Assignment No. 01

Page No.
Date

Aim:- Write a Java/C/C++/python program that contains a string (char pointer) with a value 'Hello World'. The program should AND or and XOR each character in this string with 127 and display the result.

Hardware/Software Requirements:

1. Desktop PC - Ram of 512 MB
2. Java or python or C

Theory:-

a) String :-

The string is the one dimensional array of characters terminated by a null ('\0'). Each and every character in the array consumes one type of memory and the last character must always be 0. The termination character ('\0') is used to identify where the string ends. In C language string declaration can be done in two ways.

1. By char array
2. By string literal

Eg. Declaring string by char array in C language

```
char ch[17] = {'H', 'e', 'l', 'l', 'o', ' ', 'W', 'o', 'r', 'l', 'd',
               '!', 'W', 'e', 'l', 'c', 'o', 'm', 'e', '\0'};
```

We can also define the string by the string literal in C language.

char str[] = "onlinesmartrainer"; '\0' will be appended by compiler.

B) AND Operation :

There are two inputs and one output in binary AND operation. The inputs and result to a binary AND operation can only be 0 or 1. The binary AND operation will always produce a 1 output if both inputs are 1 and will produce a 0 output if both inputs are 0 for two different inputs, the output will be 0.

AND Truth Table

Input		Output
X	Y	
0	0	0
0	1	0
1	0	0
1	1	1

c) XOR operation :

There are two inputs and one output in binary XOR (exclusive OR) Operation. It is similar to ADD operation which takes two inputs and produces one result i.e. One output. The input and result to a binary XOR operation can be 0 or 1. The binary XOR operation will always produce a 1 output if either of its inputs is 1 and will produce a 0 output if both of its inputs are 0 or 1

Page No.
Date

XOR Truth Table

Input		Output
X	Y	
0	0	0
0	1	1
1	0	1
1	1	0

Conclusion:

Thus we successfully implemented the string with 127 in AND, OR, XOR operations

Assignment No. 02

Page No.
Date

Aim:- To write a java/c/c++/python program to implement DES algorithm.

Hardware / Software Requirements:

1. Desktop PC - Ram of 512 Mb
2. Java / python / C / c++

Theory :-

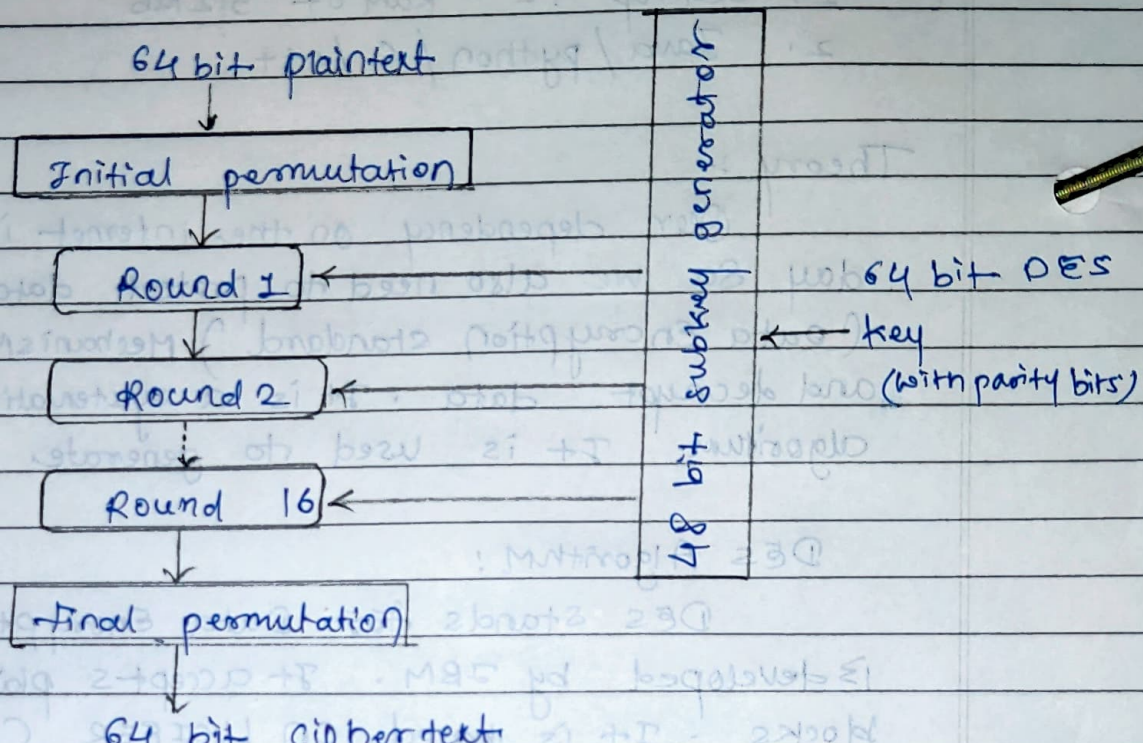
Our dependency on the internet is increasing day by day so we also need to protect data. Using DES (Data Encryption standard) Mechanism we can encrypt and decrypt data. It is a systematic symmetric algorithm. It is used to generate ciphertext.

DES Algorithm:

DES stands for Data Encryption standard. It is developed by IBM. It accepts plaintext in 64-bit blocks. It is based on LUCIFER (also known as Fiestal block cipher algorithm) which is a direct predessor of the DES algorithm. It provides high security using 128-bit key block and 128 bit block size. It uses 16-rounds of fielstal structure. The structure uses unique key for each round. Its block size is 64-bits and key sizes are 168, 112 and 56-bits respectively for keys 1, 2, & 3. It also uses the DES equivalent rounds i.e. 48. It means 16 rounds for each key.

It encrypt the data using the first key (k1) decrypts the data by using the second key (k2) and again encrypts the data by using third key (k3)

Another variant of the algorithm uses only two keys k1 & k3, where both the keys k1 & k3 are the same. It is used still but considered as a legacy algorithm.



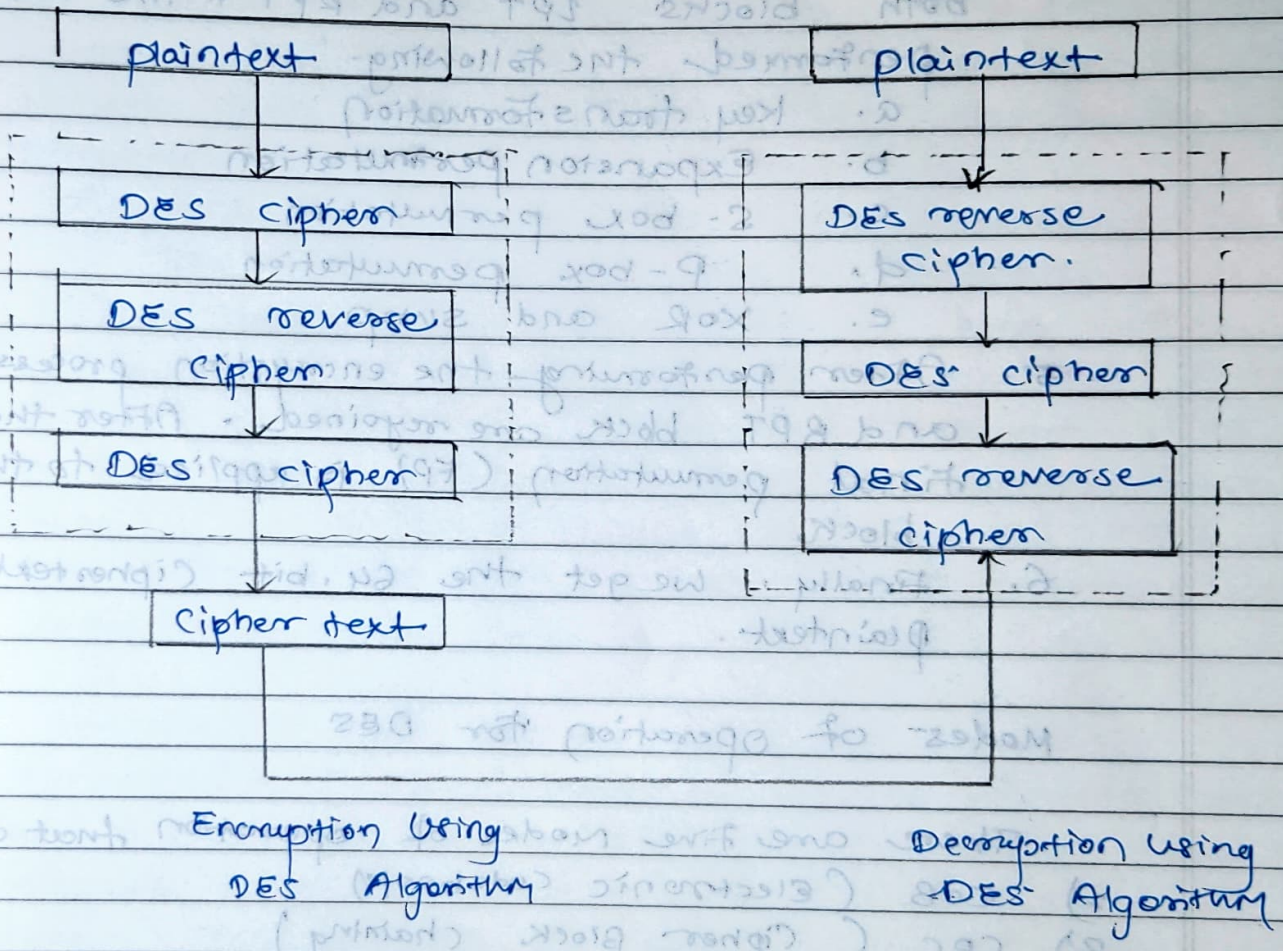
Generating keys:

The algorithm performs 16 rounds of encryption and for each round, a unique key is generated. Before moving to the steps, it is important to know that in plaintext the bits are labeled from 1 to 64 where 1 is the most significant bit. The process of generating keys are as follows.

1. First of we compress the and transpose the given 64-bit key into a 48-bit keys using the table given below.
2. Separate the result into two equal parts i.e. C and

D

3. The part C & D are left shifted circularly for encryption the 1st, 2nd, 9th and 16th round is responsible for that shifts a bit to the left by 1 bit, circularly. All the rest rounds are shifted to the left by 2-bit circularly.
4. After that, the result is compressed to 48-bits with the help of the following table.
5. The result that we get from step 3 becomes the input for the next round of the key generation.



Page No.
Date

Encryption steps of the Algorithm

1. The algorithm takes the 64-bit plain text as input
2. The text is passed into a function called the initial permutation (IP) function.
3. The IP function breaks the plain text into the two halves of the permuted block. These two blocks are known as left plain text (LPT) and Right plain text (RPT)
4. The 16 round encryption process is performed on both blocks LPT and RPT. The encryption process performed the following.
   a. Key transformation
   b. Expansion permutation
   c. S-box permutation
   d. P-box permutation
   e. XOR and swap
5. After performing the encryption process, the LPT and RPT block are rejoined. After that the final permutation (FP) is applied to the combined block
6. Finally, we get the 64-bit ciphertext of the plaintext.

Modes of operation for DES

There are five modes of operation that can be choosen

1) ECB (Electronic codebook)
2) CBC (Cipher Block chaining)
3) CFB (cipher feedback)

Page No.
Date

4) OFB (Output feedback)
5) CTR (Counter)

Conclusion:

We conclude that we have successfully completed and implemented the DES algorithm.

Assignment No. 03

Page No.
Date

Aim: Write a java/c/c++/python program to implement AES Algorithm.

Hardware / Software Requirement:

1. Desktop PC - Ram of 512 mb
2. Java or Python or C

Theory:

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.

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

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

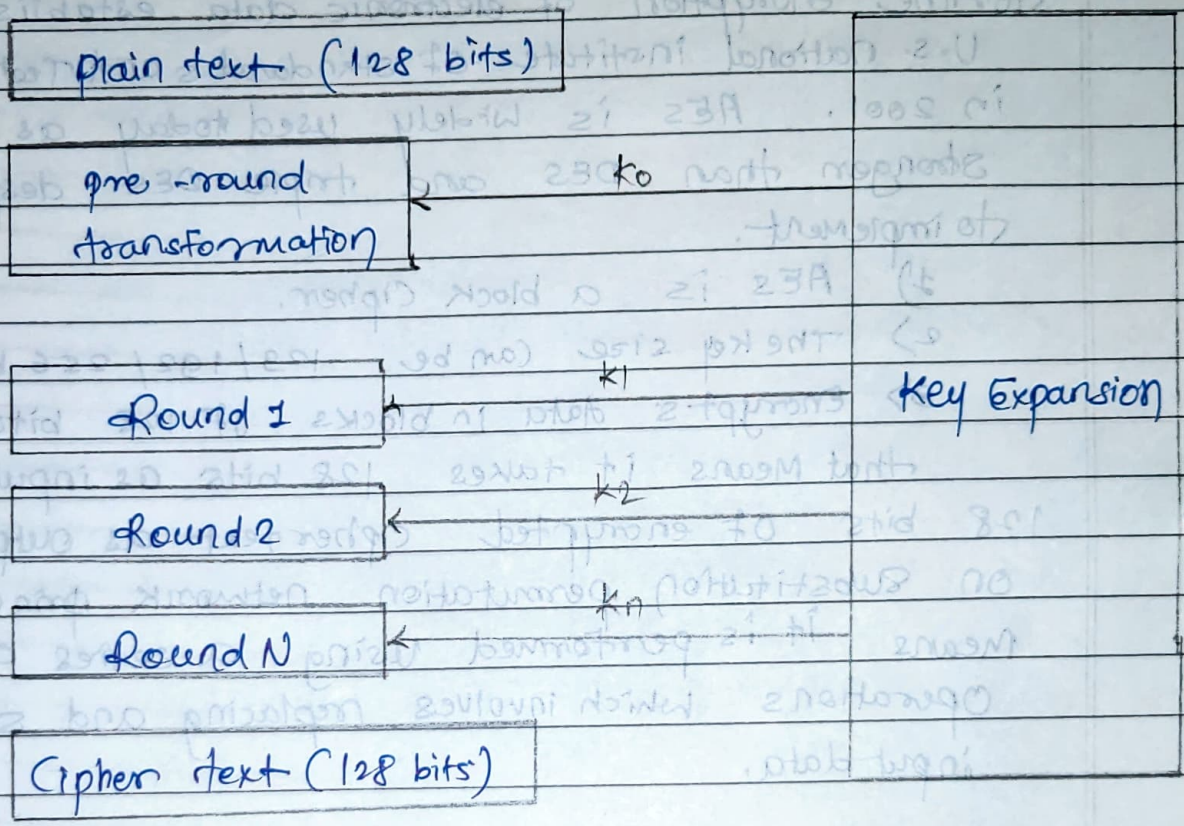
Working of the Cipher:

AES performs operations on bytes of data rather than in bits. Since the block size is 128 bits, the cipher processes 128 bits (or 16 bytes) of the input data at a time. The number of rounds depends on the key length as follows:

128 bit key – 10 rounds
192 bit key – 12 rounds
256 bit key – 14 rounds

Creation of Round keys:-

A key schedule algorithm is used to calculate all the round keys from the key. So the initial key is used to create many different round keys which will be used in the corresponding round of the encryption.



Encryption:-

AES conciders each block as a 16 byte (4 byte x 4 byte = 128) grid in a column major arrangement.

Each round comprises of 4 steps:

1) Subbytes
2) Shift rows

3) Mix Columns
4) Add Round Key

1. Sub-Bytes :- In this step each byte is substituted by another byte. Its performed using a lookup table also called the S-box. The result of this step is a 16 byte (4×4) matrix like before.

2. Shift rows :-
The first row is not shifted.
The second row is shifted once to the left.
The third row is shifted twice to the left.
The fourth row is shifted thrice to the left.

3. Mix Columns :- This steps is basically a matrix multiplication. Each column is multiplied with a specific matrix and thus the position of each byte in the column is changed as a result.

4. Add round keys :- Now the resultant output of the previous stage is XOR-ed with the corresponding round key. Here, the 16 bytes is not considered as a grid but just as 128 bits of data. After these rounds 128 bytes of encrypted data is given back as output. This process is repeated until all the data to be encrypted undergoes this process.

Decryption :-
The stages in the rounds can be easily undone as these stages have an opposite to it which when

performed reverts the changes. Each 128 blocks goes through the 10, 12 or 14 rounds depending on the key size.

4 stages:

1. Add round key
2. Inverse mix columns
3. Shift rows
4. Inverse sbab bytes

2. Inverse mix columns :- This steps is similar to the mix columns step in ecryption but differes in the matrix used to carry out the operation.

4. Inverse Sub bytes :- Inverse s-box is used as a lookup table and using which the bytes are substituted during decryption.

Conclusion :-

Hence we have sucessfully implemented AES Algorithm.

Assignment No. 04

Page No.
Date

Aim :- Write a java / c / c++ / python program to implement RSA algorithm.

Hardware / Software Requirements :

1. Desktop PC - Ram of 512 mb.
2. Java or Python or C

Theory :-

RSA Encryption Algorithm :- It is a algorithm is a type of public-key encryption algorithm. Public key encryption algorithm is also called the Assymmetric algorithm. RSA is the most common public-key algorithm, named after its inventors RNest, Shamit, and Addman (RSA)

RSA algorithm procedure to generate public and private keys :

1) Select two large prime numbers, p and q
2) multiply these numbers to find $n = p \times q$, where n is called the modulus for encryption and decryption.
3) Choose a number e less than n, such that n is relatively prime to $(p-1) \times (q-1)$. It means that e and $(p-1) \times (q-1)$ have no common factor except 1. Choose "e". such that $1 < e < \phi(n)$, ϕ is prime to $\phi(n)$.

   $gcd(e, d(n)) = 1$

4) If $n = p \times q$, then the public key is $<e, n>$. A plaintext Message m is encrypted using public key $<e, n>$. To find ciphertext from the plaintext following formula is used to get Ciphertext C

$c = m^e \bmod n$

Here, m must be less than n. A larger message (n) is treated as a concatenation of messages, each of which is encrypted separately.

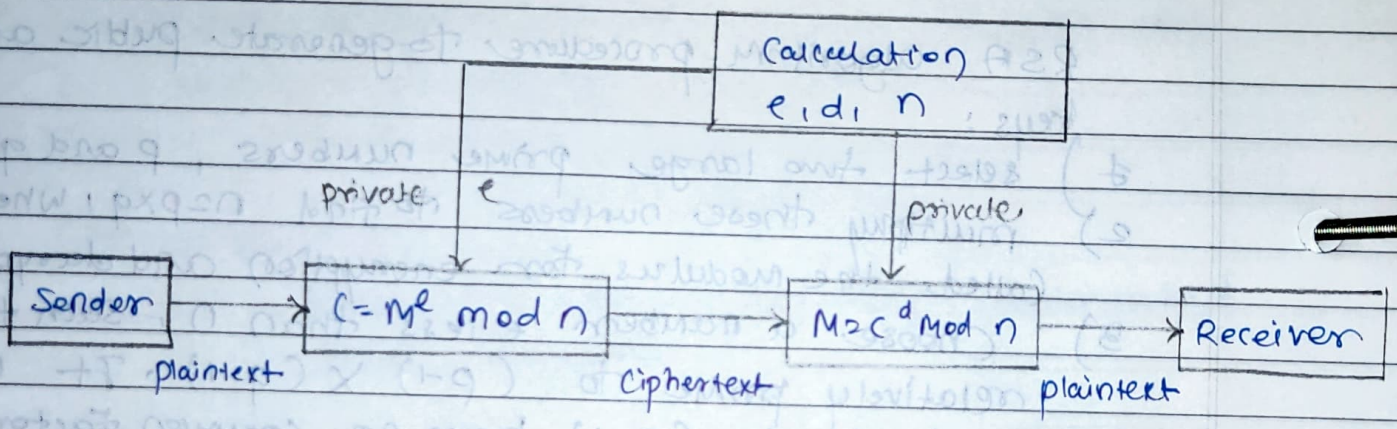
5) To determine the private key, we use the following formula to calculate the d such that:

$$De \bmod \{(p-1) \times (q-1)\} = 1$$

or $$De \bmod \varphi(n) = 1$$

6) The private key is $< d, n >$. A ciphertext message C is decrypted using private key $< d, n >$. To calculate plain text m from the ciphertext C following formula is used to get plain text m.

$$m = c^d \bmod n$$



Conclusion:

Hence we have successfully implemented RSA Algorithm.

Assignment No. 05

Page No.
Date

Aim:- Implement the Diffie-Hellman key Exchange mechanism using HTML and Javascript. Consider the end user as one of the parties (Alice) and the Javascript application as other party (bob)

Hardware/Software Requirements:-

1. Desktop PC - Ram of 512 mb
2. Java / Javascript / HTML

Theory :-

Background :- Elliptic curve cryptography (ECC) is an approach to public-key cryptography based on the algebraic structure of elliptic curves over finite fields. Ecc require a smaller key as compared to non-Ecc cryptography to provide equivalent Security (a 256-bit ECC security has equivalent security attained by 3072-bit RSA cryptography.

Deffie-Hellman algorithm:

The deffie-Hellman algorithm is being used to establish a shared secret that can be used for secret communication while exchanging data over a public network using the elliptic curve to generate points and get the secret key using the parameters.

- For the sake of simplicity and practical implementation of the algorithm. We will consider only 4 variables. One prime p and G (a primitive root of P) and two private values a & b.
- P and G are both publicly available numbers. Users (Say Alice & bob) pick private values a & b and

they generate a key and exchange it publicly. The opposite person receives the key and that generates a secret key, after which they have the same secret key to encrypt.

Explanation:

Alice	Bob
public keys available = P, G	Public keys available = P, G.
Private key selected = a	Private key selected = b
key generated = $x = G^a \bmod p$	key generated = $y = G^b \bmod p$

Exchange of generated keys takes place	
key received = y	key received = x
Generated secret key = $k_a = y^a \bmod p$	Generated secret key = $k_b = x^b \bmod p$
Algebraically it can be shown that $k_a = k_b$	
Users now have symmetric secret key to encrypt	

Example :-

Step 1 :- Alice & Bob get public numbers P = 23, G = 9

Step 2 :- Alice selected a private key a = 4 & Bob selected a private key b = 3

Page No.
Date

Step 3:- Alice & Bob compute public values
Alice : $X = ((g^4) \bmod 23) = (6561 \bmod 23) =$
Bob : $Y = ((g^3) \bmod 23) = (729 \bmod 23) = 16$

Step 4 : Alice and Bob exchanger public numbers

Step 5 : Alice receives public key $Y = 16$ and
Bob receives public key $x = 6$

Step 6: Alice and Bob compute symmetric keys
Alice : $Ka = y^a \bmod p = 65536 \bmod 23 = 9$
Bob : $Kb = x^b \bmod p = 216 \bmod 23 = 9$

Step 7 : 9 is the shared secret.

Conclusion:
Hence we have successfully implemented the Diffie-Hellman key exchange mechanism using HTML and javascript.