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

NAME: Sarvesh Patil
CLASS: D15A
ROLL NO: 52

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.

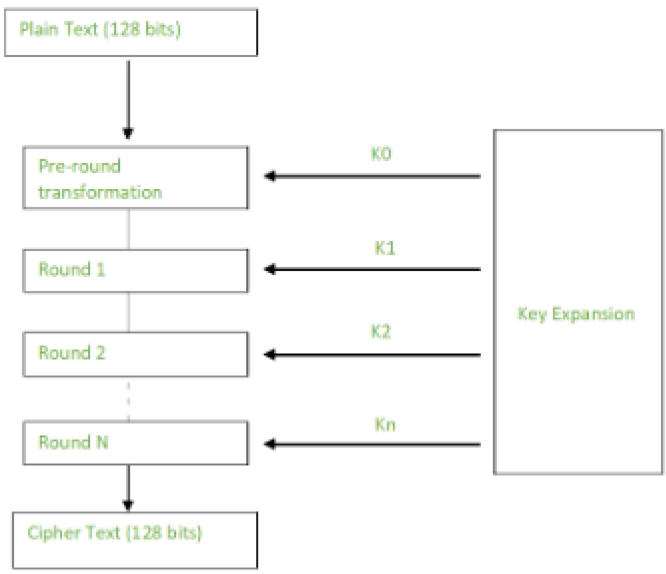
Points to remember:

- 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.



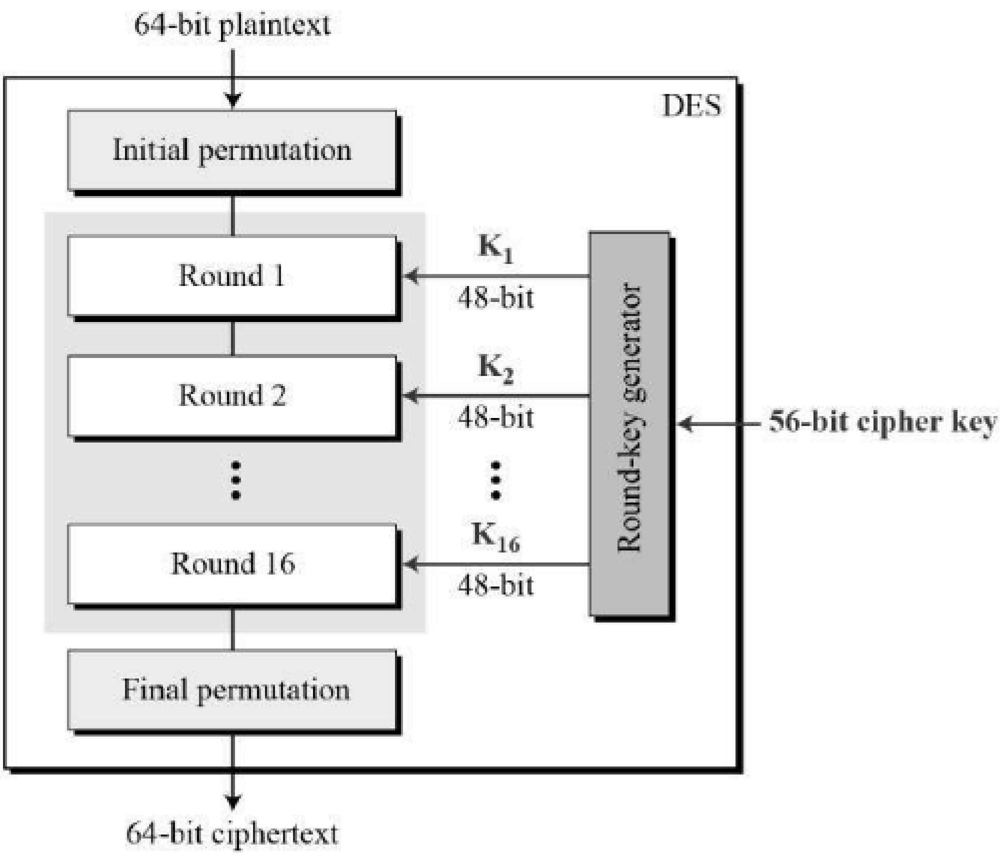
NAME: Sarvesh Patil
CLASS: D15A
ROLL NO: 52

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

<u>METHODS</u>:

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:

$$Ci = EK(Bi \oplus Ci\text{-}1)$$

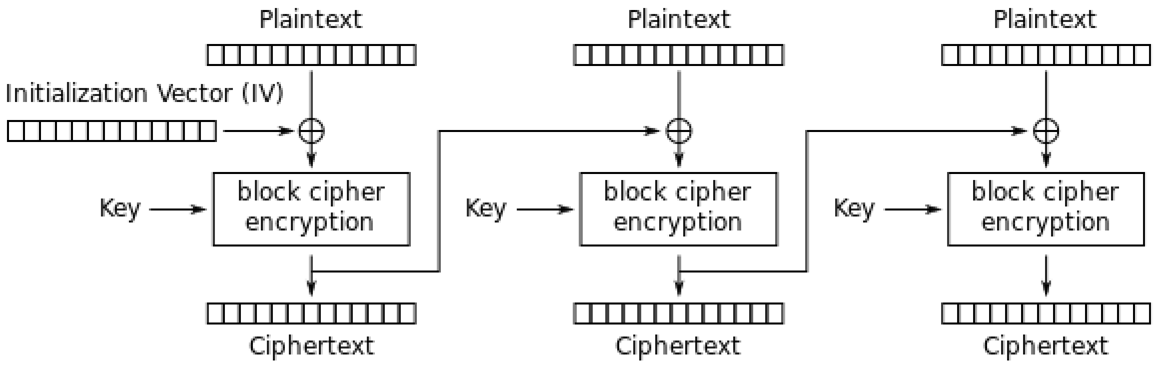
where EK denotes the block encryption algorithm using key K, and Ci-1 is the cipher corresponding to Bi-1.

Similarly, decryption using the CBC can be done using:

$$Bi = DK(Ci) \oplus (Ci\text{-}1)$$

where DK denotes the block decryption algorithm using key K.

NAME: Sarvesh Patil
CLASS: D15A
ROLL NO: 52



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

$$Cj = Pj \otimes E(K, [Cj - i \otimes Pj - 1])$$

By rearranging terms, we can demonstrate that decryption works.

$$Pj = Cj \otimes E(K, [Cj - 1 \otimes Pj - 1])$$

One advantage of the OFB method is that bit errors in transmission do not propagate. For example, if a bit error occurs in C1, only the recovered value of P1 is affected; subsequent plaintext units are not corrupted. With CFB, C1 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 cipher-text 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.

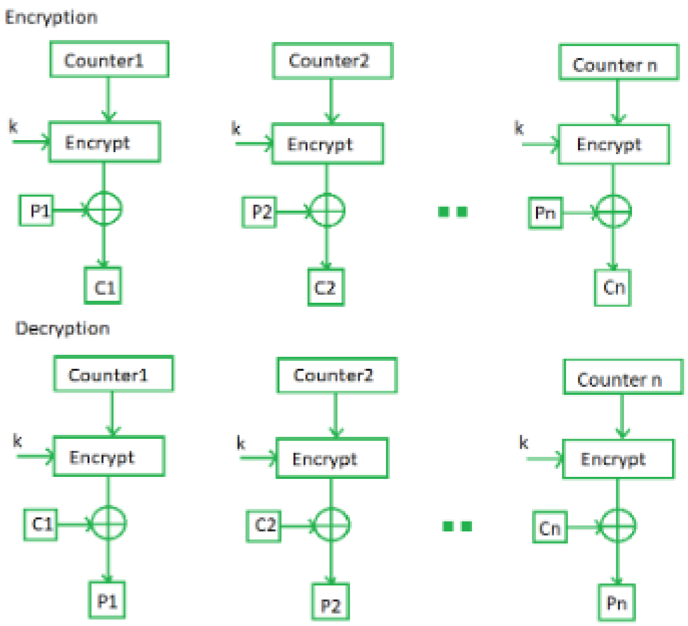
NAME: Sarvesh Patil
CLASS: D15A
ROLL NO: 52

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 –

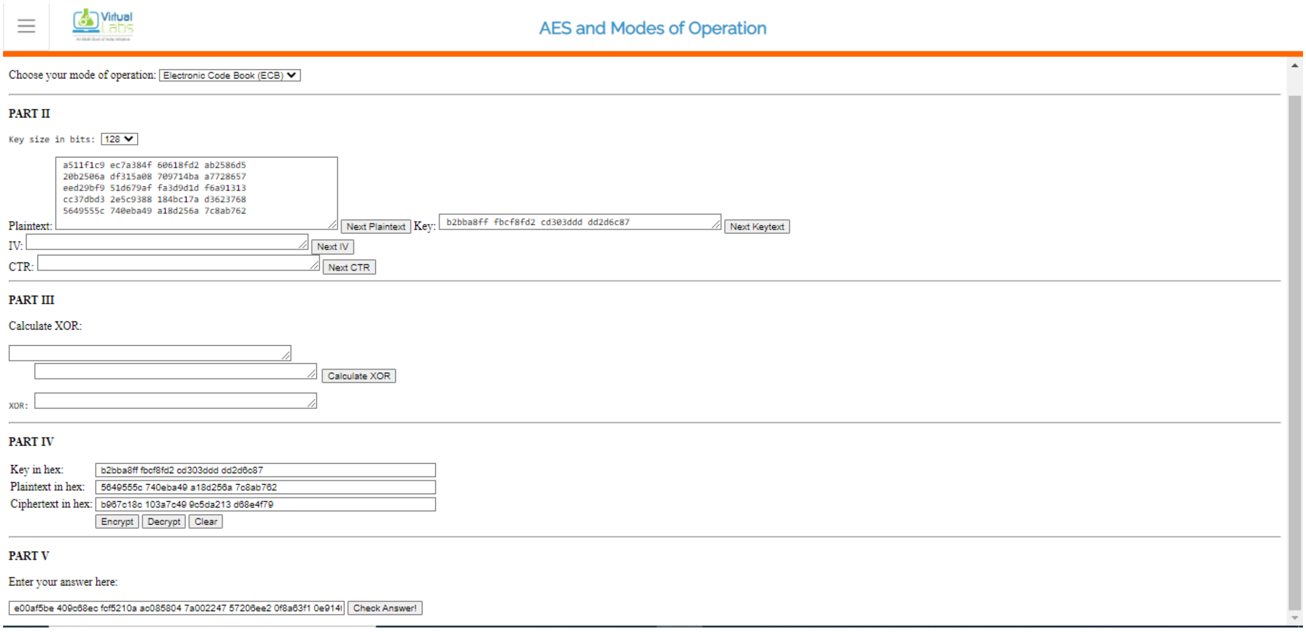
- Load the initial counter value in the top register is the same for both the sender and the receiver. It plays the same role as the IV in CFB (and CBC) mode.
- Encrypt the contents of the counter with the key and place the result in the bottom register.
- Take the first plaintext block P1 and XOR this 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.



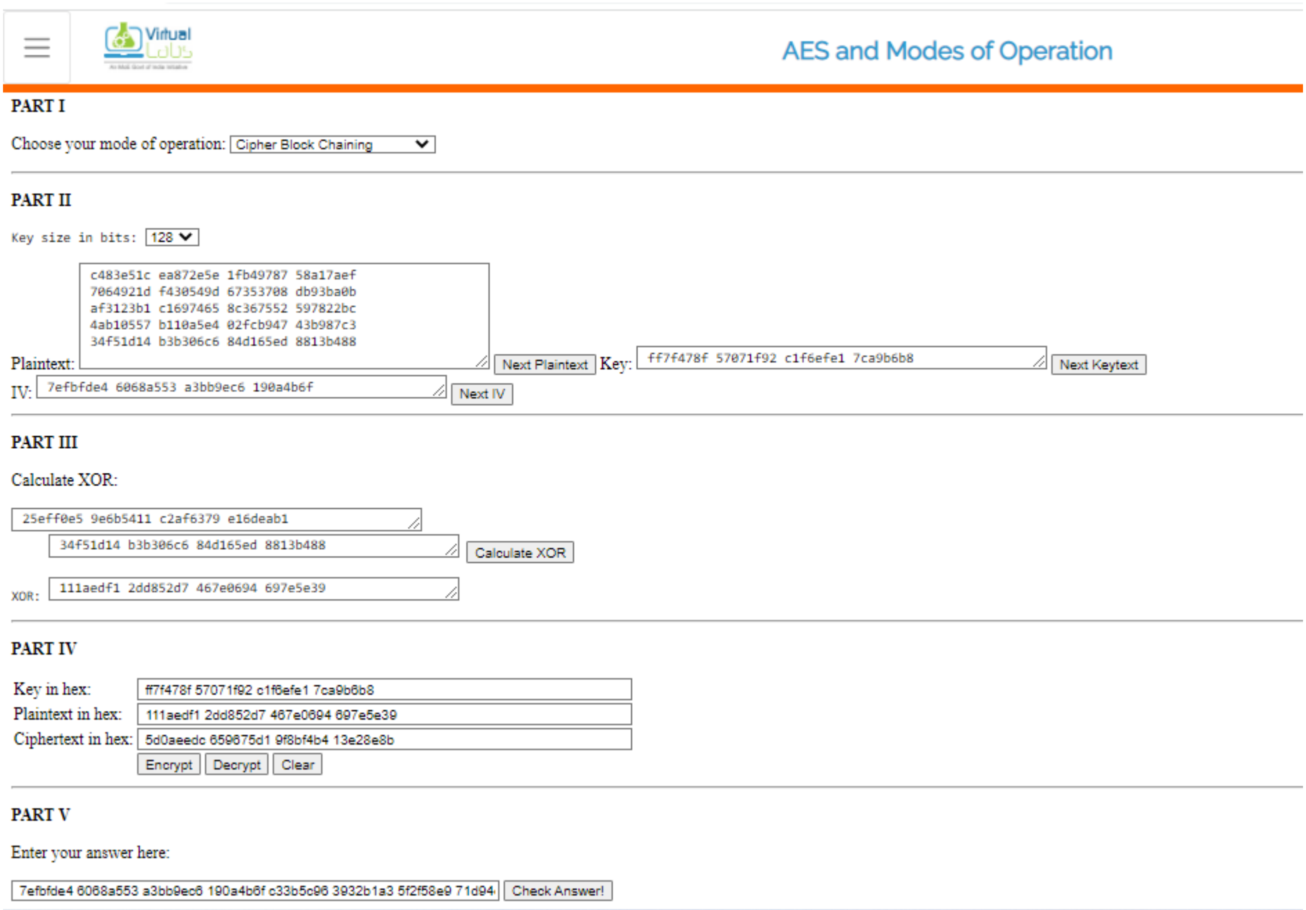
NAME: Sarvesh Patil
CLASS: D15A
ROLL NO: 52

RESULTS:

ECB:

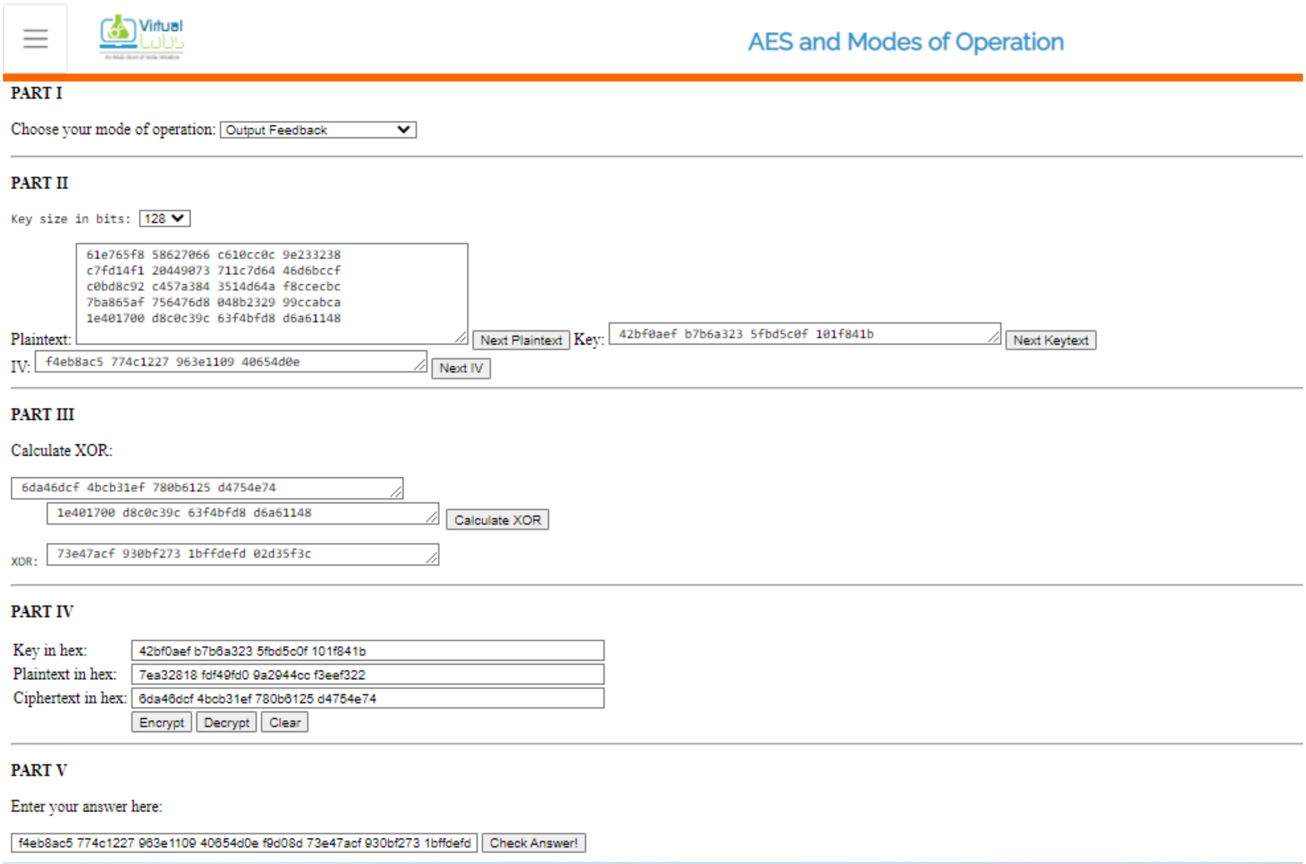


CIPHER BLOCK CHAINING:

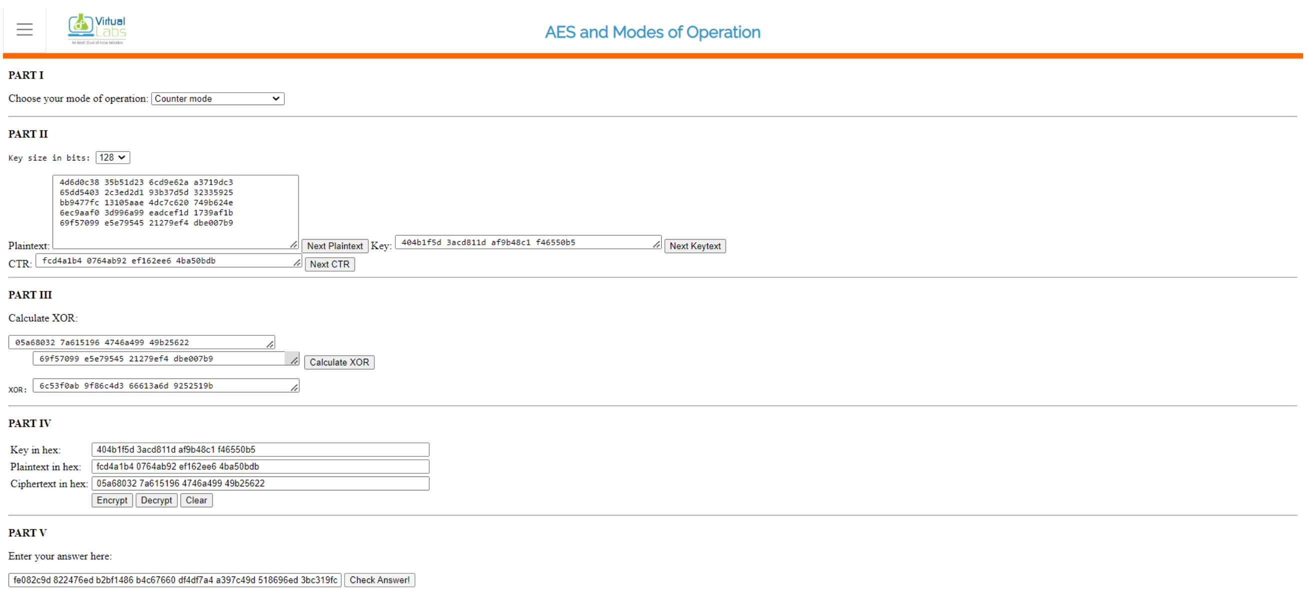


NAME: Sarvesh Patil
CLASS: D15A
ROLL NO: 52

OUTPUT FEEDBACK:

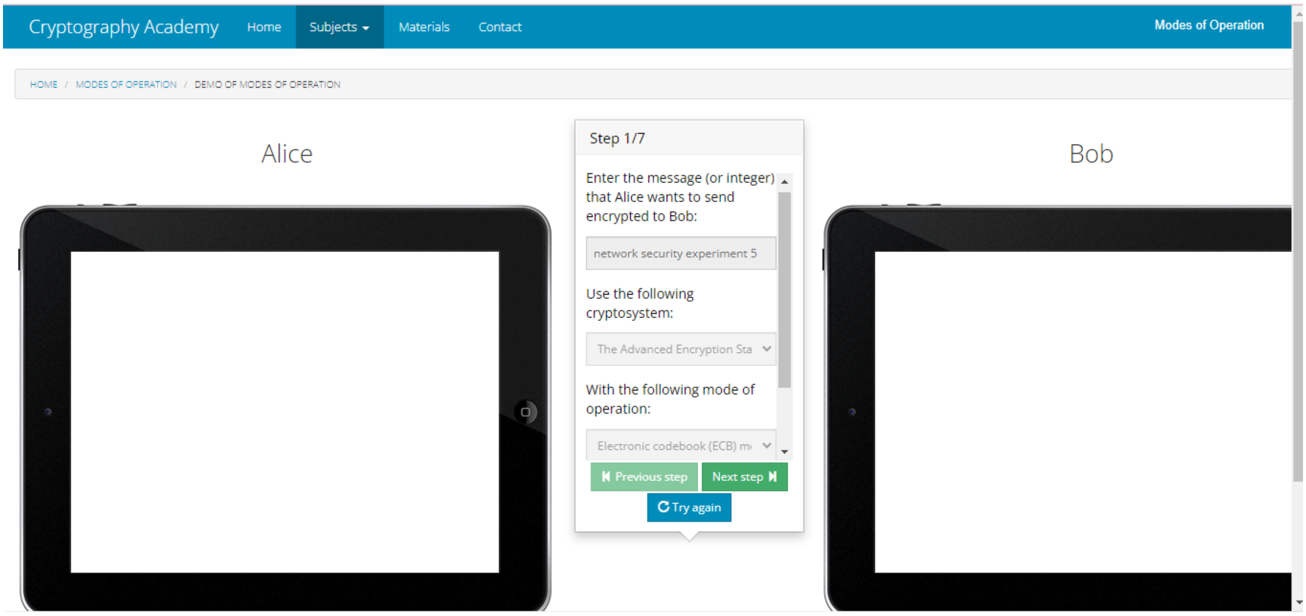


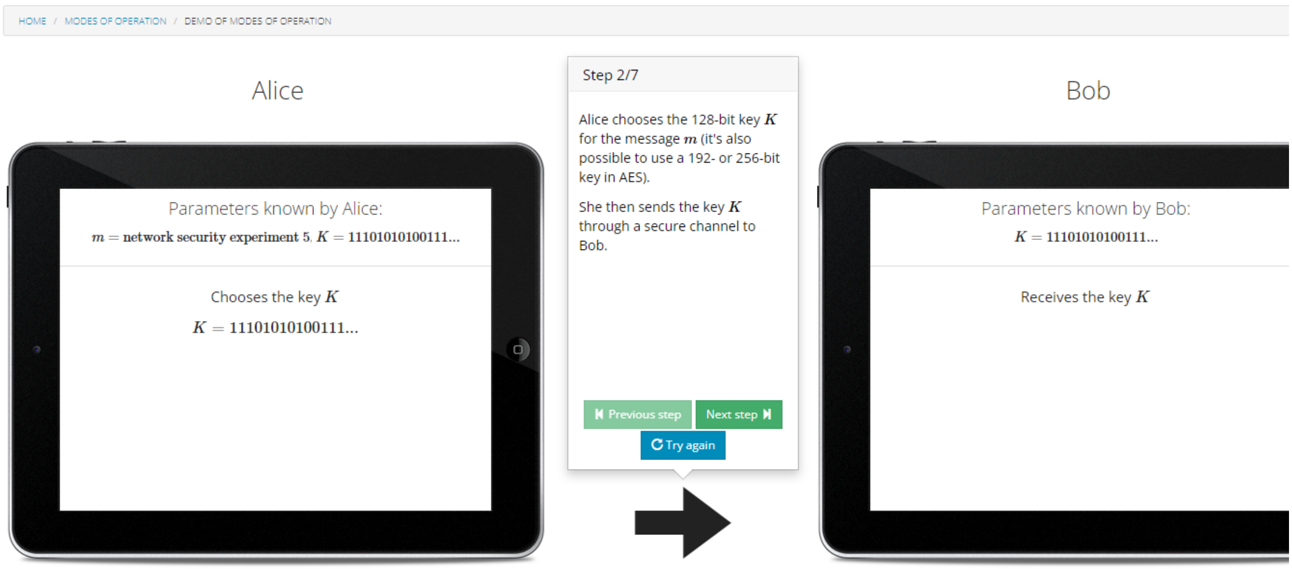
COUNTER MODE:



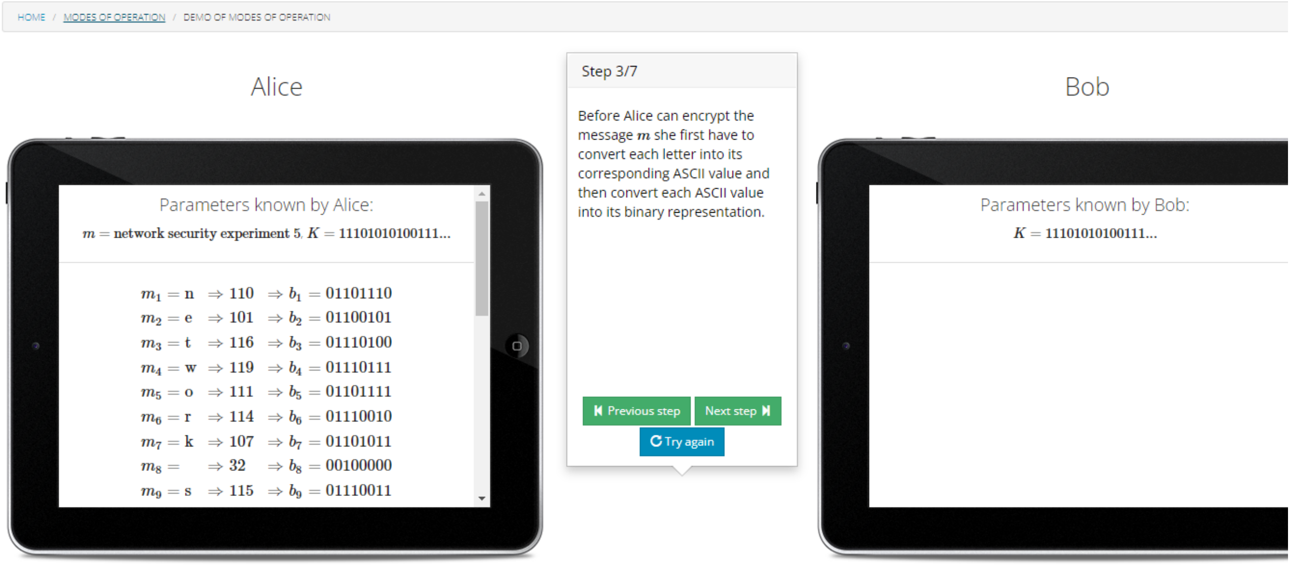
NAME: Sarvesh Patil
CLASS: D15A
ROLL NO: 52

Modes of operation (Cryptography Academy):

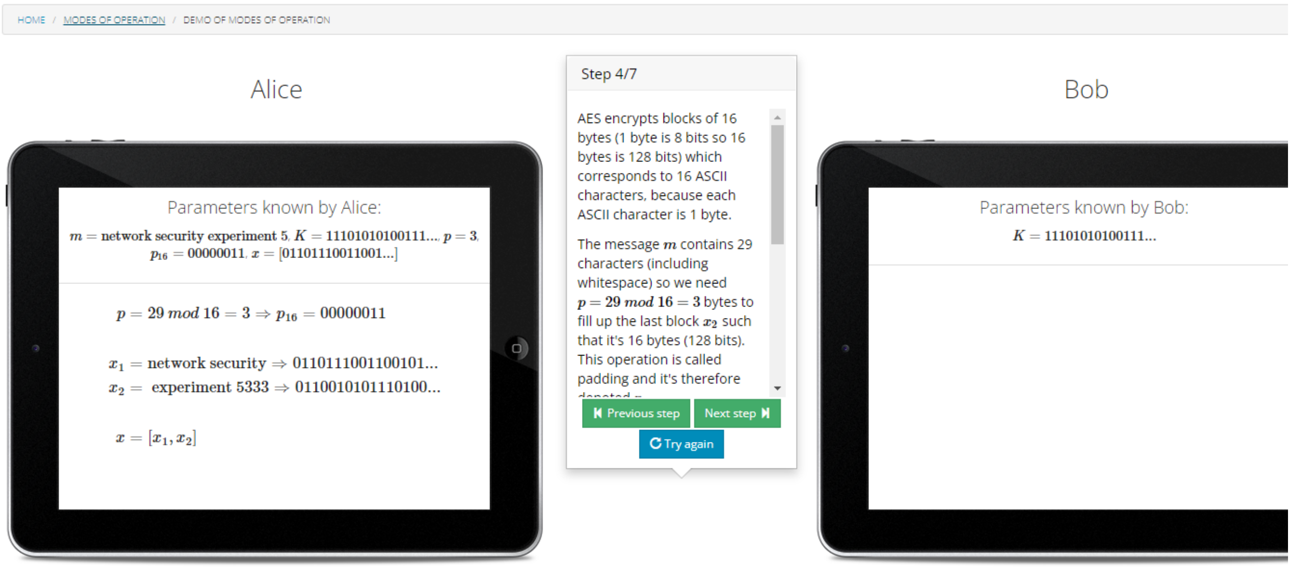
1. 

2. 

NAME: Sarvesh Patil
CLASS: D15A
ROLL NO: 52



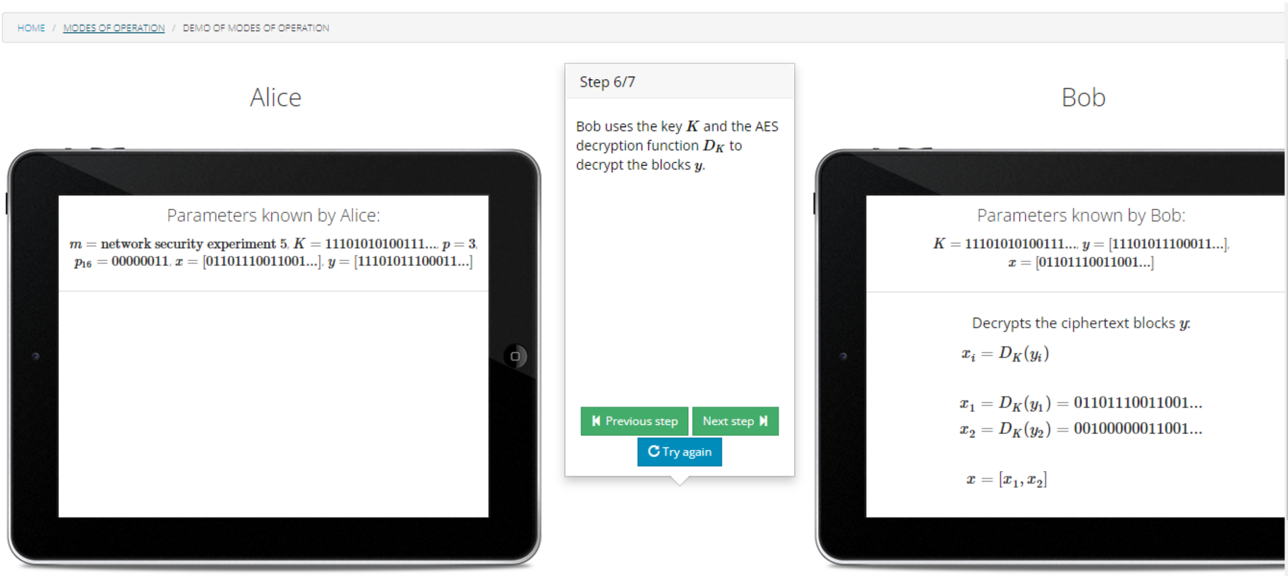
3.



4.

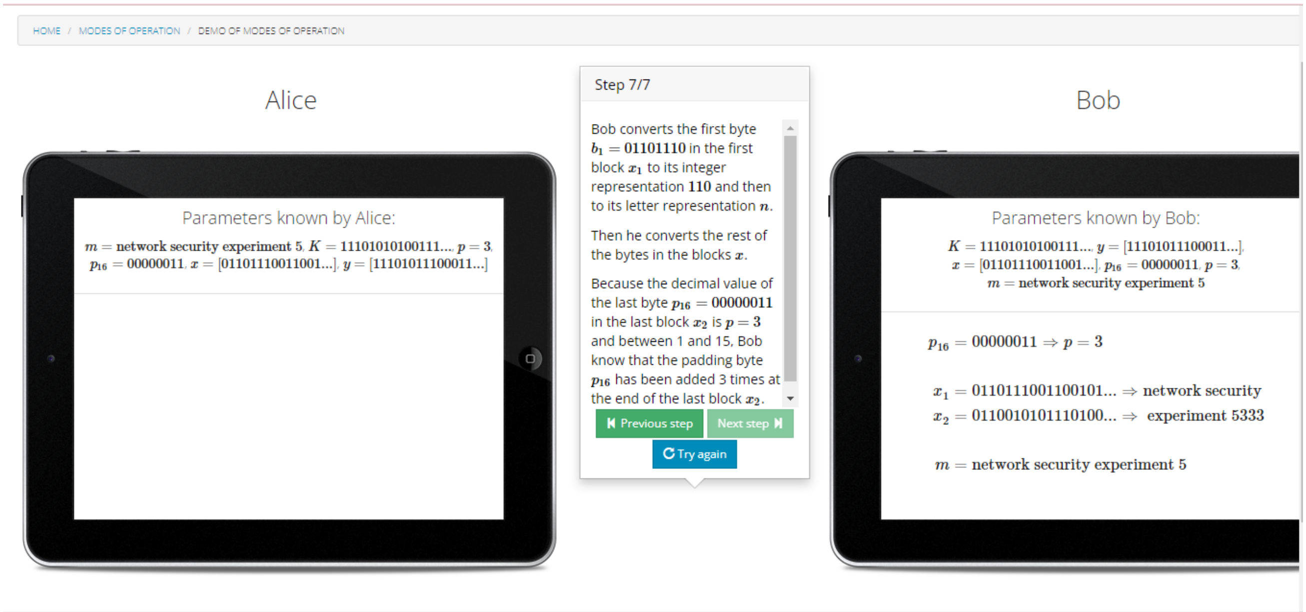
NAME: Sarvesh Patil
CLASS: D15A
ROLL NO: 52

5.



6.

NAME: Sarvesh Patil
CLASS: D15A
ROLL NO: 52

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.