# TIME MEMORY TRADEOFF ATTACK ON TRIPLE DATA ENCRYPTION STANDARD(Triple-DES)

Project report submitted in fulfilment of the requirement for the degree
of Bachelor of Technology

in

**Computer Science and Engineering**

By

Anshul Chauhan (151265)

Under the supervision of

Dr. Suman Saha

to

**Department of Computer Science & Engineering and Information Technology**

**Jaypee University of Information Technology Waknaghat, Solan-173234(HP)**

## Certificate

## Candidate's Declaration

**I hereby declare that the work presented in this report entitled TIME MEMORY TRADEOFF ATTACK ON TRIPLE DATA ENCRYPTION STANDARD(3 DES) in fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering/Information Technology submitted in the department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology, Waknaghat is an authentic record of my own work carried out over a period from August 2018 to December 2018 under the supervision of Dr. Suman Saha, Assistant Professor (Senior Grade ), Computer Science and Engineering/Information Technology.**

**The matter embodied in the report has not been submitted for the award of any other degree or diploma.**

**Anshul Chauhan, 151265**

**This is to certify that the above statement made by the candidate is true to the best of my knowledge.**

**Dr. Suman Saha**
**Assistant Professor (Senior Grade )**

**Computer Science and Engineering/Information Technology**

**Dated: 1/12/2018**

# CONTENTS

# ACKNOWLEDGEMENT

It is a certifiable joy to express our profound feeling of thanks and appreciation to our tutor, logician and guide Dr. SUMAN SAHA, Assistant Professor (Senior Grade), Department of Computer Science and Engineering and Information Technology, Jaypee University of Information Technology, Waknaghat, Himachal Pradesh. His commitment and unmistakable fascination over the entirety of his mind-boggling disposition to help his understudies had been exclusively and basically in charge of finishing our work. His opportune guidance, careful examination, academic exhortation and logical methodology have helped us to an exceptionally extraordinary degree to achieve this undertaking.

We have taken endeavors in this undertaking. Be that as it may, it would not have been conceivable without the caring help and help of numerous people and associations. We might want to stretch out our earnest because of every one of them.

We might want to offer our thanks towards our folks and Jaypee University of Information Technology for their benevolent collaboration and support which helped us in culmination of this task.

Our thanks and thanks likewise go to our partner in building up the task and individuals who have energetically bailed us out with their capacities.

# ABSTRACT

The time of innovation opens the need of security in our generation and if we consider the pace of time at which it is changing, technology is also becoming more prone to security breach and various online or network attacks. Everybody usually likes an immediate, exact, quick and simple answer for the issues of almost every kind so it is also a greater need to speed up the techs along with its security.

Now to this level of security, we have introduced the concept of triple DES algorithm which consider to be one of most secure algorithm in the today's world. However, there is one problem with this algorithm which is its speed of encryption and decryption. So, in order to speed this algorithm up, we modified this algorithm by storing some precomputed values and divided its work by time and storage sharing formula.

Now in order to check its vulnerability, we had an attack on triple DES which is Time memory data trade off attack. This attack is based on the balance between time and memory to find the key pair corresponding to the given ciphertext. This attack is very efficient and is not restricted to only cryptography but also it has some other applications too. However, in this attack, there are some anomalies also, that is why, we used the concept on rainbow tables to make ourselves ensure that there should not be any repetition in the chaining process.

# CHAPTER : 1

# INTRODUCTION

## 1.1. Introduction

People the most edified and propelled species on this planet has advanced because of the knowledge and the capacity to effectively pass on this information to his who and what is to come. Correspondence was the essential methods for accomplishing all this. Initially people traded information utilizing motions. Gradually, he began cutting data into items as images, sculptures and so forth to recollect it for a more drawn out term. Speech was a brilliant blessing invested to humans.

Be that as it may, with time we need to ensure the data from spilling to the undesirable sources additionally emerged. The science relating to adding the disarray and dispersion to the message has likewise built up a considerable measure and is presently called cryptography. Greeks originated this concept of cryptography. The word signifies "mystery writing". In the most punctual times, the cryptography was primarily constrained to the military purposes. The soonest example or confirmation that exists is from 2000 years before Crist in Egypt. This is the antiquated hieroglyphic which was considered as the holy composition since individuals were not able disentangle the pictures and images installed on it.

Later an alternate tablet was found it actually described the meaning of these images was referenced against their local language. In military, crypto was very famous since a very long time. It has been recommended that the bold armed forces of Spartans additionally utilized this workmanship to convey messages to generals. Slave's head were being shaved, messages were written on their head and then they wait for his hair to grow again. After that, they sent the captive to the different kingdoms, where the receivers shaved slave’s head and recouped the message. The Romans are also likewise set to know things about the craft of cryptography. King Caesar himself was benefitted so much with the renowned Caesar cipher. This is a basic substitution figure which includes the moving to the plaintext by the number referenced in the key. Hence, it was being said that the substitution ciphers controlled the established period of cryptography.

As military utilized this science to trade essential messages the adversaries needed to break cryptographic frameworks and gain shrouded information. This prompt the improvement of a thing called cryptanalysis. Cryptanalysis can be characterized as the investigation of the exist cryptographic frameworks and utilizing the learning picked up to break system with the access to the mystery information.

The most conspicuous strategy being the one created by the renowned mathematician Al-Kindi as prior as in 820 AD. He was first one to examine the mono-alphabetic substitution ciphers and concocted the recurrence based analysis. This progression ended up beingthe most important during the Second World War. As the greater part of the current frameworks were simply substitution ciphers the majority of them were helpless to this attack. So with this, the need to grow more secure and advance cryptosystems increased and Such framework did discharge after the mid 14th century. Leon Arberti made a polyalpahbetic cipher. He is known as "The father of western cryptography". His technique included the utilization of a cipher disk which comprised of a movable inward disk. With rotations, this plate could delineate internal content to any required plaintext.

Until now the utilization of a secret key was unclear and unidentified. In the mid 15th century, Blaise De Vigenere had a thought that would revolutionize cryptographical way then it was done earlier. His conspire included rehashing the way to change in accordance with the length of the plaintext and performing basic expansion modulo 26. The straightforwardness of the strategy alongside the confusion and dissemination it offered was amazing. In 1865 Friedrich Kasiski built up a technique called KASISKI TEST through which the Vigenere cipher was dissimilated. The cryptography remained around the equivalent till the 20th century.

In any case, by the appearance of the modern unrest and the mechanical era, the measure of information traded encountered an exponential growth. Hence, the desperation in raising the level of the innovation engaged with the trading of messages. Moreover, the World Wars played an imperative job in the headways in this field. In 1917, the British Army blocked a message sent by the Germany to the Mexico. The scrambled message was before long decoded by the British intelligence.

In the message, the German Foreign Minister Arthur Zimmerman tended to the Mexican pastors ,requesting them so that they could be benefited by joining hands for the war against United States Of America. This affected the American pioneers and lead them to take part in the World War. Now the Americans were aware of the significance of maintaining mystery and built up a superior cryptosystem, One Time Pad. One Time Pad comprised of creation of an arbitrary key with length equivalent to the length of the plaintext and xoring the both in order to get required ciphertext. This method was proposed by Gilbert Sandford Vernam in early 19th century.

Similar things likewise occurred in second world war. The Japanese had designed a machine considered PURPLE and encoded their messages with the help of this machine. USA had built up the decrypting algorithm. Unaware of this, the Japanese informed their armada posted close USA about the landing of their military chief. The American armed force grabbed this chance and killed an essential,energetic leader. But the enigma machine adventure is the most well-known and heroic. In 1932 German engineer Arthur Scherbius built up a machine called Enigma which was equipped for performing different encryption operations. Machine ended up being very helpful on the planet war for communicating messages, secretly.

The British could capture numerous messages but were unable to break those. So, they utilized a group of incredible cryptographers and mathematicians comprising of greats like Alan Turing to break it. they made numerous perceptions and made sense of numerous feeble points. They discovered that a letter in plaintext will never match to a similar letter set in the ciphertext. Using this and numerous comparablefacts the British could break the enigma machine and accumulated gigantic measure of information about the activities of the Germans, which they used to inevitably overcome the German propaganda.

Since the world war the exploration of cryptography has enhanced significantly. Computers have moved toward becoming part and packages of our life. We all have an actual existence on the web and it has turned out to be exceptionally important to anchor it against any type of threats. Hence, many prevalent cipher plans have been introduced. This incorporates numerous crypto systems. The old square ciphers, stream ciphers to the further developed elliptic curve cryptography, quantum cryptography and lattice based cryptography. All furnish us with a protected online life. The calculations like DES(Data Encryption Standard) were utilized at first to scramble the information online. But the handling power and the boundless assets are the difficulties it was not able keep up. Having a key size of just 56 bits until the point when in 1990's, the plan was broken utilizing beast compel and differential analysis due to which introduction of triple DES takes place. Due to the significance that these calculations hold, we require them to be impenetrable against any sort of attack. Hence, we investigate these algo’s against every single possible danger.

One such basic risk is the brute power attack. This includes connecting each conceivable key and searching for the privileged key. But due the substantial size of the key included our handling power ends up being feeble. So, cryptanalysts constantly needed to build up a superior technique than brute force. A strategy more productive and something that could be utilized generally to split any sort of scheme. In 1980 Martin E. Hellman proposed something that appeared as though the answer. A straightforward proposition asserting to setting up a center path between the memory used and the processing power. His arrangement guaranteed to reduce the season of beast drive from N to N1/3.The proposition was inevitably acknowledged and is a famous strategy to cryptanalyze any calculation or cryptographic scheme. Cryptanalysis is the important part of cryptology. We require our answers for be impeccable and need to be the first to discover any blame in them,if it does. What will occur if a man having abhorrent plans finds out a few flaws or escape clauses in our cryptographic scheme? Therefore, in order to make DES more secure, triple DES was introduced which reduces the chances of get caught in mid-way attacks. Attacks over triple DES doesn’t seem much feasible, so this time we are trying to use TMTO over Triple DES.

## 1.2. Problem Statement

Triple Data Encryption Standard is serving as the encryption standard for about 10 years. Explain how this algorithm was finally broken and demonstrate a Time Memory Data Trade Off Attack on it.

## 1.3. Objective

**GOAL**:

Implementation of Time Memory Data Trade-off Attack on triple DES with overall higher success rate.

### METRICS:

Time taken by the machine to perform TMTO attack.

Overall success rate involved.

Total memory required for storing encrypted texts.

### Objective 1:

Optimization and execution of triple Data Encryption Standard in C++14.

### Objective 2:

Execution of the offline phase of Time Memory Trade Off on Triple Data Encryption Standard.

### Objective 3:

Extension of the performance using rainbow and perfect tables.

### Objective 4:

Execution of the online phase of Time Memory Trade Off on triple Data Encryption Standard.

## 1.4. Methodology

In the last advance of our venture a Graphical User Interface will be created and will be combined withtheattack's execution. There will be a button for setting up a tabular format for another plaintext or for executing out the second part of TMTO attack. After executing the attack, the key will be showed on the interface. We need high performance driven machines for this attack. Therefore, we need GPU's which will comprise of some online cloud administrations.

# CHAPTER : 2

# LITERATURE SURVEY

## 2.1 A Cryptographic Time Memory Trade Off

**2.1.1 Author:**Martin E. Hellman

**2.1.2 Year:**1980

**2.1.3 Summary**: The major research paper to present finding a center path or we can say an in between method between the measure of memory utilized and the required time to use brute force attack over this whole plot. In this paper, we use two phases to separate the undertaking of brute force into two sections that is the pre-calculation procedure to prepare a tabular form in the backend and afterward the online stage to scan for the originally used key.

### 2.1.4 Advantages:

**2.1.4.1** Most initial paper to present the idea of utilizing Time Memory Trade Off.

**2.1.4.2** Reduction of overall brute force time by a huge difference.

### 2.1.5 Disadvantages:

**2.1.5.1** There can be false Alarms initialized in the online stage.

**2.1.5.2** Merged chains and circles made issues and expanded the measure of aggregate memory required.

## 2.2 Rigorous time/space tradeoffs for rearranging capacities

**2.2.1 Author:** Amos Fiat,MoniNaor

**2.2.2 Year:**1991

**2.2.3 Summary:** The underlying proposition of Hellman asserted that TMTO could be expanded further and used for reversal of one-way functions. TMTO stayed constrained up to block ciphers. its utilization for reversal was acknowledged in the following research work, that provided a powerful scheme for accomplishing this purpose.

**2.2.4 Advantages:**

**2.2.4.1** One of the initial papers to broaden the idea of utilizing Time Memory Trade Off upto inversion of one-way functions.

**2.24.2** As compared to original approach,brute force time was significantly reduced

**2.2.5** Disadvantages:

**2.2.5.1** The proposition was only theoretical one and they were not able demonstrate it with practical implementation and execution.

**2.3 Cryptanalytic Time/Memory/Data Tradeoffs for Stream Ciphers**

**2.3.1 Author:** Alex Biryukov and Adi Shamir

**2.3.2 Year:**2000

**2.3.3 Summary:** One of the main research work to propose a technique for execution of TMTO successfully over stream ciphers. Initial thought was to utilize lower examining obstruction for tradeoff on stream ciphers.

**2.3.4 Advantages:**

**2.3.4.1** Initial paper to present the idea of utilizing Time Memory Trade Off onstream ciphers.

**2.3.4.2** Time consumption of precomputation stage lowered down.Also, the add up to requirement for information diminished.

### 2.4 A Time-Memory Tradeoff Using Distinguished Points

**2.4.1 Authors:** Francois-Xavier Standaert,GaelRouvroy,Jean-JaquesQuisquater

**2.4.2 Year:**2002

**2.4.3 Summary:** Primary paper to actualize the proposition of Rivest of utilizing different ending points. Instead of producing the chains of similar lengths, we settle the end points of chains and continue creating the chain till the point when any of these end points is accomplished.

**2.4.4 Advantages:**

2.4.4.1 Initial research paper to broaden the idea of utilizing different end points for executing Time Memory Data Trade Off.

2.4.4.2This method provided us with better metrics and lessened the aggregate memory references and made usage less demanding.

### 2.5 Breaking Ciphers with COPACOBANA-A Cost-Optimized Paralled Code Breaker

**2.5.1 Authors:** Sandeep Kumar,ChristofPaar,JanPelzl,Gerd Pfeiffer

**2.5.2 Year:**2003

**2.5.3 Summary:** The following research paper was the first to provide the structure of a particular machine called COPACOBANA to successfully attack on DES. The machine could do so in around 3 days. The evaluated cost of this highly powered machine was around $10,000 which was viewed as more than expected.

**2.5.4 Advantages:**

**2.5.4.1** Initial work to provide a sufficient structure for a specific machine.

**2.5.4.2** The planned ended up being extremely productive and practical.

### 2.6 Making a Faster Cryptanalytic Time-Memory Trade-Off

**2.6.1 Author:** Philippe Oechslin

**2.6.2 Year:** 2003

**2.6.3 Summary:** Provided the idea of rainbow tables. The rainbow table is a variation of the time memory tradeoff strategy which manages different issues its predecessor had by utilizing distinctive decrease work on every progression in the chain. Philippe Oechslin demonstrated that the execution of the TMTO can be expanded radically by utilizing diverse reduction methods.

The likelihood of impact between two chains diminished to 1/t. Philippe Oechslin further expanded his work and attempted to locate the ideal estimations of the parameters m, t and l and attempted to make something many refer to as perfect tables. Perfect tables have 0 combines and no memory is trashed.

**2.6.4 Advantages:**

**2.6.4.1** Initial research to provide the possibility of rainbow tables. Perfect tables were additionally introduced to the world of cryptography in this research work as it were.

**2.6.4.2** Provided appropriate system to pick different metrics engaged with performing Time Memory Data Tradeoff.

**2.6.4.3** False alarms, merges, circles were tackled that were initially noticeable in Hellman's research work.

### 2.7 Time Memory Tradeoff Implementation on Copacobana

**2.7.1 Author:** Stefan Spitz

**2.7.2 Year:** 2007

**2.6.3 Summary:** The main purpose of this paper was to show and design a system which can reduce the time to break the cipher and get the key from the block cipher. This was done on the popular data encryption standard (DES). In this paper, Stefan used special hardware Copacobana to break the DES in the minimum amount of time.

### 2.6.4 Advantages:

**2.6.4.1** This specific application of just one task provides the chances to make this more complex a simpler one and consumes less time of breaking the DES.

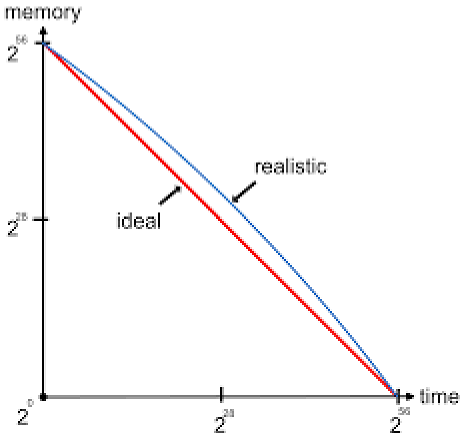
**2.6.4.2** This gives the predictability to improve or modify the design of some particular task which can be incorporated using FPGA’s.

# CHAPTER : 3

# SYSTEM DEVELOPMENT

## 3.1. Computational

The computational model will comprise of the utilization of distributed computing services. The calculation will be transferred to the cloud where elite equipment will run our calculation till the formation of the tables.



## 3.2. Numerical

### Hellman Analysis

The condition for each progression:

$C'=S'k(P)$

The yield created in this progression will be of 64 bits. So, we lessen it to the key size i.e 56 bits

$f(K)=Red[S'k(P)]$

The likelihood of accomplishment is given by the equation:

$P(S')>=(1/n)\sum i=1M\sum j=0T-1[(n-iT)/n]j+1$

The normal number of false alerts per table is given by the equation:

$E'(F)<=MT(T+1)/2*n$

Hellman proposed to pursue the connection:

$m=M*T$

$MT2=n$

$M=T=n(1/3)$

Breaking the DES by means of this technique was diminished to 2^38.

M-number of beginning stages

T-length of chain

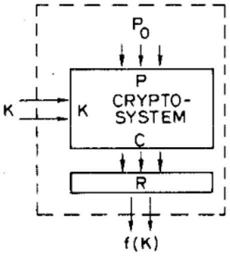
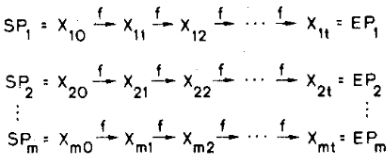


Fig. 1. Construction of the function *f*.



$$\begin{array}{l}
SP_1 = X_{10} \xrightarrow{f} X_{11} \xrightarrow{f} X_{12} \xrightarrow{f} \cdots \xrightarrow{f} X_{1t} = EP_1 \\
SP_2 = X_{20} \xrightarrow{f} X_{21} \xrightarrow{f} X_{22} \xrightarrow{f} \cdots \xrightarrow{f} X_{2t} = EP_2 \\
\vdots \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \vdots \\
SP_m = X_{m0} \xrightarrow{f} X_{m1} \xrightarrow{f} X_{m2} \xrightarrow{f} \cdots \xrightarrow{f} X_{mt} = EP_m
\end{array}$$

Fig. 2. Matrix of images under *f*.

### 3.3 Modification of Triple DES:

Triple DES is the combinational algorithm and improved implementation of DES. In triple DES, we are using DES three times with 2 different keys. In this way, we improve the security and maintain confidentiality in the data. Using key of 112 bits, it was being said that, this is nearly impossible to break the triple DES in sufficient amount of time. However, using hardcore and task devoted machines, it is now possible to break this algorithm.

Other issue with this algorithm is that it is slow compared to DES, AES and other algorithms. Reason behind this is the length of the key which is 112 bits. So in order to resolve this we have made some optimizations in this.

We used the logic of time memory data trade off in this also. If we let the machine do everything in the running phase then obviously it will take time to compute data and table on every iteration. So in order to prevent that, we computed all the tables in the precomputed phase and then used those tables to directly input the values. In this way, every time, we need not to compute everything on the spot. Hence, in this way, we can speed up the algorithm to its apex.

### 3.4 Sorting Function:

while applying TMTO, we use the inbuilt sorting function sorting function of C++. This sorting function uses a hybrid sorting algorithm (introspective sort) which is the combination of quick sort, heap sort and insertion sort. Introspective sort is formulated to increase the speed of sorting function and was built using the pros and cons of all these three sorting algorithms.

**Insertion sort** is used if elements are not greater than 16.

**heap sort** is used to heapify the elements.

**Quick sort** is used to find the partition point in the array

**Algorithm:**

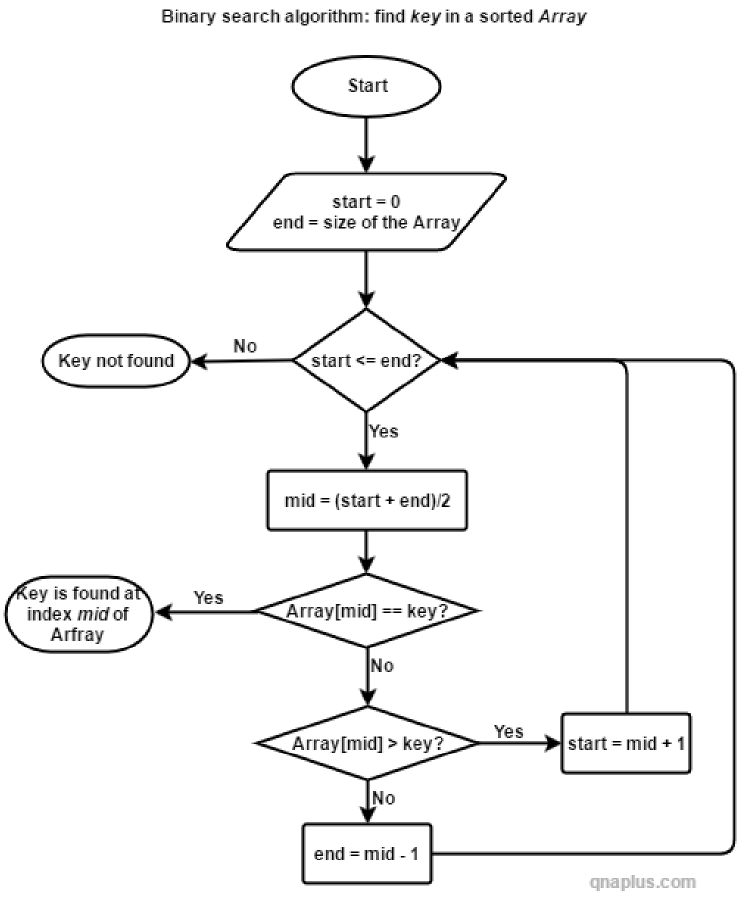
```
sorting(A : arr):
  depth= 2xfloor(log(len(A)))
introspective(A, depth)
```

```
introspective(A, depth):
    L = len(A)
    if L<=16:
        insert(A)
    if depth == 0:
heapA(A)
    else:

        part = partA(A)
        introspective(A[0:n-1], depth - 1)
        introspective(A[n+1:L], depth - 1)
```

### 3.5 Why to use binary search instead of any other search?

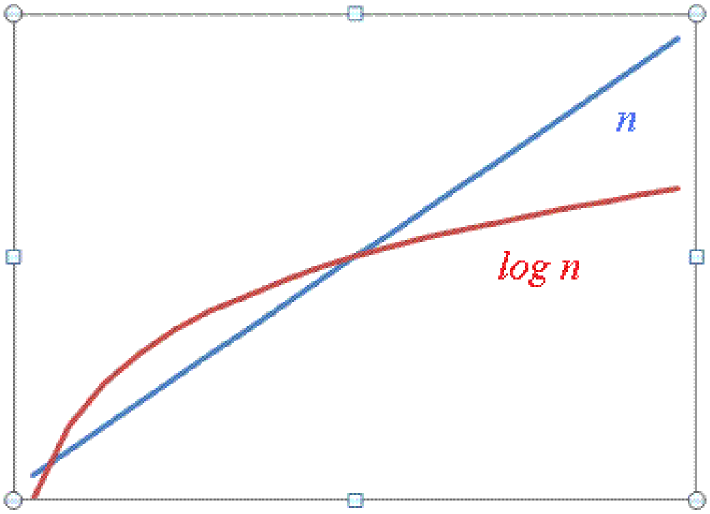
what is binary search?

Suppose we are given a sorted array of n elements and we want to find some element in the array. So, in case of binary search, we break the array in two halves, and check if the middle element is greater than or less than the searching element. If searching element is greater then go for the right part of the array and if searching element is smaller then go for the left part of an array. Repeat these steps, and eventually we will find our element.



Why we used binary search?

Reason for using binary search is its less time complexity compared to linear search. The time complexity of binary search is O(log(n)) whereas for linear search it is O(n*n). Even in the best case, it gives the result in O(1).



Why we are not using hashing algorithms?

The time complexity of hashing algorithm is O(n) and is considered to be the fastest one. But here, in case of TMTO, we can not use this one. The reason is space complexity and storage issue. When we apply TMTO, we need to store millions of computations or sometimes even billions in precomputational phase and it is already consuming so much space to store this data. But, if we go for hashing then we also need to store hashes of these million computations which doesn't seems feasible for us. This process can increase the space complexity to another level and also storing this much data just for finding one time key is not practically profitable.

Why to use iterative binary search not recursive binary search?

We know that both iterative and recursive binary searching algorithms have same time complexity that is, O(log(n)) and they generally differ in terms of code length, code complexity and most important of all 'space complexity'. Space complexity for recursive can reach to log(n) but in case of iterative, it will always be O(1). When we are using TMTO, we need to look after both space and time for the computations. We need to create a balance between the both and minimize both of them. So, it is profitable for us to use iterative binary search rather than recursive binary search.

### 3.6 Implementation of Expansion/reduction function in rainbow tables:

what is reduction function?

Reduction function is an iterative function over the output of DES to reduce the bits from 64 bits to 56 bits so that we can take that output as the key for next function of DES.

what is expansion function?

expansion function is an iterative function over the output of Triple DES to expand the bits from 64 bits to 112 bits so that we can take that output as the key for next function of triple DES.

Why do we need to use different functions at every point in rainbow tables?

If we don't use different functions in rainbow tables then it is losing the properties of rainbow tables or we can say that these will not be rainbow tables this will only be the table of hash chains which will eventually be less efficient for large tables.

How do we create reduction functions?

Main motive to create the reduction function is to generate the redundancy at every point of chain. It could be any function which can give us redundancy. We used the size of searching space or set and take the hash value module of that searching set. To make reduction function different at every point, we are also incrementing with the count of 1.

Reduction function:

```
def Red(i, a):
    return {(a + i)%(2^56)}
```

### 3.7 Distinguished Points Analysis

The calculation proposed requires to pick a DP-property of request d and a most extreme chain length t. We precalculated r tables by picking r diverse capacities. For each cover work m distinguished starting points (which are recognized) will be chosen in a random order. For every initialization point a chain will be allotted till the point that a DP is found or till the point that the length of the chain becomes is t + 1. Just starting focuses repetition of distinguished points in less than t cycles will be trashed away with the comparison of chain lengths, remaining will be trashed of. Also, if a same distinguished point is the ending point for different chains, in that situation the chain of maximum length will be given away. This comprises of a much lower memory multifaceted nature than Hellman's tradeoff. Pre-calculation procedure:Create r different tables with (SP,EP,l)- triples, arranged on the basis of end points.

1. Pick the DP-property having request D.

2. Pick R diverse veil capacities gi ,i = 1, 2, ..., R. It characterizes R distinctive f capacities: fi = gi (E’κ (p )), I = 1, 2, ..., R.

3. Pick the most extreme chain length T.

4. For i = 1 to R

(a) Choose M arbitrary begin focuses SP’1 , SP’2 , ..., SP’M .

(b) For j = 1 to M, L = 1 to T

I. Process fil(SP’j ).

ii. In the event that fil(SP’j ) is a Distinguish point, store the triple (SP’j , EP’j = fil(SP’j ), L)

what's more, take next j.

iii. In the event that L > T "overlook" SP’j and take next j.

(c) Sort triples on end focuses. On the off chance that few end focuses are indistinguishable, just store the triple with the biggest L.

(d) Store the greatest L for each table: Lmax

For the inquiry calculation, a table just must be gotten to when a DP is experienced amid an cycle which permits proficient executions of the on the web assault. Additionally, on the off chance that the experienced DP isn't in the table, one won't discover the objective key by emphasizing further. Henceforth the ebb and flow pursuit can skirt the rest of this table.

Inquiry calculation: Given C’ = E’K (p) discover K.

```
1. For i = 1 to R
    (a) Look up lmax
    (b) Y’ = gi (C’).
    (c) For j = 1 to Lmax
        I. On the off chance that Y’ is a Distinguish Point,
            A. On the off chance that Y’ in table i,
                – Take the comparing SP’(i) and length L in the table.
                – If j < l
                    • Compute forerunner K̃ = fil−1−j (SP’L ).
                    • If C’ = E’K̃ (p ) then K = K̃: STOP.
                    • If C’ 6= E’K̃ (p ), take next i.
            B. Else take next i.
        ii. Set Y’ = f (Y’ ).
```

The probablility to achieve the recognized point is given by the recipe:

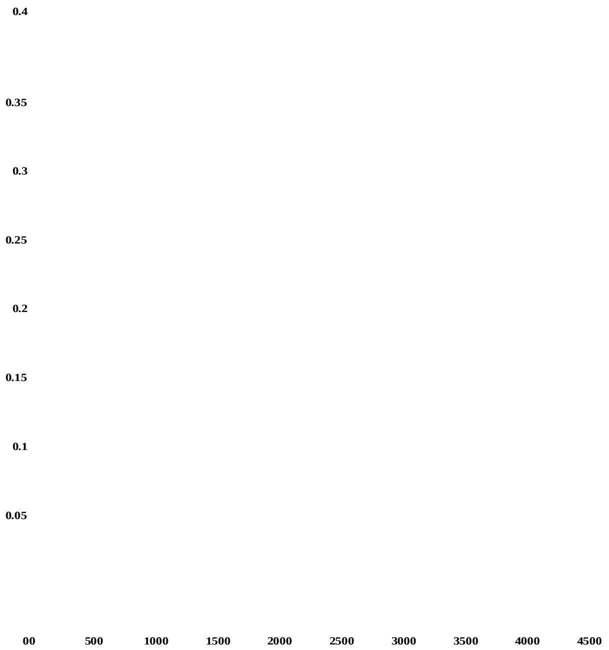
$$P_2(l)=\prod^{\mathbf{l-1}}_{\mathbf{i=0}}(1-2^{\mathbf{k-d}}/2^{\mathbf{k}}-i)$$

Choosing the average chain length β

$$P_2(l) \simeq (1 - \frac{2^{k-d}}{2^k - \frac{l-1}{2}})^l$$

$$P_1(l) = 1 - \prod_{i=0}^{l-1}(1 - \frac{2^{k-d}}{2^k - i})$$

$$P_1(l) \simeq 1 - (1 - \frac{2^{k-d}}{2^k - \frac{l-1}{2}})^l$$



**X= l, Y=lP(ADP is reached in exactly l iterations)**

## Register the normal chain length :

$$\beta = \frac{\sum_{l=t_{min}}^{t_{max}} l.P(DP.in.exactly.l.iterations)}{\sum_{l=t_{min}}^{t_{max}} P(DP.in.exactly.l.iterations)}$$

$$\gamma = \sum_{l=t_{min}}^{t_{max}} P(DP.in.exactly.l.iterations) = P_1(t_{max}) - P_1(t_{min} - 1)$$

**Numerator can be estimated as follows:**

$$\sum_{l=t_{min}}^{t_{max}} l.P(DP.in.exactly.l.iterations)$$
$$= \sum_{l=t_{min}}^{t_{max}} l.(\prod_{i=0}^{l-2}(1-\frac{2^{k-d}}{2^k-i}) - (\prod_{i=0}^{l-1}(1-\frac{2^{k-d}}{2^k-i}))$$
$$\simeq \sum_{l=t_{min}}^{t_{max}} l.((1-\frac{2^{k-d}}{2^k-\frac{t}{2}})^{l-2} - (1-\frac{2^{k-d}}{2^k-\frac{t}{2}})^{l-1})$$

Where T = (Tmaximun + Tminimum) / 2

We can rewrite this equation in simpler form as follows:

$$\sum_{l=t_{min}}^{t_{max}} l.((1-x)^{l-2} - (1-x)^{l-1})$$

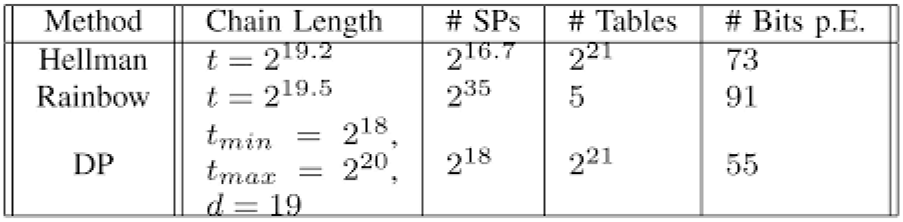
Where $x = \frac{2^{k-d}}{2^k-\frac{t}{2}}$.

$$\sum_{l=t_{min}}^{t_{max}} l.((1-x)^{l-2} - (1-x)^{l-1})$$
$$= t_{min}.(1-x)^{t_{min}-2} - t_{max}.(1-x)^{t_{max}-1} + \sum_{l=t_{min}-1}^{t_{max}-2}(1-x)^{l}$$
$$= (1-x)^{t_{min}-2}.(t_{min}+\frac{1-x}{x}) - (1-x)^{t_{max}-1}.(t_{max}+\frac{1}{x})$$

Finally the total average chain length should be :

$$\beta \simeq \frac{(1-x)^{t_{min}-2}.(t_{min}+\frac{1-x}{x})-(1-x)^{t_{max}-1}.(t_{max}+\frac{1}{x})}{\gamma}$$

evaluated dp – property,



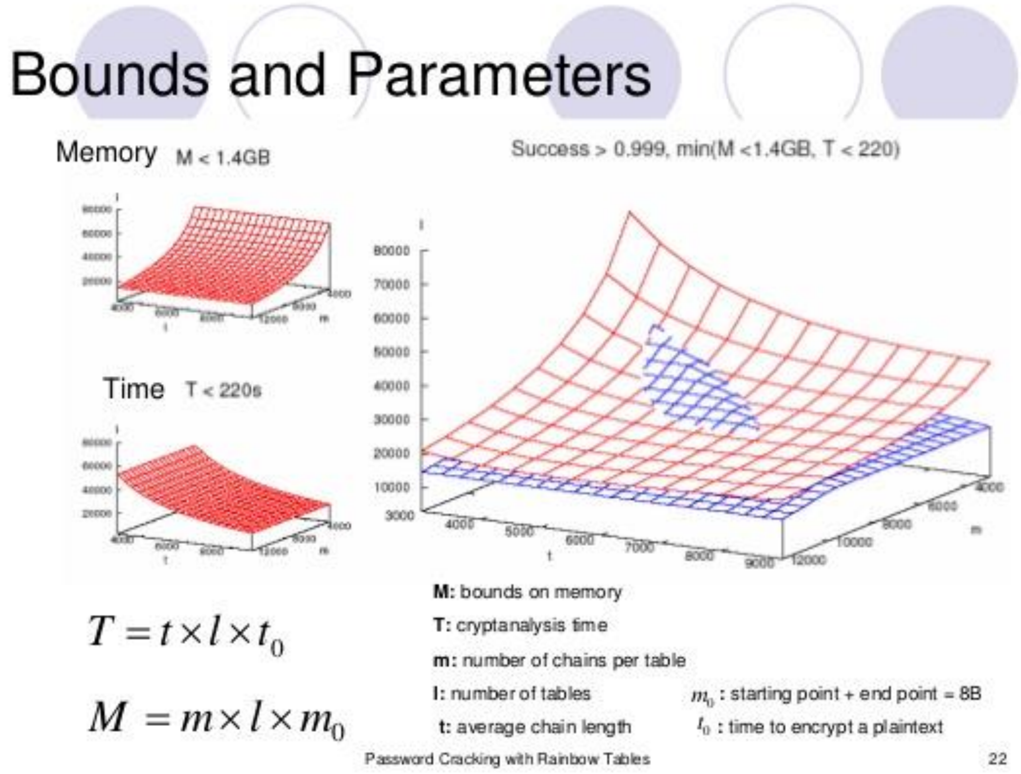
| Method | Chain Length | # SPs | # Tables | # Bits p.E. |
|---|---|---|---|---|
| Hellman | $t = 2^{19.2}$ | $2^{16.7}$ | $2^{21}$ | 73 |
| Rainbow | $t = 2^{19.5}$ | $2^{35}$ | 5 | 91 |
| DP | $t_{min} = 2^{18}$,<br>$t_{max} = 2^{20}$,<br>$d = 19$ | $2^{18}$ | $2^{21}$ | 55 |

Probability to find the key using table of M rows of T keys

$$P_{table} \geq \frac{1}{N}\sum_{i=1}^{m}\sum_{j=0}^{t-1}(1-\frac{it}{N})^{j+1}$$

The probability of having success with the help of L tables is provided as

$$P_{success} \geq 1-\left(1-\frac{1}{N}\sum_{i=1}^{m}\sum_{j=0}^{t-1}(1-\frac{it}{N})^{j+1}\right)^{\ell}$$



## Bounds and Parameters

Memory M < 1.4GB

Time T < 220s

Success > 0.999, min(M <1.4GB, T < 220)

$$T = t \times l \times t_0$$

$$M = m \times l \times m_0$$

**M:** bounds on memory

**T:** cryptanalysis time

**m:** number of chains per table

**l:** number of tables

**t:** average chain length

$m_0$ : starting point + end point = 8B

$t_0$ : time to encrypt a plaintext

Password Cracking with Rainbow Tables 22

Arrangement space for likelihood of achievement with 99.9%, most extreme size of 220 seconds and memory size of 1.4GB

Table os estimate M x T is given as

$$P_{table} = 1 - \prod_{i=1}^{t}(1 - \frac{m_i}{N})$$

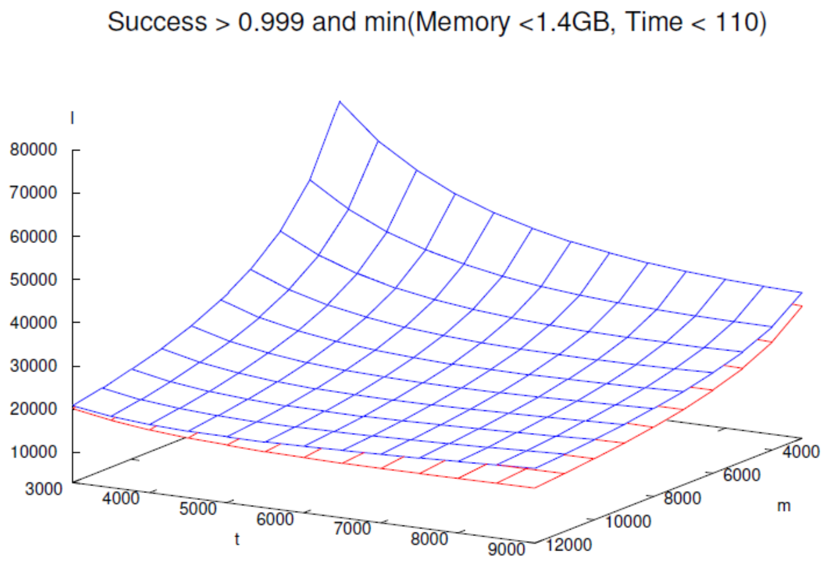
$$\text{where} \quad m_1 = m \quad \text{and} \quad m_{n+1} = N\left(1 - e^{-\frac{m_n}{N}}\right)$$

t exemplary tables having size M x T can be appeared as pursue

$$\begin{array}{cc} & \overset{t}{\xrightarrow{\hspace{5cm}}} \\ m \Big\downarrow & \begin{bmatrix} k^1_{1,1} \xrightarrow{f_1} \xrightarrow{f_1} \ldots \xrightarrow{f_1} k^1_{1,t} \\ \\ k^1_{m,1} \xrightarrow{f_1} \xrightarrow{f_1} \ldots \xrightarrow{f_1} k^1_{m,t} \end{bmatrix} \\ & \overset{t}{\xrightarrow{\hspace{5cm}}} \\ m \Big\downarrow & \begin{bmatrix} k^2_{1,1} \xrightarrow{f_2} \xrightarrow{f_2} \ldots \xrightarrow{f_2} k^2_{1,t} \\ \\ k^2_{m,1} \xrightarrow{f_2} \xrightarrow{f_2} \ldots \xrightarrow{f_2} k^2_{m,t} \end{bmatrix} \\ & \vdots \hspace{3cm} \vdots \\ & \overset{t}{\xrightarrow{\hspace{5cm}}} \\ m \Big\downarrow & \begin{bmatrix} k^{t-1}_{1,1} \xrightarrow{f_{t-1}} \xrightarrow{f_{t-1}} \ldots \xrightarrow{f_{t-1}} k^{t-1}_{1,t} \\ \\ k^{t-1}_{m,1} \xrightarrow{f_{t-1}} \xrightarrow{f_{t-1}} \ldots \xrightarrow{f_{t-1}} k^{t-1}_{m,t} \end{bmatrix} \\ & \overset{t}{\xrightarrow{\hspace{5cm}}} \\ m \Big\downarrow & \begin{bmatrix} k^{t}_{1,1} \xrightarrow{f_{t}} \xrightarrow{f_{t}} \ldots \xrightarrow{f_{t}} k^{t}_{1,t} \\ \\ k^{t}_{m,1} \xrightarrow{f_{t}} \xrightarrow{f_{t}} \ldots \xrightarrow{f_{t}} k^{t}_{m,t} \end{bmatrix} \end{array}$$

Also, rainbow table having size MT x T can be built as

$$\begin{array}{cc} & \overset{t}{\xrightarrow{\hspace{6cm}}} \\ m \times t \left\downarrow \vphantom{\begin{matrix} a \\ \\ \\ \\ \\ \\ \\ \\ b \end{matrix}} \right. & \begin{bmatrix} k_{1,1} \xrightarrow{f_1} \xrightarrow{f_2} \quad \ldots \quad \xrightarrow{f_{t-1}} k_{1,t} \\ \\ \\ \\ \\ \\ \\ \\ k_{mt,1} \xrightarrow{f_1} \xrightarrow{f_2} \quad \ldots \quad \xrightarrow{f_{t-1}} k_{mt,t} \end{bmatrix} \end{array}$$



Examination between progress rates of established and rainbow tables

In the wake of breaking of 500 secret phrase hashes, estimations for great tables with discernable focuses and for rainbow tables

| | classic with DP | rainbow |
|---|---|---|
| $t$, $m$, $\ell$ | 4666, 8192, 4666 | 4666, 38'223'872, 1 |
| predicted coverage | 75.5% | 77.5% |
| measured coverage | 75.8% | 78.8% |

Cryptanalysis measurements with tables yielding 99.9% achievement rate. We can see from the center section that rainbow table requires multiple times less counts.

| | classic with DP | rainbow | ratio | rainbow sequential | ratio |
|---|---|---|---|---|---|
| $t$, $m$, $\ell$ | 4666, 7501, 23330 | 4666, 35M, 5 | 1 | 4666, 35M, 5 | 1 |
| cryptanalysis time | 101.4s | 66.3 | 1.5 | 13.6s | **7.5** |
| hash calculations | 90.3M | 7.4M | **12** | 11.8M | 7.6 |
| false alarms (fa) | 7598 | 1311 | 5.8 | 2773 | 2.7 |
| hashes per fa | 9568 | 4321 | 2.2 | 3080 | 3.1 |
| effort spent on fa | 80% | 76% | 1.1 | 72% | 1.1 |
| success rate | 100% | 100% | 1 | 100% | 1 |

We just need to watch number of particular indicates in the last segment know the check of number of unmistakable chains

$$\hat{P}_{table} = 1 - e^{-t\frac{m_t}{N}} \quad \text{where} \quad m_1 = N \quad \text{and} \quad m_{n+1} = N\left(1 - e^{-\frac{m_n}{N}}\right)$$

### 3.8 Algorithm of triple DES:

In this venture, we are attempting to break an established encryption strategy called Triple DES. Triple DES stands triple Data Encrypt Standard. It is a symmetric key block cipher algorithm in which we apply DES three times with different keys. This algorithm can either use 2 keys or 3 keys. It was defined in various standards namely RFC (which was approved in 1995), ANSI(which was approved in 1998),FIPS(which was approved in 1999) and NIST(which was approved in 2017).

Initially we use the DES cipher which has the key size of 56 bits and was considered sufficient when this algorithm was designed. But with the advancement in technology, the availability of devices which has high computational power made it feasible to have brute force attack on it. So, after this, double DES was designed but laterally, proved to be inefficient. The for its inefficiency in the meet in the middle attack. So at last Triple DES was introduced which is considered to be more efficient. This uses a bundle of 3 DES keys. However, there are some problems with this too.

ciphertext = EK3(DK2(EK1(plaintext)))

```
void DES_Encryption(M,K1)

void DES_Decryption(M,K2);

void DES_Encryption(M,K1)
```

Triple DES Decryption Algorithm :

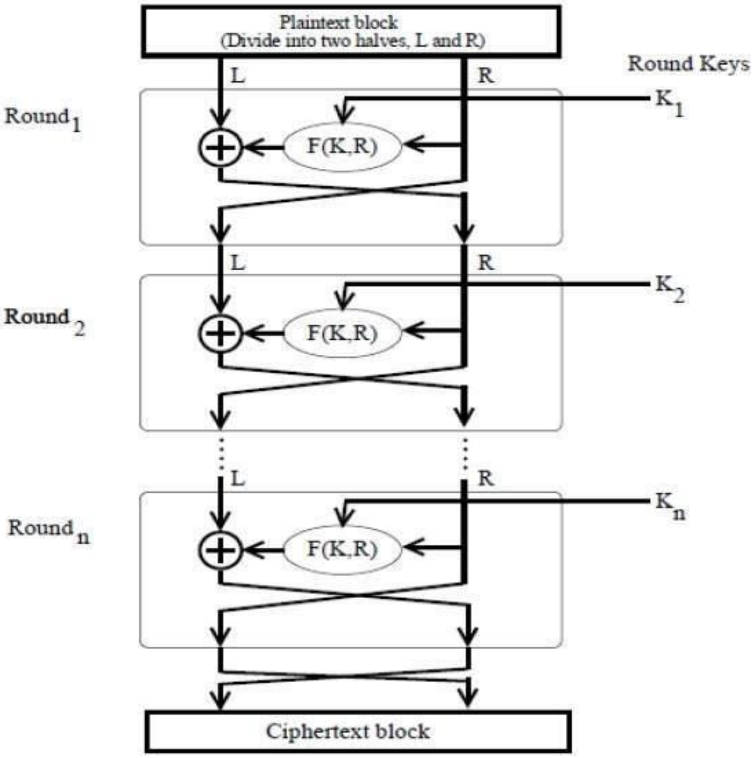
```
void DES_Decryption(M,K1)

void DES_Encryption(M,K2);
```

```
void DES_Decryption(M,K1)
```

In this case, we can also take k1=k3 because it will shorter the key length and security is also not being compromised. Therefore, the total length of key will become 112 bits.

The scrambling is done by permutation table and substitution box. These tables are predefined and can be used to scramble the data into unrecognizable format.

Triple DES uses the same fiestal network to encrypt the data. It comprises of some function which helps to provide randomness in the cipher. The organize comprises of littler squares which independently impart some perplexity. The add up to enter bits are isolated into two equivalent parts(here 32 bits each).They are both treated distinctively with the left bits going undisturbed to the correct segment of the following Feistel block. The right bits are first sustained into a capacity called Feistel work and are then xored with the left 32 bits. This process is done three times using two different keys. Firstly, encrypt by using first key then decrypt using 2nd key and after that, again encrypt using first key. This yield is then take to one side square of the following round.

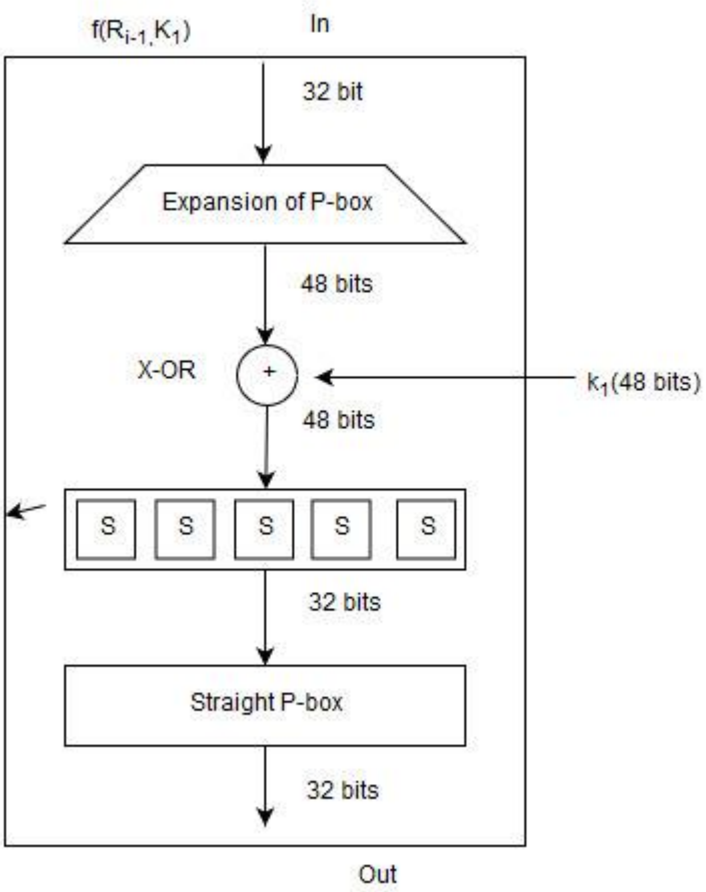


A Classical Feistel Network for triple DES

**Feistel function:**

The feistel function deals with half of the bits at a time. Each experiences through four principle steps.

1.In this progression the info's 32 bits are ventured into 48 bits. This is finished by the utilization of a stage development box. The input is first separated into eight four bits blocks. Now,each of these four bits are ventured into 6 bits.



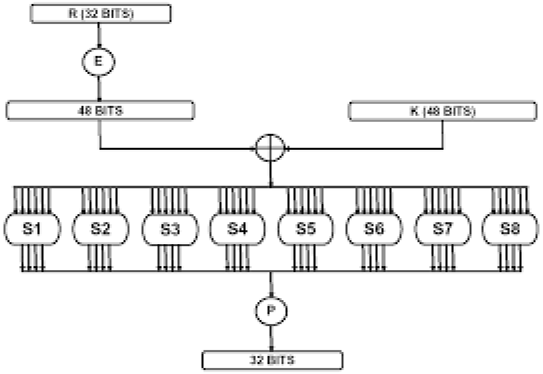
2.The over 48 bits are xored with the 48 bits of the key. It is to be noticed that the info key size was just 56 bits. So,here we utilize a pseudo arbitrary generator to really deliver the required key. The step is known as the key-blending step.

3.In this stage a substitution box is used. This is utilized to change over the 48 bits yield to 32 bits. The input estimate is isolated into 6 block,each of 8 bits. Now we have a two dimensional grid containing 4 rows and 16 columns. The first and the last piece are utilized to discover the line and the center four bits are utilized to find the column. Each cell has a 4 bit esteem and we substitute the given 6 bits with these 4 bits. After doing this for every one of the 8 squares we will have a subsequent size of 8*4 i.e. 32 bits. It is to be remembered that we have to keep any sort of linearity in the development of these s-boxes. Moreover, for each of the 8 squares of information distinctive s-box is utilized.

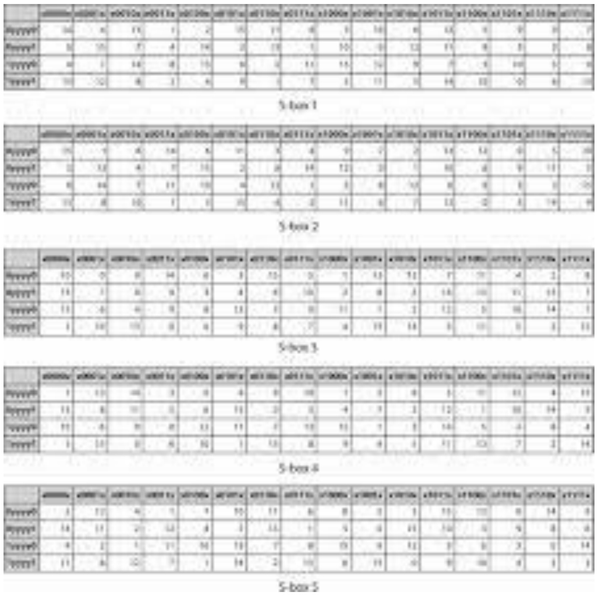
4.Now the yield 32 bits are spread out utilizing a stage block. The fundamental usefulness of a p-box is to scramble the yield and to appropriate the impact of the s-boxes to a more extensive range. This is known as the change step.

R1=L0 L1=f(r)^L0

The Key-Mixing step



Sample



Substitution Box

**Key Scheduling Algorithm**

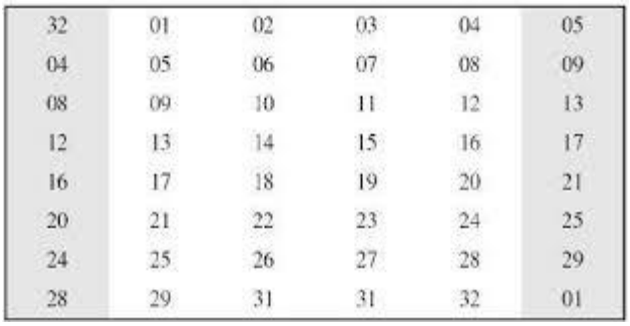
For each capacity in the feistel round we require a key of 48 bits. As a considerable measure of such advances including diverse keys can be there we can't utilize the information key directly. Hence,we require a key booking calculation.

Steps engaged with key booking:

1.The 56 bits input enter bits are isolated into two parts. These keys have a size of 28 bits.

2.The bits are moved by limited positions. The first,second,ninth and sixteenth positions are moved by 1 bit to the right. All the remaining bits are moved by 2 bits positions to right.

3.Now a D-box is used. The bits changes over 28 bits of the key into 24 bits. The left and the correct piece of the are presently joined to give us the 48 bits which are utilized for the encryption procedure.



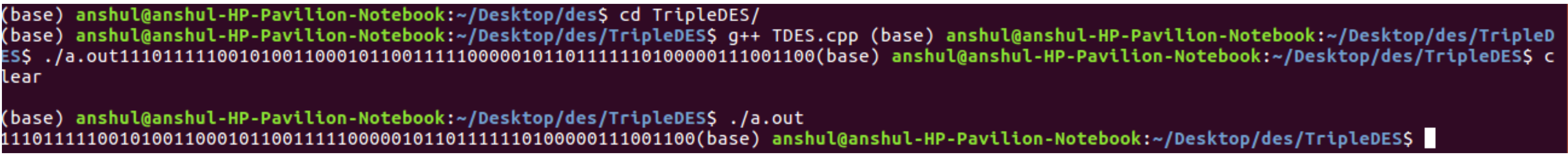
| | | | | | |
|---|---|---|---|---|---|
| 32 | 01 | 02 | 03 | 04 | 05 |
| 04 | 05 | 06 | 07 | 08 | 09 |
| 08 | 09 | 10 | 11 | 12 | 13 |
| 12 | 13 | 14 | 15 | 16 | 17 |
| 16 | 17 | 18 | 19 | 20 | 21 |
| 20 | 21 | 22 | 23 | 24 | 25 |
| 24 | 25 | 26 | 27 | 28 | 29 |
| 28 | 29 | 31 | 31 | 32 | 01 |

### A Sample D-Box

The initial three row of the table are utilized for the pressure of the left piece of the key while the lower three keys are utilized to diminish the extent of the correct part.

### Triple Data Encryption Standard (output)

This calculation is worked by the utilization of Feistel network. Sixteen Feistel squares are utilized in DES. The square size for this calculation is 64 bits. Hence,triple DES is only Feistel rounds connected multiple times.



```
(base) anshul@anshul-HP-Pavilion-Notebook:~/Desktop/des$ cd TripleDES/
(base) anshul@anshul-HP-Pavilion-Notebook:~/Desktop/des/TripleDES$ g++ TDES.cpp (base) anshul@anshul-HP-Pavilion-Notebook:~/Desktop/des/TripleD
ES$ ./a.out111011111001010011000101100111110000001011011111101000000111001100(base) anshul@anshul-HP-Pavilion-Notebook:~/Desktop/des/TripleDES$ c
lear

(base) anshul@anshul-HP-Pavilion-Notebook:~/Desktop/des/TripleDES$ ./a.out
111011111001010011000101100111110000001011011111101000000111001100(base) anshul@anshul-HP-Pavilion-Notebook:~/Desktop/des/TripleDES$ █
```

### 3.9 Hellman Analysis

In this methodology, we pick M begin focuses and settle a chain length by size T. This can be effectively distributed to reduce both time and memory. It also helps in Managing intense issues for which no effective calculation is accessible memory is saved by storing in a table just the start and end of chains and it a kind of Chosen plaintext attack.

### 3.10 Pre-calculation Phase

In this stage, we manufacture the table which we later use for enormous
power in the online stage.
In the table, we endeavor to store all the conceivable ciphertexts

Table : m*t[m beginning points,t columns]

In any case, because of space imperatives, we just spare first and last segment

$C=S_k(P)$

Now the output of the above step is of 64 bits. But the size of the key is 112 bits. So, we use a expansion function.

$f(K)=R[S_k(P)]$

**Algorithm:**

●M irregular purposes of 112 bits are chosen. These fill in as the beginning stage or key as in triple DES.

● For each beginning stage a chain of length t is created.

● The beginning stage is taken and is encouraged into the calculation.

● If the yield measure is not quite the same as the required size(112 bits as in triple DES) the yield bits are concatenated up by some initial bits of key.

● a similar technique is rehashed t times.

● The beginning and the end points will be stored.

● The above advances are rehashed for m diverse begin focuses.

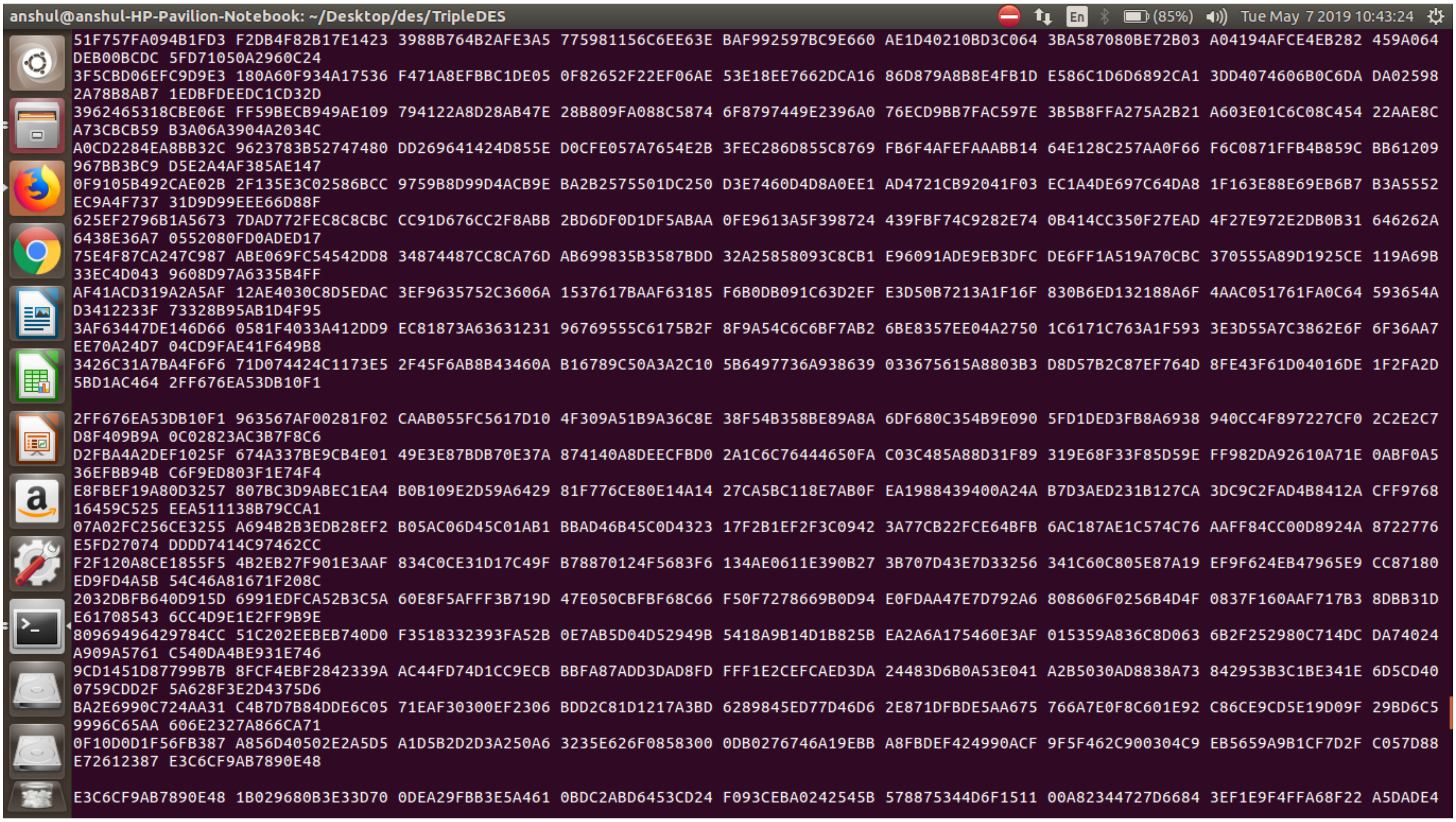
$$SP_1 = x_{10} \xrightarrow{f} x_{11} = f(x_{10}) \xrightarrow{f} x_{12} = f(x_{11}) \xrightarrow{f} \ldots \xrightarrow{f} x_{1t} = f(x_{1(t-1)}) = EP_1$$

$$SP_2 = x_{20} \xrightarrow{f} x_{21} = f(x_{20}) \xrightarrow{f} x_{22} = f(x_{21}) \xrightarrow{f} \ldots \xrightarrow{f} x_{2t} = f(x_{2(t-1)}) = EP_2$$

$$\vdots \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \vdots$$

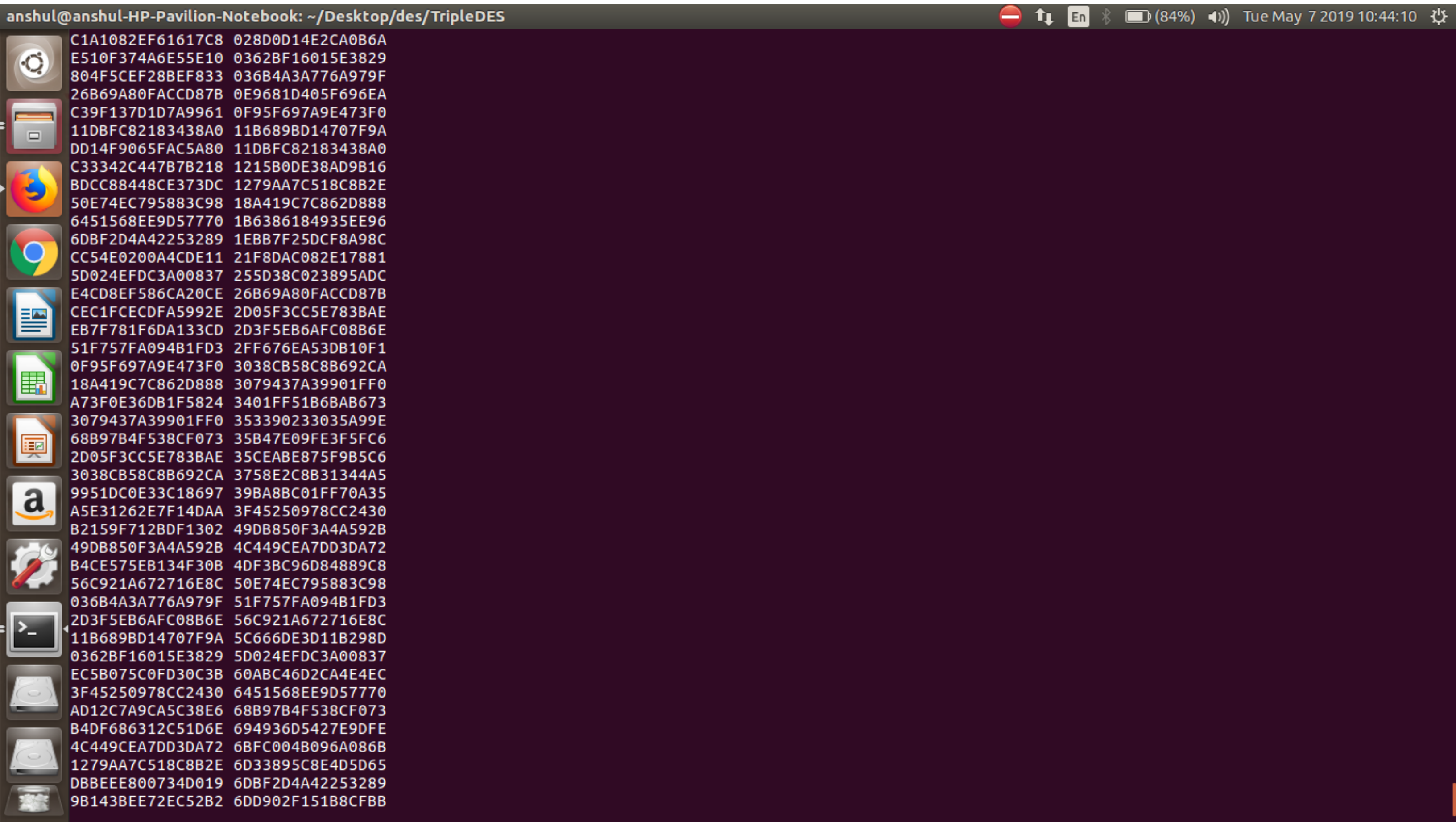
$$SP_m = x_{m0} \xrightarrow{f} x_{m1} = \xrightarrow{f} x_{m2} = \xrightarrow{f} \ldots \xrightarrow{f} x_{mt} = f(x_{m(t-1)}) =$$

# Output of Pre-calculation phase:



```
51F757FA094B1FD3 F2DB4F82B17E1423 3988B764B2AFE3A5 775981156C6EE63E BAF992597BC9E660 AE1D40210BD3C064 3BA587080BE72B03 A04194AFCE4EB282 459A064
DEB00BCDC 5FD71050A2960C24
3F5CBD06EFC9D9E3 180A60F934A17536 F471A8EFBBC1DE05 0F82652F22EF06AE 53E18EE7662DCA16 86D879A8B8E4FB1D E586C1D6D6892CA1 3DD4074606B0C6DA DA02598
2A78B8AB7 1EDBFDEEDC1CD32D
3962465318CBE06E FF59BECB949AE109 794122A8D28AB47E 28B809FA088C5874 6F8797449E2396A0 76ECD9BB7FAC597E 3B5B8FFA275A2B21 A603E01C6C08C454 22AAE8C
A73CBCB59 B3A06A3904A2034C
A0CD2284EA8BB32C 9623783B52747480 DD269641424D855E D0CFE057A7654E2B 3FEC286D855C8769 FB6F4AFEFAAABB14 64E128C257AA0F66 F6C0871FFB4B859C BB61209
967BB3BC9 D5E2A4AF385AE147
0F9105B492CAE02B 2F135E3C02586BCC 9759B8D99D4ACB9E BA2B2575501DC250 D3E7460D4D8A0EE1 AD4721CB92041F03 EC1A4DE697C64DA8 1F163E88E69EB6B7 B3A5552
EC9A4F737 31D9D99EEE66D88F
625EF2796B1A5673 7DAD772FEC8C8CBC CC91D676CC2F8ABB 2BD6DF0D1DF5ABAA 0FE9613A5F398724 439FBF74C9282E74 0B414CC350F27EAD 4F27E972E2DB0B31 646262A
6438E36A7 0552080FD0ADED17
75E4F87CA247C987 ABE069FC54542DD8 34874487CC8CA76D AB699835B3587BDD 32A25858093C8CB1 E96091ADE9EB3DFC DE6FF1A519A70CBC 370555A89D1925CE 119A69B
33EC4D043 9608D97A6335B4FF
AF41ACD319A2A5AF 12AE4030C8D5EDAC 3EF9635752C3606A 1537617BAAF63185 F6B0DB091C63D2EF E3D50B7213A1F16F 830B6ED132188A6F 4AAC051761FA0C64 593654A
D3412233F 73328B95AB1D4F95
3AF63447DE146D66 0581F4033A412DD9 EC81873A63631231 96769555C6175B2F 8F9A54C6C6BF7AB2 6BE8357EE04A2750 1C6171C763A1F593 3E3D55A7C3862E6F 6F36AA7
EE70A24D7 04CD9FAE41F649B8
3426C31A7BA4F6F6 71D074424C1173E5 2F45F6AB8B43460A B16789C50A3A2C10 5B6497736A938639 033675615A8803B3 D8D57B2C87EF764D 8FE43F61D04016DE 1F2FA2D
5BD1AC464 2FF676EA53DB10F1

2FF676EA53DB10F1 963567AF00281F02 CAAB055FC5617D10 4F309A51B9A36C8E 38F54B358BE89A8A 6DF680C354B9E090 5FD1DED3FB8A6938 940CC4F897227CF0 2C2E2C7
D8F409B9A 0C02823AC3B7F8C6
D2FBA4A2DEF1025F 674A337BE9CB4E01 49E3E87BDB70E37A 874140A8DEECFBD0 2A1C6C76444650FA C03C485A88D31F89 319E68F33F85D59E FF982DA92610A71E 0ABF0A5
36EFBB94B C6F9ED803F1E74F4
E8FBEF19A80D3257 807BC3D9ABEC1EA4 B0B109E2D59A6429 81F776CE80E14A14 27CA5BC118E7AB0F EA1988439400A24A B7D3AED231B127CA 3DC9C2FAD4B8412A CFF9768
16459C525 EEA511138B79CCA1
07A02FC256CE3255 A694B2B3EDB28EF2 B05AC06D45C01AB1 BBAD46B45C0D4323 17F2B1EF2F3C0942 3A77CB22FCE64BFB 6AC187AE1C574C76 AAFF84CC00D8924A 8722776
E5FD27074 DDDD7414C97462CC
F2F120A8CE1855F5 4B2EB27F901E3AAF 834C0CE31D17C49F B78870124F5683F6 134AE0611E390B27 3B707D43E7D33256 341C60C805E87A19 EF9F624EB47965E9 CC87180
ED9FD4A5B 54C46A81671F208C
2032DBFB640D915D 6991EDFCA52B3C5A 60E8F5AFFF3B719D 47E050CBFBF68C66 F50F7278669B0D94 E0FDAA47E7D792A6 808606F0256B4D4F 0837F160AAF717B3 8DBB31D
E61708543 6CC4D9E1E2FF9B9E
80969496429784CC 51C202EEBEB740D0 F3518332393FA52B 0E7AB5D04D52949B 5418A9B14D1B825B EA2A6A175460E3AF 015359A836C8D063 6B2F252980C714DC DA74024
A909A5761 C540DA4BE931E746
9CD1451D87799B7B 8FCF4EBF2842339A AC44FD74D1CC9ECB BBFA87ADD3DAD8FD FFF1E2CEFCAED3DA 24483D6B0A53E041 A2B5030AD8838A73 842953B3C1BE341E 6D5CD40
0759CDD2F 5A628F3E2D4375D6
BA2E6990C724AA31 C4B7D7B84DDE6C05 71EAF30300EF2306 BDD2C81D1217A3BD 6289845ED77D46D6 2E871DFBDE5AA675 766A7E0F8C601E92 C86CE9CD5E19D09F 29BD6C5
9996C65AA 606E2327A866CA71
0F10D0D1F56FB387 A856D40502E2A5D5 A1D5B2D2D3A250A6 3235E626F0858300 0DB0276746A19EBB A8FBDEF424990ACF 9F5F462C900304C9 EB5659A9B1CF7D2F C057D88
E72612387 E3C6CF9AB7890E48

E3C6CF9AB7890E48 1B029680B3E33D70 0DEA29FBB3E5A461 0BDC2ABD6453CD24 F093CEBA0242545B 578875344D6F1511 00A82344727D6684 3EF1E9F4FFA68F22 A5DADE4
```

## **Storing initial and final column of the Precomputed table:**



```
C1A1082EF61617C8 028D0D14E2CA0B6A
E510F374A6E55E10 0362BF16015E3829
804F5CEF28BEF833 036B4A3A776A979F
26B69A80FACCD87B 0E9681D405F696EA
C39F137D1D7A9961 0F95F697A9E473F0
11DBFC82183438A0 11B689BD14707F9A
DD14F9065FAC5A80 11DBFC82183438A0
C33342C447B7B218 1215B0DE38AD9B16
BDCC88448CE373DC 1279AA7C518C8B2E
50E74EC795883C98 18A419C7C862D888
6451568EE9D57770 1B6386184935EE96
6DBF2D4A42253289 1EBB7F25DCF8A98C
CC54E0200A4CDE11 21F8DAC082E17881
5D024EFDC3A00837 255D38C023895ADC
E4CD8EF586CA20CE 26B69A80FACCD87B
CEC1FCECDFA5992E 2D05F3CC5E783BAE
EB7F781F6DA133CD 2D3F5EB6AFC08B6E
51F757FA094B1FD3 2FF676EA53DB10F1
0F95F697A9E473F0 3038CB58C8B692CA
18A419C7C862D888 3079437A39901FF0
A73F0E36DB1F5824 3401FF51B6BAB673
3079437A39901FF0 353390233035A99E
68B97B4F538CF073 35B47E09FE3F5FC6
2D05F3CC5E783BAE 35CEABE875F9B5C6
3038CB58C8B692CA 3758E2C8B31344A5
9951DC0E33C18697 39BA8BC01FF70A35
A5E31262E7F14DAA 3F45250978CC2430
B2159F712BDF1302 49DB850F3A4A592B
49DB850F3A4A592B 4C449CEA7DD3DA72
B4CE575EB134F30B 4DF3BC96D84889C8
56C921A672716E8C 50E74EC795883C98
036B4A3A776A979F 51F757FA094B1FD3
2D3F5EB6AFC08B6E 56C921A672716E8C
11B689BD14707F9A 5C666DE3D11B298D
0362BF16015E3829 5D024EFDC3A00837
EC5B075C0FD30C3B 60ABC46D2CA4E4EC
3F45250978CC2430 6451568EE9D57770
AD12C7A9CA5C38E6 68B97B4F538CF073
B4DF686312C51D6E 694936D5427E9DFE
4C449CEA7DD3DA72 6BFC004B096A086B
1279AA7C518C8B2E 6D33895C8E4D5D65
DBBEEE800734D019 6DBF2D4A42253289
9B143BEE72EC52B2 6DD902F151B8CFBB
```

### 3.11 Online Phase:

This stage includes looking the table for therequired
The sections of the table is arranged.

Look time for n tables: n log(ne)

**Algorithm:**

● This stage includes scanning the table for the required

ciphertext.

● Firstly, the ciphertext is concatenated up to 112 bits.

● Now the end points of the different chains are looked for the equivalent ciphertext bits.

● If similar bits are experienced, the chain is recovered to discover the t section and the yield of this t in the wake of applying the decrease work is the key.

● If the bits don't coordinate then the diminished ciphertext bits

are encoded and lessened p time 1<=p<=T.

● Every point of time they will be matched corresponding to the required

ciphertext's bits.

● When a match is discovered ,the chain is produced T-p times

also, we get the key from that point.

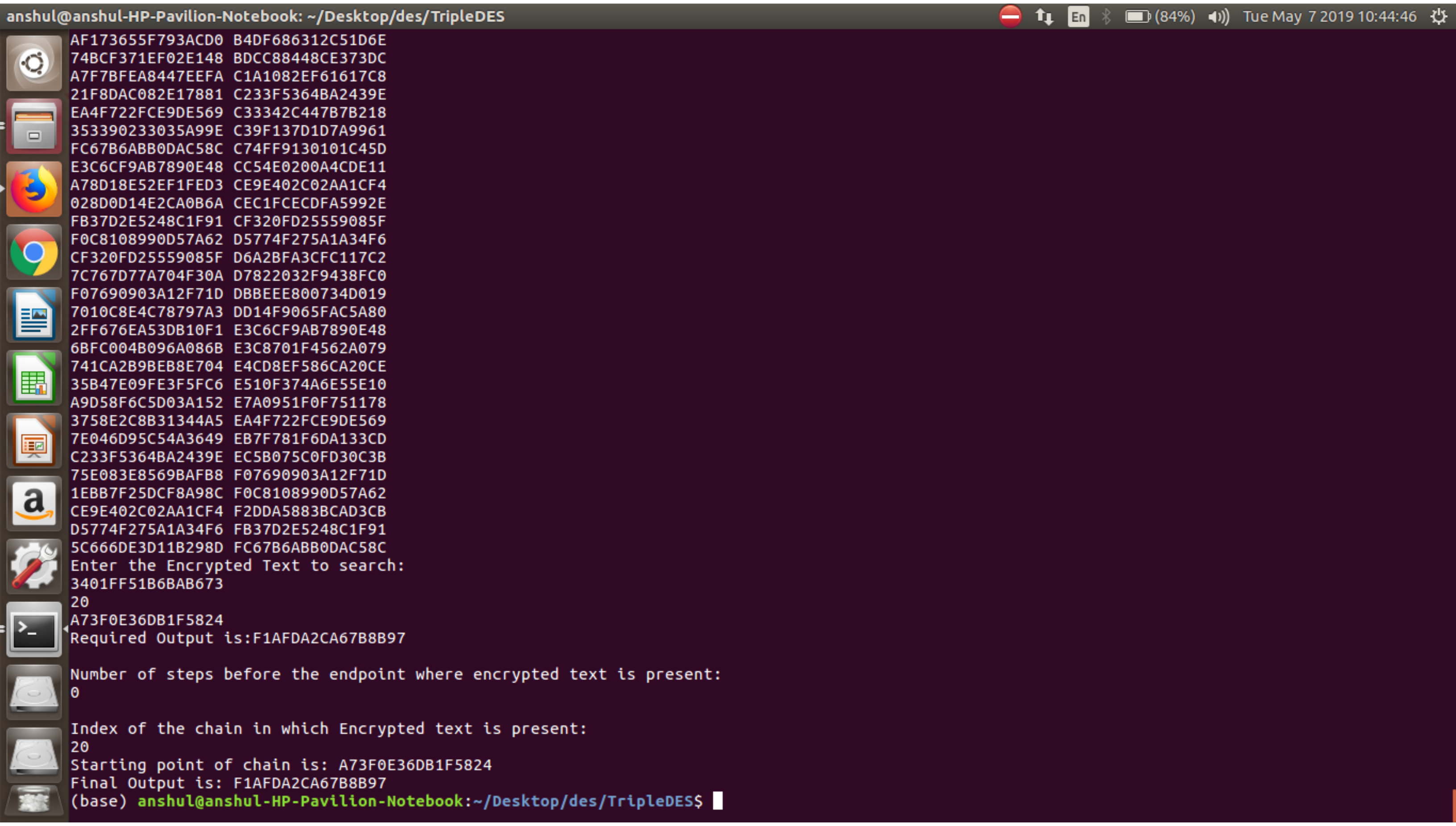
Hellman recommended to pursue the connection:

M=m*t

mt2=N

m=t=N(1/3)

# Output of Pre-calculation phase:



```
AF173655F793ACD0 B4DF686312C51D6E
74BCF371EF02E148 BDCC88448CE373DC
A7F7BFEA8447EEFA C1A1082EF61617C8
21F8DAC082E17881 C233F5364BA2439E
EA4F722FCE9DE569 C33342C447B7B218
353390233035A99E C39F137D1D7A9961
FC67B6ABB0DAC58C C74FF9130101C45D
E3C6CF9AB7890E48 CC54E0200A4CDE11
A78D18E52EF1FED3 CE9E402C02AA1CF4
028D0D14E2CA0B6A CEC1FCECDFA5992E
FB37D2E5248C1F91 CF320FD25559085F
F0C8108990D57A62 D5774F275A1A34F6
CF320FD25559085F D6A2BFA3CFC117C2
7C767D77A704F30A D7822032F9438FC0
F07690903A12F71D DBBEEE800734D019
7010C8E4C78797A3 DD14F9065FAC5A80
2FF676EA53DB10F1 E3C6CF9AB7890E48
6BFC004B096A086B E3C8701F4562A079
741CA2B9BEB8E704 E4CD8EF586CA20CE
35B47E09FE3F5FC6 E510F374A6E55E10
A9D58F6C5D03A152 E7A0951F0F751178
3758E2C8B31344A5 EA4F722FCE9DE569
7E046D95C54A3649 EB7F781F6DA133CD
C233F5364BA2439E EC5B075C0FD30C3B
75E083E8569BAFB8 F07690903A12F71D
1EBB7F25DCF8A98C F0C8108990D57A62
CE9E402C02AA1CF4 F2DDA5883BCAD3CB
D5774F275A1A34F6 FB37D2E5248C1F91
5C666DE3D11B298D FC67B6ABB0DAC58C
Enter the Encrypted Text to search:
3401FF51B6BAB673
20
A73F0E36DB1F5824
Required Output is:F1AFDA2CA67B8B97

Number of steps before the endpoint where encrypted text is present:
0

Index of the chain in which Encrypted text is present:
20
Starting point of chain is: A73F0E36DB1F5824
Final Output is: F1AFDA2CA67B8B97
(base) anshul@anshul-HP-Pavilion-Notebook:~/Desktop/des/TripleDES$ 
```

**Enhancements by Perfect Table:**

In 2003 Philippe Oechslin demonstrated that the execution of the TMTO can be expanded radically by utilizing unique expansion function. The likelihood of crash between two chains diminished to 1/t.

Rainbow chains have the accompanying points of interest:

- Rainbow Tables have the advantage the individual building those tables can pick how much stockpiling is required by choosing the quantity of connections in each chain.
- The more connections between the seed and the last esteem, the more passwords are caught. One shortcoming is that the individual structure the chains doesn't pick the passwords they catch so Rainbow Tables can't be improved for normal passwords.
- Hash tables are useful for normal passwords, Rainbow Tables are useful for intense passwords. The best methodology is recoup whatever number passwords as would be prudent utilizing hash tables or potentially customary breaking with a word reference of the top N passwords.

Given M(memory available),N, and P(required likelihood

of progress), the ideal parameters of the exchange off

that limit the cryptanalysis time are:

$L=-\{\ln(1-p)/2\}$

$m'=M'/L$

$T=(- \{N*\ln(1-P)\})/M'$

Utilizing the above parameters we attempted actualized TMTO

for DES :

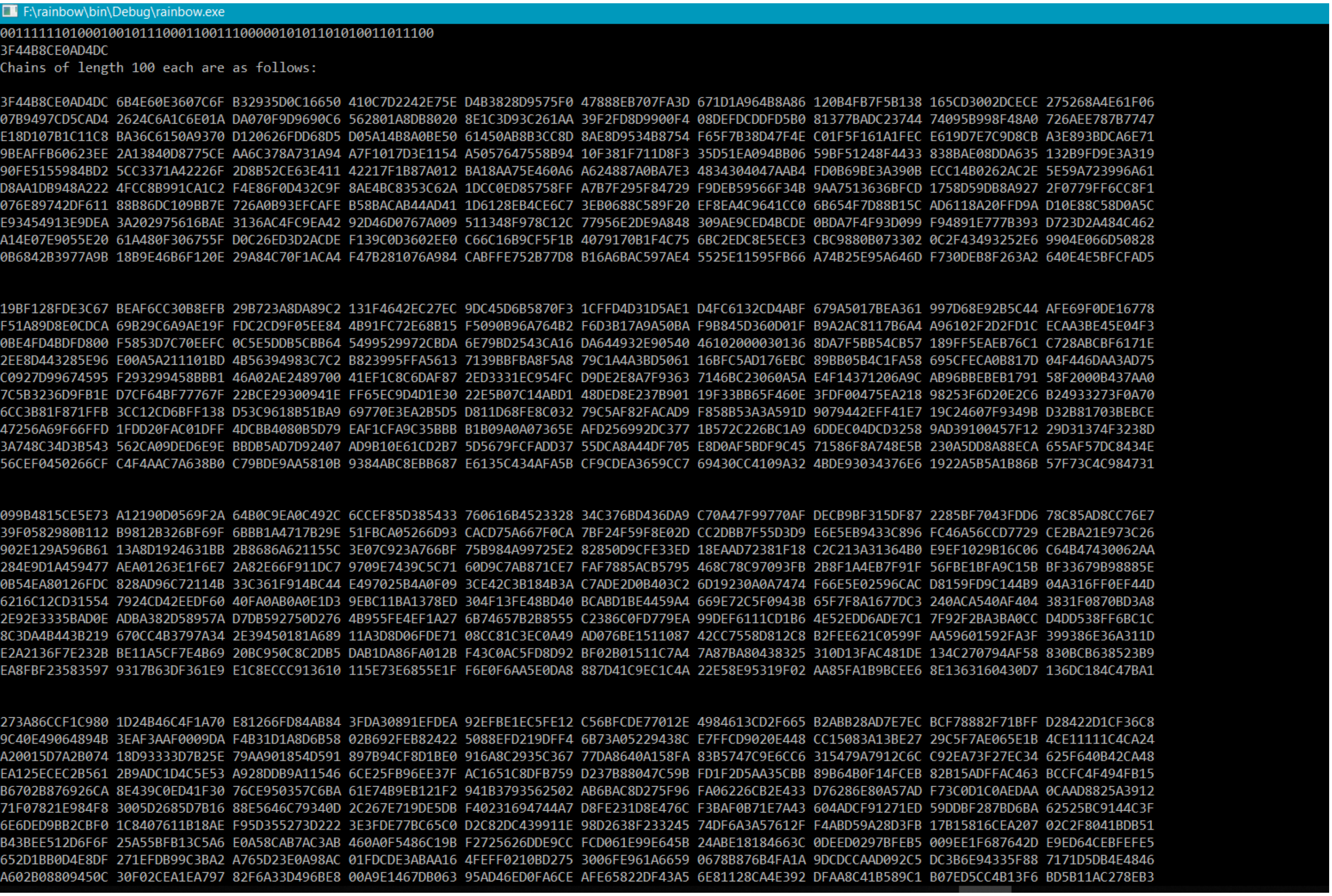
$p=.86$

$L=1$

$m'=34359738368$

$T=262144$

# Output of Rainbow table



```
00111111010001001011100011001110000010101101010011011100
3F44B8CE0AD4DC
Chains of length 100 each are as follows:

3F44B8CE0AD4DC 6B4E60E3607C6F B32935D0C16650 410C7D2242E75E D4B3828D9575F0 47888EB707FA3D 671D1A964B8A86 120B4FB7F5B138 165CD3002DCECE 275268A4E61F06
07B9497CD5CAD4 2624C6A1C6E01A DA070F9D9690C6 562801A8DB8020 8E1C3D93C261AA 39F2FD8D9900F4 08DEFDCDDFD5B0 81377BADC23744 74095B998F48A0 726AEE787B7747
E18D107B1C11C8 BA36C6150A9370 D120626FDD68D5 D05A14B8A0BE50 61450AB8B3CC8D 8AE8D9534B8754 F65F7B38D47F4E C01F5F161A1FEC E619D7E7C9D8CB A3E893BDCA6E71
9BEAFFB60623EE 2A13840D8775CE AA6C378A731A94 A7F1017D3E1154 A5057647558B94 10F381F711D8F3 35D51EA094BB06 59BF51248F4433 838BAE08DDA635 132B9FD9E3A319
90FE5155984BD2 5CC3371A42226F 2D8B52CE63E411 42217F1B87A012 BA18AA75E460A6 A624887A0BA7E3 4834304047AAB4 FD0B69BE3A390B ECC14B0262AC2E 5E59A723996A61
D8AA1DB948A222 4FCC8B991CA1C2 F4E86F0D432C9F 8AE4BC8353C62A 1DCC0ED85758FF A7B7F295F84729 F9DEB59566F34B 9AA7513636BFCD 1758D59DB8A927 2F0779FF6CC8F1
076E89742DF611 88B86DC109BB7E 726A0B93EFCAFE B58BACAB44AD41 1D6128EB4CE6C7 3EB0688C589F20 EF8EA4C9641CC0 6B654F7D88B15C AD6118A20FFD9A D10E88C58D0A5C
E93454913E9DEA 3A202975616BAE 3136AC4FC9EA42 92D46D0767A009 511348F978C12C 77956E2DE9A848 309AE9CED4BCDE 0BDA7F4F93D099 F94891E777B393 D723D2A484C462
A14E07E9055E20 61A480F306755F D0C26ED3D2ACDE F139C0D3602EE0 C66C16B9CF5F1B 4079170B1F4C75 6BC2EDC8E5ECE3 CBC9880B073302 0C2F43493252E6 9904E066D50828
0B6842B3977A9B 18B9E46B6F120E 29A84C70F1ACA4 F47B281076A984 CABFFE752B77D8 B16A6BAC597AE4 5525E11595FB66 A74B25E95A646D F730DEB8F263A2 640E4E5BFCFAD5


19BF128FDE3C67 BEAF6CC30B8EFB 29B723A8DA89C2 131F4642EC27EC 9DC45D6B5870F3 1CFFD4D31D5AE1 D4FC6132CD4ABF 679A5017BEA361 997D68E92B5C44 AFE69F0DE16778
F51A89D8E0CDCA 69B29C6A9AE19F FDC2CD9F05EE84 4B91FC72E68B15 F5090B96A764B2 F6D3B17A9A50BA F9B845D360D01F B9A2AC8117B6A4 A96102F2D2FD1C ECAA3BE45E04F3
0BE4FD4BDFD800 F5853D7C70EEFC 0C5E5DDB5CBB64 5499529972CBDA 6E79BD2543CA16 DA644932E90540 46102000030136 8DA7F5BB54CB57 189FF5EAEB76C1 C728ABCBF6171E
2EE8D443285E96 E00A5A211101BD 4B56394983C7C2 B823995FFA5613 7139BBFBA8F5A8 79C1A4A3BD5061 16BFC5AD176EBC 89BB05B4C1FA58 695CFECA0B817D 04F446DAA3AD75
C0927D99674595 F293299458BBB1 46A02AE2489700 41EF1C8C6DAF87 2ED3331EC954FC D9DE2E8A7F9363 7146BC23060A5A E4F14371206A9C AB96BBEBEB1791 58F2000B437AA0
7C5B3236D9FB1E D7CF64BF77767F 22BCE29300941E FF65EC9D4D1E30 22E5B07C14ABD1 48DED8E237B901 19F33BB65F460E 3FDF00475EA218 98253F6D20E2C6 B24933273F0A70
6CC3B81F871FFB 3CC12CD6BFF138 D53C9618B51BA9 69770E3EA2B5D5 D811D68FE8C032 79C5AF82FACAD9 F858B53A3A591D 9079442EFF41E7 19C24607F9349B D32B81703BEBCE
47256A69F66FFD 1FDD20FAC01DFF 4DCBB4080B5D79 EAF1CFA9C35BBB B1B09A0A07365E AFD256992DC377 1B572C226BC1A9 6DDEC04DCD3258 9AD39100457F12 29D31374F3238D
3A748C34D3B543 562CA09DED6E9E BBDB5AD7D92407 AD9B10E61CD2B7 5D5679FCFADD37 55DCA8A44DF705 E8D0AF5BDF9C45 71586F8A748E5B 230A5DD8A88ECA 655AF57DC8434E
56CEF0450266CF C4F4AAC7A638B0 C79BDE9AA5810B 9384ABC8EBB687 E6135C434AFA5B CF9CDEA3659CC7 69430CC4109A32 4BDE93034376E6 1922A5B5A1B86B 57F73C4C984731


099B4815CE5E73 A12190D0569F2A 64B0C9EA0C492C 6CCEF85D385433 760616B4523328 34C376BD436DA9 C70A47F99770AF DECB9BF315DF87 2285BF7043FDD6 78C85AD8CC76E7
39F0582980B112 B9812B326BF69F 6BBB1A4717B29E 51FBCA05266D93 CACD75A667F0CA 7BF24F59F8E02D CC2DBB7F55D3D9 E6E5EB9433C896 FC46A56CCD7729 CE2BA21E973C26
902E129A596B61 13A8D1924631BB 2B8686A621155C 3E07C923A766BF 75B984A99725E2 82850D9CFE33ED 18EAAD72381F18 C2C213A31364B0 E9EF1029B16C06 C64B47430062AA
284E9D1A459477 AEA01263E1F6E7 2A82E66F911DC7 9709E7439C5C71 60D9C7AB871CE7 FAF7885ACB5795 468C78C97093FB 2B8F1A4EB7F91F 56FBE1BFA9C15B BF33679B98885E
0B54EA80126FDC 828AD96C72114B 33C361F914BC44 E497025B4A0F09 3CE42C3B184B3A C7ADE2D0B403C2 6D19230A0A7474 F66E5E02596CAC D8159FD9C144B9 04A316FF0EF44D
6216C12CD31554 7924CD42EEDF60 40FA0AB0A0E1D3 9EBC11BA1378ED 304F13FE48BD40 BCABD1BE4459A4 669E72C5F0943B 65F7F8A1677DC3 240ACA540AF404 3831F0870BD3A8
2E92E3335BAD0E ADBA382D58957A D7DB592750D276 4B955FE4EF1A27 6B74657B2B8555 C2386C0FD779EA 99DEF6111CD1B6 4E52EDD6ADE7C1 7F92F2BA3BA0CC D4DD538FF6BC1C
8C3DA4B443B219 670CC4B3797A34 2E39450181A689 11A3D8D06FDE71 08CC81C3EC0A49 AD076BE1511087 42CC7558D812C8 B2FEE621C0599F AA59601592FA3F 399386E36A311D
E2A2136F7E232B BE11A5CF7E4B69 20BC950C8C2DB5 DAB1DA86FA012B F43C0AC5FD8D92 BF02B01511C7A4 7A87BA80438325 310D13FAC481DE 134C270794AF58 830BCB638523B9
EA8FBF23583597 9317B63DF361E9 E1C8ECCC913610 115E73E6855E1F F6E0F6AA5E0DA8 887D41C9EC1C4A 22E58E95319F02 AA85FA1B9BCEE6 8E1363160430D7 136DC184C47BA1


273A86CCF1C980 1D24B46C4F1A70 E81266FD84AB84 3FDA30891EFDEA 92EFBE1EC5FE12 C56BFCDE77012E 4984613CD2F665 B2ABB28AD7E7EC BCF78882F71BFF D28422D1CF36C8
9C40E49064894B 3EAF3AAF0009DA F4B31D1A8D6B58 02B692FEB82422 5088EFD219DFF4 6B73A05229438C E7FFCD9020E448 CC15083A13BE27 29C5F7AE065E1B 4CE11111C4CA24
A20015D7A2B074 18D93333D7B25E 79AA901854D591 897B94CF8D1BE0 916A8C2935C367 77DA8640A158FA 83B5747C9E6CC6 315479A7912C6C C92EA73F27EC34 625F640B42CA48
EA125ECEC2B561 2B9ADC1D4C5E53 A928DDB9A11546 6CE25FB96EE37F AC1651C8DFB759 D237B88047C59B FD1F2D5AA35CBB 89B64B0F14FCEB 82B15ADFFAC463 BCCFC4F494FB15
B6702B876926CA 8E439C0ED41F30 76CE950357C6BA 61E74B9EB121F2 941B3793562502 AB6BAC8D275F96 FA06226CB2E433 D76286E80A57AD F73C0D1C0AEDAA 0CAAD8825A3912
71F07821E984F8 3005D2685D7B16 88E5646C79340D 2C267E719DE5DB F40231694744A7 D8FE231D8E476C F3BAF0B71E7A43 604ADCF91271ED 59DDBF287BD6BA 62525BC9144C3F
6E6DED9BB2CBF0 1C8407611B18AE F95D355273D222 3E3FDE77BC65C0 D2C82DC439911E 98D2638F233245 74DF6A3A57612F F4ABD59A28D3FB 17B15816CEA207 02C2F8041BDB51
B43BEE512D6F6F 25A55BFB13C5A6 E0A58CAB7AC3AB 460A0F5486C19B F2725626DDE9CC FCD061E99E645B 24ABE18184663C 0DEED0297BFEB5 009EE1F687642D E9ED64CEBFEFE5
652D1BB0D4E8DF 271EFDB99C3BA2 A765D23E0A98AC 01FDCDE3ABAA16 4FEFF0210BD275 3006FE961A6659 0678B876B4FA1A 9DCDCCAAD092C5 DC3B6E94335F88 7171D5DB4E4846
A602B08809450C 30F02CEA1EA797 82F6A33D496BE8 00A9E1467DB063 95AD46ED0FA6CE AFE65822DF43A5 6E81128CA4E392 DFAA8C41B589C1 B07ED5CC4B13F6 BD5B11AC278EB3
```

# Chapter 5

## Performance analysis

The proposed framework needs a high handling power. We are keen on executing the perfect rainbow tables. This technique includes formation of the table for which the Triple DES encryption calculation must be utilized a few times. According to the proposed plan we pick the proper parameters and make the tables. Rainbow tables can be created in around 13 days. The expected encryption rate for our Triple DES implementation is around 78 encryptions/sec. This will create countless aggregate size of these encryptions will be about 3 * 2^42 GB. Do we mean to store all of these?No.(Even on the off chance that we do,we won't be capable to)!!We will simply be picking a restricted number of beginning stages with a settled chain length. We will spare just the sets comprising of the beginning stage and the end point.

Be that as it may, we have another inquiry before us. How are we going to accomplish 78 encryptions/sec. Our streamlined execution of triple DES joins bit slicing. A method suggested by Philipp Grabher, Johann Großsch¨adl, and Dan Page in 1996 which can significantly expand the speed. With our upgraded usage and bit cutting we achieved a speed of about 2^22 on an intel i5, fourth age machine. Now for expanding the speed we would require devoted gpu's and need to code them for our requirement. The preferred framework would be a workstation comprising of 8 GTX 1080i gpu's working simultaneously. Where would we be able to get these?Provided the circumstance we don't have a workstation with the necessities referenced we can contract a cloud framework to do this work for us. After considering a few administrations we have chosen to run with vscaler cloud services. The cost involved surely is a main consideration required here.

Following measurements for investigation ought to be utilized:

1.Time

2.Avalanche effect

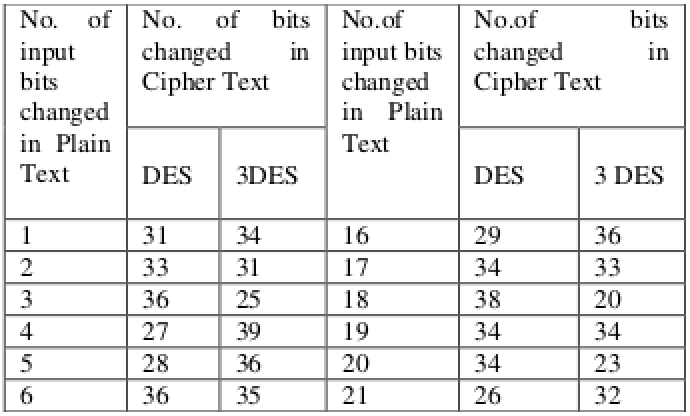
3.Money

4.Success Probability

The hardware we are using comprises of 2.3 ghz processor, 4gb RAM, 64bit OS (windows 10). Also, we have I5 5 generation processors(octa core) and the language we are using is C++.

**1. Time:**

With any encryption algorithm, time becomes an important factor to measure to determine the nature of algorithm and how good it is to be implemented. We can compare the DES and triple DES in terms of execution time. DES takes 92 milliseconds to encrypt the one block whereas in case of triple DES, it takes 176 milliseconds to encrypt one block. For decryption, DES takes 10ms and triple DES takes almost 3 times than normal DES i.e. 29ms.



| No. of input bits changed in Plain Text | No. of bits changed in Cipher Text | | No.of input bits changed in Plain Text | No.of bits changed in Cipher Text | |
|---|---|---|---|---|---|
| | DES | 3DES | | DES | 3 DES |
| 1 | 31 | 34 | 16 | 29 | 36 |
| 2 | 33 | 31 | 17 | 34 | 33 |
| 3 | 36 | 25 | 18 | 38 | 20 |
| 4 | 27 | 39 | 19 | 34 | 34 |
| 5 | 28 | 36 | 20 | 34 | 23 |
| 6 | 36 | 35 | 21 | 26 | 32 |

**Measurements If Hellman Analysis Is Used**

The preprocessing period of the Hellman strategy takes a more drawn out time due to the various issues referenced above. It takes the aggregate time of N2/3 for the preprocessing phase. Its time complexity is O(logN+t). According to the hellman analysis, the time according to today is around 250 to 300 days.

**Measurements If Distinguished Points Analysis Is Used**

The preprocessing period of the Hellman technique takes a more extended time due to the various issues referenced above. It takes the aggregate time of N2/3 for the preprocessing phase.

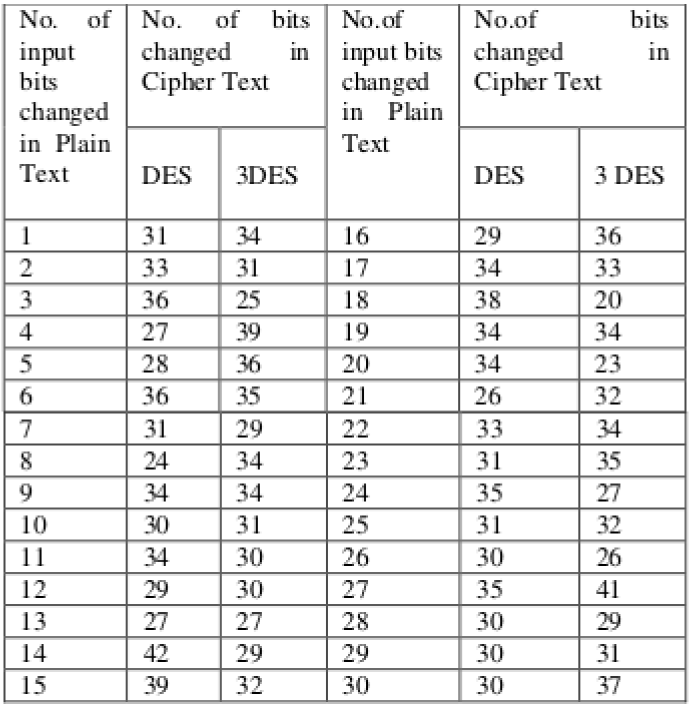
Its time complexity is O(logN+t). According to the hellman analysis, the time according to today is around 100 to 150 days.

### Measurements If Perfect Rainbow Tables Are Used

The preprocessing stage a generally takes less time as a result of absence of consolidations and loops. But some time need to be given on keeping a mind over these merges. Yet it is much lesser than previous ones. The add up to time devoured in the pre-computation stage is around 30 days for a win probability of .72.

### 2. Avalanche effect:

avalanche effect is the change in the no. of bits in output is we made 1 bit change in the input. In DES, for the consecutive change in bits from 1 to 6 it shows the change of 31,33,36,27,28,36 bits respectively and so on for upcoming bits. Whereas in case of triple DES, it shows the change of 34,31,25,39,36,35 bits respectively and so on for upcoming bits. However, average change is considered to be of 30 bits in DES and 32 bits in triple DES. This proves that triple DES better in case of security.



| No. of input bits changed in Plain Text | No. of bits changed in Cipher Text | | No.of input bits changed in Plain Text | No.of bits changed in Cipher Text | |
|---|---|---|---|---|---|
| | DES | 3DES | | DES | 3 DES |
| 1 | 31 | 34 | 16 | 29 | 36 |
| 2 | 33 | 31 | 17 | 34 | 33 |
| 3 | 36 | 25 | 18 | 38 | 20 |
| 4 | 27 | 39 | 19 | 34 | 34 |
| 5 | 28 | 36 | 20 | 34 | 23 |
| 6 | 36 | 35 | 21 | 26 | 32 |
| 7 | 31 | 29 | 22 | 33 | 34 |
| 8 | 24 | 34 | 23 | 31 | 35 |
| 9 | 34 | 34 | 24 | 35 | 27 |
| 10 | 30 | 31 | 25 | 31 | 32 |
| 11 | 34 | 30 | 26 | 30 | 26 |
| 12 | 29 | 30 | 27 | 35 | 41 |
| 13 | 27 | 27 | 28 | 30 | 29 |
| 14 | 42 | 29 | 29 | 30 | 31 |
| 15 | 39 | 32 | 30 | 30 | 37 |

### 3. Money:

#### Measurements If Hellman Analysis Is Used

In Hellman Analysis the issue of unions and circles prevails. So, we have to create countless of this. the measure of memory required expands immensely. The cost for the arrangement as referenced in the first arrangement was about $3.5M. But with time the expense of different equipment segments has diminished and the native actualized solution will cost about $12000.

#### Measurements If Distinguished Points Analysis Is Used

In Distinguished Point the issue of unions and circles still wins however to a lower extent. We need to make a comparitively more modest number of tables. The cost for the solution as referenced in the original arrangement was about $100k. But with time the expense of various equipment parts has diminished and the local executed solution will cost about $9000.

#### Measurements If Perfect Rainbow Tables Are Used

Perfect Tables handle the issue of merges, loops and false alarms. This helps in reducing the measure of memory required by a vast amount. Moreover less work should be done even during the time spent making the table.

### 4. Success Probability:

#### Measurements If Hellman Analysis Is Used

The probability of progress as proposed in the first paper is (m*t)/N. [m-the quantity of begin points,t-the length of chain, N-The aggregate hunt space]. But merges and circles diminish the achievement probability to some level. If we take adequately large m and t then a win probability of a range somewhere in the range of .63 and .75 can be accomplished.

#### Measurements If Distinguished Points Analysis Is Used

The accomplishment in the scope of .7 to .85 can be accomplished.

#### Measurements If Perfect Rainbow Tables Are Used

The accomplishment in the scope of .1 to .999 can be accomplished.

# Chapter 6

# Conclusions

## 6.1 Conclusions

Hence far in our task we have done and secured the examination part. We read and gathered all the expected data to play out this. In addition, we have developed a powerful structure which will push us to pro-actively manage the different issues. Having built up a decent work process we can play out the task gave we are given the vital assets.

## 6.2 Future extension

Being a summed up assault, it has just been built up that we can play out the Time Memory Tradeoff assault on a large portion of the calculations present out there. Hence, this methodology can be effectively connected to any issue with a characterized pursuit space.

The results of the project shows that breaking triple DES having a key of 112 bits is indeed a possible task but at some point it can take much time due to which it, sometimes, becomes pointless. Therefore, in order to break triple DES properly we need to apply probabilistic or guessing approach like van oorschot and wiener approach with some real time modifications as this approach alone is clearly not sufficient.

# REFERENCES

1.https://en.wikipedia.org/wiki/Data_Encryption_Standard

2. https://en.wikipedia.org/wiki/Time/Time_memory_tradeoff

3. https://ee.stanford.edu/~hellman/publications/36.pdf "A cryptanalytic Time-Memory Trade-Off" by Martin E. Hellman

4. https://perso.uclouvain.be/fstandae/PUBLIS/2.pdf "A Time-Memory Tradeoff using Distin. guished Points"

5.https://eprint.iacr.org/2008/054.pdf "Variants of the Distinguished Point Method for Cryptanlytic Time Memory Trade-Off"

6. https://perso.uclouvain.be/fstandae/PUBLIS/74.pdf "Time Memory Trade-offs"

7. https://crypto.junod.info/jacm08.pdf "Characterization and Improvement of Time Memory Trade-Off Based on Perfect Tables"

8. https://lasec.epfl.ch/pub/lasec/doc/Oech03.pdf "Making a faster Cryptanalytic Time Memory Trade-Off"

9. Performance Analysis of DES and Triple DES Dr. O. Srinivasa Rao Associate Professor of CSE,Department of Computer Science and Engineering, University College of Engineering (Autonomous) Kakinada, Andhra Pradesh

10. https://en.wikipedia.org/wiki/Triple_Data_Encryption_Standard