NITTE EDUCATION TRUST

N.M.A.M. INSTITUTE OF TECHNOLOGY
(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)
Nitte – 574 110, Karnataka, India
(ISO 9001:2015 Certified), Accredited with 'A' Grade by NAAC
☎: 08258 - 281039 – 281263, Fax: 08258 – 281265

Department of Computer Science and Engineering

B.E. CSE Program Accredited by NBA, New Delhi from 1-7-2018 to 30-6-2021

Report on Mini Project

Extended Triple-DES RC4 Cryptography

Course Code: 16CSE54

Course Name: Cryptography and Network Security

Submitted To,
Dr. Radhakrishna Dodmane

Submitted By:

P Pawan Bhandarkar	4NM16CS090
Prajwal S P	4NM16CS099
Prem Stevenson Pereira	4NM16CS108
Rahul D Shetty	4NM16CS111
Sushanth Kille	4NM16CS158

Date of submission:

Signature of Course Instructor

NITTE EDUCATION TRUST | N.M.A.M. INSTITUTE OF TECHNOLOGY
(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)
Nitte – 574 110, Karnataka, India
(ISO 9001:2015 Certified), Accredited with 'A' Grade by NAAC
☎: 08258 - 281039 – 281263, Fax: 08258 – 281265

Department of Computer Science and Engineering

B.E. CSE Program Accredited by NBA, New Delhi from 1-7-2018 to 30-6-2021

Department of Computer Science and Engineering

CERTIFICATE

Certified that the Mini Project work entitled **Extended Triple-DES RC4 Cryptography** carried out by **P Pawan Bhandarkar (4NM16CS090), Prajwal SP (4NM16CS099), Prem Stevenson Pereira (4NM16CS108), Rahul D Shetty (4NM16CS111)** and **Sushanth Kille (4NM16CS158**), bona fide students of **NMAM Institute of Technology**, **Nitte** has been carried out satisfactorily. The Mini Project report for the Subject **Cryptography and Network Security**, has been prepared as per the prescribed format.

Name & Signature of Guide	**Name & Signature of HOD**
Dr Radhakrishna Dodmane	Dr. K R Udaya Kumar Reddy
Associate Professor	Head of the Department
Department of CSE	Department of CSE
NMAMIT, NITTE	NMAMIT, NITTE

ACKNOWLEDGEMENT

We believe that our project will be complete only after we thank the people who have contributed to make this project successful.

First and foremost, our sincere thanks to our beloved principal, **Dr. Niranjan N. Chiplunkar** for giving us an opportunity to carry out our project work at our college and providing us with all the needed facilities.

We express our deep sense of gratitude and indebtedness to our guide **Dr. Radhakrishna Dodmane C V**, Assistant Professor Gd III, Department of Computer Science and Engineering, for his inspiring guidance, constant encouragement, support and suggestions for improvement during the course of our project.

We sincerely thank **Dr. K.R. Udaya Kumar Reddy**, Head of Department of Computer Science and Engineering, Nitte Mahalinga Adyantaya Memorial Institute of Technology, Nitte.

We also thank all those who have supported us throughout the entire duration of our project.

Finally, we thank the staff members of the Department of Computer Science and Engineering and all our friends for their honest opinions and suggestions throughout the course of our project.

P Pawan Bhandarkar	4NM16CS090
Prajwal S P	4NM16CS099
Prem Stevenson Pereira	4NM16CS108
Rahul D Shetty	4NM16CS111
Sushanth Kille	4NM16CS158

ABSTRACT

In this paper, we discuss the drawbacks of once state-of-the-art algorithms, namely the Triple DES algorithm and the RC4 Stream cipher. Having understood the limitations of these encryption mechanisms, we propose an alternative algorithm that aims to leverage and combine the best features of both these algorithms and provide much better security than either of them. The Extended Triple DES Encryption algorithm proposed in this paper, aims to combine the strength of the DES Ecosystem and the pseudo-randomness of the RC4 cipher. The resulting algorithm is one which has a key-space of 256 bytes. We also discuss about how our algorithm differs from the existing ones using frequency histograms and other mathematical techniques. It is worth mentioning that not only does our algorithm keep the individual advantages of the component algorithms but also improves upon them with added advantages of its own.

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

Communication is the act of sending a message or signal from one point to another. The art of communication is one that has evolved with mankind over the course of millennia. Cryptography, in the purest sense of the word, means to “change a message in a way that makes it safe from adversaries while keeping it understandable by the intended recipient”. The origins of cryptography are rooted deep in the history of the world, dating as far back as 1900 BCE, when the Egyptians used hieroglyphics to hide their information. As technology advanced over the years, so did the means and techniques by which one could perform encryption and consequently, so did the degree to which security could be provided to the data.

The advent of computers resulted in an exponential increase in the level of complexity and speed at which we can now protect the information being sent from attackers. In the recent years, Encryption algorithms such as the Advanced Encryption Standard, Secure Hashing Algorithms, Triple DES etc. have all become synonymous with the word “cryptography”. These current State of the Art Encryption algorithms provide security in ways that it near-impossible for unauthorized intermediaries. However, with hardware improvements such as in the case of GPU’s has made it increasingly likely that sometime in the future, these algorithms could potentially be broken. Because of the potential threat that this problem poses, the community is always exploring newer alternatives that could improve the security of user content across the internet in the long run.

It is for this very reason that we decided to explore the level of security that we could achieve by combining the crypto logical prowess of two well established algorithms - The Triple Data Encryption Standard and The RC4 Stream Cipher. The reason we chose these two algorithms is because the modularity of the DES algorithm allows us to swap out the default key generation algorithm for the RC4 algorithm. We hypothesize that the pseudo random nature of RC4 would help improve the security over the predictable 48-bit keys that would otherwise be generated by the sequence of shifts and compressions in the case of Vanilla DES.

Chapter 2

IMPLEMENTATION

The implementation of our algorithm has been divided into 4 main sections and each section comprises of relevant algorithms and diagrams explaining the different aspects of the code:

1) Data Encryption Standard
2) Triple DES
3) RC4 Stream Cipher
4) Extended Triple DES

2.1 Data Encryption Standard

The DES algorithm is a symmetric-key 64-bit block cipher that was once considered the standard for security across the internet until it was regarded as being officially broken in 1993 when an inexpensive and fast piece of hardware was proposed that was able to exhaustively search the DES key space.

The DES algorithm works by the integration of *four* main components:

1) An initial permutation
2) 16 identical round functions
3) A Key generation Algorithm
4) A Final Permutation

The **Initial and final permutations** are effectively inverses of one another and are meant to shuffle the bits around before and after the 16 rounds in order to increase security and help further reduce statistical characteristics of the cipher text.

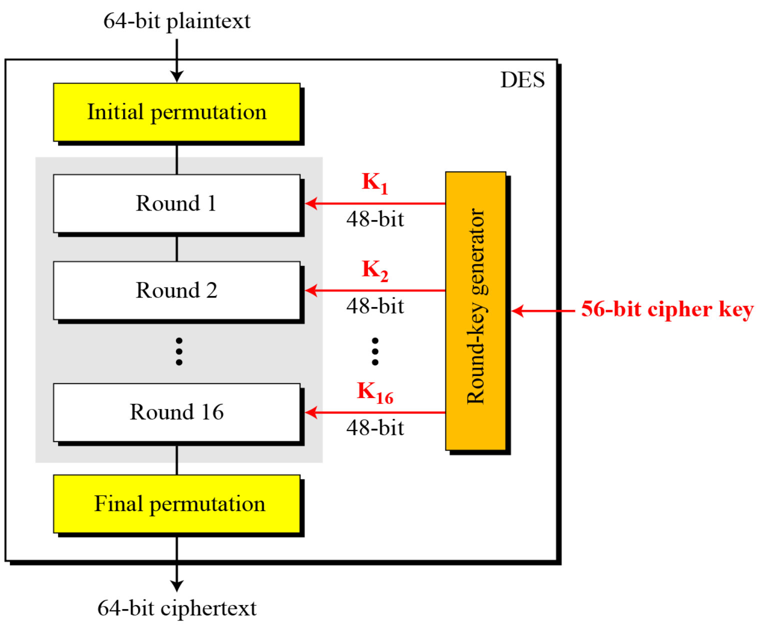


Figure 1: DES

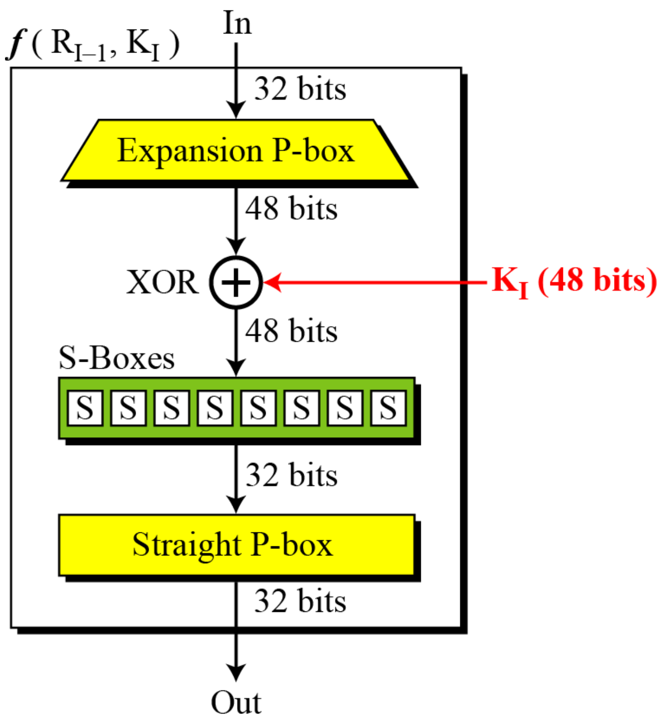


Figure 2: Round Function

The round function is at the heart of DES. It takes the rightmost 32 bits of the input and transforms it using a 48-bit key to produce a 32-bit output by applying a function to it. The function ***f*** is implemented as shown on the left. The output bits become the leftmost 32 bits of the block that is fed as input to the next round.

It comprises of:

a) *Expansion P-Box* that expands the 32 bit input into 48 bits.
b) 8 *S-boxes* that each transform 6 bits of the input into a 4-bit output.
c) A *Straight P-Box* that permutes the 32 bits for added security

The Key Generation Algorithm for Vanilla DES is implemented as shown in the diagram. It takes as input, a 64-bit key and uses it to generate 16 different round keys in order, which are then fed to the corresponding functions of the DES rounds. The 64-bit key first get converted into 56 by dropping the parity bits. It is then divided into left and right halves that are both shifted left by either one or two bits depending on the round. These shifted bits are combined in the compression P-Box whose output is sent as a round key. The rounds [*1, 2, 9, 16]* correspond to 1 bit shifts and the rest correspond to 2 bit shifts.

This diagram clearly highlights the potential weak point in the DES algorithm. By knowing the Initial key, it is possible to predict all the subsequent keys of the DES algorithm since the Compression P-Box is known.

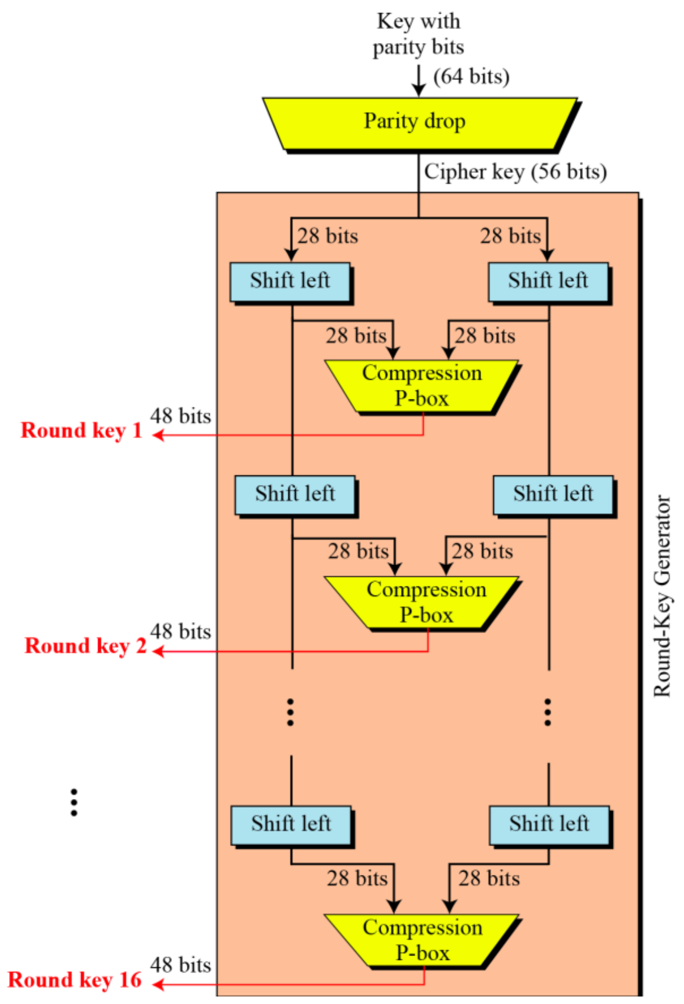


Figure 3: DES Key Generation

2.2 Triple DES

One of the main drawbacks of the DES algorithm is the fact that it has a small key space of only 56 bits. This is what caused it to be broken by exhaustive search as early as in 1993. To overcome this, Triple DES was proposed.

The main idea of Triple DES, is to improve security by making use of DES three times, each time using a different key or using two keys as follows:

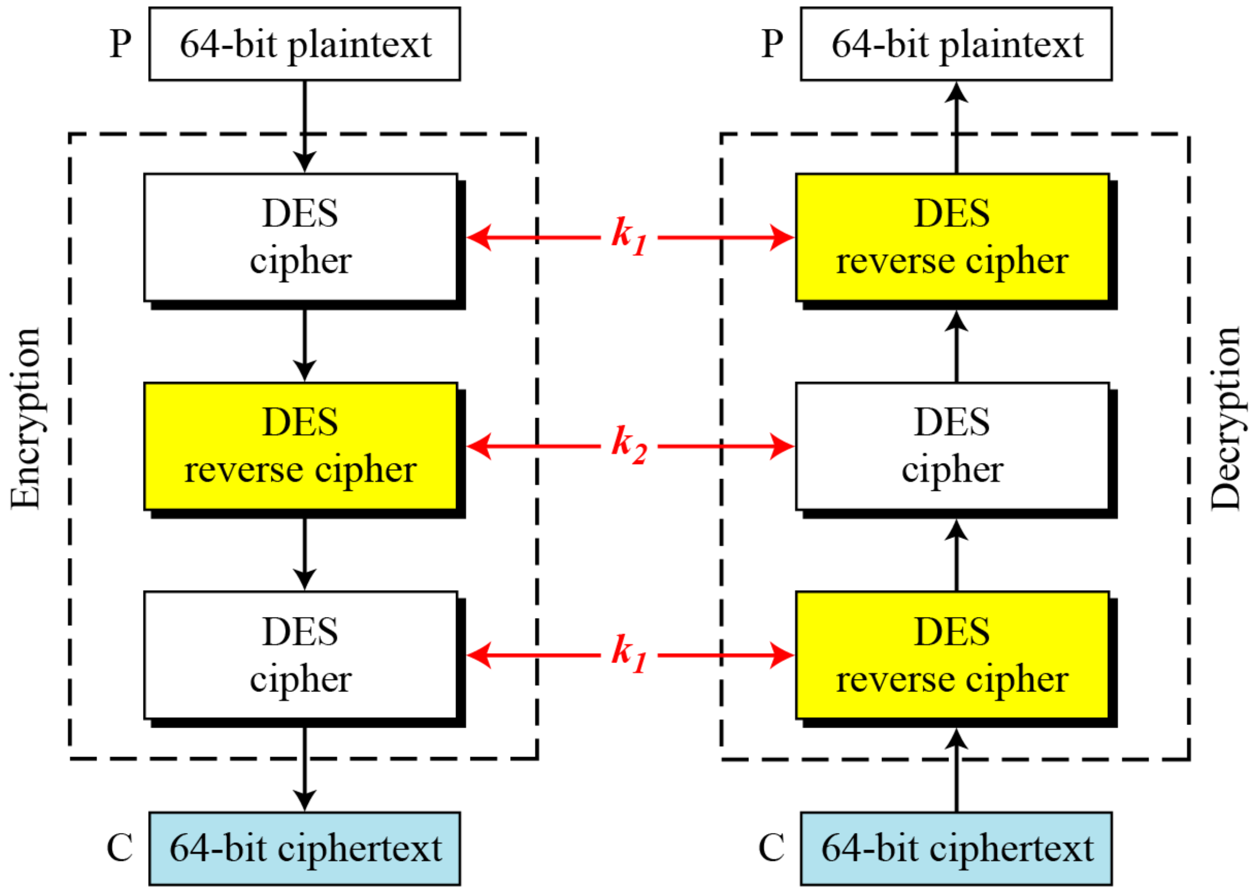


Figure 4: Triple DES

Although using three different keys should mean a key space of 156 bits, due to meet in the middle attacks, the effective security provided is reduced to only 112 bits. The second keying option means a key space of 112 bits but due to its susceptibility to certain chosen-plaintext and known-plaintext attacks, it is deemed to have only 80 bits of security. As of 2019 however, it is possible for reasonably cheap hardware to be fast enough to exhaustively search the TDES key space and it has hence been declared as broken.

2.3 RC4 Stream Cipher

RC4 (Rivest Cipher 4) is a stream cipher technique developed by Ronald Rivest of RSA. The symmetric key algorithm is used identically for encryption and decryption such that the data stream is simply XORed with the generated key sequence. This cryptography scheme is used in SSL/TLS and also in Wi-Fi security systems.

The key used is variable size ranging from 1 to 256 bytes long and the algorithm strength is usually based on strong permutation techniques used. It is known to have good resistance against cryptanalytic attacks.

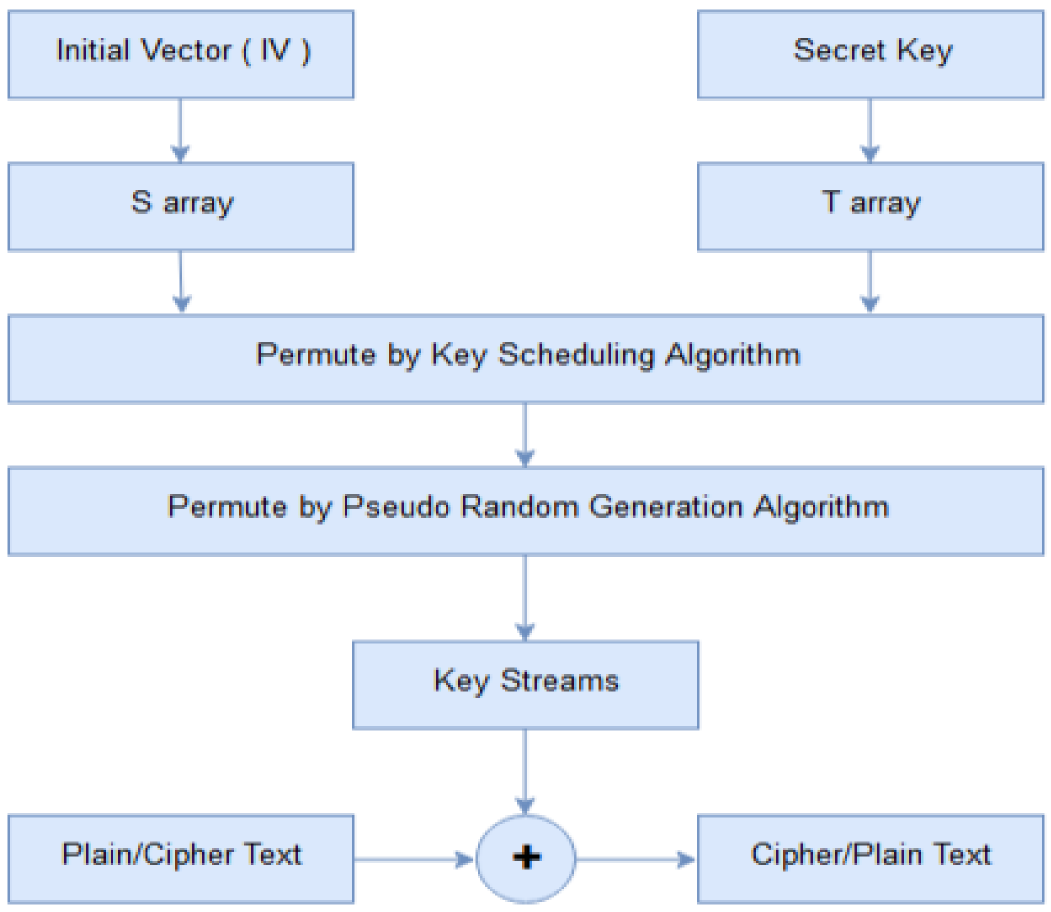


Figure 5: RC4 Cryptography

ALGORITHM: RC4

1. A 256-byte state vector is used to hold the permutation of all-8 bit numbers. The vector is initialized to a permutation based on the key which can be 1 to 255 bytes wide.
2. To calculate initial permutation, we use a temporary vector ‘T’ of 256-bytes wide that contains a repeated sequence of keys.
3. The key and the temporary vector ‘T’ are only used for initial phase and encryption/ decryption phases only make use of another vector “S”.
4. We XOR the plain text with values in “S” to generate the required Cipher text.
5. Decryption process follows the same.

2.4 Extended Triple DES-RC4 Cryptography

Now that we have established the building blocks of our algorithm, we now get into the meat of our project. From the description of how the Key Generation of the DES algorithm worked, it was clear that there was a major flaw in it’s working. If the Initial 64-bit key being given as input to the Key Generation Algorithm were to be found out, then all subsequent keys could also be broken down sequentially. It is also possible that if even one of the round keys was discovered, then the other round keys could also be found out quite easily.

We also realized the strength of the RC4 Algorithm in that it can hold its own against cryptanalytic attacks and even if the initial key is known, the round keys fed into the functions would still be pseudorandom in nature

ALGORITHM: Extended Triple DES-RC4 Cryptography

This algorithm provides a structure similar to that of Triple DES but a major difference is the keys generated for each of the DES rounds are provided by a modified RC4 algorithm. RC4 provides a pseudo random key generation scheme and using the previous or initial key as a seed we generate the Keys.

1. Use RC4 pseudo random generation to generate 16 set of sub keys from an initial key K1 and for generation K1j’th key of the 16 rounds use the K1(j-1)’th key as input for the RC4 function.
2. XOR the 16 keys with an initial vector key set (IV) and produce a new 16 keys of 48-bits which are the input for each of the rounds of DES.

3. Pass in the intermediate cipher text through Decryption and Encryption rounds by applying the same procedures as in Rule 1 and 2 and produce the final Cipher Text.
4. Decryption process follows the same but providing the keys in reverse order for the different DES rounds.

The below figure represents the entire structure of Extended Triple DES.

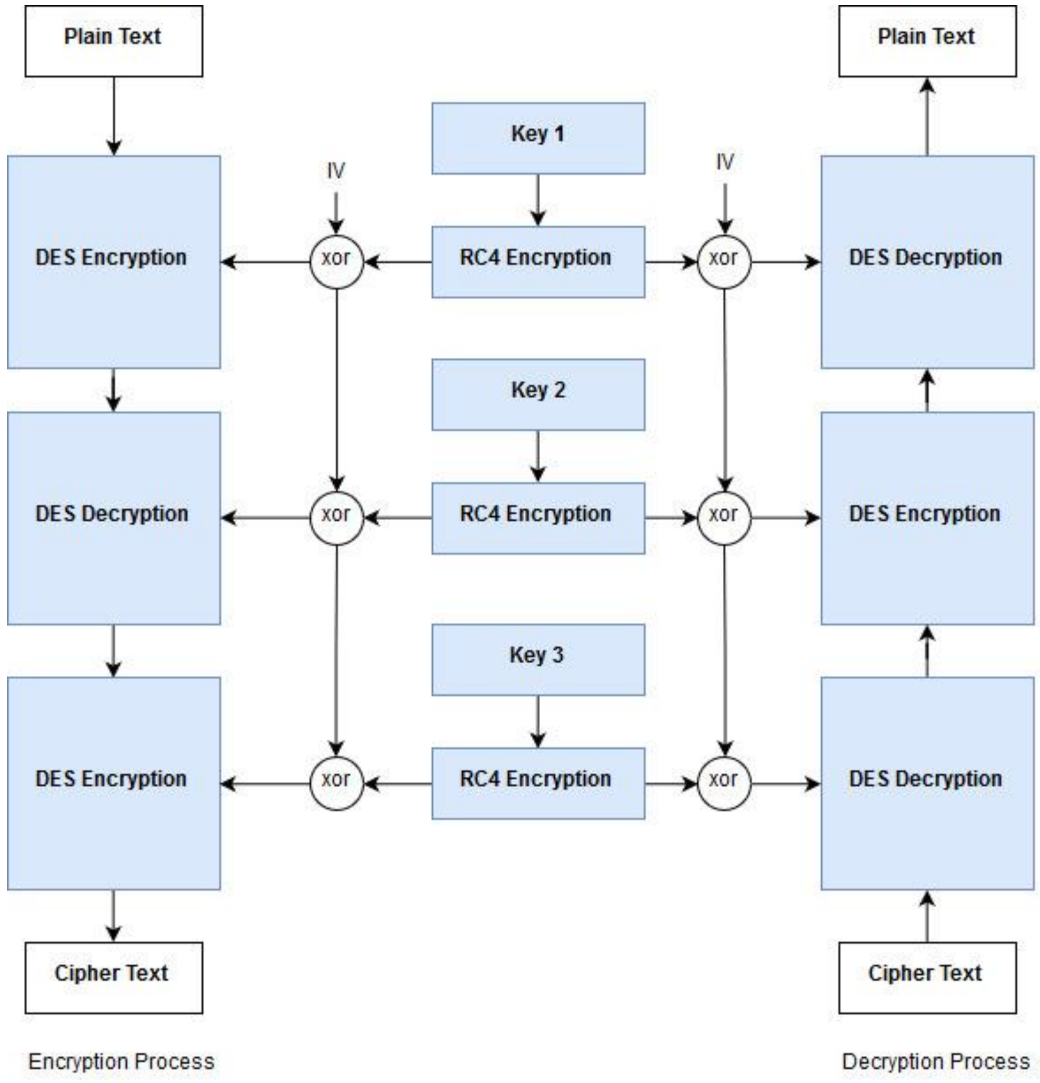


Figure 6: Extended Triple DES-RC4 Cryptography

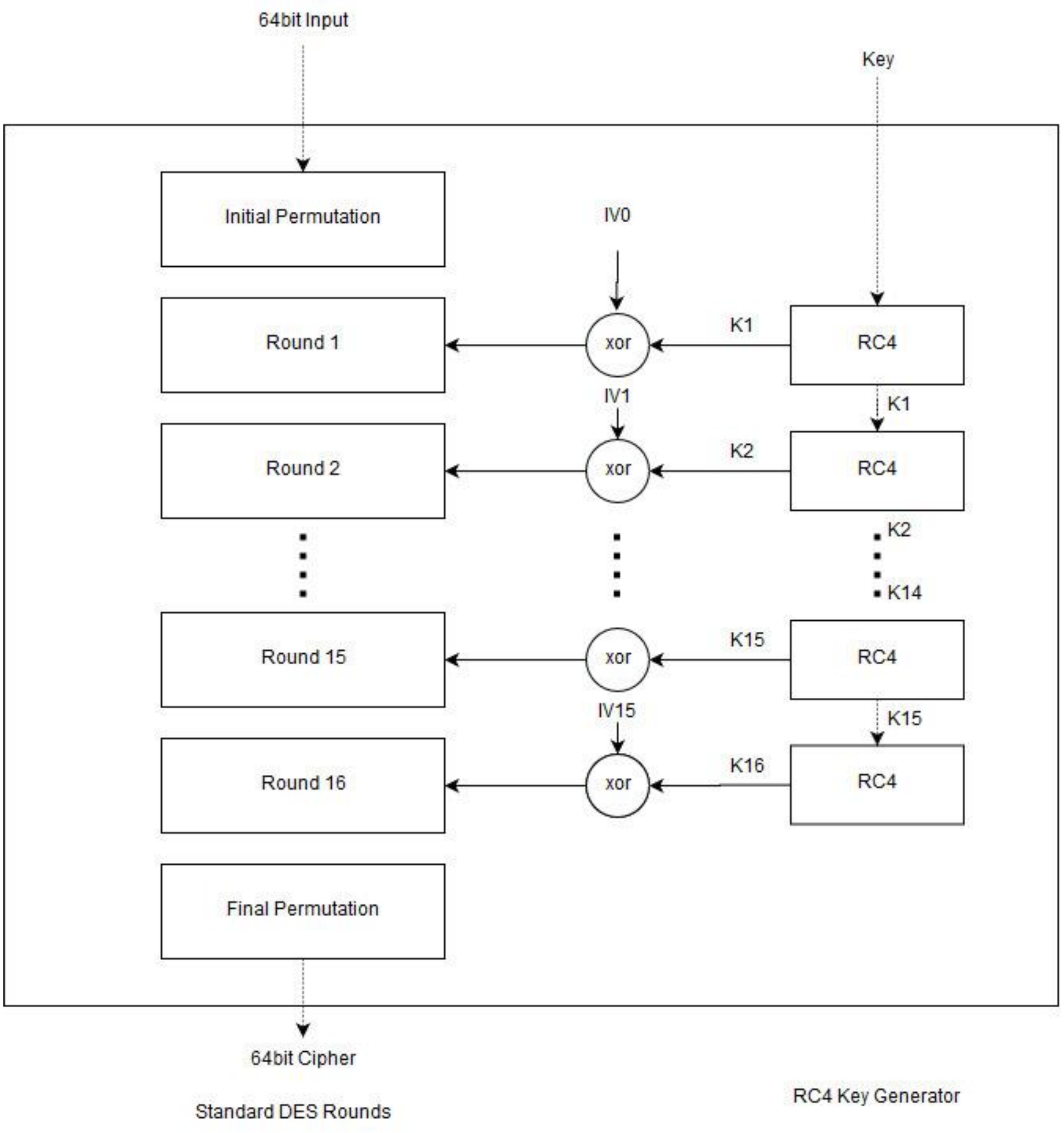


Figure 7: Key inputs for each DES round

The above figure shows how each rounds of DES obtains key from the RC4 key generation function. By sharing the three keys Key1, Key2 and Key3 between the sender and the receiver, the same pseudorandom sequence of round keys can be generated and can hence be used for encryption and decryption respectively. This provides security in a way that ensures that knowledge of any of the intermediate round keys cannot be used to obtain the original key.

2.5 Program

For the purpose of demonstration, we implemented the entire code using Python. Python provides a rich set of functions along with its flexible programming paradigm for faster and efficient coding. Due to these features Python was selected for the demonstration of Extended Triple DES-RC4 Cryptography Algorithm.

Modules for each are implemented by making each of the cryptography technique into an independent class. DES, RC4 and TDES are the 3 main sub-class that were created which represents the appropriate technique. Then a modification was done for DES and TDES classes to incorporate the key generation using RC4 and for propagating vector keys between the different DES process.

```
from DES import *
from utility import *
from TDES import *
from RC4 import *

class XTDES:

    @staticmethod
    def apply(text,keys,mode=ENCRYPT):
        rc4 = RC4()
        tdes = TDES()
        keys = [ rc4.genKey(x,8) for x in keys ]
        res=""
        if mode==ENCRYPT:
            res=tdes.encrypt(text,keys)
        else:
            res=tdes.decrypt(text,keys)
        return res

    @staticmethod
    def encrypt(text,keys):
        return XTDES.apply(text,keys,ENCRYPT)

    @staticmethod
    def decrypt(text,keys):
        return XTDES.apply(text,keys,DECRYPT)
```

The above python class is used to create Extended Triple DES-RC4 objects using which we can use to encrypt/decrypt messages. The function *encrypt* () and *decrypt* () takes in 2 parameters. One of them is a text and other one is a list of 3 keys. The text parameter for encrypt is the plain text and on applying that function we obtain the ciphered message in text form. The text parameter for decrypt is the reverse, it takes in a cipher text and decrypts it to produce the original message by using the given keys. Padding is done while converting for proper output.

Chapter 3

OBSERVATION

3.1 Comparison with Triple DES

Similarities
Brute Forcing 2^{150} combinations of keys could produce a solution, but this is currently infeasible with the existing technology.
Uses three rounds of Encryption/Decryption/Encryption DES process for encryption and reverse of each for decryption.
A Secured channel is needed to share all the 3 keys.
Easier to implement in hardware as both uses straight forward logical functions.

Triple DES	Extended Triple DES-RC4
Provides no security for the keys.	Keys generated are pseudo random.
Iterative trace back method can be used for obtaining the keys.	Keys obtained are not the final one.
Keys which are given to the DES are independent of each other.	Each key which are input to the DES are dependent on previous keys.

3.2 Security Features

In order to describe about how well the Extended Triple DES-RC4 encrypts when compared to normal Triple DES we use the following test scenarios:

1. Distribution of Frequencies Characters.
2. Variation in Output when keys are changed.
3. Variation in Outputs with same keys.

Distribution of Frequencies of Characters

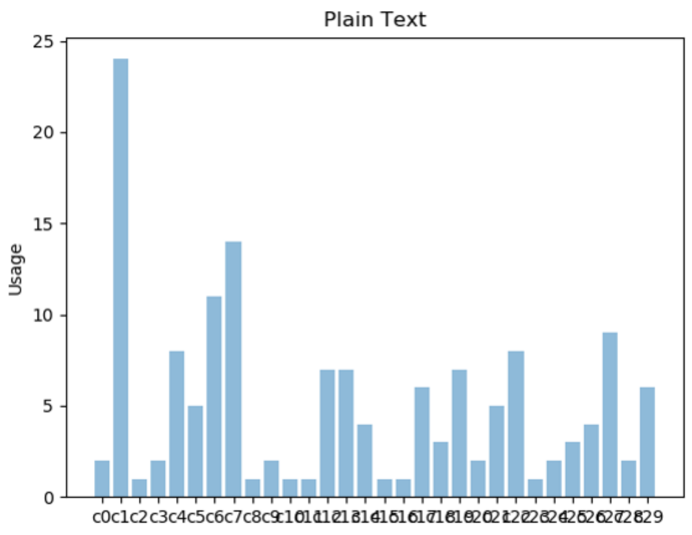


Figure 8: Input Text

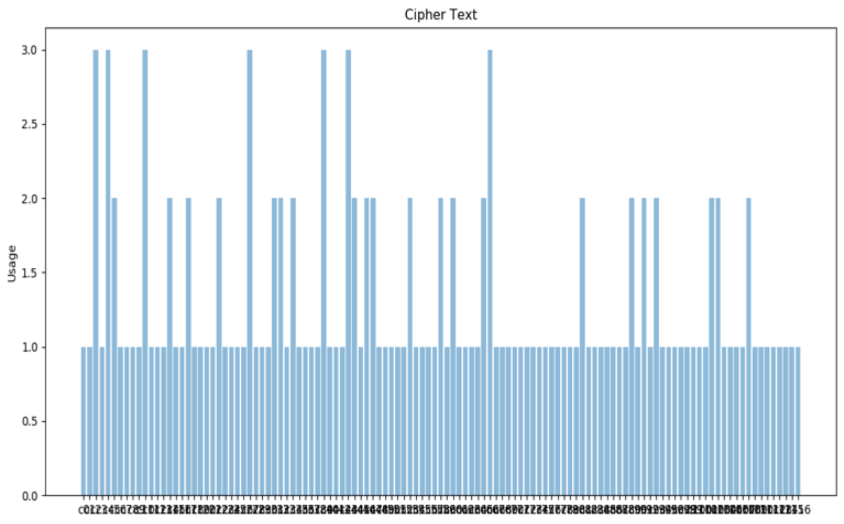


Figure 9: Output Text

The input text graph shows major variation in the frequencies of each character for a sample text while the output text graph represents how different characters are used to map single characters and how there is a uniform and distribution of frequencies of all the characters.

Variation in Output when keys are changed

In the case we use a same text sample “India” and try to vary a key1 by different bit amounts. We can notice that by varying few bit amounts we can produce a large variation in the cipher text bit. This effect of variation is called Avalanche Effect.

Key 1	Cipher Text	Key bit variance	Cipher bit variance
1	Øy€ m5		
2	‡ A%íP	2	19
3	Ñ-ö8f@3ã	1	18
8	µ["Œ+{Ç	3	24
31	)JÛq§ôP	3	26

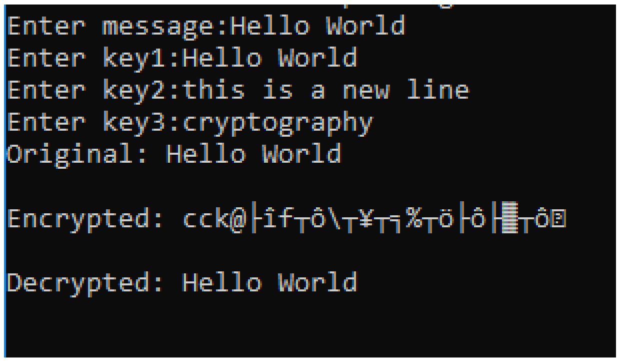
Variation in Output when keys are changed

The following keys were for encrypting a text sample Key1:1 Key2:2 Key3:3. Then we try to change the input key by varying it bit by bit. We can notice how the Cipher Text formed also varies in its output.

input	Cipher	Input bit variance	Cipher bit variance
1	³k!€ Ì÷#R		
2	Ð$pqŸQ¹Ž	2	26
3	MÇëèÉ L	1	19
8	e+àÍ¤ú=ø	3	25
31	¨!ˆ GK • ¢‹	3	29

3.3 Results

The first sample run uses “Hello World” as input text and takes in 3 keys “Hello World”, “this is a new line” and “cryptography”. This produces the following output below:



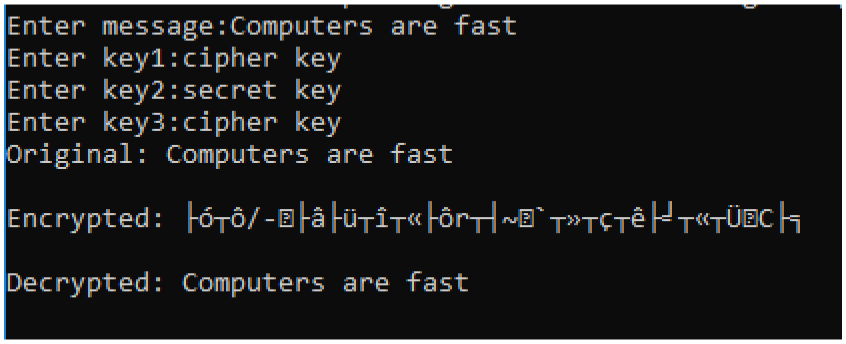
```
Enter message:Hello World
Enter key1:Hello World
Enter key2:this is a new line
Enter key3:cryptography
Original: Hello World

Encrypted: cck@├îf┬ô\┬¥┬╕%┬ö├ô├▓┬ô▯

Decrypted: Hello World
```

Figure 10: Case 1

The second sample run uses “Computers are fast” as input text and takes in 3 keys “cipher key”, “secret key” and “cipher key”. This produces the following output below:



```
Enter message:Computers are fast
Enter key1:cipher key
Enter key2:secret key
Enter key3:cipher key
Original: Computers are fast

Encrypted: ├ó┬ô/-▯├â├ü┬î┬«├ôr┬┤~▯`┬»┬ç┬ê├┘┬«┬Ü▯C├╕

Decrypted: Computers are fast
```

Figure 11: Case 2

Chapter 4

CONCLUSION

Throughout this paper, we explored the various ways in which the algorithms of yesterday fell short. With our learnings and findings, we were able to come up with a new alternative that proved to be superior to the constituent algorithms in every way. Using the RC4 algorithm as the key generator of the Triple DES algorithm, we can effectively increase the key space from a mere 64 bits to (256 * 8) bits, making it that much harder to crack via simple cryptanalysis methods. In addition to this, we also made use of an initial vector that added to the security by making sure that the knowledge of once key would not compromise the secrecy of the others. The RC4 Stream Cipher as a key generator is reminiscent of the working of a one-time pad algorithm, where a pseudorandom keystream is generated and used for encryption.

Having successfully designed and implemented this project, we have a few improvements that we intend on applying in the near future including but not limited to trying out alternate schemes such as an Output feedback mechanism, Cipher block chaining and Electronic code blocks. If time permits, it would be worth considering these options and choosing the one that provides the best results.

Chapter 5

REFERENCES

1. https://www.geeksforgeeks.org/computer-network-rc4-encryption-algorithm/
2. https://en.wikipedia.org/wiki/Triple_DES
3. https://www.quora.com/The-DES-encryption-was-broken-in-1999-Why-and-how-did-it-happen
4. https://pdfs.semanticscholar.org/c72a/2d25070b6271b58c57227f21402ee7d3d1a0.pdf
5. http://www.crypto-it.net/eng/symmetric/rc4.html?tab=0
6. https://en.wikipedia.org/wiki/RC4
7. Diagrams - Behrouz Forouzan Slides on DES and TDES