VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Computer Science and Engineering

Cryptography Fundamentals- CSE 1011
PROJECT COMPONENT

COMPARISON AND IMPLEMENTATION OF PROTECTION OF FILE SYSTEM USING VARIOUS ENCRYPTION METHODS

SUBMITTED TO:

Dr. Govinda K

SUBMITTED BY:

Rohan Allen- 18BCI0247
Shashank Gummuluru- 18BCI0207

Acknowledgement:

We would like to express our special thanks and gratitude to Dr Govinda K for his guidance and support in the completion of this project. We would also like to extend our gratitude to the Vellore Institute of Technology, Vellore for providing us with the facilities required in the completion of this project.

Abstract:

Data security is a very serious issue which we are facing today in this digital world of communication. There are numerous hackers in every nook and corner of this globe in search of our data for their personal advantage. Thus, we need to come up with a methodology to enable the secure transmission of sensitive data. Our project aims to encrypt and decrypt confidential files so that they do not end up in the wrong hands. We are encrypting and decrypting the files using the AES encryption algorithm.

Introduction:

If we develop some basic cryptosystem that has the objective to encrypt and decrypt the given data then few fundamentals that must be kept in mind are as follows:

• Plaintext: It is the data which has to be secured during the transfer of data.

• Encryption Algorithm: It is a mathematical function that will generate a ciphertext depending on the plaintext and the encryption key. Its main purpose is to take plaintext and an encryption key as input to produce a ciphertext.

• Ciphertext: The ciphertext is a jumbled up version of the plaintext that is determined based on the encryption key.The ciphertext is not secured. It passes through a public channel. It can be manipulated or compromised by anyone who has access to the communication channel.

• Decryption Algorithm: It is a mathematical function,that generates the plaintext for any given ciphertext and decryption

key. It is a cryptographic algorithm that takes a ciphertext and a decryption key as input, and outputs a plaintext.

• Encryption Key: Its value is always known to the sender. The sender enters the encryption key into the encryption algorithm with the plaintext to compute the ciphertext.

• Decryption Key: Its value is always known to the receiver. The decryption key is opposite to the encryption key. The receiver enters the decryption key into the decryption algorithm with the ciphertext to compute the plaintext.

Literature Review:

AES:

The Advanced Encryption Standard is an encryption algorithm established by the U.S National Institute of Standards and Technology (NIST) in 2001. It is currently the most famous and popularly used symmetric encryption algorithm. It is found to be at least six times faster than DES.

AES is an iterative security algorithm rather than an algorithm which follows the Feistel structure. It is based on 'substitution–permutation network'. It comprises of a series of linked operations, some of which involve replacing inputs by specific outputs (substitutions) and others involve shuffling bits around (permutations).

Astoundingly, AES performs all its computations on bytes rather than bits. Therefore, AES treats the 128 bits of a plaintext block as 16 bytes. These 16 bytes are arranged in four columns and four rows for processing as a matrix −

Unlike DES, the number of rounds in AES is variable and depends on the length of the key. AES uses 10 rounds for 128-bit keys, 12 rounds for 192-bit keys and 14 rounds for 256-bit keys. Each of these rounds uses a different 128-bit round key, which is calculated from the original AES key.

The features of AES are as follows:

Block Size: 128 bit Plaintext [16 Bytes]

Number of Rounds: 10

Key Size: 128 bit [16 bytes]

Number of Subkeys: 44

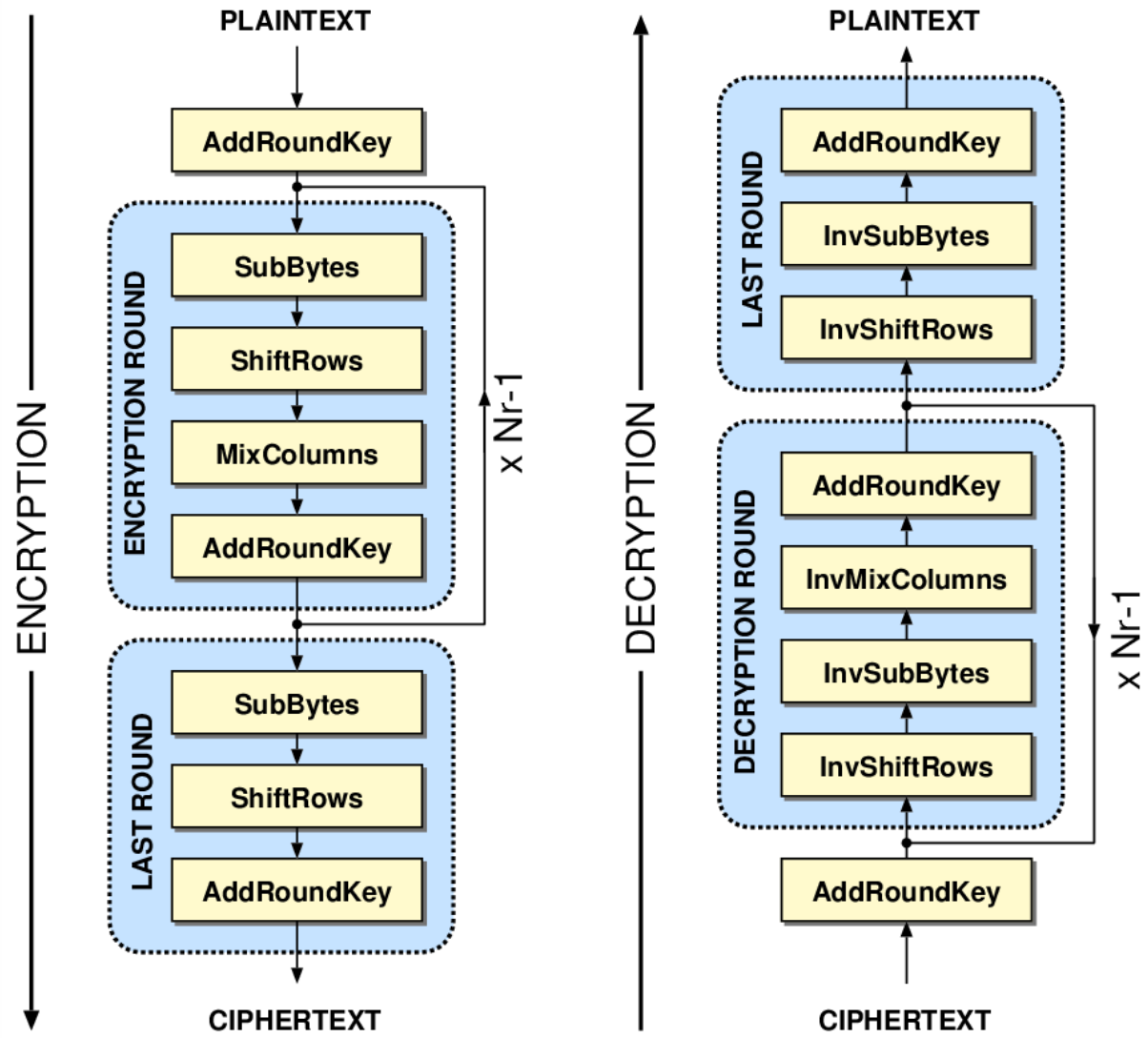
Each Subkey Size: 4 bytes

In Each Round 4 Subkeys are used. In the pre-round calculation also 4 subkeys are used.

Cipher Text: 128 bit

AES can be broken down into 4 steps:

1. Sub Bytes
2. Shift Rows
3. Mix Columns
4. Add Round Key


ENCRYPTION
PLAINTEXT
AddRoundKey
ENCRYPTION ROUND
SubBytes
ShiftRows
MixColumns
AddRoundKey
x Nr-1
LAST ROUND
SubBytes
ShiftRows
AddRoundKey
CIPHERTEXT
DECRYPTION
PLAINTEXT
LAST ROUND
AddRoundKey
InvSubBytes
InvShiftRows
DECRYPTION ROUND
AddRoundKey
InvMixColumns
InvSubBytes
InvShiftRows
x Nr-1
AddRoundKey
CIPHERTEXT

These 4 steps make up a round. Before the first round, Add Round Key is performed, and in the last round, Mix Columns is not performed.

Each round has a certain key of its own, derived from the key given. The algorithm to obtain the round keys is called the key schedule.

With a larger key size, AES has a larger number of rounds. The reason is that given more key bits, there is a need for more rounds so as to ensure that there is more confusion and thus more security.

Sub bytes:

A fixed S-box is used, taking in a byte as input and also producing a byte. The 16 input bytes are substituted by looking up a fixed table (S-box) given in design. The result is in a matrix of four rows and four columns.

Shift Rows and Mix Columns

Each of the four rows of the matrix is shifted to the left. The leftover entries are re-inserted on the right side of row. Shift is carried out as follows −

- First row is not shifted.
- Second row is shifted one (byte) position to the left.
- Third row is shifted two positions to the left.
- Fourth row is shifted three positions to the left.
- The result is a new matrix consisting of the same 16 bytes but shifted with respect to each other.

Mix Columns performs a transformation on each column of the matrix. It multiplies the matrix of bytes with:

Each column of four bytes is now transformed using a special mathematical function. This function takes as input the four bytes of one column and outputs four completely new bytes, which replace the original column. The result is another new matrix consisting of 16 new bytes. It should be noted that this step is not performed in the last round.

AddRoundKey

The 16 bytes of the matrix are now considered as 128 bits and are XORed to the 128 bits of the round key. If this is the last round then the output is the ciphertext. Otherwise, the resulting 128 bits are interpreted as 16 bytes and we begin another similar round.

Decryption Process

The process of decryption of an AES ciphertext is similar to the encryption process in the reverse order. Each round consists of the four processes conducted in the reverse order −

- Add round key
- Mix columns
- Shift rows
- Byte substitution

Since sub-processes in each round are in reverse manner, unlike for a Feistel Cipher, the encryption and decryption algorithms needs to be separately implemented, although they are very closely related.

Encryption Code:

```
from Crypto.Cipher import AES
from Crypto.Random import get_random_bytes
import os

fileName = raw_input("Enter name of the file (with extension) that
needs to be encrypted\t")

try:
    f = open(str(fileName), "r")
    if f.mode == "r":
        data = f.read()
except (IOError):
    print('File does not exist')
    exit()

key = raw_input("Enter a key of length 16\t")
while(1):
    if(len(str(key)) == 16):
        break
    else:
        print("Key must be of length 16\n")
        key = input("Enter a key of length 16\t")
keyBytes = bytes(key)

cipher = AES.new(keyBytes, AES.MODE_EAX)
ciphertext, tag = cipher.encrypt_and_digest(data)
encryptedFileName = raw_input("Enter name of the file where you
want to save the encrypted data\t")
```

```
eName = str(encryptedFileName) +".bin"
file_out = open(eName, "wb")
[ file_out.write(x) for x in (cipher.nonce, tag, ciphertext) ]
os.remove(str(fileName))
```

Decryption Code:

```
from Crypto.Cipher import AES

fileNameTemp = raw_input("Enter name of file to be decrypted\t")
fileName = fileNameTemp + ".bin"

file_in = open(str(fileName), "rb")
nonce, tag, ciphertext = [ file_in.read(x) for x in (16, 16, -1) ]
key = raw_input("Enter your secret key\t")
keyBytes = bytes(key)
try:
    cipher = AES.new(keyBytes, AES.MODE_EAX, nonce)
    data = cipher.decrypt_and_verify(ciphertext, tag)
    print(data)
except ValueError:
    print("Key or message is corrupt")
```

Screenshots of code

```python
from Crypto.Cipher import AES
from Crypto.Random import get_random_bytes
import os

fileName = raw_input("Enter name of the file (with extension) that needs to be encrypted\t")

try:
    f = open(str(fileName), "r")
    if f.mode == "r":
        data = f.read()
except (IOError):
    print('File does not exist')
    exit()

key = raw_input("Enter a key of length 16\t")
while(1):
    if(len(str(key)) == 16):
        break
    else:
        print("Key must be of length 16\n")
        key = input("Enter a key of length 16\t")
keyBytes = bytes(key)

cipher = AES.new(keyBytes, AES.MODE_EAX)
ciphertext, tag = cipher.encrypt_and_digest(data)
encryptedFileName = raw_input("Enter name of the file where you want to save the encrypted data\t")

eName = str(encryptedFileName) +".bin"
file_out = open(eName, "wb")
[ file_out.write(x) for x in (cipher.nonce, tag, ciphertext) ]
os.remove(str(fileName))

```

```python
from Crypto.Cipher import AES

fileNameTemp = raw_input("Enter name of file to be decrypted\t")
fileName = fileNameTemp + ".bin"

file_in = open(str(fileName), "rb")
nonce, tag, ciphertext = [ file_in.read(x) for x in (16, 16, -1) ]
key = raw_input("Enter your secret key\t")
keyBytes = bytes(key)
try:
    cipher = AES.new(keyBytes, AES.MODE_EAX, nonce)
    data = cipher.decrypt_and_verify(ciphertext, tag)
    print(data)
except ValueError:
    print("Key or message is corrupt")

```

Explanation of code:

Encryption:
The input is the file name that is to be encrypted. Along with this, the user inputs their own unique 16bit password which is used to encrypt the file. Along with this, the name of the file where the encrypted binary file is to be stored is also inputted by the user. The file is then encrypted and saved with the original file being deleted. The pycryptodome and os libraries are used to help encrypt the files and delete the file from the computer.

Decryption:
At the time of decryption, the file name where the encrypted data is stored is given as input. The key that was used at the time of encryption is also asked. Incorrect key results in the termination of the program. When the correct key is given the file is decrypted and the ciphertext that has been converted to plaintext is displayed directly on the terminal. The pycryptodome library is used to decrypt the given encrypted file.

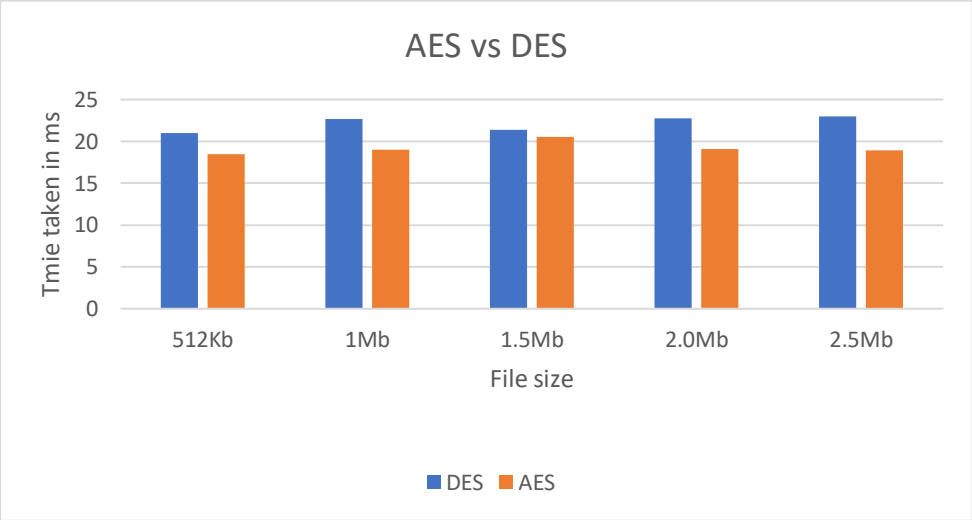
Result and Analysis:

Graph:

This graph shows the time AES and DES security algorithms take for file encryption of various sizes. Thus AES is more secure than DES and more efficient.

File Size	AES	DES
512kB	18.5	21.02
1Mb	19	22.7

1.5Mb	20.52	21.35
2.0Mb	19.08	22.77
2.5Mb	18.92	23.01



AES Analysis

In current times AES is widely adopted and used by multiple organisations and business vendors all across the world . At the current time of writing, no practical cryptanalytic attacks against AES has been discovered. Also, AES has various possible key lengths which can be selected according to the situation required, which allows AES to adapt to various technological breakthroughs and advances. Thus AES is one of the most secure and speedy encryption algorithms at the moment

However, the AES security is possible only if it is implemented correctly and the key is managed properly.

References:

- *"Biclique Cryptanalysis of the Full AES" (PDF). Archived from the original(PDF) on March 6, 2016. Retrieved sept. 23, 2019*

- *En, Ng Wei. "Why AES Is Secure." Why AES Is Secure, 17 Jan. 2017, weien.io/posts/crypto/why-aes-is-secure.html.*

- *Nyberg K. (1991) Perfect nonlinear S-boxes. In: Davies D.W. (eds) Advances in Cryptology — EUROCRYPT '91. EUROCRYPT 1991. Lecture Notes in Computer Science, vol 547. Springer, Berlin, Heidelberg*

- *Jie Cui; Liusheng Huang; Hong Zhong; Chinchen Chang; Wei Yang (May 2011). "AN IMPROVED AES S-BOX AND ITS PERFORMANCE ANALYSIS" (PDF).*

- *William Stallings. Cryptography and Network security: Principles and Practices. Prentice Hall Inc., second edition, 1999.*

- *Behrouz A. Forouzan. Cryptography and Network Security. Tata McGraw-Hill, 2007.*

Annex 1: Plagiarism report