

NAME: Sarvesh Patil

CLASS: D15A

ROLL NO: 52

LAB 05

LAB 05: To understand how to Encrypt long messages using various modes of operation using AES or DES.

ROLL NO	52
NAME	Sarvesh Patil
CLASS	D15A
SUBJECT	Internet Security Lab
LO MAPPED	LO1: To apply the knowledge of symmetric cryptography to implement classical ciphers

AIM:

To understand how to Encrypt long messages using various modes of operation using AES or DES.

INTRODUCTION:**AES:**

Advanced Encryption Standard (AES) is a specification for the encryption of electronic data established by the U.S National Institute of Standards and Technology (NIST) in 2001. AES is widely used today as it is a much stronger than DES and triple DES despite being harder to implement.

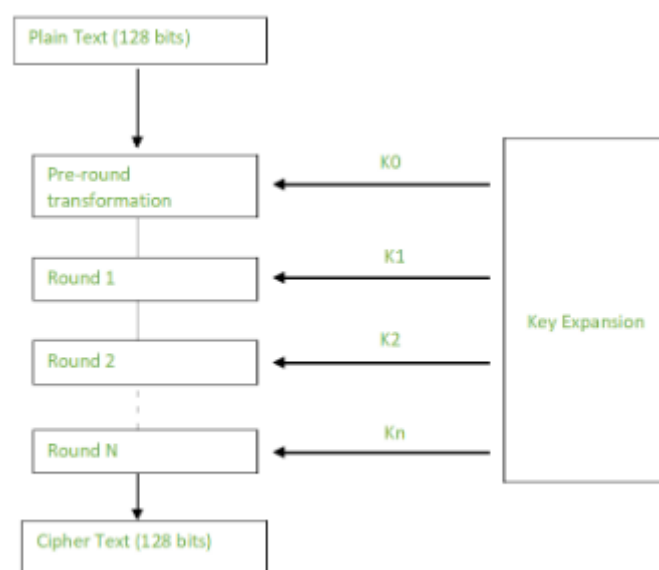
Points to remember:

- AES is a block cipher.
- The key size can be 128/192/256 bits.
- Encrypts data in blocks of 128 bits each.

That means it takes 128 bits as input and outputs 128 bits of encrypted cipher text as output. AES relies on the substitution-permutation network principle which means it is performed using a series of linked operations that involves replacing and shuffling the input data.

AES is an iterative rather than a Feistel cipher. It is based on a 'substitution-permutation network'. It comprises a series of linked operations, some of which involve replacing inputs with specific outputs (substitutions) and others involve shuffling bits around (permutations). Interestingly, AES performs all its computations on bytes rather than bits. Hence, 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.

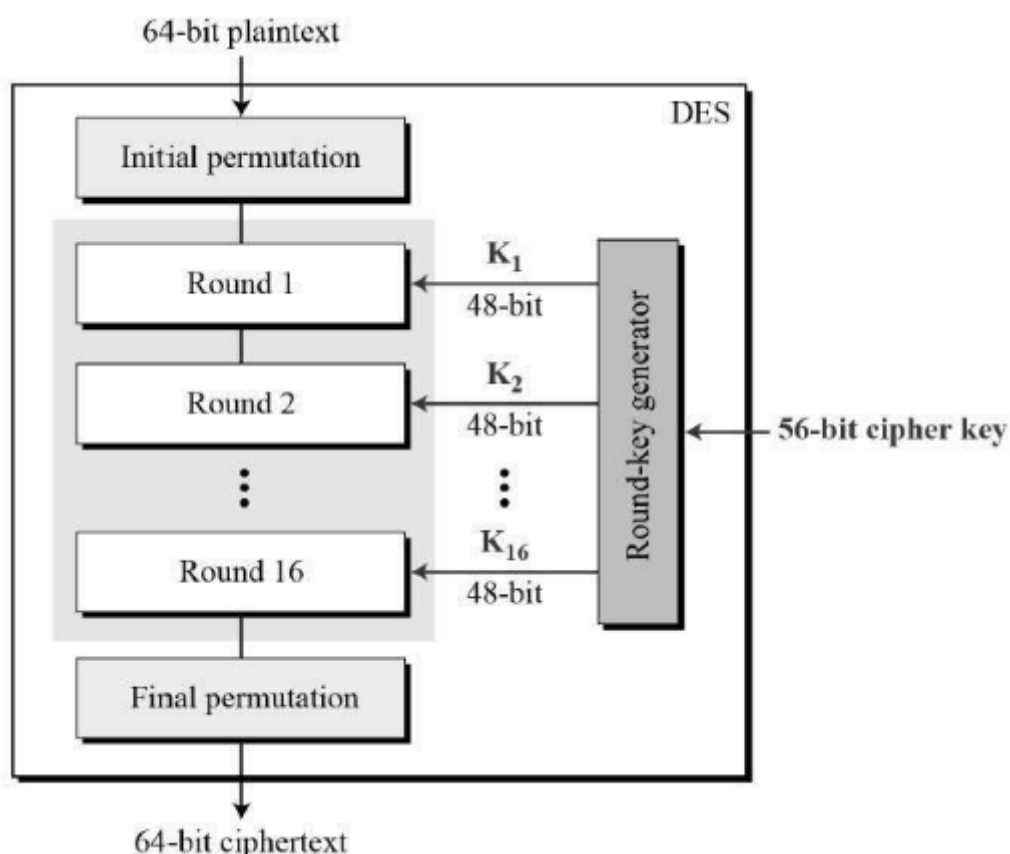


DES:

Over the last decade the world has seen an astounding growth of information technology that has resulted in significant advances in cryptography to protect the integrity and confidentiality of data. The most affected are the financial sector and e-commerce over the Internet, which requires secure data transfer, or handling during any transaction of business. Secrecy is the heart of cryptography. Encryption is a practical means to achieve information secrecy. In this aspect DES (Data Encryption Standard)- A symmetric key cryptography and its variant triple DES, has over the last three decades played a major role in securing data in this sector of the economy and within other governmental and private sector security agencies. In this study, the theory and implementation of DES and its suitability for securing data in the future will be described. Comparison with other symmetric key crypto-algorithm will also be considered.

The Data Encryption Standard (DES) is a symmetric-key block cipher published by the National Institute of Standards and Technology (NIST).

DES is an implementation of a Feistel Cipher. It uses 16 round Feistel structure. The block size is 64-bit. Though the key length is 64-bit, DES has an effective key length of 56 bits since 8 of the 64 bits of the key are not used by the encryption algorithm (function as check bits only). The General Structure of DES is depicted in the following illustration –



Since DES is based on the Feistel Cipher, all that is required to specify DES is –

- Round function
- Key schedule

- Any additional processing – Initial and final permutation

METHODS:

ECB:

Electronic Code Book (ECB) is a simple mode of operation with a block cipher that's mostly used with symmetric key encryption. It is a straightforward way of processing a series of sequentially listed message blocks. The input plaintext is broken into numerous blocks. The blocks are individually and independently encrypted (ciphertext) using the encryption key. As a result, each encrypted block can also be decrypted individually. ECB can support a separate encryption key for each block type.

In ECB, each block of plaintext has a defined corresponding ciphertext value and vice versa. So, identical plaintexts with identical keys always encrypt to identical ciphertexts. This means that if plaintext blocks P1, P2, and so on are encrypted multiple times under the same key, the output ciphertext blocks will always be the same. In other words, the same plaintext value will always result in the same ciphertext value. This also applies to plaintexts with partial identical portions. For instance, plaintexts containing identical headers of a letter and encrypted with the same key will have partially identical ciphertext portions.

For any given key, a codebook of ciphertexts can be created for all possible plaintext blocks. With the ECB mode, encryption entails only looking up the plaintext(s) and selecting the corresponding ciphertext(s). This operation is like assigning code words in a codebook. In fact, the term "code book" derives from the cryptographic codebooks used during the United States Civil War (1861-1865). In terms of error correction, any bit errors in a ciphertext block will only affect the decryption of that block. Chaining dependency is not an issue. Any reordering of the ciphertext blocks will only reorder the corresponding plaintext blocks. It won't affect decryption.

CBC:

CBC (short for cipher-block chaining) is an AES block cipher mode that trumps the ECB mode in hiding away patterns in the plaintext. CBC mode achieves this by XOR-ing the first plaintext block (B1) with an initialization vector before encrypting it. CBC also involves block chaining as every subsequent plaintext block is XOR-ed with the ciphertext of the previous block.

If we summarize this process into a formula, it would look like this:

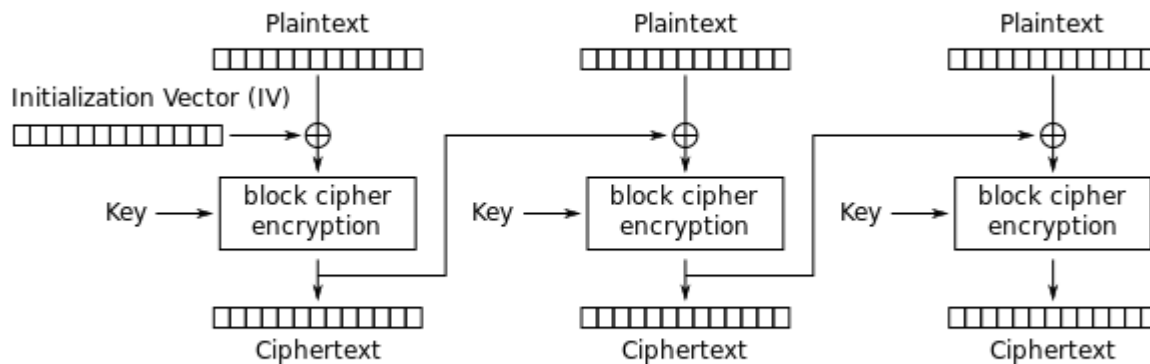
$$C_i = EK(B_i \oplus C_{i-1})$$

where EK denotes the block encryption algorithm using key K, and C_{i-1} is the cipher corresponding to B_{i-1} .

Similarly, decryption using the CBC can be done using:

$$B_i = DK(C_i) \oplus (C_{i-1})$$

where DK denotes the block decryption algorithm using key K.



Cipher Block Chaining (CBC) mode encryption

OUTPUT FEEDBACK:

The output feedback (OFB) mode is similar in structure to that of CFB. As can be seen in Figure 6.6, it is the output of the encryption function that is fed back to the shift register in OFB, whereas in CFB, the ciphertext unit is fed back to the shift register. The other difference is that the OFB mode operates on full blocks of plaintext and ciphertext, not on an s-bit subset. Encryption can be expressed as

$$C_j = P_j \otimes E(K, [C_j - i \otimes P_j - 1])$$

By rearranging terms, we can demonstrate that decryption works.

$$P_j = C_j \otimes E(K, [C_j - 1 \otimes P_j - 1])$$

One advantage of the OFB method is that bit errors in transmission do not propagate. For example, if a bit error occurs in C_1 , only the recovered value of P_1 is affected; subsequent plaintext units are not corrupted. With CFB, C_1 also serves as input to the shift register and therefore causes additional corruption downstream. The disadvantage of OFB is that it is more vulnerable to a message stream modification attack than is CFB. Consider that complementing a bit in the ciphertext complements the corresponding bit in the recovered plaintext. Thus, controlled changes to the recovered plaintext can be made. This may make it possible for an opponent, by making the necessary changes to the checksum portion of the message as well as to the data portion, to alter the ciphertext in such a way that it is not detected by an error-correcting code.

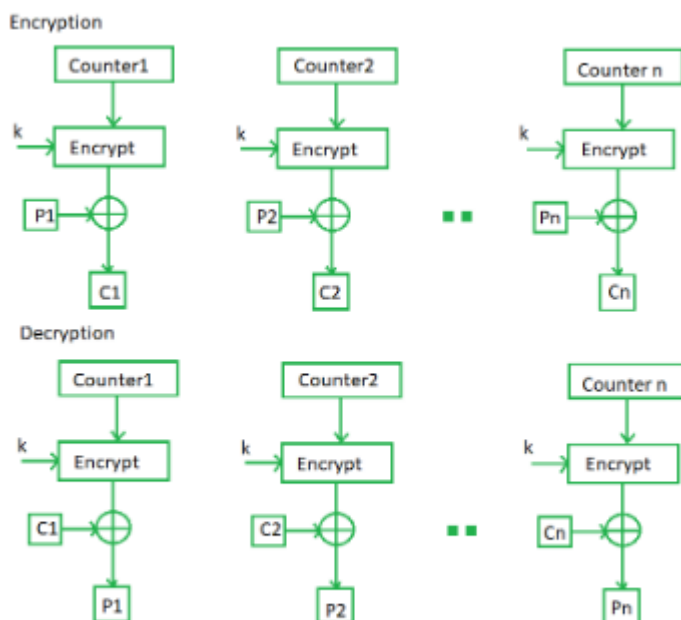
OFB has the structure of a typical stream cipher because the cipher generates a stream of bits as a function of an initial value and a key, and that stream of bits is XORed with the plaintext bits (see Figure 3.1). The generated stream that is XORed with the plaintext is itself independent of the plaintext; this is highlighted by dashed boxes in Figure 6.6. One distinction from the stream ciphers we discuss in Chapter 7 is that OFB encrypts plaintext a full block at a time, where typically a block is 64 or 128 bits. Many stream ciphers encrypt one byte at a time.

COUNTER MODE:

The Counter Mode or CTR is a simple counter-based block cipher implementation. Every time a counter-initiated value is encrypted and given as input to XOR with plaintext it results in a ciphertext block. The CTR mode is independent of feedback use and thus can be implemented in parallel.

Both encryption and decryption in CTR mode are depicted in the following illustration. Steps in operation are –

- Load the initial counter value in the top register is the same for both the sender and the receiver. It plays the same role as the IV in CFB (and CBC) mode.
- Encrypt the contents of the counter with the key and place the result in the bottom register.
- Take the first plaintext block P1 and XOR this to the contents of the bottom register. The result of this is C1. Send C1 to the receiver and update the counter. The counter update replaces the ciphertext feedback in CFB mode.
- Continue in this manner until the last plaintext block has been encrypted.
- The decryption is the reverse process. The ciphertext block is XORed with the output of encrypted contents of the counter value. After decryption of each ciphertext, block counter is updated as in the case of encryption.



RESULTS:

ECB:

Virtual Labs

AES and Modes of Operation

Choose your mode of operation: Electronic Code Book (ECB)

PART II

Key size in bits: 128

a511f1c9 ec7a384f 60618fd2 ab2586d5
2062506a df315a08 7097140a a7728637
eed29bf9 51d679af fa3d9d1d f6a91313
cc37dbd3 2e5c9388 184bc17a d3623768
5649555c 740eba49 a18d256a 7c8ab762

Next Plaintext

Key: b2bba8ff fbc8fd2 cd303ddd dd2dc87

Next Keytext

IV:

Next IV

CTR:

Next CTR

PART III

Calculate XOR:

Calculate XOR

XOR:

PART IV

Key in hex: b2bba8ff fbc8fd2 cd303ddd dd2dc87

Plaintext in hex: 5649555c 740eba49 a18d256a 7c8ab762

Ciphertext in hex: b097c18c 103a7c49 0c5da213 a08e4779

Encrypt Decrypt Clear

PART V

Enter your answer here:

e00af50e 409c88ec f05210a ac085804 7a002247 57206ee2 0f6a93f1 0e9141 Check Answer!

CIPHER BLOCK CHAINING:

Virtual Labs

AES and Modes of Operation

Choose your mode of operation: Cipher Block Chaining

PART II

Key size in bits: 128

c483e51c ea872e5e 1fb49787 58a17aef
7064921d f430549d 67353708 db93ba0b
af3123b1 c1697465 8c367552 597822bc
4ab10557 b110a5e4 02fcb947 43b987c3
34f51d14 b3b306c6 84d165ed 8813b488

Next Plaintext

Key: ff7f478f 57071f92 c1f6efe1 7ca9b6b8

Next Keytext

IV: 7efbfde4 6068a553 a3bb9ec6 190a4b6f

Next IV

PART III

Calculate XOR:

25eff0e5 9e6b5411 c2af6379 e16deab1

34f51d14 b3b306c6 84d165ed 8813b488

Calculate XOR

XOR: 111aef1 2dd852d7 467e0694 697e5e39

PART IV

Key in hex: ff7f478f 57071f92 c1f6efe1 7ca9b6b8

Plaintext in hex: 111aef1 2dd852d7 467e0694 697e5e39

Ciphertext in hex: 5d0aeedc 659075d1 9f8bf4b4 13e28e8b

Encrypt Decrypt Clear

PART V

Enter your answer here:

7efbfde4 6068a553 a3bb9ec6 190a4b6f c33b5c90 3932b1a3 5f2f58e9 71d941 Check Answer!

NAME: Sarvesh Patil

CLASS: D15A

ROLL NO: 52

OUTPUT FEEDBACK:

Virtual Labs
An AICTE Sankat of India Initiative

AES and Modes of Operation

PART I
Choose your mode of operation: Output Feedback

PART II
Key size in bits: 128

61e765f8 58627066 c610cc0c 9e233238
c7fd14f1 20449073 711c7d64 46d6bccf
c0bd8c92 c457a384 3514d64a f8cccebc
7ba865af 756476d8 048b2329 99ccabca
1e401700 d8c0c39c 63f4bfd8 d6a61148

Plaintext: f4eb8ac5 774c1227 963e1109 40654d0e Next Plaintext Key: 42bf0aef b7b6a323 5fbd5c0f 101f841b Next Keytext
IV: f4eb8ac5 774c1227 963e1109 40654d0e Next IV

PART III
Calculate XOR:

6da46dcf 4bcb31ef 780b6125 d4754e74
1e401700 d8c0c39c 63f4bfd8 d6a61148

Calculate XOR

XOR: 73e47acf 930bf273 1bffdefd 02d35f3c

PART IV
Key in hex: 42bf0aef b7b6a323 5fbd5c0f 101f841b
Plaintext in hex: 7ea32818 fd49fd0 9a2044cc f3eef322
Ciphertext in hex: 0da40dcf 4bcb31ef 780b6125 d4754e74
Encrypt Decrypt Clear

PART V
Enter your answer here:

f4eb8ac5 774c1227 963e1109 40654d0e f0d08d 73e47acf 930bf273 1bffdefd Check Answer!

COUNTER MODE:

Virtual Labs
An AICTE Sankat of India Initiative

AES and Modes of Operation

PART I
Choose your mode of operation: Counter mode

PART II
Key size in bits: 128

4d600c38 35051023 6cd9e62a e3719dc3
65d05403 2c3eaf201 9303705d 32339325
bb9477fc 13105aee 4dc7c620 7490624e
6ec9aaf0 3d996a99 eadcf1d 1730a71b
69f57099 e5e79545 21279ef4 d0e00709

Plaintext: fc04a1b4 0764ab92 ef162ee6 4ba50bdb Next Plaintext Key: 404b1f5d 3acd011d af9b48c1 f46550b5 Next Keytext
CTR: fc04a1b4 0764ab92 ef162ee6 4ba50bdb Next CTR

PART III
Calculate XOR:

05a60032 7a615196 4746a499 49b25622
69f57099 e5e79545 21279ef4 d0e00709

Calculate XOR

XOR: 6c53f0ab 9f86c4d3 66613a6d 9252519b

PART IV
Key in hex: 404b1f5d 3acd011d af9b48c1 f46550b5
Plaintext in hex: fc04a1b4 0764ab92 ef162ee6 4ba50bdb
Ciphertext in hex: 05a60032 7a615196 4746a499 49b25622
Encrypt Decrypt Clear

PART V
Enter your answer here:

fe082c9d 022476ed b2bf1406 b4c67660 d44d7a4 a397c49d 510696ed 3bc319fc Check Answer!

Modes of operation (Cryptography Academy):

1.

Cryptography Academy Home Subjects Materials Contact Modes of Operation

HOME / MODES OF OPERATION / DEMO OF MODES OF OPERATION

Alice

Bob

Step 1/7

Enter the message (or integer) that Alice wants to send encrypted to Bob:

network security experiment 5

Use the following cryptosystem:

The Advanced Encryption Sta

With the following mode of operation:

Electronic codebook (ECB) m

Previous step Next step

Try again

2.

HOME / MODES OF OPERATION / DEMO OF MODES OF OPERATION

Alice

Bob

Step 2/7

Alice chooses the 128-bit key K for the message m (it's also possible to use a 192- or 256-bit key in AES).

She then sends the key K through a secure channel to Bob.

Parameters known by Alice:

$m = \text{network security experiment 5}$ $K = 11101010100111...$

Chooses the key K

$K = 11101010100111...$

Parameters known by Bob:

$K = 11101010100111...$

Receives the key K

Previous step Next step

Try again

HOME / MODES OF OPERATION / DEMO OF MODES OF OPERATION

Alice

Parameters known by Alice:
 $m = \text{network security experiment 5}$, $K = 11101010100111\dots$

$$\begin{aligned} m_1 = n &\Rightarrow 110 \Rightarrow b_1 = 01101110 \\ m_2 = e &\Rightarrow 101 \Rightarrow b_2 = 01100101 \\ m_3 = t &\Rightarrow 116 \Rightarrow b_3 = 01110100 \\ m_4 = w &\Rightarrow 119 \Rightarrow b_4 = 01110111 \\ m_5 = o &\Rightarrow 111 \Rightarrow b_5 = 01101111 \\ m_6 = r &\Rightarrow 114 \Rightarrow b_6 = 01110010 \\ m_7 = k &\Rightarrow 107 \Rightarrow b_7 = 01101011 \\ m_8 = &\Rightarrow 32 \Rightarrow b_8 = 00100000 \\ m_9 = s &\Rightarrow 115 \Rightarrow b_9 = 01110011 \end{aligned}$$

Step 3/7

Before Alice can encrypt the message m she first have to convert each letter into its corresponding ASCII value and then convert each ASCII value into its binary representation.

Previous step Next step Try again

Bob

Parameters known by Bob:
 $K = 11101010100111\dots$

3.

HOME / MODES OF OPERATION / DEMO OF MODES OF OPERATION

Alice

Parameters known by Alice:
 $m = \text{network security experiment 5}$, $K = 11101010100111\dots$, $p = 3$,
 $p_{16} = 00000011$, $x = [01101110011001\dots]$

$$p = 29 \bmod 16 = 3 \Rightarrow p_{16} = 00000011$$

$$\begin{aligned} x_1 &= \text{network security} \Rightarrow 0110111001100101\dots \\ x_2 &= \text{experiment 5333} \Rightarrow 0110010101110100\dots \end{aligned}$$

$$x = [x_1, x_2]$$

Step 4/7

AES encrypts blocks of 16 bytes (1 byte is 8 bits so 16 bytes is 128 bits) which corresponds to 16 ASCII characters, because each ASCII character is 1 byte.

The message m contains 29 characters (including whitespace) so we need $p = 29 \bmod 16 = 3$ bytes to fill up the last block x_2 such that it's 16 bytes (128 bits). This operation is called padding and it's therefore

Previous step Next step Try again

Bob

Parameters known by Bob:
 $K = 11101010100111\dots$

4.

HOME / MODES OF OPERATION / DEMO OF MODES OF OPERATION

Alice

Parameters known by Alice:

$m = \text{network security experiment 5}$ $K = 11101010100111\dots$ $p = 3$
 $p_{16} = 00000011$ $x = [01101110011001\dots]$ $y = [11101011100011\dots]$

Encrypts the plaintext blocks x :

$y_i = E_K(x_i)$

$y_1 = E_K(x_1) = 11101011100011\dots$
 $y_2 = E_K(x_2) = 01100101011101\dots$

$y = [y_1, y_2]$

Step 5/7

Alice uses the key K and the AES encryption function E_K to encrypt the blocks x . She then sends the ciphertext blocks y to Bob.

Previous step

Next step

Try again

Bob

Parameters known by Bob:

$K = 11101010100111\dots$ $y = [11101011100011\dots]$

Receives the ciphertext blocks y

5.

HOME / MODES OF OPERATION / DEMO OF MODES OF OPERATION

Alice

Parameters known by Alice:

$m = \text{network security experiment 5}$ $K = 11101010100111\dots$ $p = 3$
 $p_{16} = 00000011$ $x = [01101110011001\dots]$ $y = [11101011100011\dots]$

Step 6/7

Bob uses the key K and the AES decryption function D_K to decrypt the blocks y .

Previous step

Next step

Try again

Bob

Parameters known by Bob:

$K = 11101010100111\dots$ $y = [11101011100011\dots]$
 $x = [01101110011001\dots]$

Decrypts the ciphertext blocks y :

$x_i = D_K(y_i)$

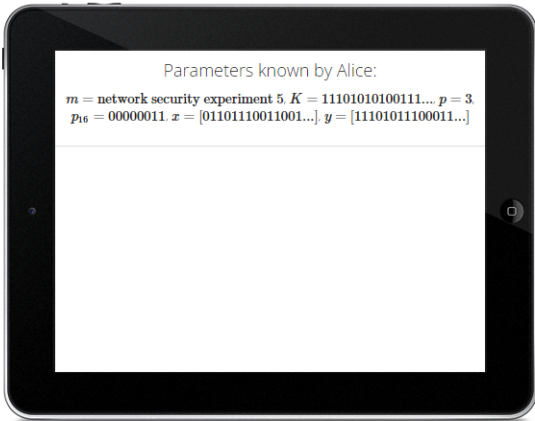
$x_1 = D_K(y_1) = 01101110011001\dots$
 $x_2 = D_K(y_2) = 00100000011001\dots$

$x = [x_1, x_2]$

6.

HOME / MODES OF OPERATION / DEMO OF MODES OF OPERATION

Alice



Step 7/7

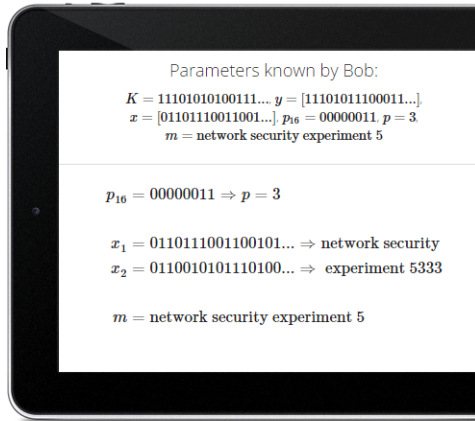
Bob converts the first byte $b_1 = 01101110$ in the first block x_1 to its integer representation **110** and then to its letter representation **n**.

Then he converts the rest of the bytes in the blocks x .

Because the decimal value of the last byte $p_{16} = 00000011$ in the last block x_2 is $p = 3$ and between 1 and 15, Bob know that the padding byte p_{16} has been added 3 times at the end of the last block x_2 .

[Previous step](#)
[Next step](#)
[Try again](#)

Bob



7.

CONCLUSION:

We have successfully understood the different Encipherment modes of operation using AES and also implemented all of them using virtual labs and obtained the output.