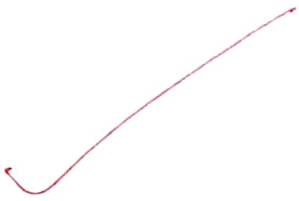
Practical No.-1

Aim: Write a Program to implement caesar cipher

Objectives:

1. To implement one of the encryption technique.
2. To use one of the Substitution technique.
3. To implement classical cipher



Practical No. - 1

Aim :- To demonstrate classical cipher.

Inlab-aim :- Write a Program to implement Caesar cipher.

Objectives :-

1. To implement one of the encryption technique.
2. To use one of the substitution technique.
3. To implement classical cipher.

Theory :-

In cryptography, a Caesar Cipher, also known as Caesar's cipher, the shift cipher, Caesar's Codeor Caesar shift, is one of the simplest & most widely known encryption techniques. It is a type of substitution cipher in which each letter in the plaintext is replaced by a letter some fixed number of positions down the alphabet. For example, with a left shift of 3, D would be replaced by A, E would become B, & so on.

Code :-

```
#include <stdio.h>
int main () {
    int k;
    char msg[100], ch;
    Printf("\n Enter message:");
      gets(msg);
    Printf("\n Enter key :");
     Scanf("%d", &K);
          int i=0;
```

Conclusion :-

Hence, we have successfully implemented Caesar cipher using classical cipher.

```
for(i=0; msg[i]!='\0'; i++){
    ch=msg[i];
  if(ch>='a' && ch<='z'){
    msg[i]=ch+k;
     if(ch>'z')
      msg[i]=ch-26;
    }
} printf("The encrypted message:");
   puts(msg);
    for(i=0; msg[i]!='\0'; i++){
      ch=msg[i];
     if(ch>='a' && ch<='z'){
       msg[i]=ch-k;
       if(ch<'a')
        msg[i]=ch+26;
     }
  }printf("the decrypted message:");
      puts(msg);
      return 0;
}
```

Output :- Enter Message: attack is tomorrow
Enter Key : 3
The Encrypted Message: dwwdfn lv wrpruurz
The decrypted message: attack is tommarrow

Conclusion :-
Hence, we have successfully implemented Caesar cipher using classical cipher.

Practical No. -2

Aim :- Write a Program to implement Playfair cipher.

Objectives :-

1. To implement one of the encryption technique
2. To use one of the substitution technique.
3. To implement playfair cipher

Practical No.-2

Page No. 4
DATE / /20

Aim :- To demostrate Substitution ciphers.

Inlab Aim :- Write a Program to implement Playfair cipher.

Objectives :-

1. To implement one of the encryption technique.
2. To use one of the Substitution technique.
3. To implement playfair cipher.

Theory :-

Playfair cipher is one of the Substitution cipher technique. After that the blank cell is filled with letters A-Z which are not already there in the matrix. It works as follows:

1. Observe the given plaintext if it contains any consecutive repeated letters than insert a filler character 'x' in between the repeated letters.
2. observe the given plaintext if it contains odd numbers of letters after step-1 then add one filler letter 'x' at the end.
3. Make pair of characters to form a diagrams from given Plaintext.
4. Encrypt the plaintext diagrams as follows:
   a. If a diagram contains characters in same column then replace the character by character adjacent right cell of matrix.
   b. If a diagram contains characters in same column then replace the characters by the character available in the adjacent cell below.

Code :-

```
#include<stdio.h>
int main(){
char arr[5][5] = {"MONAR","CHYBD","EFGIK",
                 "LPQST","UVWXZ"};
char pt[10];
int i,j,r1=0,r2=0;c1=0,c2=0;
Printf("playfair keymatrix \n'==\n");
for(i=0;i<5;i++){
    for(j=0;j<5;j++)
        Printf("%c",arr[i][j]);
        Printf("\n");
}Printf("Enter your Plaintext:");
Scanf("%s",pt);
Printf("Your plaintext =%s",pt);
for(i=0;i<5;i++){
    for(j=0;j<5;j++){
        if(arr[i][j]==pt[0]){
            r1=i; c1=j;
    }
    if(arr[i][j]==pt[i]){
            r2=i; c2=j;
}}}if(r1==r2){
    if(r2==4)
Printf("ciphertext = %c%c\n",
    arr[n][c1+1] arr[n][0]);
else
    Printf("ciphertext = %c%c\n",..
    arr[r2][c2+1], arr[n][c1+1]);
```

Conclusion :-

Hence, we successfully demonstrated Substitution cipher & implemented playfair cipher

```
} if (c1 == c2) {
      if (r2 == 4)
        printf ("ciphertext = %c %c \n",
          arr[r1+1][c1], arr[r2][c2]);
  else
   printf ("cipher text = %c %c \n", arr[r1+1][c1],
                                arr[r2+1][c2]);
{
 if (r1 != r2 && c1 != c2) {
    Printf ("\n ciphertext = %c %c \n", arr[r1][c2], arr[r2][c1]);
 }
 return 0;
}
```

Output :-

```
M O N A R
C H Y B D
E F G I K
L P Q S T
U V W X Z
Enter your plain text: IN
    Your Plain text = IN
```

Conclusion :-

Hence, we successfully demonstrated Substitution cipher & implemented playfair cipher

Practical No. - 3

Page No. 7
DATE / /20

Aim :- To demonstrate Transposition cipher.

Inlab- Aim :- Write a program to implement Railfence cipher.

Objectives :-

1. To implement one of the encryption technique.
2. To use Transposition technique
3. To implement Railfence cipher.

Theory :-

- In the rail fence cipher, the plain-text is written downwards & diagonally on successive rails of an imaginary fence.
- When we reach the bottom rail, we traverse upwards moving diagonally, after reaching the top rail, the direction is changed again. Thus the alphabets of the message are written in a zig-zag manner.
- After each alphabet has been written, the individual rows are combined to obtain the cipher-text.

Code :-

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
int main() {
int i, j, len, rails, count, code[100][1000];
char str[1000];
Printf("Enter Secret Message \n");
gets(str);
```

ACE

```
len = strlen(str);
Printf("Enter no. of rails \n");
Scanf("%d", &rails);
for( i=0; i<rails; i++){
   for(j=0; j<len; j++){
        code[i][j]=0;
    }
} count=0;
   j=0;
  while (j<len){
   if (count %2 ==0)
     for (i=0; i<rails; j++){
       code[i][j] = (int) str[i];
        j++;
     }
} else {
     for(i= rails -2; i>0; j++){
       code[i][j] = (int)str[j];
         j++;
   } }
   count++;
{ for (i=0; i<rails; i++) {
    for(j=0; j<len; j++){
      if (code[i][j] != 0)
        Printf("%c", code[i][j]);
   } }
    Printf("\n");
  }
   return 0;
}
```

Output :- Enter Secret Message
Hello World
Enter No. of rails
2
HORELOLLWD

Conclusion :-
Hence, we successfully demonstrated transposition cipher & implement Rail fence cipher

24/01/20

Practical No. - 4

Aim :- To demonstrate block cipher technique

Inlab Aim :- Write a Program to implement AES

Objectives :-

1. To implement block cipher.
2. To implement AES

Theory :-

AES stands for Advance encryption algorithm designed for encrypting a fixed size block of data. AES was designed by Rijmen-Daemen in Belgium & it uses 128/192/256 bit keys, 128 bit data. It Processes data as block of 4 columns of 4 bytes operates on entire data block in every round. It designed to have resistance against known attacks, speed & code compactness on many CPUs, design simplicity. The data block of 4 columns of 4 bytes. is called state.

- byte Substitution.
- shift rows
- Mix columns
- add round key
- view as alternating XOR key & Scramble data bytes.

Code :-

```
# include <stdio.h>
#include <stdlib.h>
# include <string.h>
# include <mcrypt.h>
```

```
#include<stdint.h>
#includ<stdlib.h>
int encrypt(
    void * buffer,
    int buffer_len,
    char * IV,
    char * key,
    int key_len
){
MCRYPT td = mcrypt_module_open("rijndael-128",
    NULL, "cbc", NULL);
 int blocksize = mcrypt_enc_get_block_size(td);
 if (buffer_len % blocksize != 0) { return 1; }
 mcrypt_generic_init(td, key, key_len, IV);
 mcrypt_generic (td, buffer, buffer_len);
 mcrypt_generic_deinit(td);
 mcrypt_module_close(td);
return 0;
}
int decrypt(
        void* buffer,
        int buffer_len,
        char* IV,
        char * key,
        int key_len
){
    MCRYPT td = mcrypt_module_open("rijndael-128", NULL,
```

```
"cbc", NULL);
int blocksize = mcrypt_enc_get_block_size (td);
if (buffer_len % blocksize != 0) { return 1; }
mcrypt_generic_init (td, key, key_len, IV);
mdecrypt_generic (td, key, key_len, IV);
mcrypt_generic_deinit (td);
mcrypt_module_close(td);
return 0;
}
void display (char* ciphertext, int len) {
    int v;
  for (v=0; v<len; v++) {
     Printf("%d", ciphertext[v]);
  } printf ("\n");
}
int main () {
  MCRYPT td, td2;
  char* plaintext = " test text 123";
  char * IV = "AAAAAAAAAAAAAAAA";
  char *key = "0123456789abcdef";
  int keysize = 16;
  char* buffer;
  int buffer_len = 16;
  buffer = calloc (1, buffer_len);
  strncpy (buffer, plaintext, buffer_len);
  printf ("==c==\n");
  Printf ("plain: %s\n", plaintext);
  encrypt (buffer, buffer_len, IV, key, keysize);
  Printf ("cipher: ");
```

```
    display (buffer, buffer_len);
decrypt( buffer , buffer_len ; IV , key , keysize);
    printf (" decrypt :  %s \n", buffer);
return 0;
}
```

Output :-

```
Plain : Test Text 123
Cipher : 16-124 41-33-16-123 61-64-15
Decrypt : Test Text 123
```

Conclusion :-

Hence, we successfully demonstrated block cipher by implementing AES algorithm.

Done 24/9/20

Practical No. - 5

Page No. 14
Date : / /

Aim :- To demonstrate Symmetric secret key cipher.
Inlab Aim :- Write a program to implement DES.

Objectives :-

1. To implement Symmetric Secret key cipher.
2. To implement DES
3. To use DES in real world examples.

Theory :-

The DES encryption algorithm takes a 64-bit block of Plaintext & a 64-bit key as input & Produces an 64-bit block of ciphertext as output. The DES decryption algorithm takes an 64-bit block of ciphertext & the same 64-bit key used to Produce that ciphertext as input & Produces the original 64-bit block of Plaintext. The encryption algorithm involves function : an initial Permutation (IP); a complex function labeled fk, which involves Expansions, S-Boxes, permutation, XOR & Switching 32 bits halves of the data; the function fk again, & finally a permutation function that is the inverse of the initial permutation (IP-1).

Code :-

```
#include <iostream>
#include <string.h>
#include <stdio.h>
#include <math.h>
using namespce std;
```

```
Void setB(int s; int in; int loc){
    int h1=0;
  h1=1<<B1,-(loc-1);
    if(in);
      i=i/h1;
    else
        i=i&-h1;
} int getB(int i, int loc){
     int Sum=0;
     int t=0;
     for(int i=e; i>=b; i--){
        Sum t=getB(in, i)<<t;
          t+=1;
   }
  return Sum;
} Void PKEY(int·& , int &r)
       int Sum=0;
       int t=0;
       for(int i=e; i>=b; i--){
         Sum t=getB(in, i)<<t;
    int map[56]={57, 49, 41, 33, 25, 17, 9, 1, 58,
    50, 42, 34, 26, 18, 10, 2, 59, 51, 43, 35, 27, 19, 11, 3, 60,
    52, 44, 36, 63, 55, 47, 39, 31, 23, 46, 38, 30, 22};
       int hl=l;
       int ·hr=r;
    for(int t=0; t<56; t++){
       if(map[t]>32){
         if(t>27){
      setB(hr, getB(&, map[t]-32), t-27);
```

```
}else{
SetB(h1, getB(r, map[t]-32, t++);
}}
}else{
    if(t>27){
     SetB(hr, getB(l, mp[t]); t-27);
  }else{
     SetB(h1. getB(l, map[t], t++);
  }
}}  r=hr;
    S = h1;
    r=r>>4;
    r = r<<4;
    r = r>>8;
    l = l<<8;
{void funct(int.&l, int &r){
   int map[48] = {32,1,2,3,4,5,5,6,7,8,9,8,9,10,11,12,
   13,12,13,12,13,14,15,16,17,16,17,18,19,20,21,20,
   21,22,23,24,24,25,26,27,28,29,30}
     int h1 = l;
     int h2 = l;
   for(int t=0; t<48; i++){
      if(t>23){
   SetB(hr, getB(r, map[t], t-23)
  }else{
      SetB(hl, getB(r.map[t], t+1);
  }
{  r=hr;
   l=hl;
```

```
r = r >> 8;
r = r << 8;
l = l >> 8;
l = l << 8;
} void finalP(int *l, int *r) {
int map[64] = {40, 8, 48, 16, 56, 24, 64, 32, 39, 7, 47,
15, 55, 23, 31, 38, 6, 46, 14, 54, 22, 62, 30, 37, 5, 45, 13,
53, 21, 61, 29, 4, 44, 12, 52, 20, 60, 18, 33, 3}
int hl = l;
int hr = r;
for (int t = 0; t < 64; t++) {
    if (map(t) > 32) {
        if (t > 31) {
    SetB(hr, getB(r, map[t] - 32, t - 3));
} else {
    SetB(hl, getB(r, map[t] - 32, t + 1));
}}
else{
    if (t > 31) {
    Set B(hr, get B(l, map[t], t - 31);
} else {
    Set B(hl, get B(r, map[t], t + 1));
}}}
    r = hr;
    l = hl;
} void Cipher(long, long int msg, long long int key) {
    int f = 0;
int key r = (int)(key);
int key l = (int)(key >> 32);
```

```
Count << "key left int :=" << key lo << " key right
    int :=" << keys << endl;
    int lo=0 do=0 ;
     key ( key l, key r);
         w = key l;
         do = key r;
Count << " msg left int :=" << l << "msg
 right int =" << ro << endl;
   P i ][ lo, ro);
for(int i=1; i<17;i++){
       int t=2;
if(i==11; i == 2 || i == 9 || i==16){
        t=1;
} K1 =lo;
  Kr =do;
PSUBKEY ( K1, Kr);
    int plo=do;
      lo =ro;
    ro = plo funct(ro, K1, Kr);
} int rl =0;
    rt = lo;
    ro = lo;
     lo = rt;
final P( lo, ro)
  long long int final = lo;
      final = final <<32;
       final t=ro;
} int main (){
long long int key[n[8] = {56, 18, 12, 16, 10, 107, 101,
                          121}
```

```
long long int msga[8] = {109, 101, 115, 115, 97, 103, 101, 48};
    long long int msg = 0;
    long long int key = 0;
  for (int i=0 ; i<8 ; i++){
        msg += (msg a[i] << ((7-i)*8));
}
  for (int i=0; i<8; i++){
      key += (key a[i] << ((7-i)*8));
}
   cout << "[your msg : " << msg << "]" << endl;
    cout << "[your key : " << key << "]" << endl;
       Cipher (msg, key);
  return 0;
}
```

output :-

```
[your msg : 78828336621745 20622]
[your key : 4062943354666310308 1]
   key left int : = 945977716
   key right int : = 1701537
   key left int := 7835365235
   msg right int := 1634166 0
```

Conclusion :-

Hence, we successfully implemented DES algorithm.

10/07/20

Practical No. - 6

Aim :- To demonstrate assymmetric public key cipher

Inlab Aim :- Write a program to implement RSA algorithm.

Objectives :-

- To implement assymmetric public key cryptography
- To implement RSA algorithm.

Theory :-

- It is an assymmetric cryptographic algorithm Asymmetric means that there are two different keys. This is also called public key cryptography, because one of them can be given to everyone. The other key must be kept private.

Code :-

```
#include <iostream.h>
#include <math.h>
using namespace std;
  int gcd (int a, int h){
    int temp;
    while (i) {
      temp = a % h;
      if (temp == 0)
        return h;
        a = h;
        h = temp;
  }
}
```

```
int main () {
  double p=3;
  double q= 4;
  double n = p*q;
   double count;
double totient = (p-1) *(q-1);
 double e=2;
 while( e < totient) {
   count=gcd (e, totient);
   if(count==1)
      break;
   else
      e++;
 } double d;
   double k=2;
   d= (1+(k * totient)) /e;
    double msg =12;
    double c= pow(msg, e);
    double m= pow(c, d);
        c= fmod(c, n);
        m= fmod(m, n);
cout<<"message data ="<<msg;
Cout<<"\n"<<"p="<<p;
Cout<<"\n"<<"q="<<q;
Cout<<"\n"<<"n= pq."<<n;
Cout<<"\n"<<"totient<<"tolient";
  Cout<<"\n"<<"e.z "<<e;
 Cout<<"n"<<" encrypted date= "<<c;
 Cout<<"\n "<<"original message sent = "<<m;
}
```

```
return 0;
}
```

Output :-

Message data = 12
p = 3, q = 7;
n = pq = 21
totient = 12
e = 5
d = 5
encrypted data = 3
Original message sent = 12

Conclusion :-

Hence, we successfully demonstrated asymmetric public key cipher & implemented RSA algorithm.

10/2/20

Practical No. - 7.

Aim :- To demonstrate Secure Key exchange.

In lab Aim :- Write a Program to implement Diffie - Hellman Key exchange.

Objectives :-

- To use Secure Key exchange
- To encrypt & decrypt Key Diffie-Hellman Key exchange.

Theory :-

Diffie-Hellman Key exchange, also called exponential key exchange, is a method of digital encryption that uses numbers raised to specific powers to produce decryption keys on the basis of components that are never directly transmitted, making the task of a would be code breakers mathematically overwhelming.

To implement Diffie-Hellman the two end users Alice & Bob, while communicating over a channel they know to be private, mutually agree on positive whole members - P & q, such that P is is the prime number & q is a generator of P. the generator q is a number that, when raised to positive whole-number powers less than P, never produces the same result for any two such whole numbers. The values of P may be large but the value of q is usually small.

Code :-

```
#include <stdio.h>
int compute (int a, int m, int n){
    int r;
    int y=1;
    while(m>0){
    r=m%2;
    if(r==1)
      y=(y*a)%n;
      a=a*a%n;
      m=m/2;
  } return y;
} int main(){
    int p=23, g=5, b, A, B, a=6;
    A=compute(g,a,p);
    b=15;
    B=compute(g,b,p);
    int key A= Compute(B,a,p);
    int key B= Compute(A,b,p);
Printf("A's Secret key is %d \n B's Secret key is
          %d", key A, key B);
  return 0;
}
```

Output :-

A's Secret key is 2.
B's Secret key is 4

Conclusion :-

Hence, we successfully demonstrated & implemented Diffie-Hellman key exchange algorithm

Practical No. - 8

Aim :- To demonstrate hash technique for Security

Inlab Aim :- Write a Progream to implement message digest.

Objectives :-

1. To use message digest.
2. To produce message digest for given data.
3. To implement any message digest algorithm.

Theory :-

Message digest is created by applying one of the message digest algorithm. The message digest can be calculated with the help of MD5, SHA, RIPEMD-10 etc algorithm. Message digest algorithm takes a block of data of variable size or fixed size & produces a fixed size message digest output. Any message digest algorithm does not uses key but in place of that it uses many constant available in registers.

Code :-

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <math.h>
typedef union uwb {
        unsigned w;
        unsigned char b[4];
```

```
} mus Union;
unsigned func'(unsigned abcd[]){
return(abcd[1] & abcd[2]) || (n abtd[i] & abcd[3]);
} unsigned funcl (unsigned 'abcd[]){
  return (abcd[3] & abcd[1]) || (aabcd[3] & abcd[2]);
} typedef unsigned (* Dgst funct) (unsigned a[]);
   unsigned* calculate (unsigned * K){
      double S, Ptr;
       int i;
       Pwr = Pow(2,32);
      for(i=0; i<64; i++){
         S=false (sin (i++));
       K[i]= unsigned (S * pwr);
   } return k;
}unsigned* mod s (const char *msg, int mlen){
   static Digest Array h;
     Digest Array abcd;
     Digest fetn fetn;
       short * rath;
     union {
        unsigned w[16];
        char b[64];
     }; m m;
       int as=0;
    int grp, geps, g, P;
   unsigned char p msg 2;
   for (gep=0; gep <geps ; grp++){
     mm cpy (mm, b, msg 2+ as :'64);
```

```
for (int q=0; q<16; q++){
    q = (m*q*0)%16;
    abcd[0] = abcd[3];
    abcd[3] = abcd[2];
    abcd[2] = abcd[2];
     abcd[1] = f;
    }
} for (p=0; p<4; p++)
      [p]t = abcd[p];
        os+=64;
  } return h;
} int main(){
    int j, k;
Const char *msg = message digest:";
Printf("your hashcode generated by md5 is:%d", md5);
return 0;
}
```

Output:-

Your hashcode generated by md5 is:
2 3 4 b g c 1 7 8 b 2 c g b e 4 e g g b l & l d
5 b c b b f f 6

Conclusion :-

Hence, we Successfully demonstrated hash technique for security & implemented message digest.

Done
28/12/20

Practical No. - 9

Aim :- To demostrate classical cryptography

In lab Aim :- write a Program to implement Hill Cipher

Objectives :-

1. To implement one of the encryption technique
2. To implement one of the classical Substitution cipher technique
3. To implement Hill cipher.

Theory :-

Invented by Lester S. Hill in 1929, the Hill cipher is a Substitution cipher based on linear algebra. Hill used matrices & matrix multiplication to mix up the plaintext. To encrypt a message, each block of n letters is multiplied by an invertible n x n matrix, against modulus 26. To decrypt the message, each block is multiplied by the inverse of the matrix used for encryption.

Code :-

```
# include <iostream>
# include <math.h>
using namespace std;
float encrypt[3][1], decrypt[3][1],
  a[3][3], b[3][3], mes[3][1], c[3][3];
void encryption();
```

```
Void decryption();
Void inverse();
int main(){
    getKeyMessage();
    Encryption();
    decryption();
}
Void encryption(){
  int i, j, k;
for(i=0; i<3; i++)
  for(j=0; j<1; j++){
    for(k=0; k<3; k++)
      encrypt[i][j] = encrypt[i][j] + a[i][k]
                    * mes[k][j];
Cout<<"\n encrypted string is: ";
  for(i=0; i<3; i++)
    cout<<(char)(fmod[encrypt[i][0], 26)+97);
}
Void decryption(){
    int i, j, k;
    inverse();
  for(i=0; i<3; i++)
    for(j=0; j<1; j++)
      for(k=0; k<3; k++)
        decrypt[i][j] = decrypt[i][j] + b[i][k]
                      * encrypt[k][j];
  Cout<<"\n decrypted string is: ";
    for(i=0; i<3; i++)
  cout<<(char)(fmod(decrypt[i][0], 26)+97);
```

```
cout<<"\n";
} void getkeymessage() {
    int i,j;
  char msg[3];
 cout<<" Enter 3x3 matrix for key :\n"}
    for(i=0; i<3 ; i++)
     for(j=0; j<3; j++)
        cin>>a[i][j];
        c[i][j] = a[i][j];
 } Void inverse () {
     int i,j, k;
     float p,q;
  for( i=0; i<3; i++)
   for(j=0; j<3; j++){
      if( i==j)
          b[i][j]=1;
      else
          b[i][j]=0;
  } for(k=0; k<3; k++){
     for(i=0; i<3; i++){
        p = c[i][k];
        q=c[k][k];
      for(j=0 ; j<3; j++ ){
          if(i!=k){
       c[i][j] =c[i][j] * q- p*c[k][j];
       b[i][j]= b[i][j] q - q* b[k][j];
 }}}
} for(i=0; i<3; i++)
      for(j=0; j<3 ; j++)
          b[i][j]=b[i][j]/c[i][i];
```

```
cout<<"\n\n Inverse matrix is:\n";
    for(i=0; i<3; i++){
      cout<<b[i][j]<<" ";
      cout<<"\n";
    }
```

output :- Enter 3x 3 matrix for key

6 24 1
13 16 10
20 17 15

Enter a 3 letter string : act
Encrypted string is : poh
Inverse matrix is :
0.15873 -0.777228
-0.011337 0.15873
-0.22449 0.057143
decrypted string is : act.

Conclusion :-

Hence, we successfully demonstrated classical cryptography & implemented Hill Cipher

06/03/20

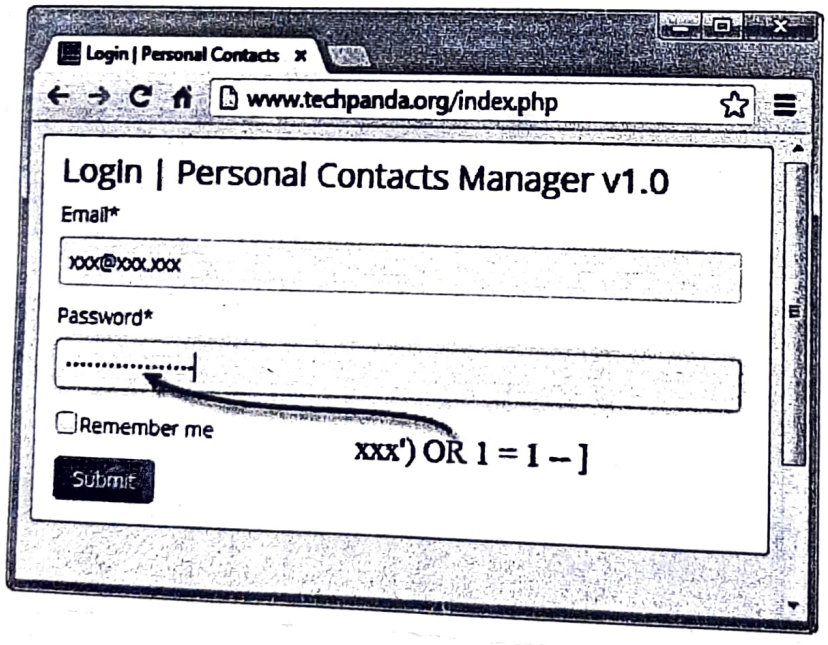
Practical No. -10

Aim :- Write a Program to implement SQL injection.

Objectives :-

- To implement software Vulnerability.
- To implement SQL injection.

O/P :-



Practical No. -10

Aim : To demonstrate Software Vulnerability

Inlab Aim :- Write a Program to implement SQL injection.

Objectives :-

1. To implement Software Vulnerability
2. To implement SQL injection.

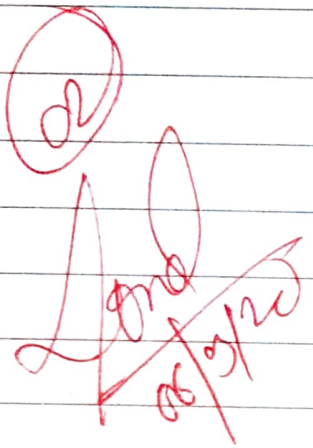
Theory :-

SQL injection is a technique of attacking on a web poretal to gain access to the Portal without knowing the credentials of the portal access. It uses the coding properties of SQL & finds a week point to gain access to the Portal without the knowledge of user id & user password. It also used to test the security of a website or database.

Code :-

```
UName = getRequestString ("Username");
UPass = getRequestString ("Userpassward");
  Sql = "SELECT * from User WHERE Name=" "
 Sql or=" = " AND Pass= " " or " = "
   Sql="SELECT * from User WHERE
     Name=" " + UName + " " "AND
     Pass = " " + UPass + " "
```

Conclusion :- Hence, we successfully demonstrated the Program for sql injection.



Practical No. - 11

Aim :- To demonstrate various encryption Techniques using GUI

Inlab Aim :- write a program to implement key agreement using Elliptic curve based Algorithm.

Objectives :-

- To understand the concept of Elliptic curve based algorithm.
- To demonstrate various encryption Techniques.

Theory :-

Elliptic-curve cryptography is an approach to Public-key cryptography based on the algebric structure of elliptic curves over finite fields. ECC requires smaller keys compared to non-EC cryptography to provide equivalent security. Elliptic curves are applicable for key agreement, digital signatures, Pseudo random generation & other tasks. elliptic curves that have application on cryptography.

Code :-

```
#include <stdio.h>
#include <math.h>
#define CT
typedef struct {double x, y } pt.
Pt zero (void) { return (Pt) {INFINITY,
                              INFINITY};};
int is_zero (Pt.p) { return p.x >= 1e20 || p.x <= -1e20;}
```

```
Pt neg(Pt p){return (Pt){-p.x,-p.y};}
db (Pt p){
    if (iszero(p))
        return p;
    Pt r;
double l=-(3.p.x.p.x)/(2.p.y);
    r.x = l.l-x.p.x;
    r.y = l.(p.x-r.x)-p.y;
    return r;
}
Pt add (Pt p, Pt q){
    if(p.x==q.x && p.y==q.y) return db(p);
  if(is_zero(p)) return q;
  if(is_zero(q)) return p;
    Pt r;
double l=(q.y-p.y)/(q.x-p.x);
    r.x = l*l-p.x-q.x;
    r.y = l*(p.x-r.x)-p.y;
    return r;
}
  Pt mul(Pt p, int n){
    int i;
   Pt r=zero();
 for(i=1; i<=n; i<<=1){
   if(i&n)  r=add(r,p);
      p=db(p);
   } return r;
} void show(const char *s, Pt p){
    Printf("%s", s);
Pt(is_zero(p)?"zero\n":"(%3f,%3f)\n", p.x, p.y);
```

```
Pt from_y(double y){
    Pt r;
    r.x = pow(y*y*y, 1.0/3);
    r.y = y;
    return r;
}

int main(void){
    Pt a, b, c, d;
    a = from_y(1);
    b = from_y(2);
    show("a = ", a);
    show("b = ", b);
    show("c = a+b = ", c = add(a,b));
    show("d = -c = ", d = neg(c));
    show("c+d = ", add(c,d));
    show("a+b+d = ", add(a, add(b,d)));
    show("a * 12345 = ", mul(a, 12345));
    return 0;
}
```

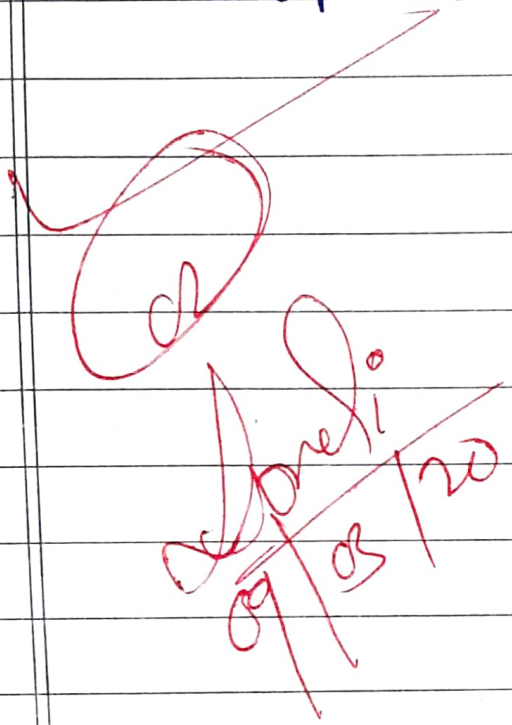
Output :-

a = (-1.817, 1.000)
b = (-1.442, 2.000)
c = a + b = (10.375, -33.525)
d = -c = (10.375, 33.525)
c+d = zero
a+b+d = zero
a * 12345 = (10.789, 33.387)

~~3 pt from y (double y) {~~

Conclusion:-

Thus, we have implemented Elliptical curve cryptography.



Practical No. - 12

Aim :- To demonstrate user authentication

In Lab Aim :- WAP to implement user authentication based on user id & password.

Theory :-

Authentication id the act of confirming the truth of attribute of a single price, data claimed true by on entry in central with identification which refues to the act of stating or otherwise indicating a claim Propated by at testing to a person or things identify authentication is the process of confirm that identity. It might involve verifying the authenticating of a website with digital cablicate data mining the age of an use' aulfact by carbon dating or ensuring that a product as what its Packaging & labeling claim to be in other words, authenticating after involves verifying the validity of atleast one of identification

Code :-

```
# include <stdio.h>
int main () {
    char Password [10], Username [10], ch;
    int i;
Printf (" Enter User Name");
gets ( username);
```

```
Printf("Enter the Password:");
for(i=0; i<8; i++){
        ch= getchar();
        password[i]=ch;
        ch='*';
        Printf("%c",ch);
}
Password[i]='\0';
Printf("\n Your password is:");
   for(i=0; i<8; i++){
        Printf("%c", password[i]);
   }
return 0;
}
```

Output :-

Enter User Name : Damini@123
Enter the password <any 8 character> :- *****
Your Passward is : damini@124

Conclusion :-

Hence, we successfully implemented user authentication based on user id & password.

29/9/20

Practical No. -13

Aim :- To implement wireshark Open Source Tool

Objectives :-

- To implement wireshark open source Tool.
- To know the use of wireshark tool.

Theory :-

Wireshark is a free & open-source packet analysis it is used for network trouble shooting, analysis, s/w & communicating Protocol.

Wireshark intercepts traffic & converts that binary traffic into human-readable format. This makes it easy to identify what traffic is crossing your network, how much of it, how frequently, how much latency there is between certain hops, so forth.

While Wireshark supports more than two thousand network protocols, many of them esteric uncommon, or old, the mordern security professional will find analyzing IP packets to be of most immediate usefulness. The majority of the packets on your network are likely to be TCP, UDP, & ICMP.

Given the large volume of traffic are what make it easily useful.

Teacher's Signature _____

Name of Practical

Features :-

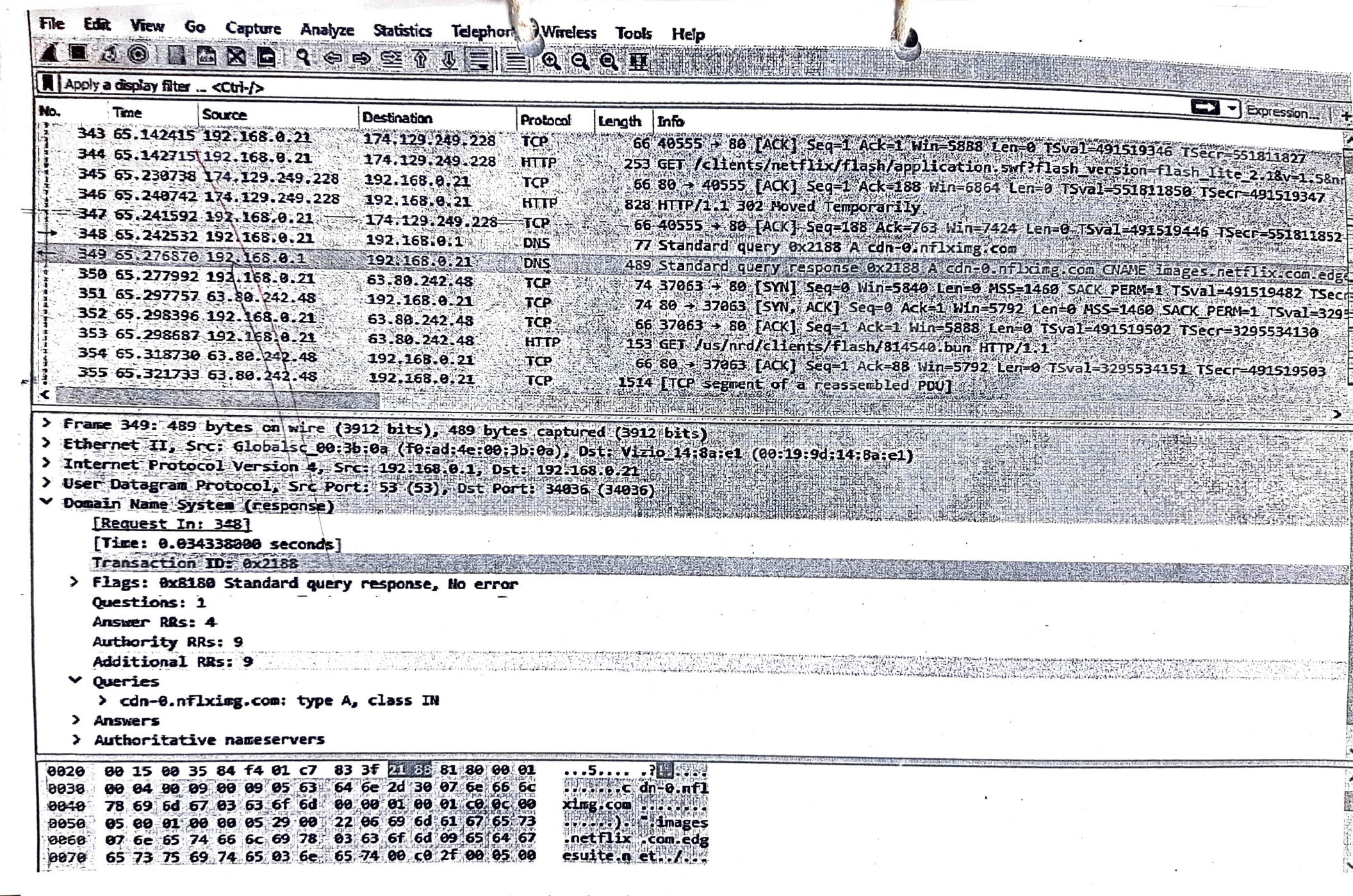
Wireshark is a data capturing program that understands the structure of different networking protocols. Wireshark uses pcap to capture packets. So it can only capture packets on the types of network that pcap supports.

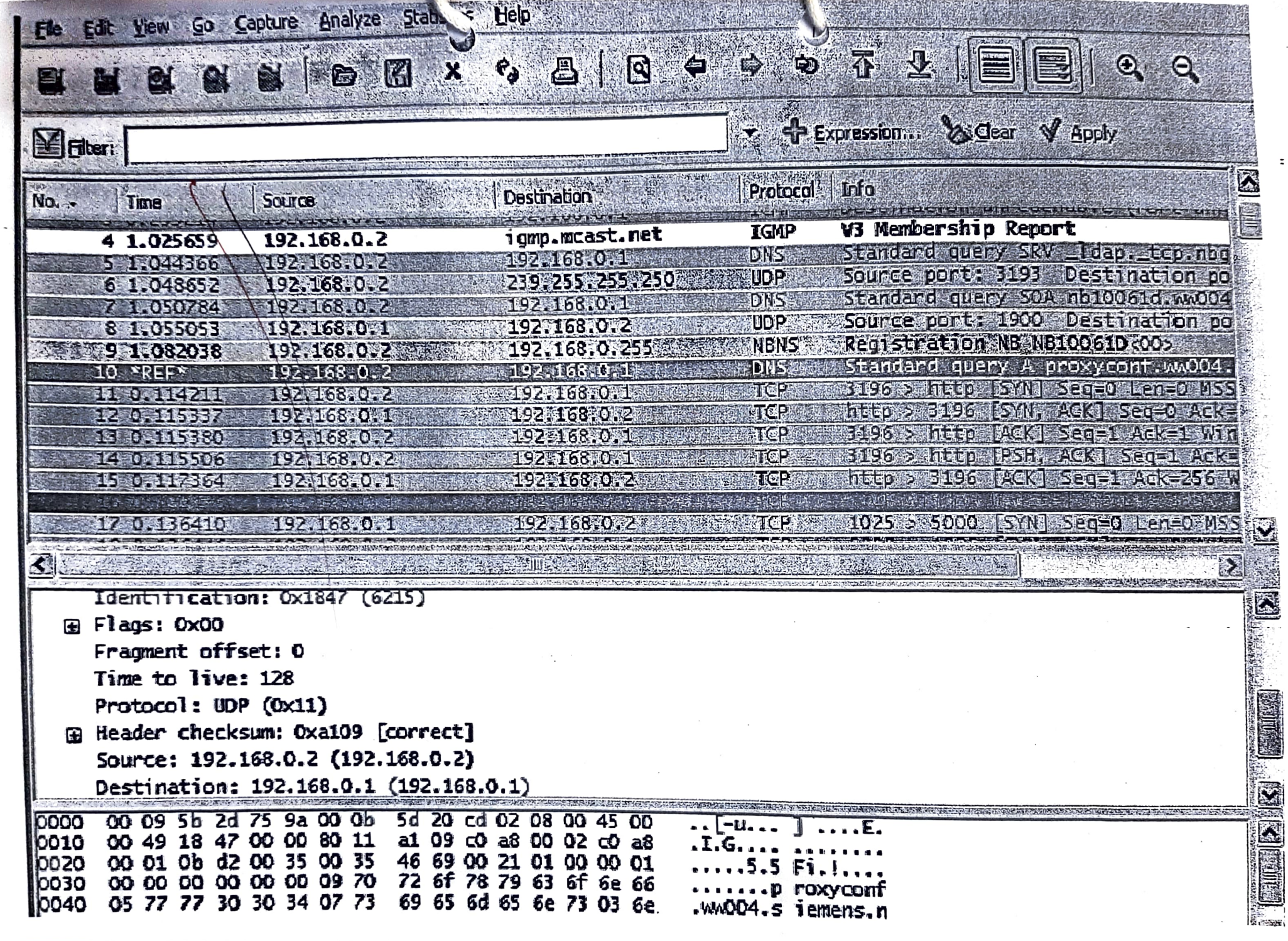
1) Data can be captured from the wire from a live network connection or read from a file or already-captured packets.
2) Captured N/W later can be inspected via GUI
3) Data display can be refined using a display filter.
4) Plug-ins can be created for dissecting new protocols.
5) VoIP calls in the captured traffic can be deleted.
6) Wireless connection can be also as long as they travers the ethernet.

Output :- On the blank page

Conclusion :- Hence, we have successfully implemented wireshark tool.

Teacher's Signature .....


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
No. Time Source Destination Protocol Length Info
343 65.142415 192.168.0.21 174.129.249.228 TCP 66 40555 → 80 [ACK] Seq=1 Ack=1 Win=5888 Len=0 TSval=491519346 TSecr=551811827
344 65.142715 192.168.0.21 174.129.249.228 HTTP 253 GET /clients/netflix/flash/application.swf?flash_version=flash_lite_2.1&v=1.5&nr
345 65.230738 174.129.249.228 192.168.0.21 TCP 66 80 → 40555 [ACK] Seq=1 Ack=188 Win=6864 Len=0 TSval=551811850 TSecr=491519347
346 65.240742 174.129.249.228 192.168.0.21 HTTP 828 HTTP/1.1 302 Moved Temporarily
347 65.241592 192.168.0.21 174.129.249.228 TCP 66 40555 → 80 [ACK] Seq=188 Ack=763 Win=7424 Len=0 TSval=491519446 TSecr=551811852
348 65.242532 192.168.0.21 192.168.0.1 DNS 77 Standard query 0x2188 A cdn-0.nflximg.com
349 65.276870 192.168.0.1 192.168.0.21 DNS 489 Standard query response 0x2188 A cdn-0.nflximg.com CNAME images.netflix.com.edg
350 65.277992 192.168.0.21 63.80.242.48 TCP 74 37063 → 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 TSval=491519482 TSecr
351 65.297757 63.80.242.48 192.168.0.21 TCP 74 80 → 37063 [SYN, ACK] Seq=0 Ack=1 Win=5792 Len=0 MSS=1460 SACK_PERM=1 TSval=329
352 65.298396 192.168.0.21 63.80.242.48 TCP 66 37063 → 80 [ACK] Seq=1 Ack=1 Win=5888 Len=0 TSval=491519502 TSecr=3295534130
353 65.298687 192.168.0.21 63.80.242.48 HTTP 153 GET /us/nrd/clients/flash/814540.bun HTTP/1.1
354 65.318730 63.80.242.48 192.168.0.21 TCP 66 80 → 37063 [ACK] Seq=1 Ack=88 Win=5792 Len=0 TSval=3295534151 TSecr=491519503
355 65.321733 63.80.242.48 192.168.0.21 TCP 1514 [TCP segment of a reassembled PDU]
Frame 349: 489 bytes on wire (3912 bits), 489 bytes captured (3912 bits)
Ethernet II, Src: Globalsc_00:3b:0a (f0:ad:4e:00:3b:0a), Dst: Vizio_14:8a:e1 (00:19:9d:14:8a:e1)
Internet Protocol Version 4, Src: 192.168.0.1, Dst: 192.168.0.21
User Datagram Protocol, Src Port: 53 (53), Dst Port: 34036 (34036)
Domain Name System (response)
[Request In: 348]
[Time: 0.034338000 seconds]
Transaction ID: 0x2188
Flags: 0x8180 Standard query response, No error
Questions: 1
Answer RRs: 4
Authority RRs: 9
Additional RRs: 9
Queries
cdn-0.nflximg.com: type A, class IN
Answers
Authoritative nameservers
0020 00 15 00 35 84 f4 01 c7 83 3f 21 88 81 80 00 01 ...5.... .?!.....
0030 00 04 00 09 00 09 05 63 64 6e 2d 30 07 6e 66 6c ....c dn-0.nfl
0040 78 69 6d 67 03 63 6f 6d 00 00 01 00 01 c0 0c 00 ximg.com ....
0050 05 00 01 00 00 05 29 00 22 06 69 6d 61 67 65 73 ......). ".images
0060 07 6e 65 74 66 6c 69 78 03 63 6f 6d 09 65 64 67 .netflix .com.edg
0070 65 73 75 69 74 65 03 6e 65 74 00 c0 2f 00 05 00 esuite.n et../...


File Edit View Go Capture Analyze Help
Filter:
Expression... Clear Apply
No. Time Source Destination Protocol Info
4 1.025659 192.168.0.2 igmp.mcast.net IGMP V3 Membership Report
5 1.044366 192.168.0.2 192.168.0.1 DNS Standard query SRV _ldap._tcp.nbg
6 1.048652 192.168.0.2 239.255.255.250 UDP Source port: 3193 Destination po
7 1.050784 192.168.0.2 192.168.0.1 DNS Standard query SOA nb10061d.ww004
8 1.055053 192.168.0.1 192.168.0.2 UDP Source port: 1900 Destination po
9 1.082038 192.168.0.2 192.168.0.255 NBNS Registration NB NB10061D<00>
10 *REF* 192.168.0.2 192.168.0.1 DNS Standard query A proxyconf.ww004.
11 0.114211 192.168.0.2 192.168.0.1 TCP 3196 > http [SYN] Seq=0 Len=0 MSS
12 0.115337 192.168.0.1 192.168.0.2 TCP http > 3196 [SYN, ACK] Seq=0 Ack=
13 0.115380 192.168.0.2 192.168.0.1 TCP 3196 > http [ACK] Seq=1 Ack=1 Win
14 0.115506 192.168.0.2 192.168.0.1 TCP 3196 > http [PSH, ACK] Seq=1 Ack=
15 0.117364 192.168.0.1 192.168.0.2 TCP http > 3196 [ACK] Seq=1 Ack=256 W
17 0.136410 192.168.0.1 192.168.0.2 TCP 1025 > 5000 [SYN] Seq=0 Len=0 MSS
Identification: 0x1847 (6215)
Flags: 0x00
Fragment offset: 0
Time to live: 128
Protocol: UDP (0x11)
Header checksum: 0xa109 [correct]
Source: 192.168.0.2 (192.168.0.2)
Destination: 192.168.0.1 (192.168.0.1)
0000 00 09 5b 2d 75 9a 00 0b 5d 20 cd 02 08 00 45 00 ..[-u...] ....E.
0010 00 49 18 47 00 00 80 11 a1 09 c0 a8 00 02 c0 a8 .I.G.... ....
0020 00 01 0b d2 00 35 00 35 46 69 00 21 01 00 00 01 .....5.5 Fi.!....
0030 00 00 00 00 00 00 09 70 72 6f 78 79 63 6f 6e 66 ....p roxyconf
0040 05 77 77 30 30 34 07 73 69 65 6d 65 6e 73 03 6e .ww004.s iemens.n