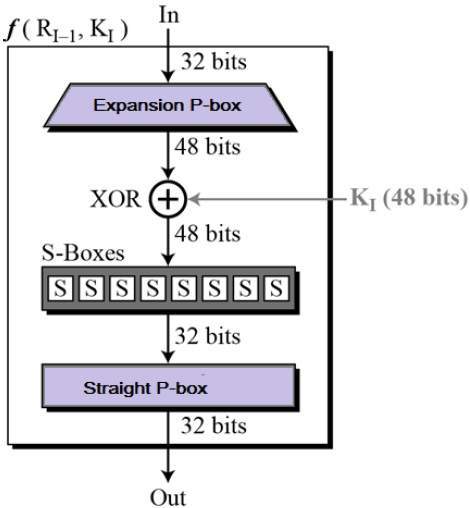
Left: **00010111 00000100 11111011 01101011**

Right: **01101111 00101100 10010100 10101100**

64 Bit Binary String

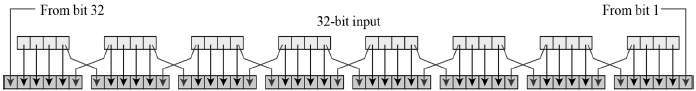
## Sixteen Rounds

When we get the two halves, left and right, the two binary codes go into 16 bits of round. All rounds are similar. In those rounds, there is task of function. In that function these steps are performed.



## 4.5.1 Expansion

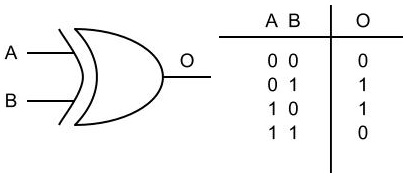
In this step, the Right 32-Bit String is expanded to 48-bit string. This expansion is takes by adding a bit to the start of and at the end of each block from its next and previous block. Look at the diagram for better understanding.



**Result: 48 Bit Right String**

## 4.5.2 Key and Right 48 Bit XOR

In this step, XOR of the 48-bit key and 48-bit right string is done.   
XOR takes place in this way.



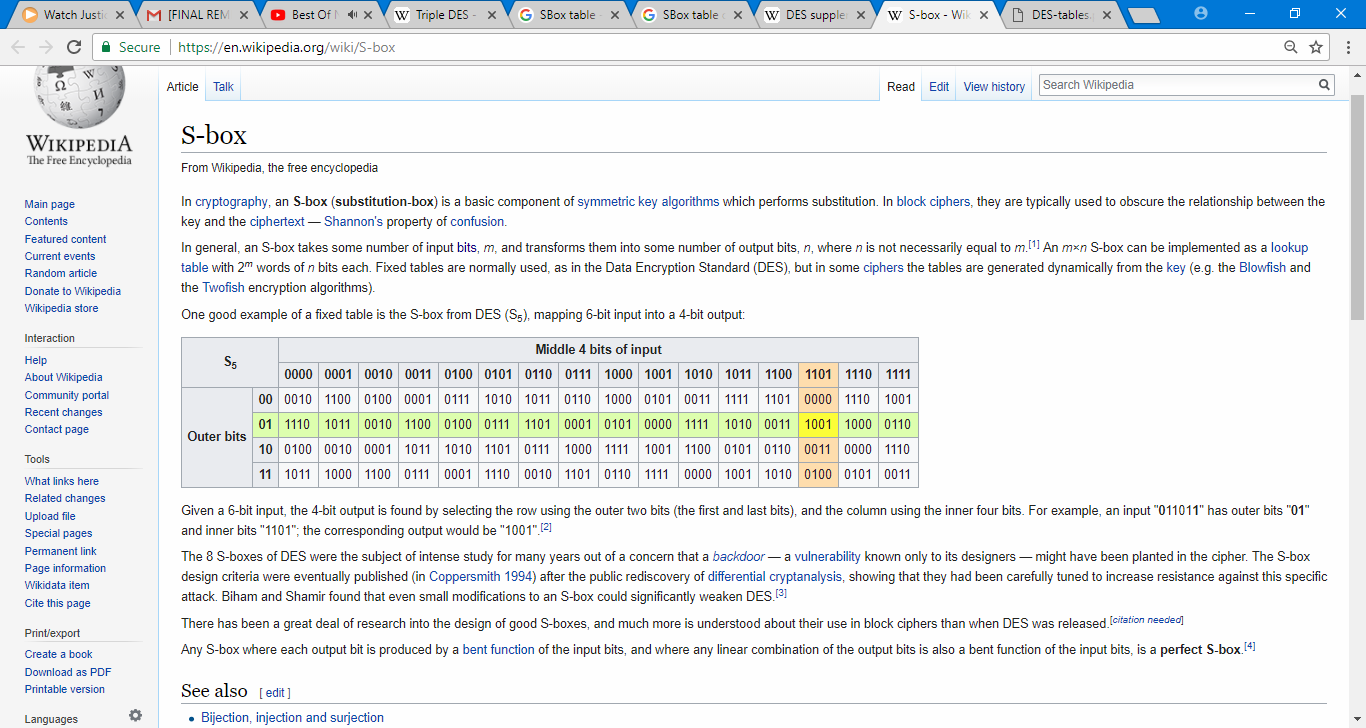
**Result: 48-Bit String**

## 4.5.3 SBoxing

SBoxing is done by converting the 48-bit string into 32-bit string. The SBoxing is done using SBox table.

Consider this first block of binary code 101100 from the 48-bit string. In this string, starting bit and ending bit is considered as row and the middle 4-bits are considered as columns. Therefore, our row is 10 and column is 0110. When we convert this row and column into decimal number it would be: 2 and 6.

Now, there will be a fixed SBox that we would be maintaining. In our application we are making this SBox table. Different applications can have different SBox table. Our SBox table looks like:



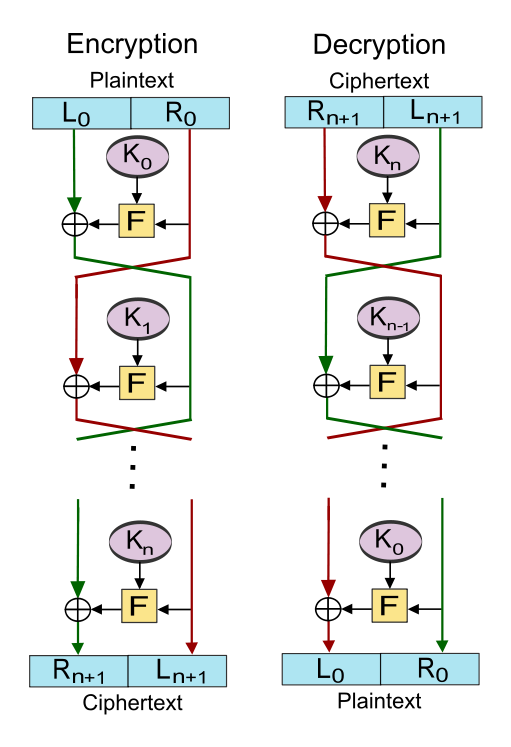
Thus, the result of 2nd row and 6th column would be 0111. In this way if we do the SBoxing of all the 8 blocks of 48-bit, then we get 8 blocks of 32-bits.

## 4.5.4 XOR of the Right 32-bit string and left 32-bit string

In this step, the left 32-bit string is XORed with the 32-bit string that we got, and this string of 32-bit is saved for the next round R2 and the right 32-bit at the previous step is considered as L2

## Exchange of blocks

In this, at the end of 16 rounds, the left 32-bit and right 32-bit are exchanged and joined together.



## Final permutation of string

In this step, the 64-bit binary string is reversed again.

## Repeat the whole above steps 2 more times

All the steps from 4.1 to 4.7 are repeated 3 times.

Decryption

It is the same as the encryption, only difference is that they keys are used in reverse order.

# Run time

## 5.1 Encryption & Decryption for Triple DES

When tried to encrypt 542686 characters – 58109 words, the run time for,

**Encryption Time:**

Nanoseconds: 437418473069

Minutes: 7.2903

**Encrypted Field:** The encryption for this process converted into 4341507 characters of binary.

**Decryption time:**

Nanoseconds: 90795876005

Minutes: 1.513

We conclude that the encryption time taken for the process is much longer than the decryption time. Encryption takes about 7times more than the decryption process.

## 5.2 Encryption & Decryption for DES

When tried to encrypt 542686 characters – 58109 words, the run time for,

**Encryption Time:**

Nanoseconds: 233479460054

Minutes: 3.891

**Encrypted Field:** The encryption for this process converted into 4341507 characters of binary.

Decryption time for the same process:

Nanoseconds: 23232131345

Minutes: 0.38720218908333

We tried to do this to experiment to find how much time it takes to encrypt such a big document using DES and to find a conclusion whether such a heavy encryption is necessary or not

## 5.3 Conclusion

We conclude that the type of encryption depends upon the material we need to encrypt. If the material is very important, then Triple DES would be an excellent solution as it undergoes 3 times. That means, the encryption process and the rounds will go under 48 times whereas DES would be done 16 times. Triple DES would give 3 times more security. The only problem, Triple DES would give is the time taken to encrypt. Therefore, choosing the type of encryption depends on the end user and its goal.

# References

1. <https://www.tutorialspoint.com/cryptography/triple_des.htm>
2. <https://www.tutorialspoint.com/cryptography/data_encryption_standard.htm>
3. <https://en.wikipedia.org/wiki/Block_cipher>
4. <https://en.wikipedia.org/wiki/S-box>
5. <http://www.vlsiinterviewquestions.org/2012/04/17/xor-gate-using-21-mux/>