MINOR-1 PROJECT

END TERM REPORT

For

Hybrid Compression-Encryption Algorithm

Submitted By

Specialization	SAP ID	Name
CCVT	500087456	Mohit Bishesh
CCVT	500086385	Shiv Pratap Singh
CCVT	500087109	Yuvraj Pundir
CCVT	500084917	Anant Garg

Department of Systemics

School Of Computer Science

UNIVERSITY OF PETROLEUM & ENERGY STUDIES,

DEHRADUN- 248007. Uttarakhand

Prof. Sandeep Pratap Singh.
Project Guide

Dr. Neelu J.
Cluster Head

Project Title

Hybrid Compression-Encryption Algorithm

Abstract

Encryption is the method by which information is converted into secret code that hides the information's true meaning. The science of encrypting and decrypting information is called cryptography. In computing, unencrypted data is also known as plaintext which is then converted into encrypted data, called ciphertext, with the help of some algorithm. Advanced Encryption Standard (AES) is one of the most powerful encryption algorithms that are widely used in various researches and implemented in hardware and software. But this algorithm has negative side effects, which can increase file size. We can use data compression as a countermeasure.

Data compression is the process of modifying, encoding or converting the bits structure of data in such a way that it consumes less space on disk. It enables reducing the storage size of one or more data instances or elements. The lossless Huffman compression algorithm can be used to implement compression.

ACKNOWLEDGEMENT

We wish to express our deep gratitude to our mentor Prof. Sandeep Pratap Singh, for all advice, encouragement, and constant support he has given us throughout our project work. This work would not have been possible without his support and valuable suggestions.

We sincerely thanks to our cluster head, Prof. (Dr.) Neelu J. Ahuja, for her great support in doing our project at SoCS.

We are also grateful to Dr. Ravi S. Iyer Dean, SoCS, UPES for giving us the necessary facilities to carry out our project work successfully.

We would like to thank all faculties from UPES for their help and constructive criticism during our project work. Finally, we have no words to express our sincere gratitude to our parents who have shown us this world and for every support they have given us.

Mohit Bishesh	500087456
Yuvraj Pundir	500087109
Shiv Pratap Singh	500086385
Anant Garg	500084917

TABLE OF CONTENTS

Sr. No.	Contents	Page No.
1.	INTRODUCTION	06
2.	LITERATURE REVIEW	07
3.	PROBLEM STATEMENT	07
4.	OBJECTIVE	07
5.	DESIGN METHODOLOGY	08
6.	PERT chart	08
7.	IMPLEMENTATION	09-14
8.	RESULT	15
9.	CONCLUSION AND FUTURE SCOPE	16
10.	REFERENCES	16
11.	GitHub Link	17

LIST OF FIGURES

Sr. No.	Figures	Page No.
1.	Methodology	08
2.	PERT chart	08
3.	Implementation	
	3.1. Objective 1	09-10
	3.2 Objective 2	11-12
	3.3 Objective 3 and 4	13-14
4.	Results	15

LIST OF TABLES

Sr. No.	Table	Page No.
1.	Comparison Table	15

1. Introduction

With the advancement of time, there is rapid innovation and development in the field of technology and tech industries. Many new Software and applications have been developed especially when we talk about social media or other software connecting people.

So here comes cryptography which plays a vital role in securing the data and maintaining the privacy of the users. Technically, cryptography refers to the encryption technique which helps us to convert the plaintext to cyphertext with the help of appropriate algorithms. This technique can be used to apply to almost all kinds of soft or digital files available including PDF, word, images, etc.

Advanced encryption standard (AES), 3DES, and Data encryption standard (DES), are the common but very famous symmetric- cryptographic algorithms being used.
Whereas the advanced encryption standard is one of the best cryptographic algorithms and hence it is being used as a security standard for the National institute of standards and technology(NIST).

But it must be noted that some of the researchers claim that AES has a slight side effect on the file size as it increases the size of the file after encryption. This side effect can be negated with the help of compression algorithms. There are two types of compression techniques- Lossless compression and Lossy compression. Lossless compression is a technique in which there is no data loss and the data of the content of the file is completely reversed during the decryption process.

Whereas lossy compression is a compression technique in which some non-critical parts of the data are removed and there is a certain loss in the data. For better accuracy, especially when we have critical data, lossless compression is always the first choice.

So finally, the integration of a cryptographic algorithm i.e. AES (used here), and a compression technique (like Huffman Encoding) is a better idea that not only reduces the file size but also increases the security and is hence helpful in maintaining the security and privacy of the users in case of connecting platforms or if we are sharing our file contents.

2. Literature Review

In the research done by author [1] it was proven that after AES encryption, the size of the file was increased by 25% but after applying Huffman encoding the encrypted file code decreased by 30%. Also, a considerable amount of change in entropy and Avalanche effect is observed after applying Huffman encoding.

In the research done by author [2] it was shown that AES has higher space complexity as compared to DES, while DES has higher time complexity as compared to AES.

3. Problem Statement

Advanced Encryption Standard is a widely used and recognized encryption algorithm. Although the algorithm provides strong and reliable encryption, it does so on the size of making the file size slightly larger. Therefore, there is a need to negate this drawback. Huffman encoding can be coupled with AES to produce compressed lossless encrypted files without compromising security.

4. Objectives

- To implement AES and DES to generate ciphertext from plaintext.
- To compare file size of ciphertexts generated by AES and DES respectively.
- To compress plain text using Huffman Encoding and then perform encryption.
- To compare the size of compressed ciphertext with normal ciphertext.

5. Methodology

- Input plaintext will be taken.
- Generating cipher text of input string using AES or DES algorithm.
- File size of generated ciphertext will be noted.
- Compression of input plaintext using Huffman encoding.
- Encryption of compressed input plaintext.
- Calculation of compression factor for each ciphertext generated with respect to the encryption technique used.

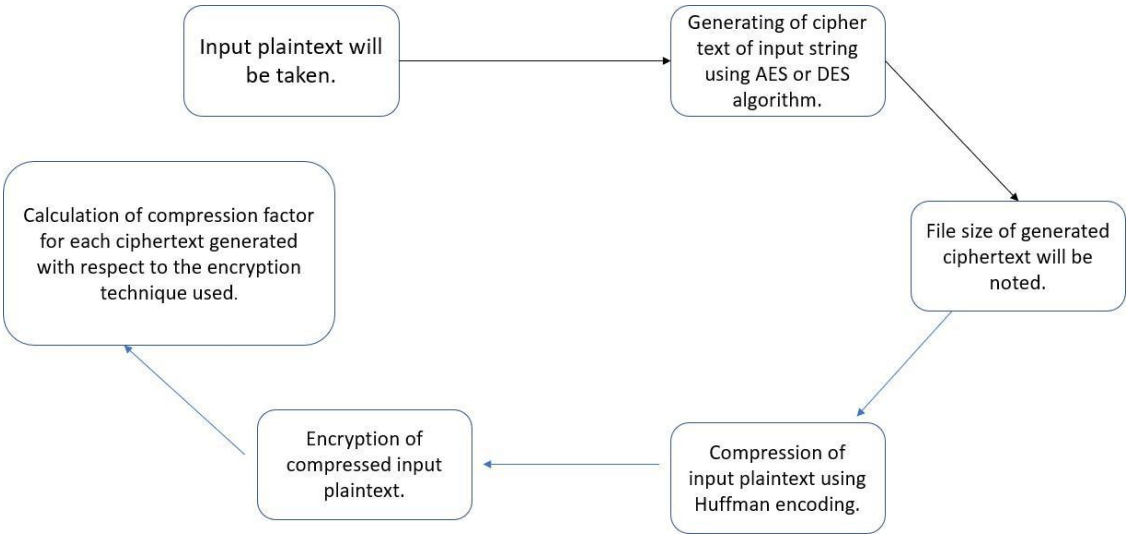

Figure 1: Methodology

6. PERT Chart

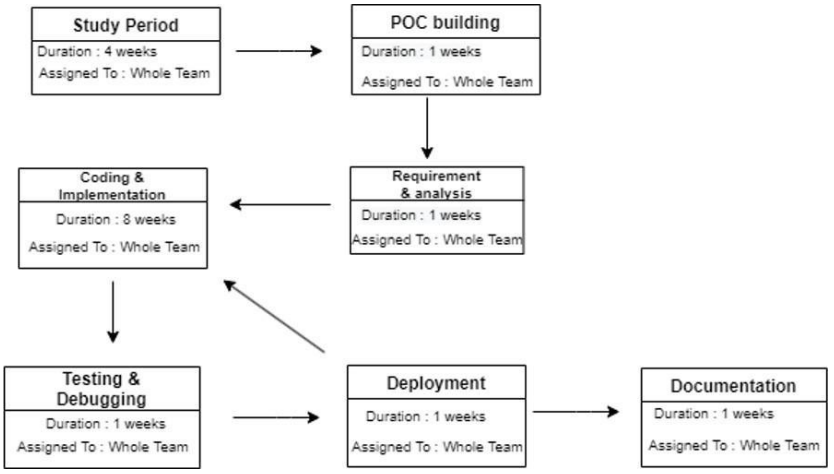

Figure 2: PERT chart.

7. Implementation

Algorithms

- AES (Advanced encryption standard).
- Huffman encoding.
- DES (Data encryption standard).

Data Structures

- Priority queue (for building Huffman tree).
- Minimum heap (to implement functionality of priority queue).
- Array.
- Linked list.

Objective 1: To implement AES and DES to generate ciphertext from plaintext.

1) Input plain text for DES

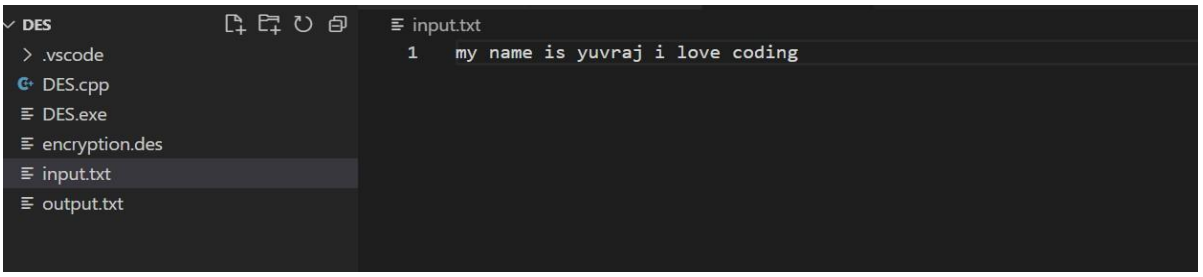

Figure 3

2) Input plain text for AES

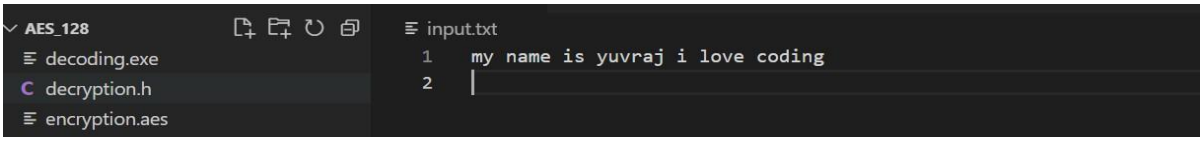

Figure 4

3) Execution of AES

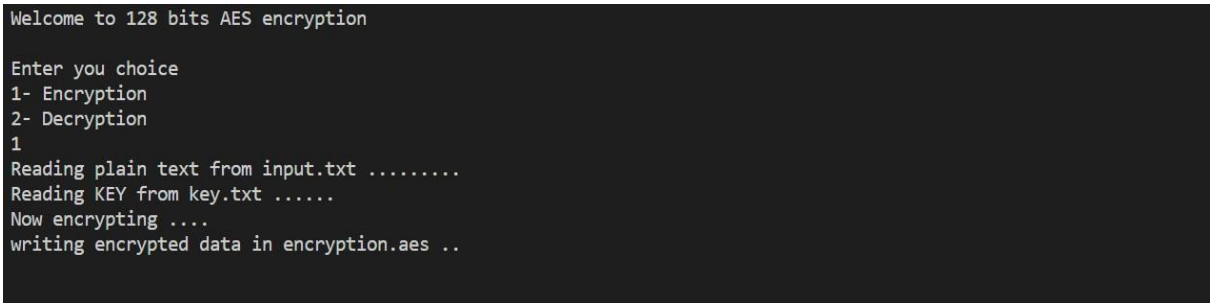

Figure 5

4) Execution of DES

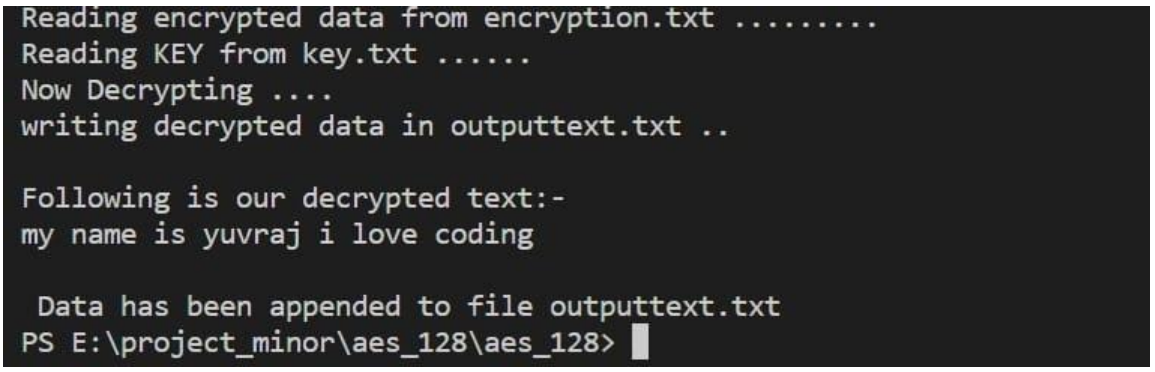

Figure 6

5) Ciphertext generated by DES.

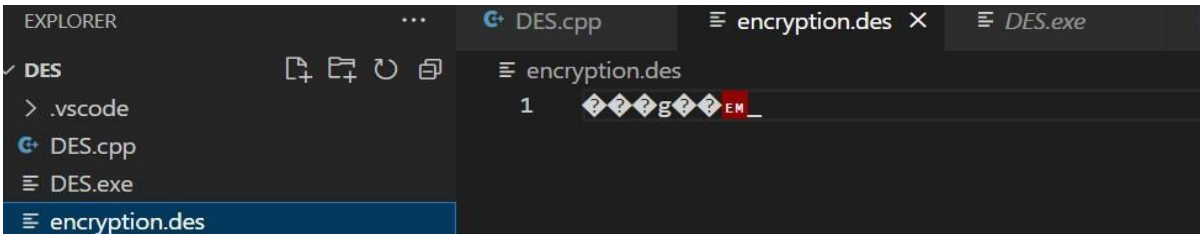

Figure 7

6) Ciphertext generated by DES.

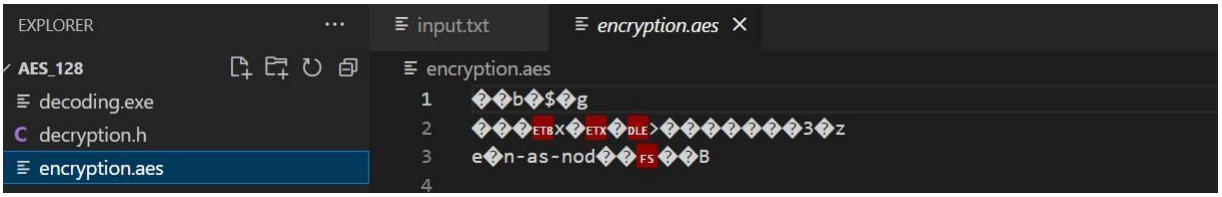

Figure 8

Objective 2: To compare file size of ciphertexts generated by AES and DES respectively.

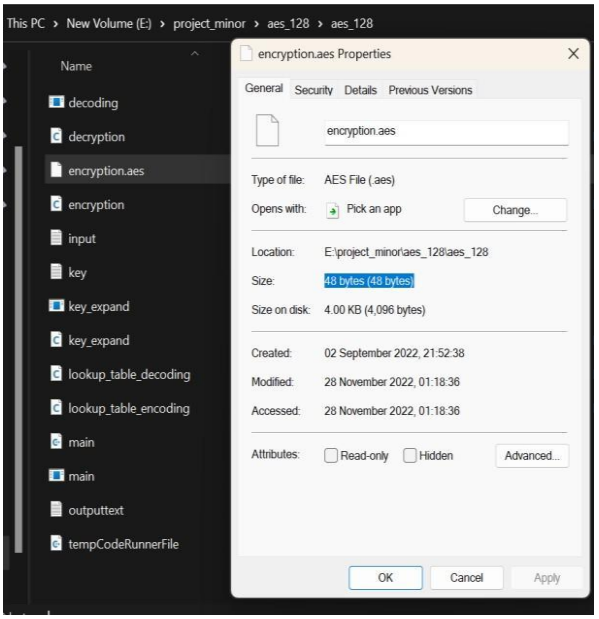

Figure 9 : AES ciphertext size

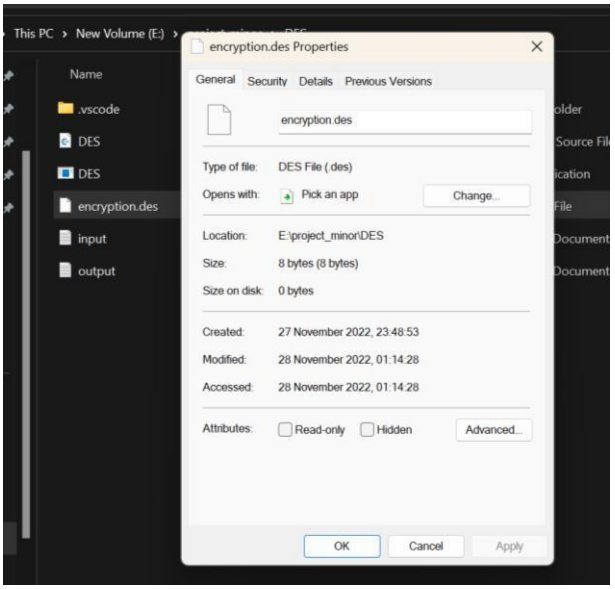

Figure 10 : DES ciphertext size

Objective 3 & 4: To compress plain text using Huffman Encoding and then perform encryption.

1) Input text for encoding and encryption.

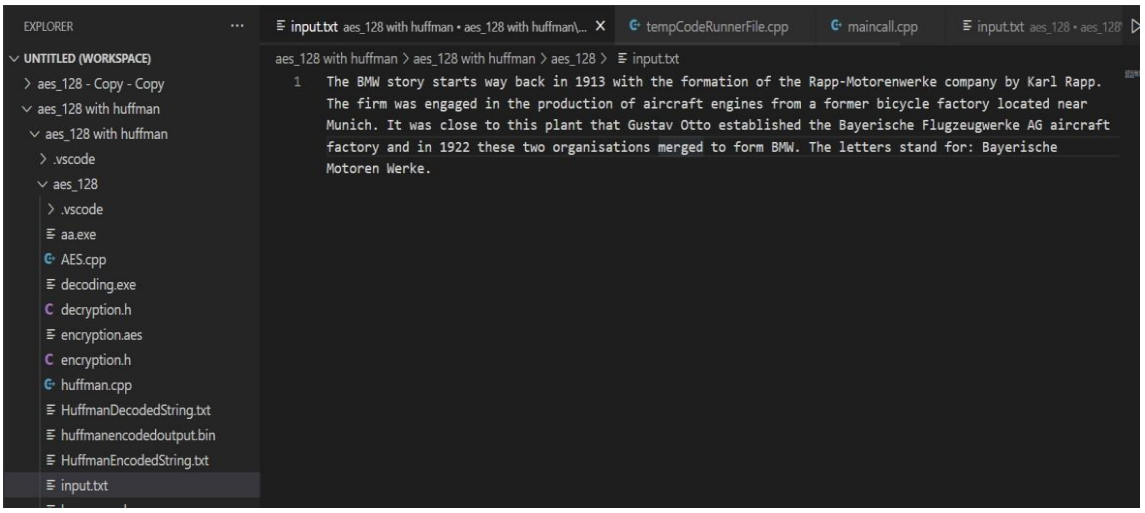

Figure 11

2) Initially the size of our input.txt file is 381 bytes.

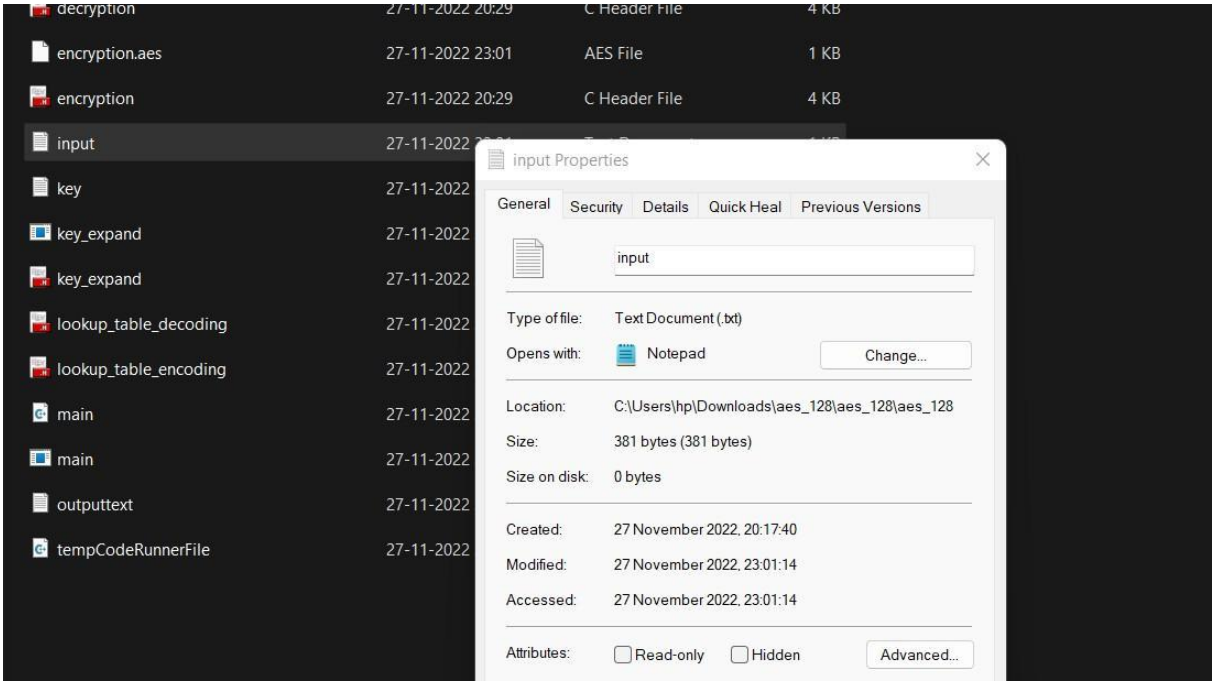

Figure 12

3) Size after encryption using only AES is still 381 Bytes.

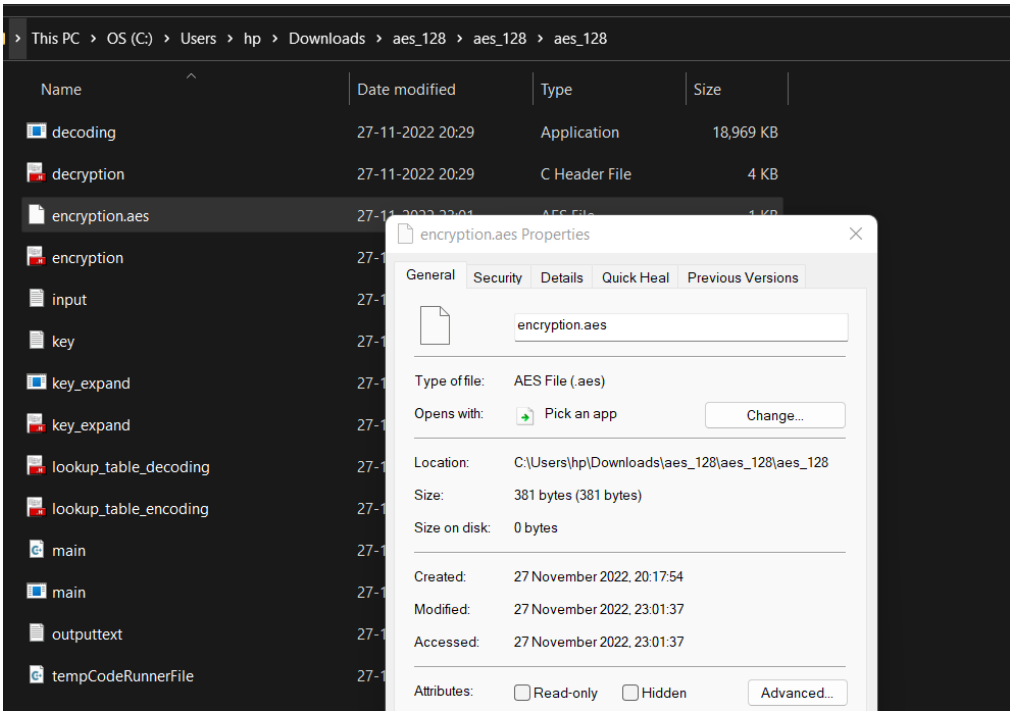

Figure 13

4) Size of encrypted file after using Huffman with AES.

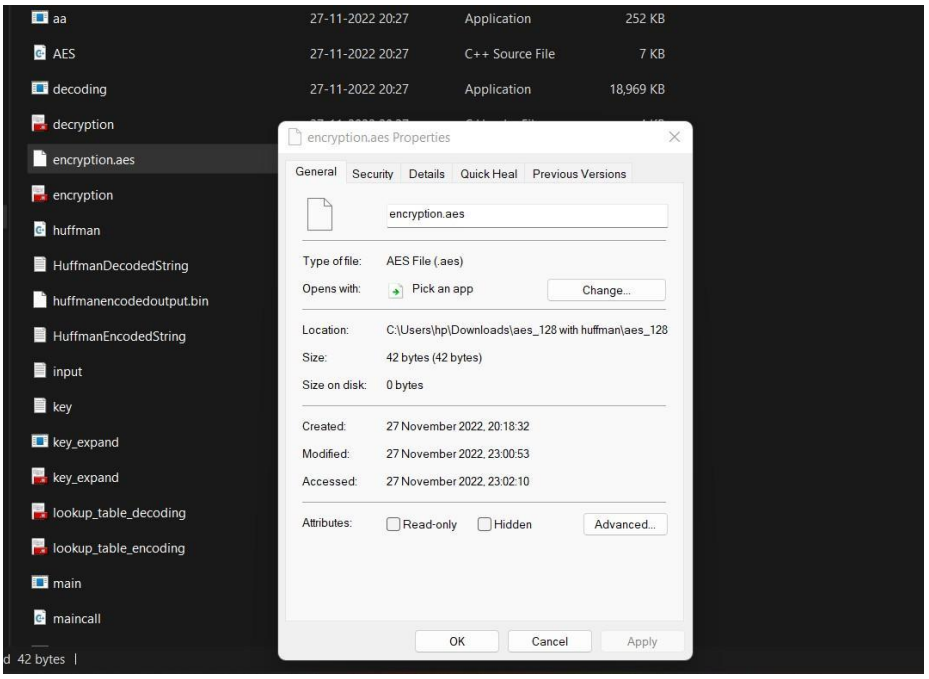

Figure 14

7. Results

The size of the ciphertext is noted twice. First, when only AES is performed to generate normal ciphertext. Second, when Huffman encoding is performed with AES to generate compressed ciphertext.
Finally, the compression factor is calculated.

After AES

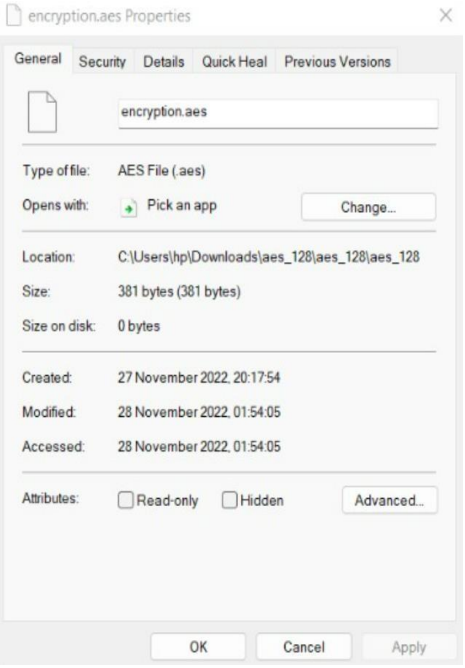

Figure 15

After AES with Huffman

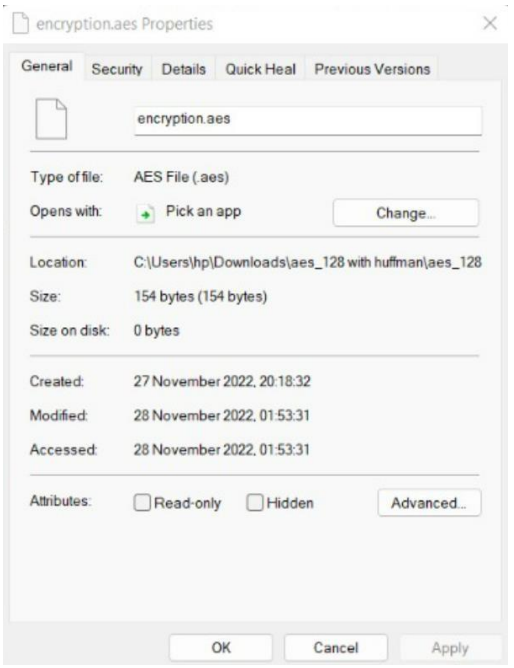

Figure 16

Based on the same procedure, compression factor for the different ciphertext is calculated.

Size of normal ciphertext	Size of compressed ciphertext	Compression factor
382	154	2.48
1108	518	2.13
2769	617	4.48
208	107	1.94
258	104	2.48

(Table 1: Comparison Table.)

8. Conclusion and future scope:

By performing our project, we have successfully concluded that we can use a combination of advanced encryption standard (AES) and huffman encoding to encrypt our files as well as to reduce the size of our file which in turn increases the efficiency and security of our file data. This combinational approach also reduces the bandwidth requirement, time of delivery and other resources needed for the transmission of loss less data.
This approach can be further extended to compress and encrypt the file of different types like PDF, images, DOCs, etc.

References

[1] M. R. Ashila, N. Atikah, D. R. Ignatius Moses Setiadi, E. H. Rachmawanto and C. A. Sari, "Hybrid AES-Huffman Coding for Secure Lossless Transmission," 2019 Fourth International Conference on Informatics and Computing (ICIC), 2019, pp. 1-5, doi: 10.1109/ICIC47613.2019.8985899.

[2] B. Bhat, A. W. Ali and A. Gupta, "DES and AES performance evaluation," International Conference on Computing, Communication & Automation, 2015, pp. 887-890, doi: 10.1109/CCAA.2015.7148500.

GitHub Link

github.com/Yuvraj7788/Hybrid-Encryption-Compression-Algorithm/