VIT®

Vellore Institute of Technology

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

ECE4007 - INFORMATION THEORY AND CODING

J component Report

Title: Text encryption and decryption using AES and DES Algorithm

Slot: C1+TC1

Submitted by:

Nikhil Subramanian 18BEC0711
Zainab Shakruwala 18BEC0744

Abstract:
With the advancements of technology towards IoT, it very important to have proper Information security to safe guard information. In this project, we will be using Advanced Encryption Standard (AES) and Data Encryption Standard (DES) algorithm to encrypt and decrypt texts. We will be comparing them.

Introduction:

The Advanced Encryption Standard or AES, also known as Rjindael is a technique used for the encryption of electronic data and was established by the United States National Institute of Standards and Technology in 2001. The information is encrypted using AES. The cryptanalysis of the algorithm is then performed and is proved to be secure. The proposed algorithm is then implemented using python and the results are discussed along with the possible future modifications. Python has an extensive library for Cryptography called the "Pycryptodome", which enables us to apply different encryption schemes without writing huge snippets of code.

Data Encryption Standard (DES) was developed in 1977. It follows the principal of Feistal Structure. In DES the plain text is divided into two halves before processing. It contains a fixed number of 16 rounds. It is a block cipher and encrypts data in blocks of size of 64 bits each. Which means 64 bits of plain text goes as the input to DES, which produces 64 bits of ciphertext. The key length is 56 bits. DES is based on the two fundamental attributes of cryptography called substitution and transposition.

Literature Survey:

Authors	Na Su , Yi Zhang, Mingyue Li	Guerrero-Sanchez, A.E.; Rivas-Araiza, E.A.; Gonzalez-Cordoba, J.L.; Toledano-Ayala, M.; Takacs, A.	Saracevic, M. H., Adamovic, S. Z., Miskovic, V. A., Elhoseny, M., Macek, N. D., Selim, M. M., & Shankar, K.
Year of Publishing	2019	2020	2020
Title	Research on Data Encryption Standard Based on AES Algorithm in Internet of Things Environment	Blockchain Mechanism and Symmetric Encryption in A Wireless Sensor Network	Data Encryption for Internet of Things Applications Based on Catalan Objects and Two Combinatorial Structures
Methodology	DESI	Decentralized Approach based on a private blockchain method using AES	DES based on Catalan object
Metrics and Remarks	Comparison with AES and DESI for runtime and data size	Hardware performance and Security Analysis	NIST Quality assurance test

Tools Used:

Python
Google Colab

Methodology and Block Diagram:

1) AES (18BEC0711)

Algorithm:

1. Import Random and Cipher modules
2. Import the base64 module to encrypt and decrypt strings into byte like objects.
3. From the cipher modules import AES
4. Define the length of the key and also define two Lambda functions to define pad and unpad the data to be encrypted. This is done because AES works with 16 bytes of data or multiples of 16.
5. Create a class which will take the key as an argument and inside the class declare 2 methods
6. First method calls the pad function and encrypts the data.
7. Second method will decrypt the encryption and unpad the data.
8. Finally call the methods and verify the result.

Encryption:

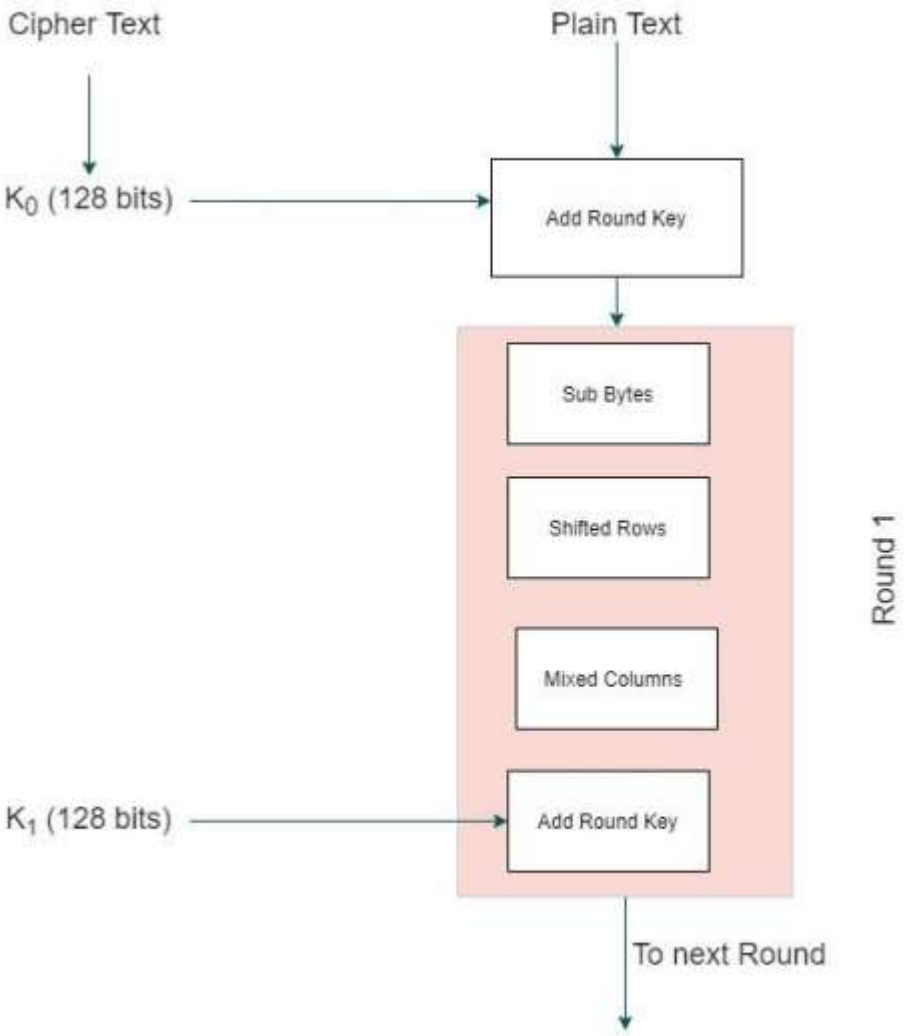

Figure 1. AES Encryption

AES performs its operations in bytes rather than bits. Hence, a text of 128 bits is treated as 16 bytes of data and is converted to 4×4 matrix. AES is based on a substitution-permutation network. Basically there are various rounds of shuffling the matrix. These number of rounds are based on the AES key provided by the user. AES uses 10 rounds for 128 bit key, 192 bit key uses 12 rounds and 256 key uses 14 rounds. Each round uses a different 128 bit key which is calculated from the original AES key. The image shown above is just one round of the many rounds that produces the final output. Each round is divided into 4 operations namely, SubBytes, Shift Rows, Mix Columns and AddroundKey.

SubBytes

The conversion of 16 byte in a four by four matrix takes place in this step.

Shift Rows

This operation comprises of 4 steps

1. First row is not shifted

2. Second row is shifted to the left by 1
3. Third row is shifted to the left by 2
4. Fourth row is shifted to the left by 3

Mix Columns

Each column of the matrix is taken individually and is input to a mathematical function. The output of the mathematical function is another 4 bytes which is substituted in the matrix in place of the input column.
Thus we get a whole new matrix. Here it is to be noted that this step is skipped in the last round.

AddroundKey

In this step the matrix obtained is again converted into 128 bits and this is then XORed with the original key to get the 128 bits key for the next round. If this is the last round then this is the encrypted data that we had to obtain finally. Else the 128 bits is the key for the next round the same operations are performed.

Decryption

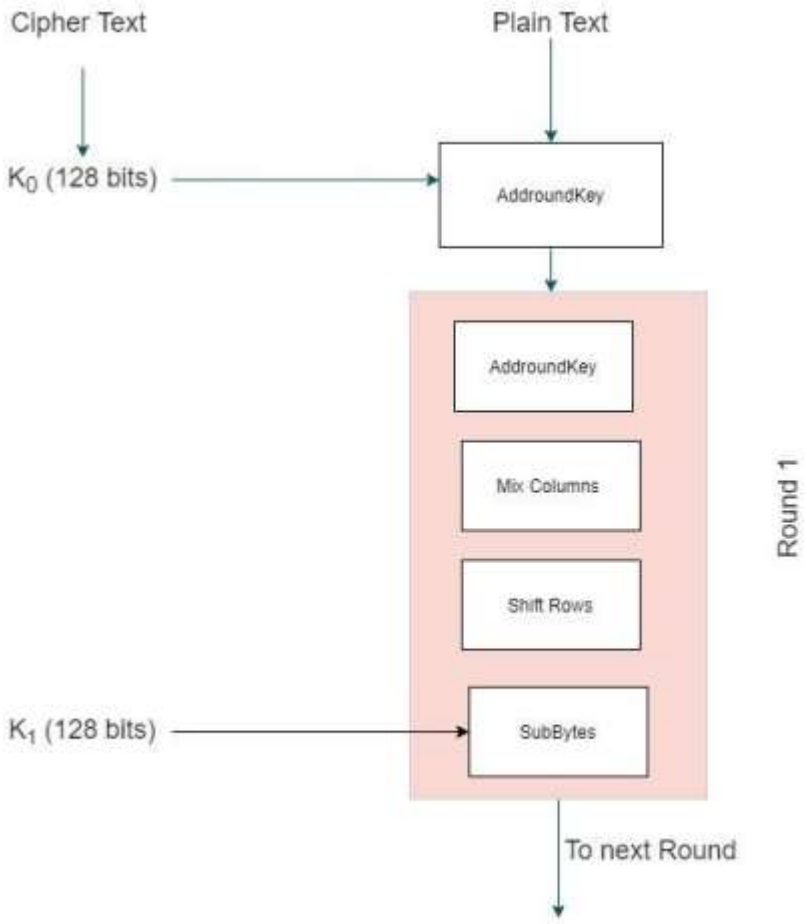

Figure 2. AES Decryption

The procedure for decryption is the reverse of the encryption process. It means now the operation are in the
following order:
1. AddroundKey
2. Mix columns
3. Shift rows
4. SubBytes

2) DES (18BEC0744)

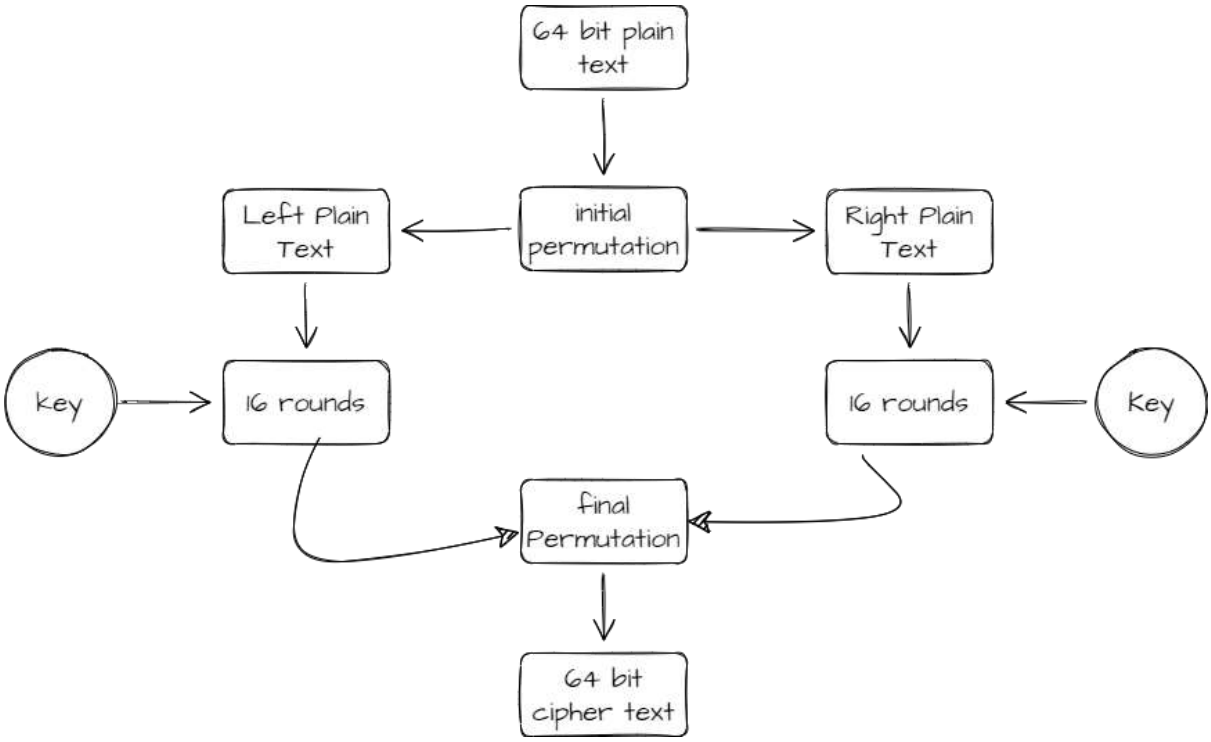

Figure 3. A brief summary of DES

The initial key in DES consists of 64 bits. But before the DES process starts every 8th bit of the key is discarded to produce a 56 bit key. Hence bit positions 8, 16, 24,32,40,48,56, and 64 are discarded.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64

Encryption:

1. In the first step, the 64-bit plain text block is handed over to an initial Permutation (IP) function.
2. The IP function is performed on the text.

3. Next, the IP two halves of the permuted block; say Left Plain Text(LPT) and Right Plain Text(RPT).
4. Each LPT and RPT goes through 16 rounds of encryption process
5. Finally, LPT and RPT are rejoined and a final permutation is performed on the combined block.
6. The result is a 64 bit cypher text.

Initial Permutation:

IP happens only once and it happens before the first round. It suggests how the transportation should proceed. For e.g. let the first bit be replaced by 58, next bit by 50 and so on.

58	50	42	34	26	18	10	2	60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6	64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1	59	51	43	35	27	19	11	3
61	33	45	37	29	21	13	5	63	55	47	39	31	23	15	7

The resulting text block will be divided into two half blocks. Each block has of 32 bits. In each round the following steps take place.

1. Key Transformation
2. Expansion Permutation
3. S-box Permutation
4. P-box Permutation
5. XOR and swap

Key Transformation:

For the available 56 bit key a different 48 bit key is generated during each round using a process called key transformation. For this, the 56 bit key is divided into two halves each of 28 bits. These halves are circularly shifted left by one or two positions, depending on the round. For example, if the round numbers 1, 2, 9, or 16 the shift is done by only position for other rounds, the circular shift is done by two positions.

After an appropriate shift, 48 of the 56 bits are selected. This is known as compression permutation. Because of this technique a different subset key bits is used in each round. That makes DES not easy to crack.

Expansion Permutation:

RPT is expanded from 32 bits to 48 bits. These bits are permuted as well. This happens as we divide RPT into 8 blocks, with each block consisting of 4 bits. Then each 4-bit block is expanded to a 6 bit block. i.e. per 4 bits, 2 more bits are added.

This process results in expansion as well as a permutation of the input bit while creating output. The key transformation process compresses the 56-bit key to 48 bits. Then the expansion permutation process expands the 32-bit RPT to 48-bits. Now the 48-bit key is XOR with 48-bit RPT and the resulting output is given to the next step, which is the S-Box substitution.

Results and Discussions:

We ran the two algorithms and ran them with the following results. We encrypted and decrypted the text and found that AES was the better algorithm.

```
        def encrypt( self, raw ):
            raw = pad(raw)
            iv = Random.new().read( AES.block_size )
            cipher = AES.new(self.key, AES.MODE_CBC, iv )
            file_out = open(self.key_location, "wb") # wb = write bytes
            file_out.write(self.key)
            file_out.close()
            return base64.b64encode( iv + cipher.encrypt( raw ) )

        def decrypt( self, enc ):
            enc = base64.b64decode(enc)
            iv = enc[:16]
            file_in = open(self.key_location, "rb")
            key_from_file = file_in.read()
            file_in.close()
            cipher = AES.new(key_from_file, AES.MODE_CBC, iv )
            return unpad(cipher.decrypt( enc[16:] )).decode('utf8')

cipher = AESCipher('mysecretpasswordmysecretpassword','/content/AES Key.txt')
encrypted = cipher.encrypt('SmartPhones')
decrypted = cipher.decrypt(encrypted)
print(encrypted)
print(decrypted)
```

```
b'6WgYRsh0TiJXkJX52bBGFUpqdk99+T6D5c8oJY/eCAg='
SmartPhones
```

Figure 4. Result of AES

```
Enter the message to be encrypted:
AB4C8C2BC3FBC5E3
No padding required
Message after padding:  AB4C8C2BC3FBC5E3
Enter the 64bit key for encryption:
74631BBDCA8
Padding required
Key after padding:  74631BBDCA800000
Encryption
Message after initial permutation:  F22046F9F5A92FB9
The converted 56bit key is:  00111000000100110000101100000001011000001001000111001101

Round: Left key part: Right key part: SubKey used:
01       F5A92FB9        22132F79     16AB20801DC9
02       22132F79        9DF2FECE     54412204E41E
03       9DF2FECE        21DD0CEA     10C4D8788942
04       21DD0CEA        B17D0A46     4930C4372660
05       B17D0A46        B19E934C     231E0184B313
06       B19E934C        A0D071FC     942303B208CA
07       A0D071FC        3B67962B     80C9B8746225
08       3B67962B        E1DA5EA7     494060887282
09       E1DA5EA7        38D6DC84     4814910716B4
10       38D6DC84        C9B13378     23270490981F
11       C9B13378        AD331A02     A46803610AE8
12       AD331A02        A506590C     108972C57034
13       A506590C        F7D4C67D     40D08808191A
14       F7D4C67D        A58AB628     69064C734D28
15       A58AB628        31FA5C5F     8A34034AB231
16       6A1A9764        31FA5C5F     A289056D34C0
Cipher text is:  86760F7ABEE16B24
```

Figure 5. Encryption using DES

```
Enter the message to be encrypted:
86760F7ABEE16B24
No padding required
Message after padding:  86760F7ABEE16B24
Enter the 64bit key for encryption:
74631BBDCA8
Padding required
Key after padding:  74631BBDCA800000
Encryption
Message after initial permutation:  6A1A976431FA5C5F
The converted 56bit key is:  00111000000100110000101100000001011000001001000111001101

Round: Left key part: Right key part: SubKey used:
01       31FA5C5F        A58AB628     A289056D34C0
02       A58AB628        F7D4C67D     8A34034AB231
03       F7D4C67D        A506590C     69064C734D28
04       A506590C        AD331A02     40D08808191A
05       AD331A02        C9B13378     108972C57034
06       C9B13378        38D6DC84     A46803610AE8
07       38D6DC84        E1DA5EA7     23270490981F
08       E1DA5EA7        3B67962B     48149107 16B4
09       3B67962B        A0D071FC     494060887282
10       A0D071FC        B19E934C     80C9B8746225
11       B19E934C        B17D0A46     942303B208CA
12       B17D0A46        21DD0CEA     231E0184B313
13       21DD0CEA        9DF2FECE     4930C4372660
14       9DF2FECE        22132F79     10C4D8788942
15       22132F79        F5A92FB9     54412204E41E
16       F22046F9        F5A92FB9     16AB20801DC9
Cipher text is:  AB4C8C2BC3FBC5E3
```

Figure 5. Decryption using DES

Comparison:

Basis of Comparison	DES	AES
Developed	DES was developed in 1977	AES was developed in 2001
Full-Form	DES stands for Data Encryption Standard	AES stands for Advanced Encryption Standard
Principle	DES follows the principle of Feistel Structure	AES s based on the principle of Substitution and Permutation
Plaintext	The plaintext is of 64 bits.	The plaintext can be 128, 192, 256 bits.
Ciphertext	Generate Ciphertext of 64 bits	Can Generate Ciphertext of 128, 192, 256 bits
Key Length	The key length is 56 bits.	Key length can be 128, 192, 256 bits.
Rounds	DES contains a fixed number of rounds, i.e. 16	AES contains a variable number of rounds depending on the size of the input, i.e. 10 rounds for 128 bit, 12 rounds for 192 bit and 14 rounds for 256 bits
Security	DES is less secure and hardly used now	AES is much more secure than DES and is widely used nowadays.
Speed	DES is comparatively slower than AES	AES is faster than DES

Key Comparisons:

- The main difference between DES vs AES is the process of encrypting. In DES, the plaintext is divided into two halves before further processing, whereas in AES whole block, there is no division, and the whole block is processed together to produce the ciphertext.
- AES is comparatively much faster than DES and can encrypt large files in a fraction of seconds compared to DES
- DES is considered more vulnerable to brute-force attacks, whereas AES has not encountered any serious attacks.
- AES is practically efficient with both hardware and software implementations, unlike DES, which was initially efficient with only hardware.
- AES is practically efficient with both hardware and software implementations, unlike DES, which was initially efficient with only hardware.

References:

[1] N. Su, Y. Zhang and M. Li, "Research on Data Encryption Standard Based on AES Algorithm in Internet of Things Environment," 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), 2019, pp. 2071-2075, doi: 10.1109/ITNEC.2019.8729488.

[2]A. E. Guerrero-Sanchez, E. A. Rivas-Araiza, J. L. Gonzalez-Cordoba, M. Toledano-Ayala, and A. Takacs, “Blockchain Mechanism and Symmetric Encryption in A Wireless Sensor Network,” Sensors, vol. 20, no. 10, p. 2798, May 2020.

[3]M. H. Saračević et al., "Data Encryption for Internet of Things Applications Based on Catalan Objects and Two Combinatorial Structures," in IEEE Transactions on Reliability, vol. 70, no. 2, pp. 819-830, June 2021, doi: 10.1109/TR.2020.3010973**.**