EXPERIMENT-1

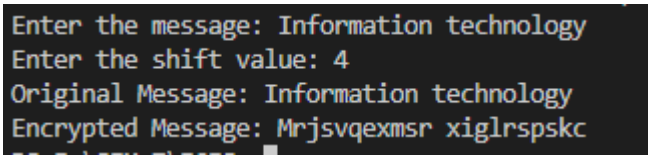
Aim:-Write program for Ceaser cipher Mono alphabetic cipher.
Tools / Apparatus: O.S.: Microsoft Windows (any) / Linux / DOS
Packages: Turbo/Borland/GNU - C/C++
Procedure:

1.Ceaser cipher

```
#include <iostream>
#include <string>
#include <cctype>

using namespace std;

string caesarCipher(const string &message, int shift)
{
   string encryptedMessage = "";

   for (char ch : message) {
      if (isalpha(ch)) {
         char base = (isupper(ch)) ? 'A' : 'a';
         char encryptedChar = (ch - base + shift) % 26 + base;
         encryptedMessage += encryptedChar;
      } else {
         encryptedMessage += ch;
      }
   }

   return encryptedMessage;
}
```

```
int main() {
    string message;
    int shift;

    cout << "Enter the message: ";
    getline(cin, message);

    cout << "Enter the shift value: ";
    cin >> shift;

    string encryptedMessage = caesarCipher(message, shift);

    cout << "Original Message: " << message << endl;
    cout << "Encrypted Message: " << encryptedMessage << endl;

    return 0;
}
```

Output:-



2.Monoalphabetic

```
#include<iostream>
#include<string>
#include<vector>
using namespace std;
int main(){
	string s;
	cout<<"Enter plain text :";
	getline(cin,s);
	cout<<"Enter key :";
	string key;
	getline(cin,key);
	vector<int> pt(26,-1);
	vector<int> ky(26,-1);
	cout<<"Encrypted message is : ";
	int j=0;
	for(int i=0;i<s.length();i++){
		if(s[i]==' '){
			cout<<s[i];
			continue;
		}
		if(s[i]>='a' && s[i]<='z'){
			if(pt[s[i]-'a']!=-1){
				cout<<(char)(pt[s[i]-'a']);
				continue;
			}
			while(key[j]==' '){
				j++;
				if(j==key.length()){
					j=0;
					for(int l=0;l<26;l++) ky[l]=-1;
				}
			}
			while(j<key.length() && ky[key[j]-'a']!=-1){
				j++;
				if(j==key.length()){
					j=0;
					for(int l=0;l<26;l++) ky[l]=-1;
				}
			}
			ky[key[j]-'a']=1;
			pt[s[i]-'a']=key[j];
```

```
                    cout<<key[j];
                    j++;
                }
        }
        cout<<endl;
        return 0;
}
```

Output:

```
Enter plain text :tommy was hungry and ate kitty
Enter key :i have done all of the assignments yesterday
Encrypted message is : ihaav edo nlftsv dfg dim yriiv
```

EXPERIMENT-2

Aim:-Implementation of Play Fair cipher.

Tools / Apparatus: O.S.: Microsoft Windows (any) / Linux / DOS
Packages: Turbo/Borland/GNU - C/C++

Procedure:

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
   string pt, key;
   cout << "Enter plain text : ";
   getline(cin, pt);
   cout << "Enter key : ";
   getline(cin, key);

   // matrix creation
   vector<int> keyCheck(26, 0);
   vector<pair<int, int>> indexMat(26);
   vector<vector<char>> mat(5, vector<char>(5));
   int a = 0, b = 0;
   for (int i = 0; i < key.length(); i++)
   {
      if (key[i] == 'i' || key[i] == 'j')
      {
         if (key[i] == 'i' && keyCheck[key[i] - 'a'] != 1)
         {
            mat[a][b] = key[i];
            indexMat[key[i] - 'a'] = {a, b};
            b++;
            if (b == 5)
            {
               b = 0;
               a++;
            }
            keyCheck['i' - 'a'] = 1;
            keyCheck['j' - 'a'] = 1;
         }
         else if (key[i] == 'j' && keyCheck[key[i] - 'a'] != 1)
         {
            mat[a][b] = 'i';
            indexMat[key[i] - 'a'] = {a, b};
            b++;
            if (b == 5)
```

```
            {
                b = 0;
                a++;
            }
            keyCheck['i' - 'a'] = 1;
            keyCheck['j' - 'a'] = 1;
        }
    }
    else
    {
        if (key[i] >= 'a' && key[i] <= 'z' && keyCheck[key[i] - 'a'] != 1)
        {
            mat[a][b] = key[i];
            indexMat[key[i] - 'a'] = {a, b};
            b++;
            if (b == 5)
            {
                b = 0;
                a++;
            }
            keyCheck[key[i] - 'a'] = 1;
        }
    }
}

for (int i = 0; i < 26; i++)
{
    if (keyCheck[i] != 1)
    {
        if (i == 8)
        {
            mat[a][b] = 'i';
            indexMat[i] = {a, b};
            b++;
            if (b == 5)
            {
                b = 0;
                a++;
            }
            keyCheck[i] = 1;
            keyCheck[i + 1] = 1;
        }
        mat[a][b] = (char)(i + 'a');
        indexMat[i] = {a, b};
        b++;
        if (b == 5)
```

```
        {
            b = 0;
            a++;
        }
        keyCheck[i] = 1;
    }
}

cout << "\nMatrix is : \n"
     << endl;
for (int i = 0; i < 5; i++)
{
    for (int j = 0; j < 5; j++)
    {
        cout << mat[i][j] << " ";
    }
    cout << endl;
}

// encryption

string ans;
char a1, a2;
for (int i = 0; i < pt.length(); i++)
{
    a1 = '$';
    a2 = '$';
    while (pt[i] == ' ' && i < pt.length())
    {
        i++;
    }
    if (i >= pt.length())
    {
        break;
    }
    a1 = pt[i];
    i++;
    while (pt[i] == ' ' && i < pt.length())
    {
        i++;
    }
    if (i == pt.length())
    {
        a2 = 'x';
    }
    else if (pt[i] == a1)
```

```
    {
        a2 = 'x';
        i--;
    }
    else
    {
        a2 = pt[i];
    }

    auto t1 = indexMat[a1 - 'a'];
    auto t2 = indexMat[a2 - 'a'];
    if (t1.first == t2.first)
    {
        ans.push_back(mat[t1.first][(t1.second + 1) % 5]);
        ans.push_back(mat[t2.first][(t2.second + 1) % 5]);
    }
    else if (t1.second == t2.second)
    {
        ans.push_back(mat[(t1.first + 1) % 5][t1.second]);
        ans.push_back(mat[(t2.first + 1) % 5][t2.second]);
    }
    else
    {
        ans.push_back(mat[t1.first][t2.second]);
        ans.push_back(mat[t2.first][t1.second]);
    }
}

cout << "\nYour encryption is : ";
cout << ans;

string decans;
cout<< "\nYour decryption is : ";
for (int i = 0; i < ans.length(); i++)
{
    a1 = '$';
    a2 = '$';
    while (ans[i] == ' ' && i < ans.length())
    {
        i++;
    }
    if (i >= ans.length())
        break;
    a1 = ans[i];
    i++;
    while (ans[i] == ' ' && i < ans.length())
```

```
            {
               i++;
            }
            if (i == ans.length())
               a2 = 'x';
            else if (ans[i] == a1)
            {
               a2 = 'x';
               i--;
            }
            else
               a2 = ans[i];

            auto t1 = indexMat[a1 - 'a'];
            auto t2 = indexMat[a2 - 'a'];
            if (t1.first == t2.first)
            {
               decans.push_back(mat[t1.first][((t1.second - 1)==-1)?4:(t1.second - 1)]);
               decans.push_back(mat[t2.first][((t2.second - 1)==-1)?4:(t2.second - 1)]);
            }
            else if (t1.second == t2.second)
            {
               decans.push_back(mat[((t1.first - 1)==-1)?4:(t1.first-1)][t1.second]);
               decans.push_back(mat[((t2.first - 1)==-1)?4:(t2.first-1)][t2.second]);
            }
            else
            {
               decans.push_back(mat[t1.first][t2.second]);
               decans.push_back(mat[t2.first][t1.second]);
            }
         }
         cout<<decans;
         return 0;
      }
```

Output:

```
Enter plain text : hidethegoldunderthecarpet
Enter key : information

Matrix is :

i n f o r
m a t b c
d e g h k
l p q s u
v w x y z

Your encryption is : doegbgghiskliеknbgkacnwpgf
Your decryption is : hidethegoldunderthecarpetx
```

EXPERIMENT-3

Aim:-Write program for Hill cipher.
Tools / Apparatus: O.S.: Microsoft Windows (any) / Linux / DOS
Packages: Turbo/Borland/GNU - C/C++
Procedure:

```
#include <bits/stdc++.h>
using namespace std;

int determinantOfMatrix(int mat[2][2], int n)
{
   int num1, num2, det = 1, index, total = 1; // Initialize result

   // temporary array for storing row
   int temp[n + 1];

   // loop for traversing the diagonal elements
   for (int i = 0; i < n; i++)
   {
      index = i; // initialize the index

      // finding the index which has non zero value
      while (index < n && mat[index][i] == 0)
      {
         index++;
      }
      if (index == n) // if there is non zero element
      {
         // the determinant of matrix as zero
         continue;
      }
      if (index != i)
      {
         // loop for swapping the diagonal element row and
         // index row
         for (int j = 0; j < n; j++)
         {
            swap(mat[index][j], mat[i][j]);
         }
         // determinant sign changes when we shift rows
         // go through determinant properties
         det = det * pow(-1, index - i);
```

```
        }

        // storing the values of diagonal row elements
        for (int j = 0; j < n; j++)
        {
            temp[j] = mat[i][j];
        }
        // traversing every row below the diagonal element
        for (int j = i + 1; j < n; j++)
        {
            num1 = temp[i];   // value of diagonal element
            num2 = mat[j][i]; // value of next row element

            // traversing every column of row
            // and multiplying to every row
            for (int k = 0; k < n; k++)
            {
                // multiplying to make the diagonal
                // element and next row element equal
                mat[j][k] = (num1 * mat[j][k]) - (num2 * temp[k]);
            }
            total = total * num1; // Det(kA)=kDet(A);
        }
    }

    // multiplying the diagonal elements to get determinant
    for (int i = 0; i < n; i++)
    {
        det = det * mat[i][i];
    }
    return (det / total); // Det(kA)/k=Det(A);
}

int main()
{
    string pt;
    string key;
    cout << "Enter plain text : ";
    getline(cin, pt);
    cout << "Enter key text(4 character) : ";
    getline(cin, key);

    string s;
    for (int i = 0; i < pt.length(); i++)
    {
        if (pt[i] >= 'a' && pt[i] <= 'z')
```

```
    {
        s.push_back(pt[i]);
    }
}

string ct;

for (int i = 0; i < s.length(); i += 2)
{
    if (i + 1 == s.length())
    {
        s.push_back('x');
    }
    if (s[i] >= 'a' && s[i] <= 'z' && s[i + 1] >= 'a' && s[i + 1] <= 'z')
    {
        int ans = ((s[i] - 'a') * (key[0] - 'a')) + ((s[i + 1] - 'a') * (key[2] - 'a'));
        ans = ans % 26;
        ct.push_back((char)(ans + 'a'));

        ans = ((s[i] - 'a') * (key[1] - 'a')) + ((s[i + 1] - 'a') * (key[3] - 'a'));
        ans = ans % 26;
        ct.push_back((char)(ans + 'a'));
    }
}

cout << "\nYour Encryption is : " << ct << "\n"
     << endl;
// for (int i = 0; i < ct.length(); i += 2)
// {
//     cout << (ct[i] - 'a') << " " << (ct[i + 1] - 'a') << endl;
// }clear

int q, a, b, r, t1, t2, t;
a = 26;
int mat[2][2] = {{(key[0] - 'a'), (key[1] - 'a')}, {(key[2] - 'a'), (key[3] - 'a')}};
b = determinantOfMatrix(mat, 2);

t1 = 0;
t2 = 1;

do
{
    q = a / b;
    r = a % b;
    t = t1 - (t2 * q);
    a = b;
```

```
        b = r;
        t1 = t2;
        t2 = t;
    } while (r != 0);

    // cout << t1 << endl;

    s = ct;
    string pta;
    for (int i = 0; i < s.length(); i += 2)
    {
        if (s[i] >= 'a' && s[i] <= 'z' && s[i + 1] >= 'a' && s[i + 1] <= 'z')
        {
            int ans = ((s[i] - 'a') * (key[3] - 'a')) + ((s[i + 1] - 'a') * (-1) * (key[2] - 'a'));
            ans *= t1;
            ans = ans % 26;
            if (ans < 0)
            {
                ans = 26 + ans;
            }
            // cout << ans << " ";
            pta.push_back((char)(ans + 'a'));

            ans = ((s[i] - 'a') * (-1) * (key[1] - 'a')) + ((s[i + 1] - 'a') * (key[0] - 'a'));
            ans *= t1;
            ans = ans % 26;
            if (ans < 0)
            {
                ans = 26 + ans;
            }
            // cout << ans << " ";
            pta.push_back((char)(ans + 'a'));
        }
    }
    cout << "\nYour plain text : " << pta << "\n"
         << endl;
    return 0;
}
```

Output:-

```
Enter plain text : information
Enter key text(4 character) : ddit

Your Encryption is : ylxvrtwxgepi

Your plain text : informationx
```

EXPERIMENT-4

Aim:- S-DES algorithm for data encryption along with key generation of S-DES.
Tools / Apparatus: O.S.: Microsoft Windows (any) / Linux / DOS
Packages: Turbo/Borland/GNU - C/C++
Procedure:

```
#include <bits/stdc++.h>
using namespace std;

int findRowColS1(int s1[4][4], string a)
{
   int i, j;
   if (a[1] == '0' && a[2] == '0')
   {
      i = 0;
   }
   else if (a[1] == '1' && a[2] == '0')
   {
      i = 2;
   }
   else if (a[1] == '0' && a[2] == '1')
   {
      i = 1;
   }
   else
   {
      i = 3;
   }

   if (a[0] == '0' && a[3] == '0')
   {
      j = 0;
   }
   else if (a[0] == '1' && a[3] == '0')
   {
      j = 2;
   }
   else if (a[0] == '0' && a[3] == '1')
   {
      j = 1;
   }
   else
   {
      j = 3;
```

```
    }

    return s1[j][i];
}

int findRowColS0(int s0[4][4], string a)
{
    int i, j;
    if (a[1] == '0' && a[2] == '0')
    {
        i = 0;
    }
    else if (a[1] == '1' && a[2] == '0')
    {
        i = 2;
    }
    else if (a[1] == '0' && a[2] == '1')
    {
        i = 1;
    }
    else
    {
        i = 3;
    }

    if (a[0] == '0' && a[3] == '0')
    {
        j = 0;
    }
    else if (a[0] == '1' && a[3] == '0')
    {
        j = 2;
    }
    else if (a[0] == '0' && a[3] == '1')
    {
        j = 1;
    }
    else
    {
        j = 3;
    }

    return s0[j][i];
}

string xorString(string a, string b, int n)
```

```
{
    string ans;
    for (int i = 0; i < n; i++)
    {
        if ((a[i] == '0' && b[i] == '0') || (a[i] == '1' && b[i] == '1'))
        {
            ans.push_back('0');
        }
        else
        {
            ans.push_back('1');
        }
    }
    return ans;
}
int main()
{
    // 00001011
    string pt, keyTemp;
    cout << "Enter plain text : ";
    cin >> pt;
    cout << "Enter key : ";
    cin >> keyTemp;

    string key;
    vector<int> p10(10);
    cout << "Enter p10 (in 10 number): ";
    for (int i = 0; i < 10; i++)
    {
        int x;
        cin >> x;
        p10[i] = x;
        key.push_back(keyTemp[x - 1]);
    }

    // 10 bit key partition

    string fbit = key.substr(0, 5);
    string sbit = key.substr(5, 5);

    // 1 shifting
    string lcs1fbit, lcs1sbit;
    for (int i = 0; i < 5; i++)
    {
        lcs1fbit.push_back(fbit[(i + 1) % 5]);
        lcs1sbit.push_back(sbit[(i + 1) % 5]);
```

```
}
string lcs1 = lcs1fbit + lcs1sbit;
// 2 shifting
string lcs2fbit, lcs2sbit;
for (int i = 0; i < 5; i++)
{
   lcs2fbit.push_back(lcs1fbit[(i + 2) % 5]);
   lcs2sbit.push_back(lcs1sbit[(i + 2) % 5]);
}
string lcs2 = lcs2fbit + lcs2sbit;

// calculating k1,k2
string k1, k2;
vector<int> p8(8);

cout << "Enter P8 (in 8 number): ";
for (int i = 0; i < 8; i++)
{
   int x;
   cin >> x;
   p8[i] = x;
   k1.push_back(lcs1[x - 1]);
   k2.push_back(lcs2[x - 1]);
}

cout << "k1 : " << k1 << endl;
cout << "k2 : " << k2 << endl;

// Encryption

vector<int> ip(8);
string ptip;
cout << "Enter IP (in 8 number) : ";
for (int i = 0; i < 8; i++)
{
   int x;
   cin >> x;
   ip[i] = x;
   ptip.push_back(pt[x - 1]);
}

vector<int> exp(8);
string rightExp;
cout << "Enter expanded permutation (in 8 number) : ";
for (int i = 0; i < 8; i++)
{
```

```
    int x;
    cin >> x;
    exp[i] = x;
    rightExp.push_back(ptip[x - 1 + 4]);
}

// rightExp XOR k1
string afterf1 = xorString(rightExp, k1, 8);
int s0[4][4] = {{1, 0, 3, 2}, {3, 2, 1, 0}, {0, 2, 1, 3}, {3, 1, 3, 2}};
int s1[4][4] = {{0, 1, 2, 3}, {2, 0, 1, 3}, {3, 0, 1, 0}, {2, 1, 0, 3}};

int a, b;
a = findRowColS0(s0, afterf1.substr(0, 4));
b = findRowColS1(s1, afterf1.substr(4, 4));
cout << "a b : " << a << " " << b << endl;

string afterS0S1;

if (a == 0)
{
    afterS0S1.push_back('0');
    afterS0S1.push_back('0');
}
else if (a == 1)
{
    afterS0S1.push_back('0');
    afterS0S1.push_back('1');
}
else if (a == 2)
{
    afterS0S1.push_back('1');
    afterS0S1.push_back('0');
}
else
{
    afterS0S1.push_back('1');
    afterS0S1.push_back('1');
}

if (b == 0)
{
    afterS0S1.push_back('0');
    afterS0S1.push_back('0');
}
else if (b == 1)
{
```

```
    afterS0S1.push_back('0');
    afterS0S1.push_back('1');
}
else if (b == 2)
{
    afterS0S1.push_back('1');
    afterS0S1.push_back('0');
}
else
{
    afterS0S1.push_back('1');
    afterS0S1.push_back('1');
}
cout << "afterS0S1 : " << afterS0S1 << endl;
cout << "Enter p4 (in 4 number) : ";
vector<int> p4(4);
string rightWithoutXor;
for (int i = 0; i < 4; i++)
{
    int x;
    cin >> x;
    p4[i] = x;
    rightWithoutXor.push_back(afterS0S1[x - 1]);
}

string finalRight = xorString(rightWithoutXor, ptip.substr(0, 4), 4);
cout << "finalRight : " << finalRight << endl;

// swap done
string afterFun1 = ptip.substr(4, 4) + finalRight;

cout << "afterFun1 : " << afterFun1 << endl;

// now ptip is afterFun1
ptip = afterFun1;
string rightExp2;
for (int i = 0; i < 8; i++)
{
    rightExp2.push_back(ptip[exp[i] - 1 + 4]);
}

cout << "rightExp2 : " << rightExp2 << endl;

string afterf2 = xorString(rightExp2, k2, 8);
cout << "afterf2 : " << afterf2 << endl;
```

```
a = findRowColS0(s0, afterf2.substr(0, 4));
b = findRowColS1(s1, afterf2.substr(4, 4));

string afterS0S12nd;

if (a == 0)
{
   afterS0S12nd.push_back('0');
   afterS0S12nd.push_back('0');
}
else if (a == 1)
{
   afterS0S12nd.push_back('0');
   afterS0S12nd.push_back('1');
}
else if (a == 2)
{
   afterS0S12nd.push_back('1');
   afterS0S12nd.push_back('0');
}
else
{
   afterS0S12nd.push_back('1');
   afterS0S12nd.push_back('1');
}

if (b == 0)
{
   afterS0S12nd.push_back('0');
   afterS0S12nd.push_back('0');
}
else if (b == 1)
{
   afterS0S12nd.push_back('0');
   afterS0S12nd.push_back('1');
}
else if (b == 2)
{
   afterS0S12nd.push_back('1');
   afterS0S12nd.push_back('0');
}
else
{
   afterS0S12nd.push_back('1');
   afterS0S12nd.push_back('1');
}
```

```
        cout << "afterS0S12nd : " << afterS0S12nd << endl;

        string rightWithoutXor2nd;
        for (int i = 0; i < 4; i++)
        {
            rightWithoutXor2nd.push_back(afterS0S12nd[p4[i] - 1]);
        }
        cout << "rightWithoutXor2nd : " << rightWithoutXor2nd << endl;

        string finalRight2nd = xorString(rightWithoutXor2nd, ptip.substr(0, 4), 4);
        cout << "finalRight2nd : " << finalRight2nd << endl;

        string afterFun2 = finalRight2nd + ptip.substr(4, 4);
        cout << "afterFun2 : " << afterFun2 << endl;

        string ans;
        cout << "Enter IP^(-1) : ";
        vector<int> ip1(8);
        for (int i = 0; i < 8; i++)
        {
            int x;
            cin >> x;
            ip1[i] = x;
            ans.push_back(afterFun2[x - 1]);
        }
        cout << "Final Encryption is : " << ans << endl;
        return 0;
    }
```

Output:-

```
Enter plain text : 10111101
Enter key : 1010000010
Enter p10 (in 10 number): 3 5 2 7 4 10 1 9 8 6
Enter P8 (in 8 number): 6 3 7 4 8 5 10 9
k1 : 10100100
k2 : 01000011
Enter IP (in 8 number) : 2 6 3 1 4 8 5 7
Enter expanded permutation (in 8 number) : 4 1 2 3 2 3 4 1
a b : 3 2
afterS0S1 : 1110
Enter p4 (in 4 number) : 2 4 3 1
finalRight : 1100
afterFun1 : 11101100
rightExp2 : 01101001
afterf2 : 00101010
afterS0S12nd : 0000
rightWithoutXor2nd : 0000
finalRight2nd : 1110
afterFun2 : 11101100
Enter IP^(-1) : 4 1 3 5 7 2 8 6
Final Encryption is : 01110101
```

EXPERIMENT-5

Aim:- Write a program to generate and exchange public keys using client server mechanism
Tools / Apparatus: O.S.: Microsoft Windows (any) / Linux / DOS
Packages: Turbo/Borland/GNU - C/C++

Server.java

```
import java.io.*;
import java.net.*;
import java.util.Scanner;


public class Server {
 public static boolean isPrime(int n) {
       if (n <= 1) {
          return false;
       }
       if (n <= 3) {
          return true;
       }
       if (n % 2 == 0 || n % 3 == 0) {
          return false;
       }
       for (int i = 5; i * i <= n; i += 6) {
          if (n % i == 0 || n % (i + 2) == 0) {
             return false;
          }
       }
       return true;
}
   public static void main(String[] args) {
       try {
```

```
        ServerSocket serverSocket = new ServerSocket(12345);
        System.out.println("Waiting for client...");

        Socket clientSocket = serverSocket.accept();
        System.out.println("Client connected!");

        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter prime number p: ");
        int p = scanner.nextInt();

        System.out.print("Enter prime number q: ");
        int q = scanner.nextInt();
        System.out.print("Enter prime number e: ");
        int e=scanner.nextInt();
        if(isPrime(p) && isPrime(q)){
        int n = p * q;
        int phi_n = (p - 1) * (q - 1);
        int d = calculateModInverse(e, phi_n);
           System.out.println("Value of d is:"+d);

        DataOutputStream out = new DataOutputStream(clientSocket.getOutputStream());
        out.writeInt(n);
        out.writeInt(e);
        out.writeInt(d);

        clientSocket.close();
        serverSocket.close();
    }else{
           System.out.println("Check whether p and q are ");
```

```
        } }

        catch (Exception e) {
            e.printStackTrace();
        }


    }
    private static int calculateModInverse(int a, int m) {
        a = a % m;
        for (int x = 1; x < m; x++) {
            if ((a * x) % m == 1) {
                return x;
            }
        }
        return 1;
    }
}
```

Client.cpp:

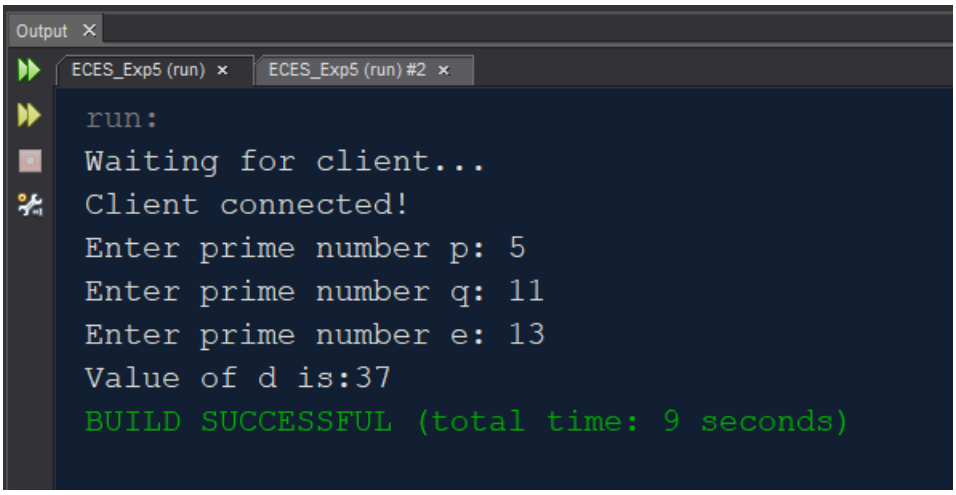
```
import java.io.*;
import java.net.*;
import java.math.BigInteger;
import java.util.Scanner;

public class Client {
    public static void main(String[] args) {
        try {
            Socket socket = new Socket("127.0.0.1", 12345);
```

```
        DataInputStream in = new DataInputStream(socket.getInputStream());
        int n = in.readInt();
        int e = in.readInt();
        int d=in.readInt();
        System.out.println("Public Key{e,n}: {" + e + ", " + n + "}");
        System.out.println("Public Key{d,n}: {" + d + ", " + n + "}");

        // Use the received public key for encryption
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter message to encrypt: ");
        double message = scanner.nextDouble();

        double encryptedMessage = encrypt(message, e, n);

        System.out.println("Encrypted Message: " + encryptedMessage);
        double decryptedMessage=decrypt(encryptedMessage,d,n);
        System.out.println("Decrypted Message: " + decryptedMessage);
        socket.close();
    } catch (Exception e) {
        e.printStackTrace();
    }
}

private static double encrypt(double message, int e, int n) {
   double encrypted=modPow(message,e,n);
    return encrypted;
}
public static double modPow(double base, double exponent, double modulus) {
    double result = 1.0;
    base = base % modulus;
```

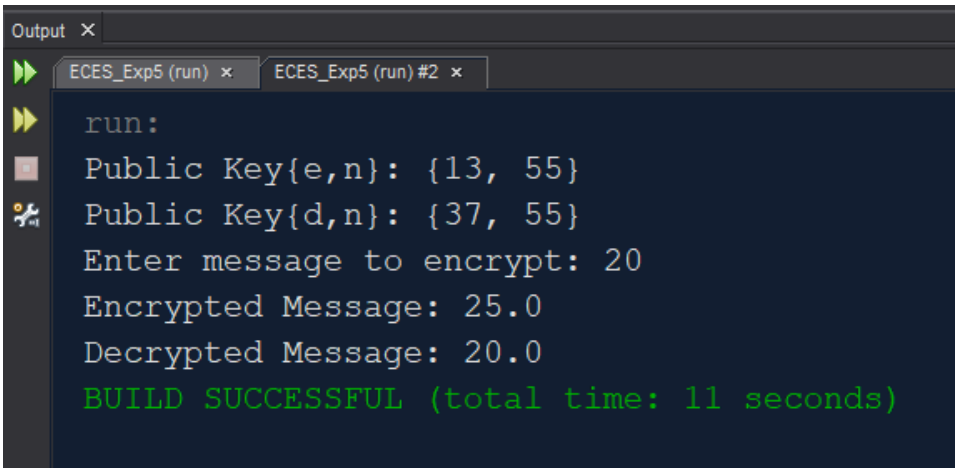
```
        while (exponent > 0) {
            if (exponent % 2 == 1) {
                result = (result * base) % modulus;
            }
            exponent = Math.floor(exponent / 2);  // Use Math.floor to ensure integer division
            base = (base * base) % modulus;
        }

        return result;
    }
    private static double decrypt(double message, int d, int n) {
      double decrypted=modPow(message,d,n);

        return decrypted;
    }
}
```

Output:

Server:



Client:



EXPERIMENT-6

Aim:- Perform Encryption, Authentication and both using RSA. (Use public key shared in above practical).

Tools / Apparatus: O.S.: Microsoft Windows (any) / Linux / DOS
Packages: Turbo/Borland/GNU - C/C++

Procedure:

```
/* Encrypt pain text "HI" using rsa algoritham for the given data p=53,q=59 */

#include <bits/stdc++.h>
using namespace std;

int power(int x, int y, int p)
{
   int res = 1; // Initialize result
   x = x % p; // Update x if it is more than or equal to p

   if (x == 0)
      return 0; // In case x is divisible by p;

   while (y > 0)
   {
      // If y is odd, multiply x with result
      if (y & 1)
         res = (res * x) % p;

      // y must be even now
      y = y >> 1; // y = y/2
      x = (x * x) % p;
   }
   return res;
}

int main()
{
   int m = 0;
   string pt;
   cout << "Enter plain text : ";
   cin >> pt;

   string ptm;
   for (int i = 0; i < pt.length(); i++)
   {
      ptm += to_string((int)(pt[i] - 'a'));
```

```
}

m = stoi(ptm);
cout << "your plain text in number is : " << m << endl;

int p, q, d;
cout << "Enter value of p : ";
cin >> p;
cout << "Enter value of q : ";
cin >> q;

int n, fn, e = 2;

n = p * q;
fn = (p - 1) * (q - 1);

// gcd(fn,e)=17
while (true)
{
   if (__gcd(fn, e) == 1)
   {
      break;
   }
   e++;
}

cout << "e is : " << e << endl;

// e*d = 1 mod fn
// find using inverse module function

int t1 = 0;
int t2 = 1;

int a = fn;
int b = e;

int r, qo, t;

do
{
   qo = a / b;
   r = a % b;
   t = t1 - (t2 * qo);
   a = b;
   b = r;
```

```
    t1 = t2;
    t2 = t;
} while (r != 0);

d = t1;

if (d < 0)
{
    d = fn + d;
}
cout << "d is : " << d << endl;

cout << "Public key : { " << e << " , " << n << " }" << endl;
cout << "Private key : { " << d << " , " << n << " }" << endl;

// Encryption
// c=m^e mod n

int c = power(m, e, n);
string cstr;
string cipher;
cstr = to_string(c);
if (cstr.length() == 2)
{
    cipher.push_back((char)(cstr[0] - '0' + 'a'));
    cipher.push_back((char)(cstr[1] - '0' + 'a'));
    cout << "Cipher text is : " << cipher << endl;
    cout << "Cipher text is : " << c << endl;
}
else if (cstr.length() == 3)
{
    cout << "Can't convert into text." << endl;
    cout << "Cipher text is : " << c << endl;
}
else
{
    int x, y;
    x = ((int)(cstr[0] - '0') * 10) + (int)(cstr[1] - '0');
    y = ((int)(cstr[2] - '0') * 10) + (int)(cstr[3] - '0');
    x = x % 26;
    y = y % 26;
    cipher.push_back((char)(x + 'a'));
    cipher.push_back((char)(y + 'a'));
    cout << "Cipher text is : " << cipher << endl;
    cout << "Cipher text is : " << c << endl;
}
```

```
        // Decryption
        // m=c^d mod n

        int m1 = power(c, d, n);

        string plain;
        cstr = to_string(m1);
        if (cstr.length() == 2)
        {
           plain.push_back((char)(cstr[0] - '0' + 'a'));
           plain.push_back((char)(cstr[1] - '0' + 'a'));
           cout << "Plain text is : " << plain << endl;
           cout << "Plain text is : " << m1 << endl;
        }
        else if (cstr.length() == 3)
        {
           cout << "Can't convert into text." << endl;
           cout << "Plain text is : " << m1 << endl;
        }
        else
        {
           int x, y;
           x = ((int)(cstr[0] - '0') * 10) + (int)(cstr[1] - '0');
           y = ((int)(cstr[2] - '0') * 10) + (int)(cstr[3] - '0');
           x = x % 26;
           y = y % 26;
           plain.push_back((char)(x + 'a'));
           plain.push_back((char)(y + 'a'));
           cout << "Plain text is : " << plain << endl;
           cout << "Plain text is : " << m1 << endl;
        }
        return 0;
    }
```

Output:-

```
Enter plain text : hi
your plain text in number is : 78
Enter value of p : 53
Enter value of q : 59
e is : 3
d is : 2011
Public key : { 3 , 3127 }
Private key : { 2011 , 3127 }
Cipher text is : xx
Cipher text is : 2375
Plain text is : hi
Plain text is : 78
```

EXPERIMENT-7

Aim:- Write a program to implement Diffie-Hellman Key exchange algorithm and perform encryption and decryption.

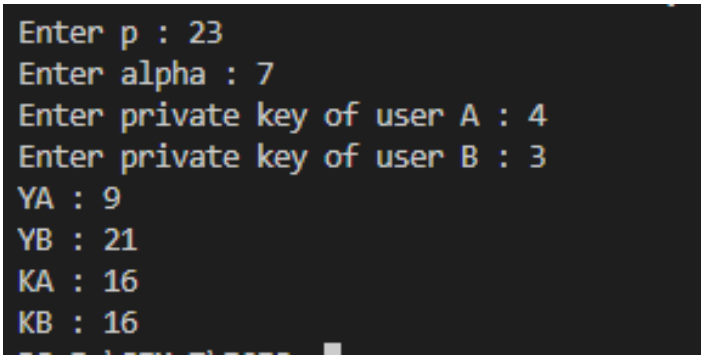
Tools / Apparatus: O.S.: Microsoft Windows (any) / Linux / DOS
Packages: Turbo/Borland/GNU - C/C++

Program :

```
#include <bits/stdc++.h>
using namespace std;

int power(int x, int y, int p)
{
   int res = 1;
   x = x % p;
   if (x == 0)
      return 0;
   while (y > 0)
   {
      if (y & 1)
         res = (res * x) % p;
      y = y >> 1;
      x = (x * x) % p;
   }
   return res;
}
int main()
{
   int alpha, p, xa, xb, ya, yb, ka, kb;
   cout << "Enter p : ";
```

```
    cin >> p;
    cout << "Enter alpha : ";
    cin >> alpha;
    cout << "Enter private key of user A : ";
    cin >> xa;
    cout << "Enter private key of user B : ";
    cin >> xb;

    ya = power(alpha, xa, p);
    yb = power(alpha, xb, p);
    ka = power(yb, xa, p);
    kb = power(ya, xb, p);

    cout << "YA : " << ya << endl;
    cout << "YB : " << yb << endl;
    cout << "KA : " << ka << endl;
    cout << "KB : " << kb << endl;
    return 0;
}
```

Output :



EXPERIMENT-8

Aim:- Write a program to authenticate a user with system using MD5 or SHA-1 Hashing Technique.

Tools / Apparatus: O.S.: Microsoft Windows (any) / Linux / DOS
Packages: Turbo/Borland/GNU - C/C++

Procedure:

```
import java.math.BigInteger;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.util.*;

public class Md5 {
   public static String generateMd5(String s) {
      try {
         MessageDigest md = MessageDigest.getInstance("MD5");
         byte[] bit = md.digest(s.getBytes());
         BigInteger bi = new BigInteger(1, bit);
         String hashValue = bi.toString(16);
         while (hashValue.length() < 32) {
            hashValue = "0" + hashValue;
         }
         return hashValue;
      } catch (NoSuchAlgorithmException e) {
         throw new RuntimeException(e);
      }
   }

   public static void main(String args[]) throws
NoSuchAlgorithmException {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter Message : ");
      String str = sc.nextLine();
      System.out.println("Your HashCode Generated by MD5 is : " +
generateMd5(str));
      sc.close();
   }
}
```

Output:

```
Enter Message : Information
Your HashCode Generated by MD5 is : a82be0f551b8708bc08eb33cd9ded0cf
```

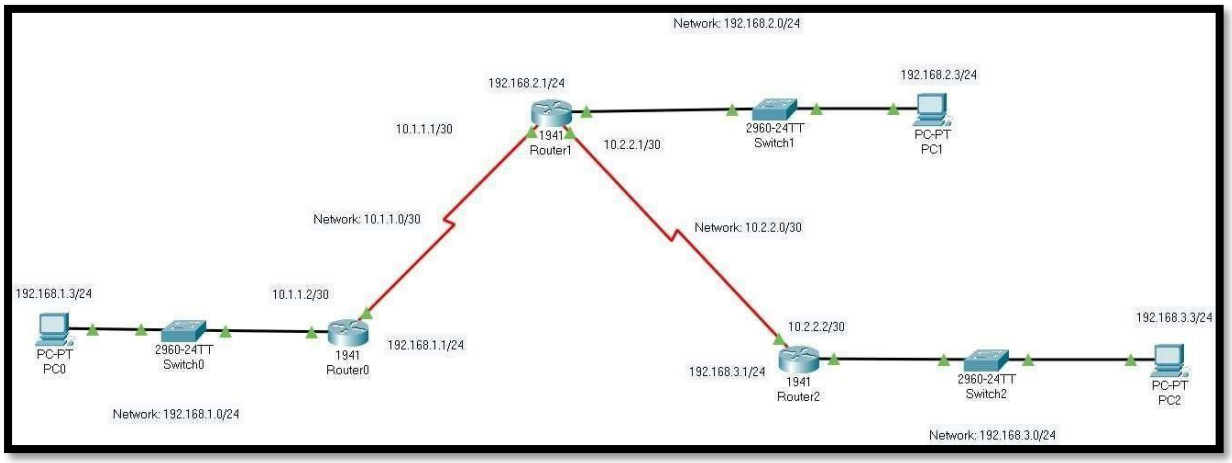
EXPERIMENT-9

Aim:- Configure VPN using Packet Tracer and demonstrate the importance of IPSec**.**

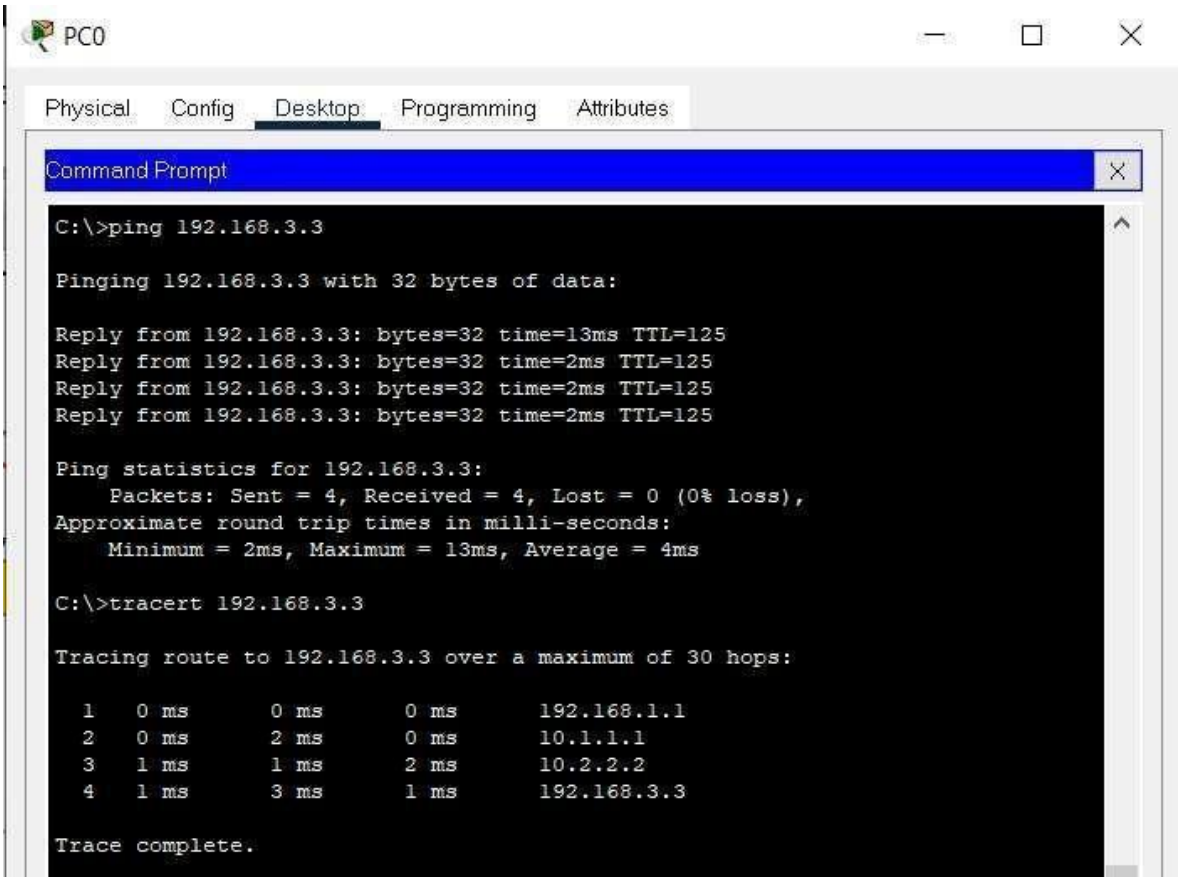
Tools / Apparatus: O.S.: Microsoft Windows (any) / Linux / DOS
Packages: Turbo/Borland/GNU - C/C++

Procedure:

Topology for the configuration



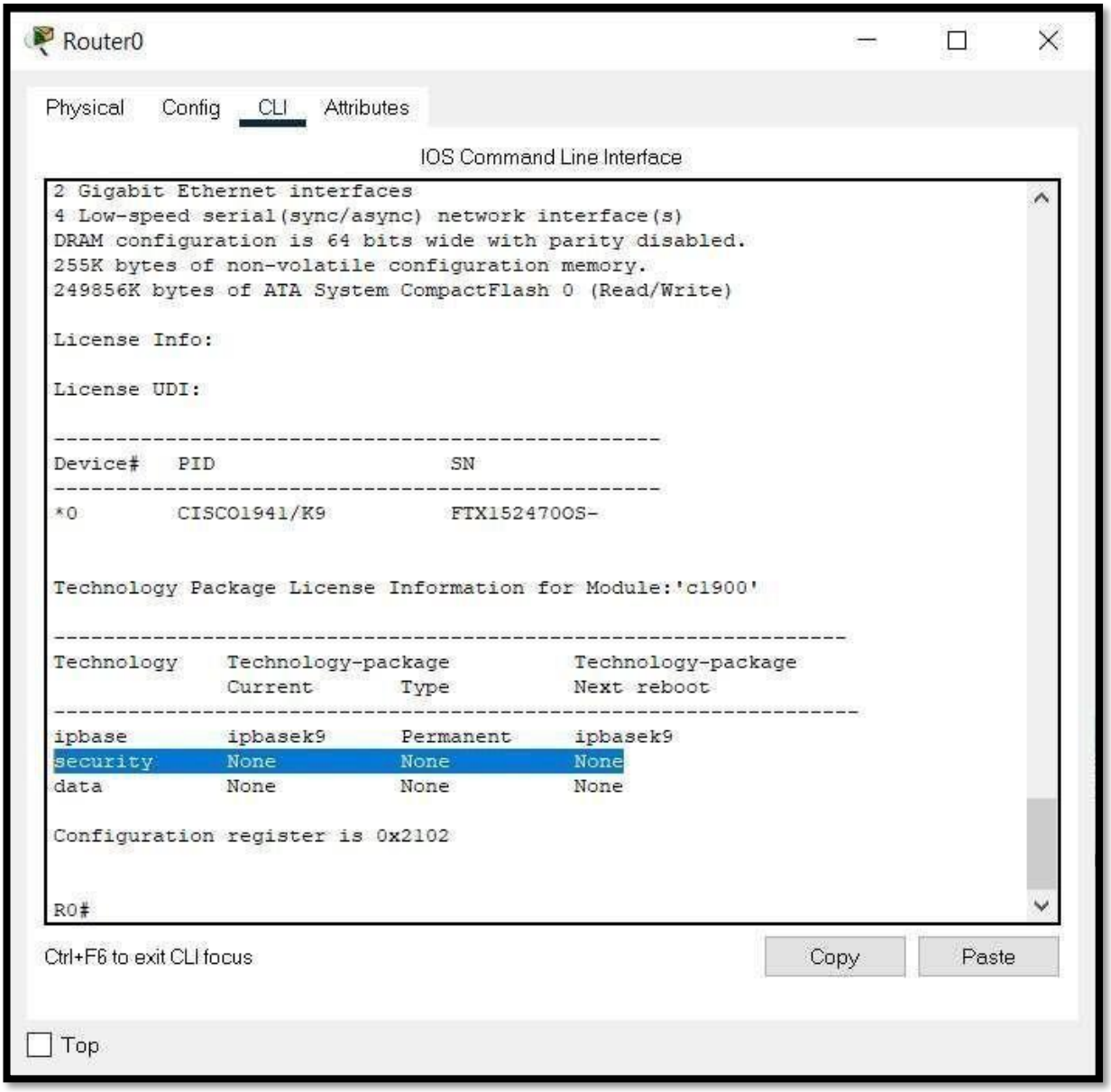
Connection between 2 endpoints



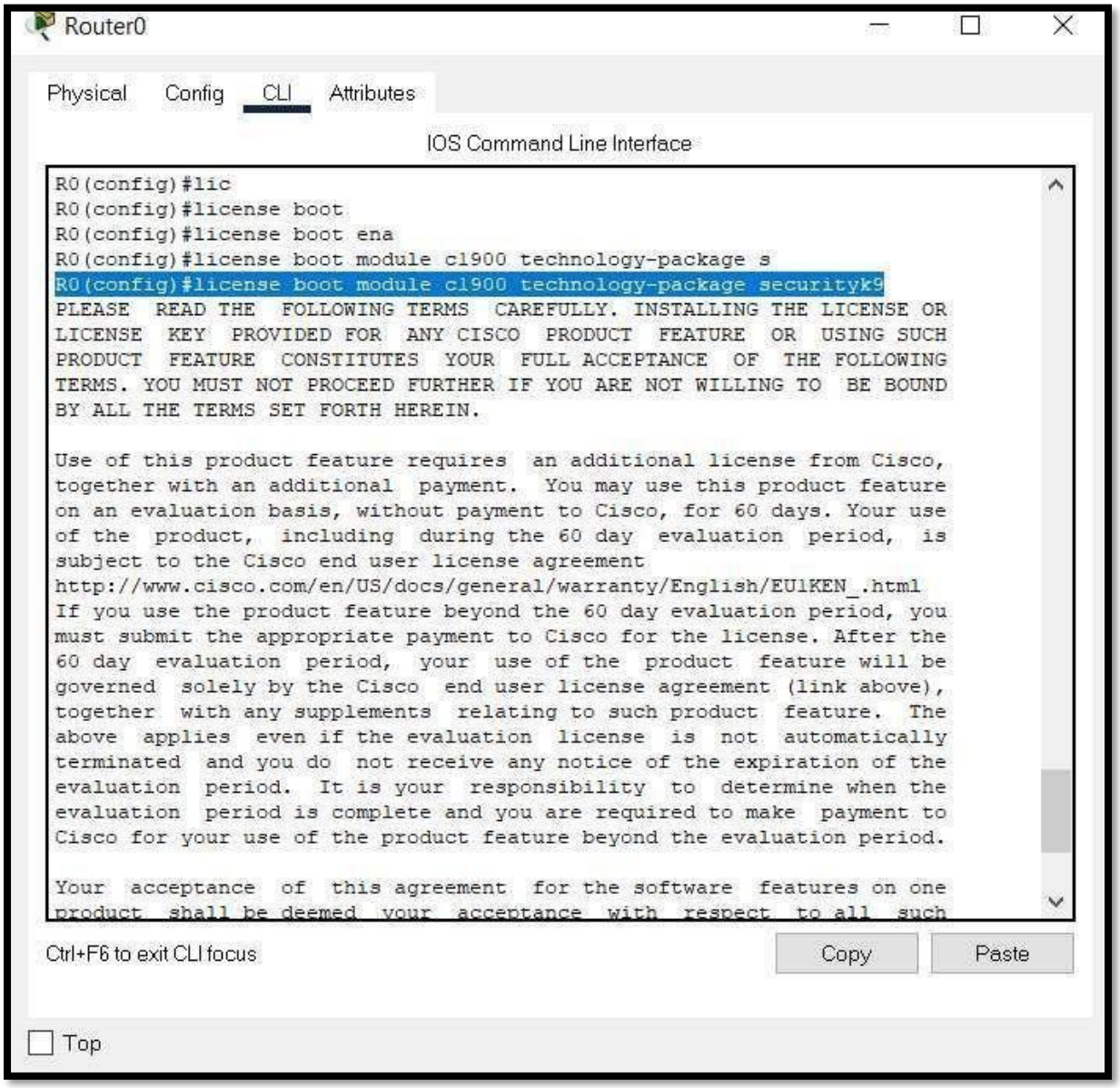
Part 1: Enable security features

1. Issue the show version command in the user EXEC or privileged EXEC mode to verifythat the security technology package license is activated.
2. If not, activate the securityk9 module for the next boot of the router, accept the license, save the configuration, and reboot R1(config)# license boot module c2900technology-package securityk9 R1(config)# end R1# copy running-config startup- config R1# reload
3. After the reloading is completed, issue the show version again to verify the securitytechnology package license activation. Do in Router R3.

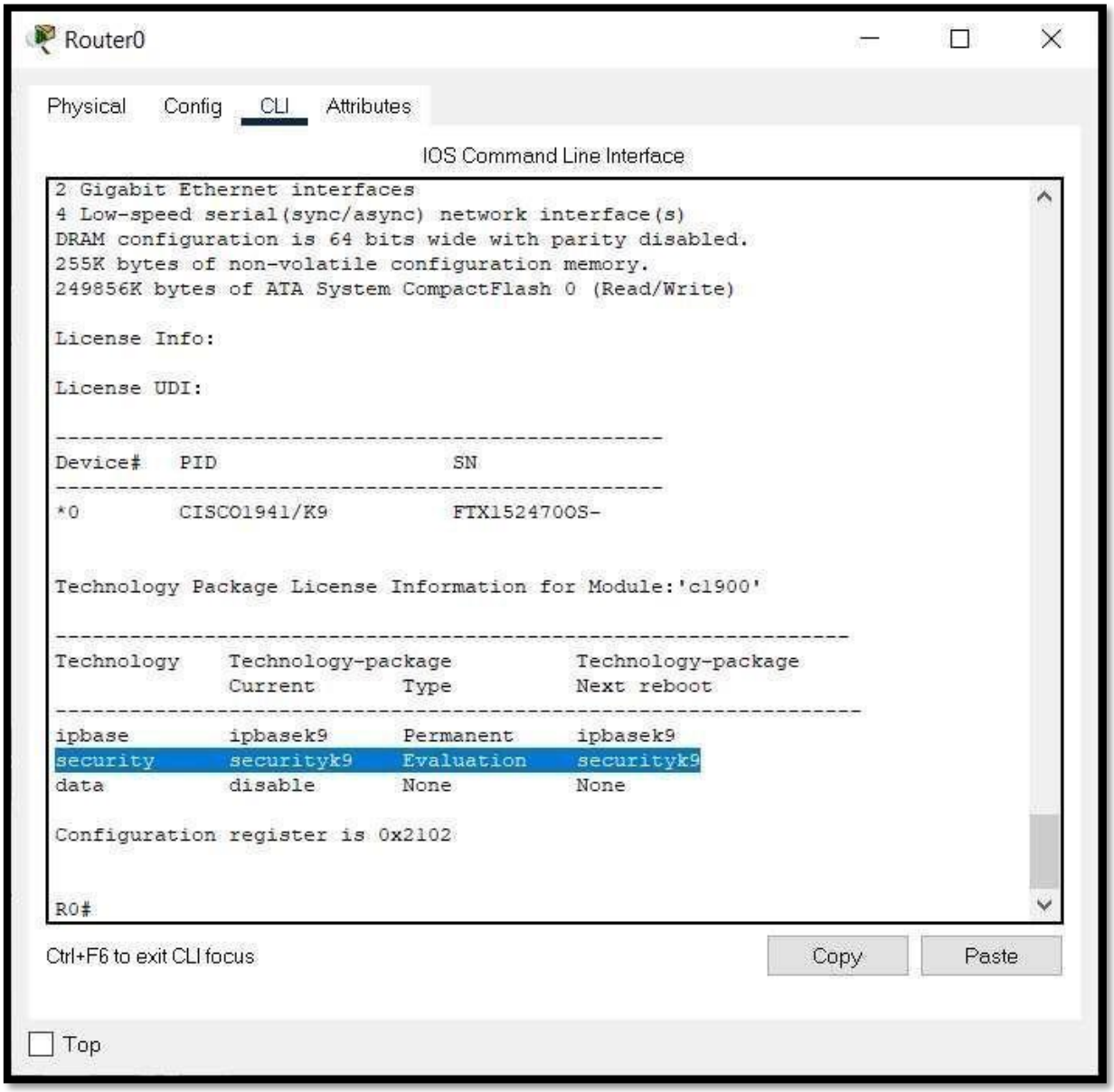
Security is disabled in Router0



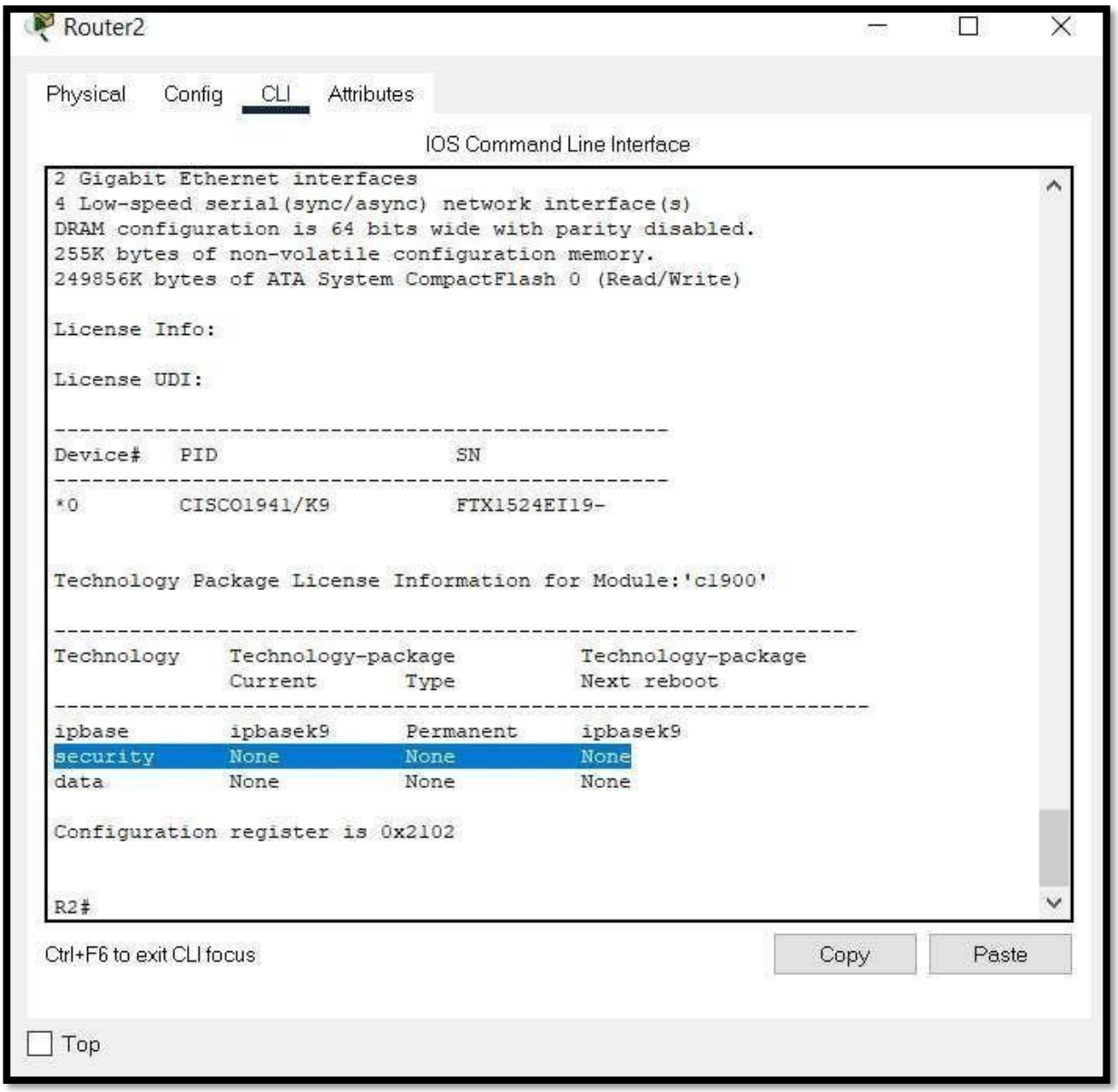
Enabling Security



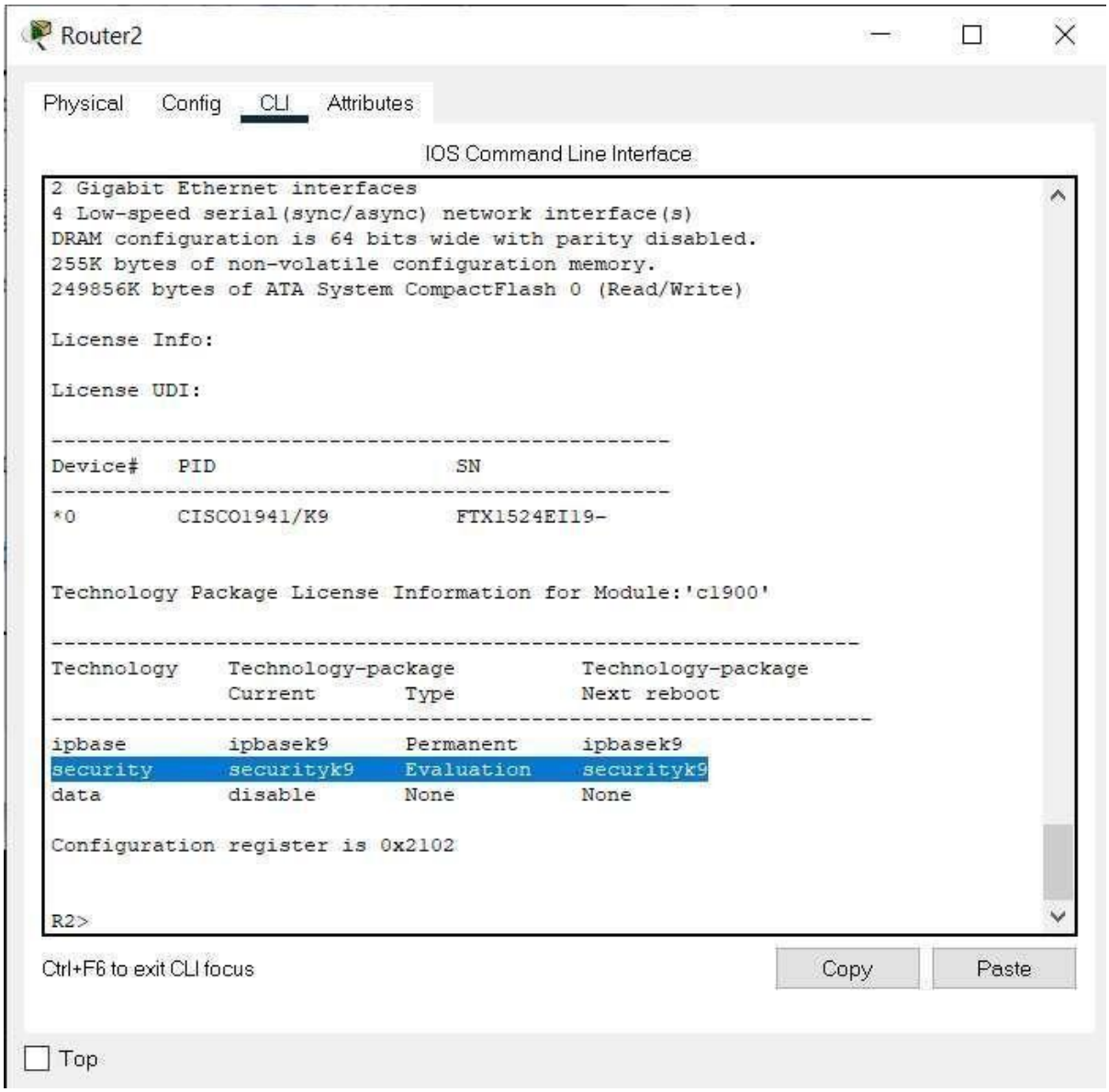
Security enabled



Security is disabled in Router0



Security enabled

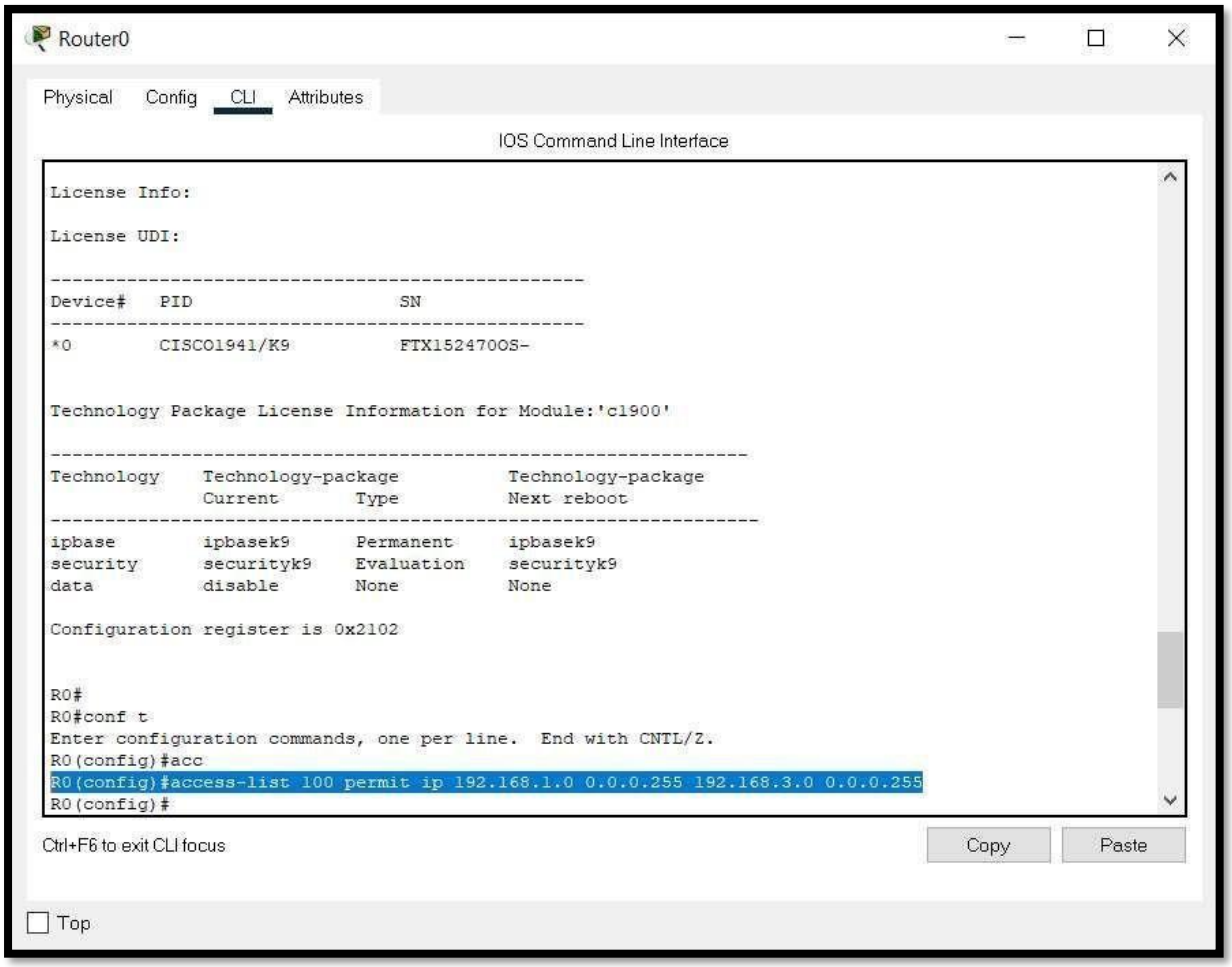


Part 2: Configure IPSec Parameters on R1 and R3

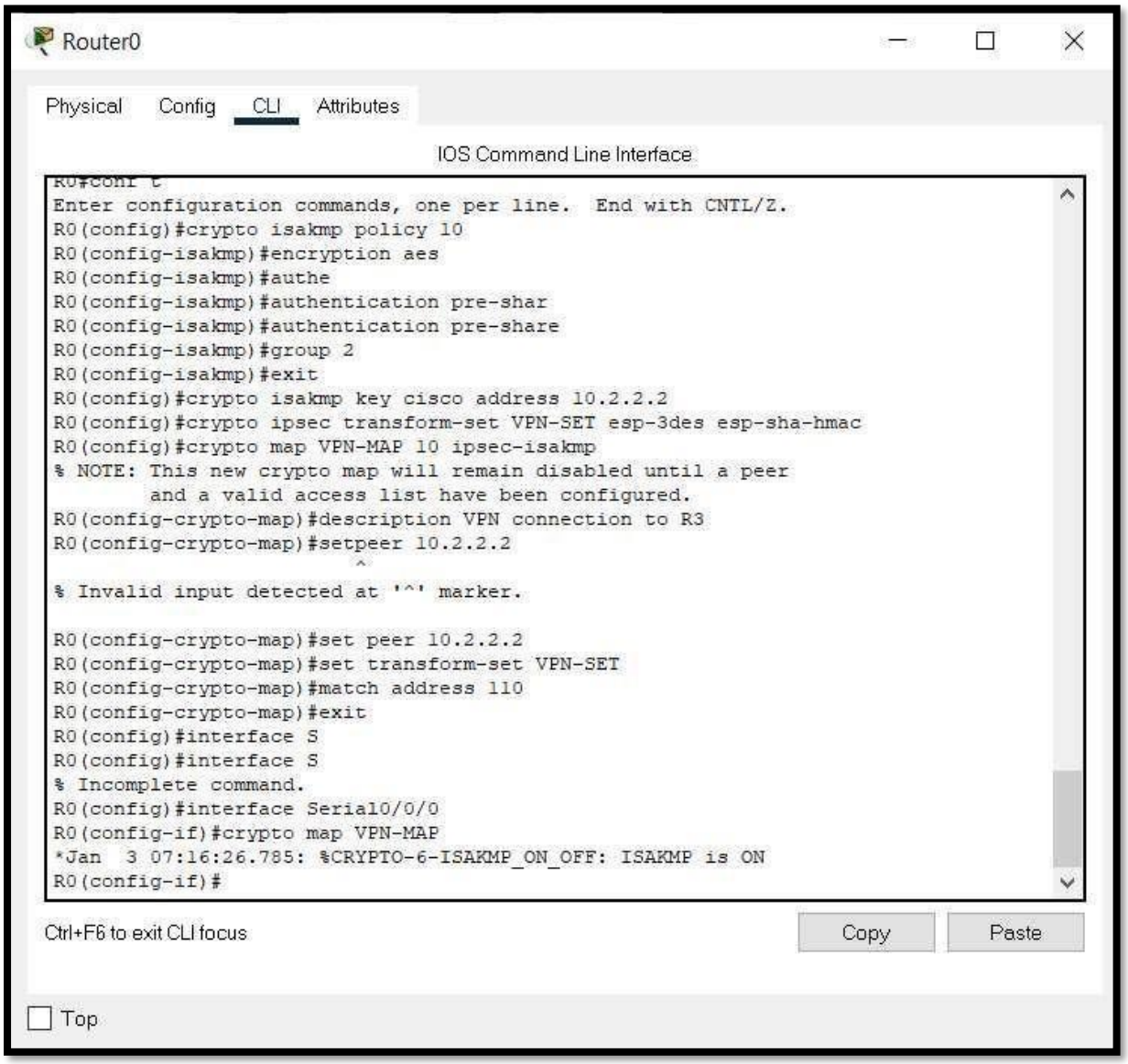
1. Test connectivity
2. Identify interesting traffic on R1. R1(config)# access-list 110 permit ip 192.168.1.0 0.0.0.255 192.168.3.00.0.0.255
3. Configure the ISAKMP Phase 1 properties on R1. R1(config)# crypto isakmp policy 10 R1(config-isakmp)# encryption aes R1(config-isakmp)# authentication preshare R1(config-isakmp)# group 2 R1(config-isakmp)# exitR1(config)# crypto isakmp key cisco address 10.2.2.2
4. Configure the ISAKMP Phase 2 properties on R1. R1(config)# crypto ipsec transform-set VPN-SET esp-3des esp-sha-hmac R1(config)# crypto map VPN-MAP 10 ipsec-isakmp R1(config-crypto-map)# description VPN connection to R3 R1(config-crypto-map)# set peer 10.2.2.2 R1(config-crypto-map)# set transform-set VPN-SET R1(config-crypto-map)# match address 110 R1(config-crypto-map)# exit

5. Configure the crypto map on the outgoing interface R1(config)# interfaceS0/0/0
   R1(config-if)# crypto map VPN-MAP
6. Configure IPSec Parameters on R3 same as R1

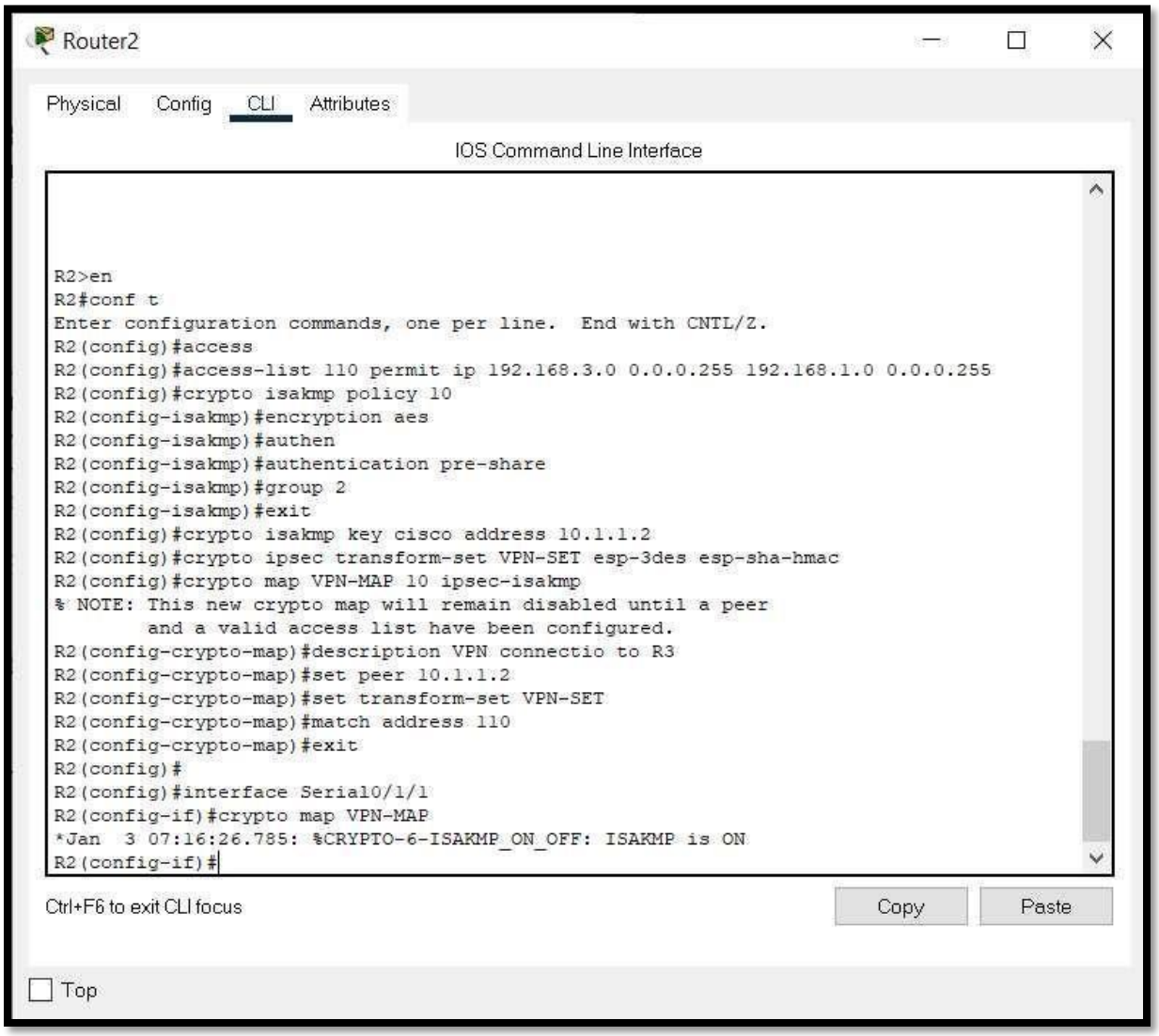
Generating interesting traffic



Executing part 2 in Router0



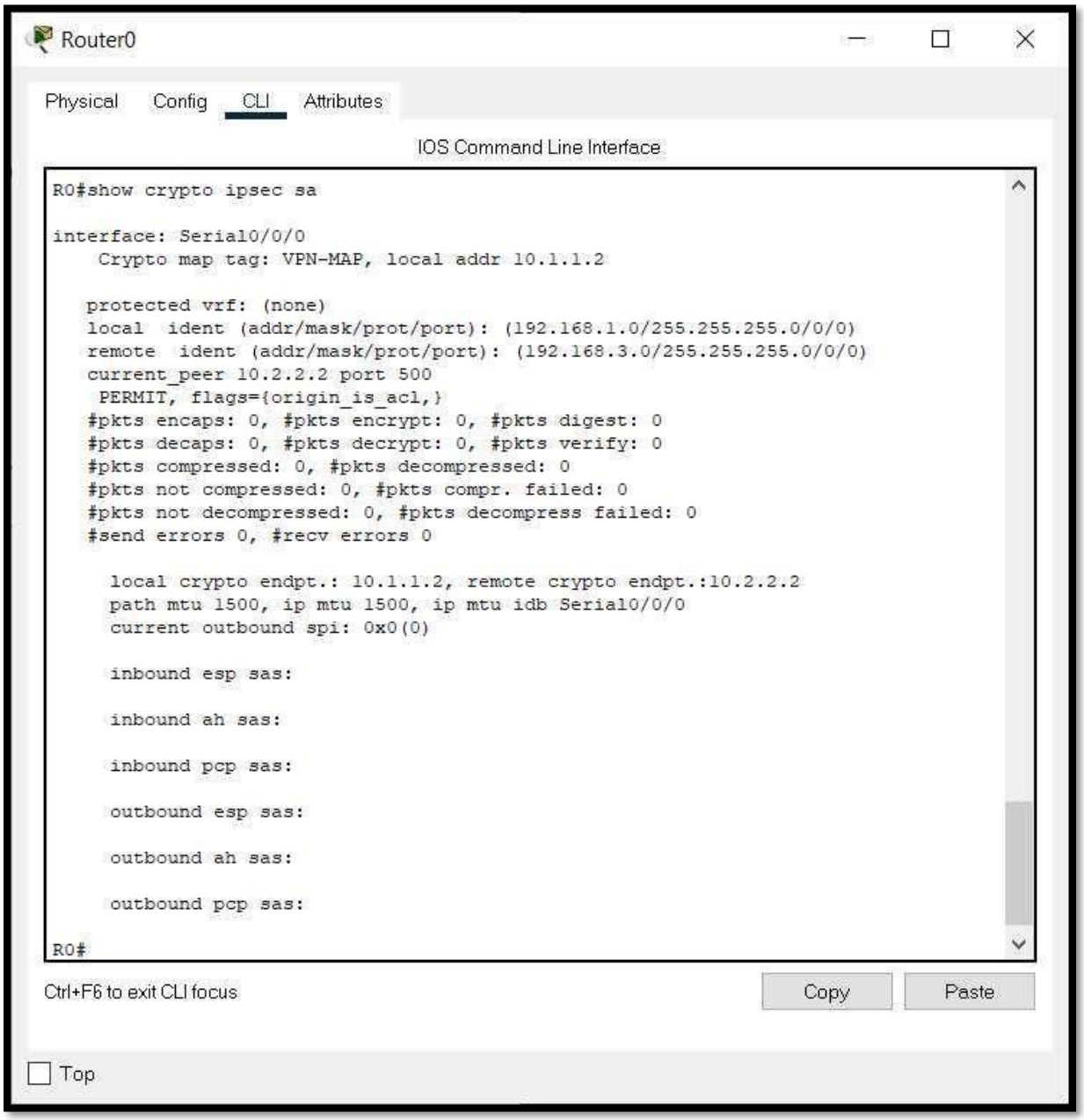
Executing part 2 in Router2



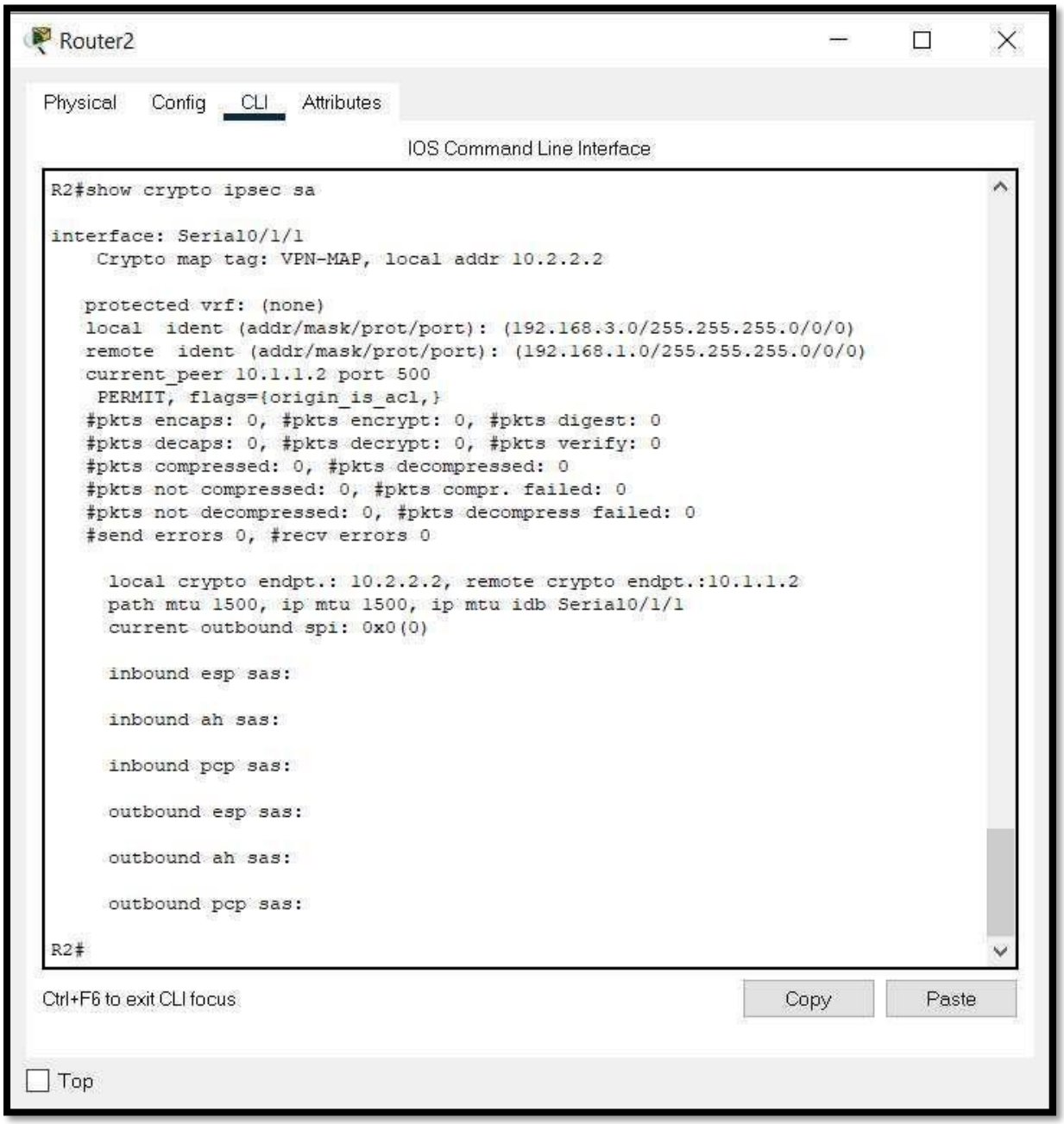
Part 3: Verify the IPSec VPN

1. Verify the tunnel prior to interesting traffic
2. R1# show crypto ipsec sa

Executing part 3 in Router0



Executing part 3 in Router2



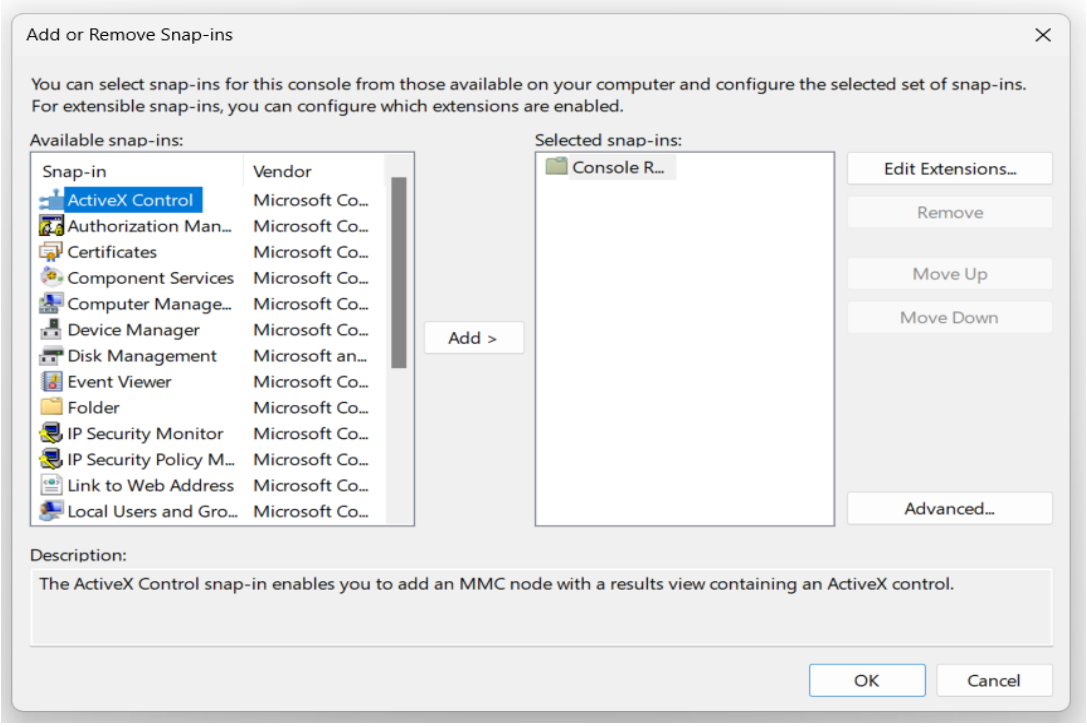
EXPERIMENT-10

Aim:- Create Self Signed Certificate and configure it for website.
Tools / Apparatus: O.S.: Microsoft Windows (any) / Linux / DOS
Packages: Turbo/Borland/GNU - C/C++
Procedure:

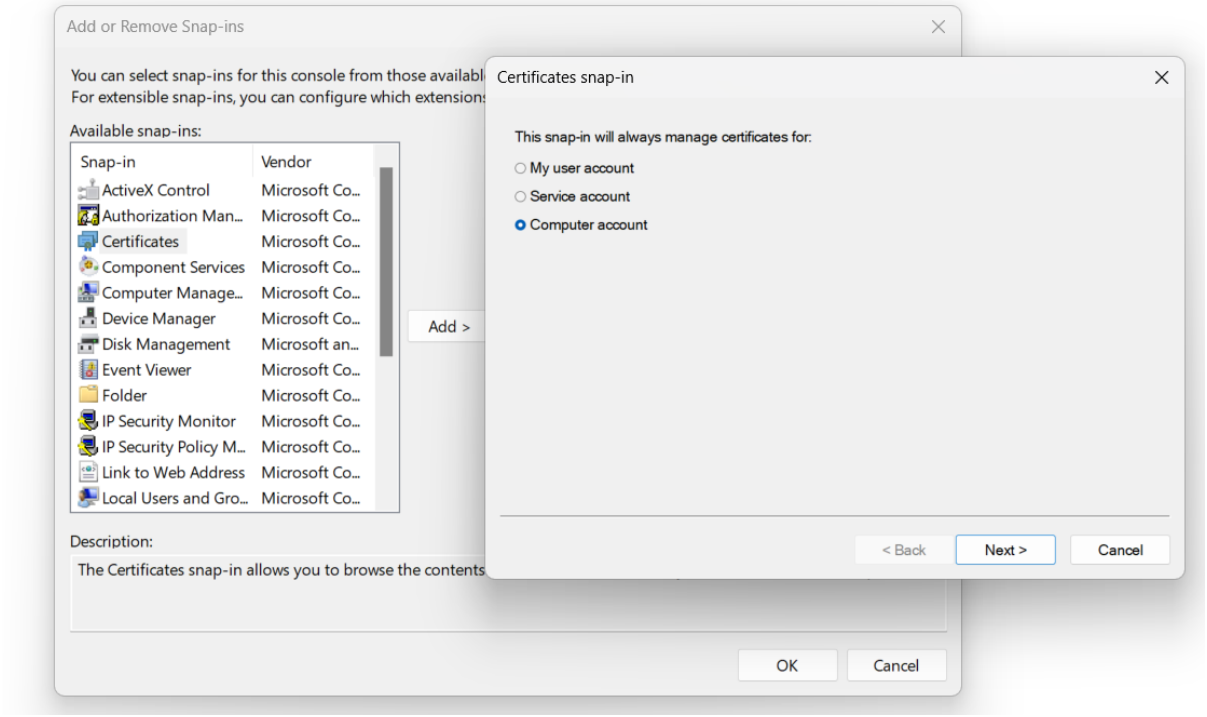
To install a self-signed certificate in the Trusted Root Certification Authorities

1. Open the certificate snap-in.
2. View certificates in the MMC snap-in
3. Select Run from the Start menu, and then enter mmc. The MMC appears.
4. From the File menu, select Add/Remove Snap In.
5. The Add or Remove Snap-ins window appears.
6. From the Available snap-ins list, choose Certificates, then select Add.
7. In the Certificates snap-in window, select Computer account, and then select Next.
8. Optionally, you can select My user account for the current user or Service account for a particular service.
9. In the Select Computer window, leave Local computer selected, and then select Finish.
10. In the Add or Remove Snap-in window, select OK.
11. Open the folder to store the certificate, either the Local Computer or the Current User.
12. Open the Trusted Root Certification Authorities folder.
13. Right-click the Certificates folder and click All Tasks, then click Import.
14. Follow the on-screen wizard instructions to import the RootCA.pfx into the store.

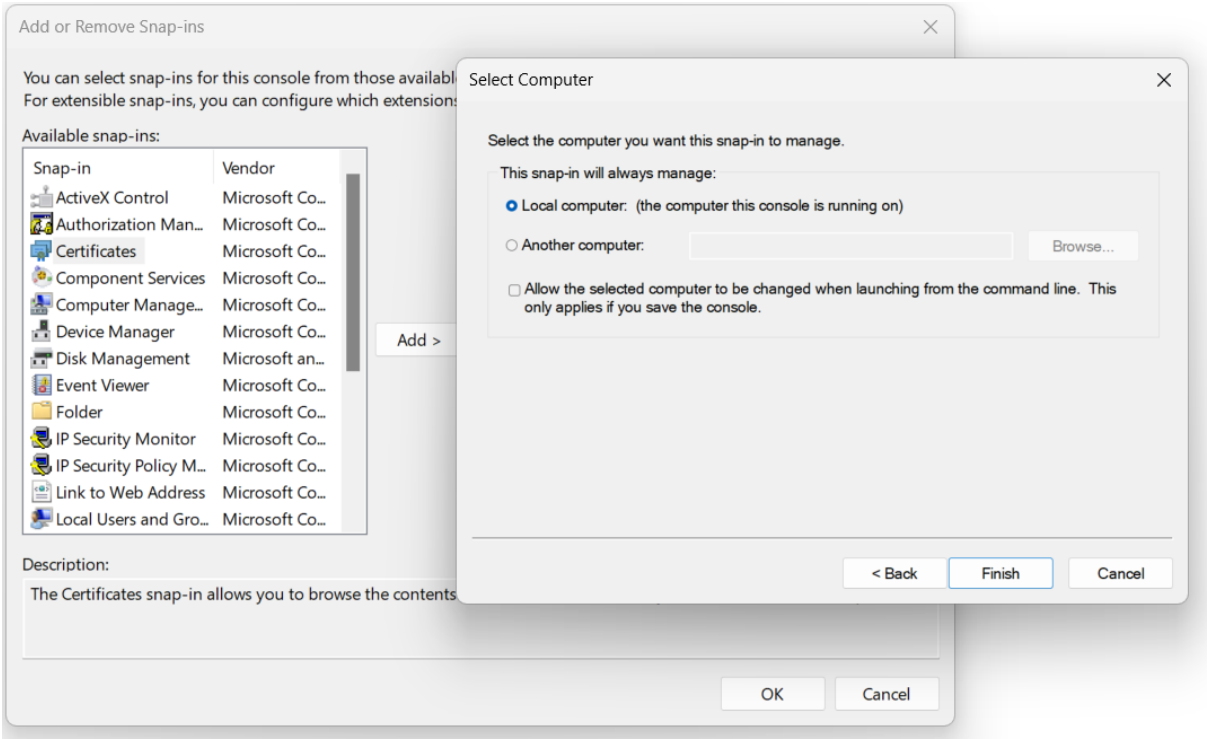
Step 1:-



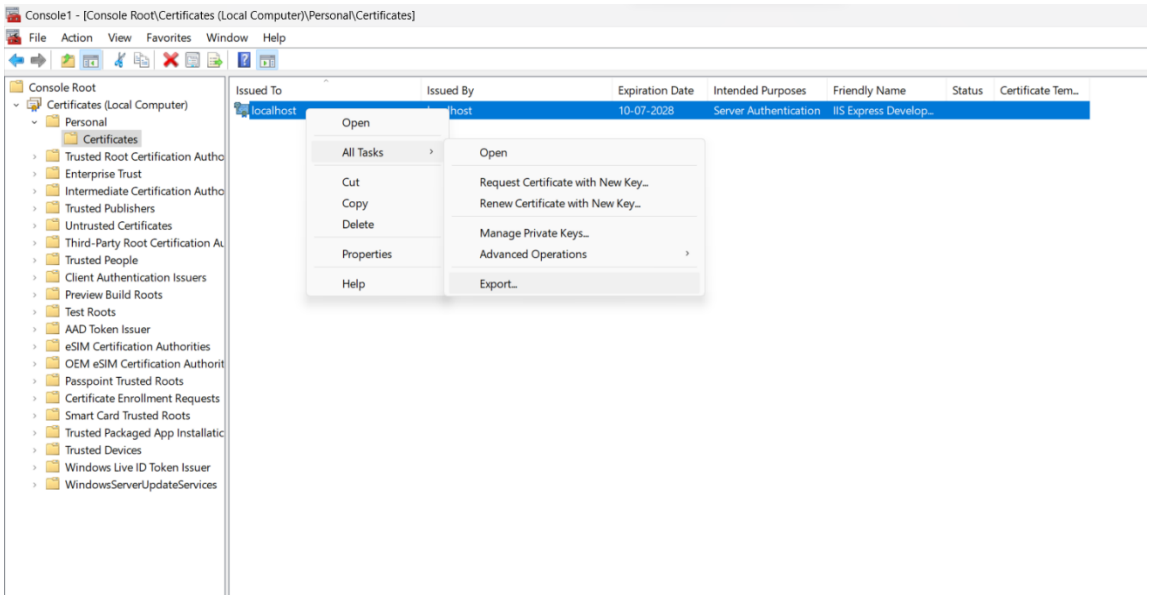
Step 2:-


Add or Remove Snap-ins
You can select snap-ins for this console from those availabl
For extensible snap-ins, you can configure which extension
Available snap-ins:
Snap-in
Vendor
ActiveX Control
Microsoft Co...
Authorization Man...
Microsoft Co...
Certificates
Microsoft Co...
Component Services
Microsoft Co...
Computer Manage...
Microsoft Co...
Device Manager
Microsoft Co...
Disk Management
Microsoft an...
Event Viewer
Microsoft Co...
Folder
Microsoft Co...
IP Security Monitor
Microsoft Co...
IP Security Policy M...
Microsoft Co...
Link to Web Address
Microsoft Co...
Local Users and Gro...
Microsoft Co...
Add >
Description:
The Certificates snap-in allows you to browse the contents
OK
Cancel
Certificates snap-in
This snap-in will always manage certificates for:
My user account
Service account
Computer account
< Back
Next >
Cancel

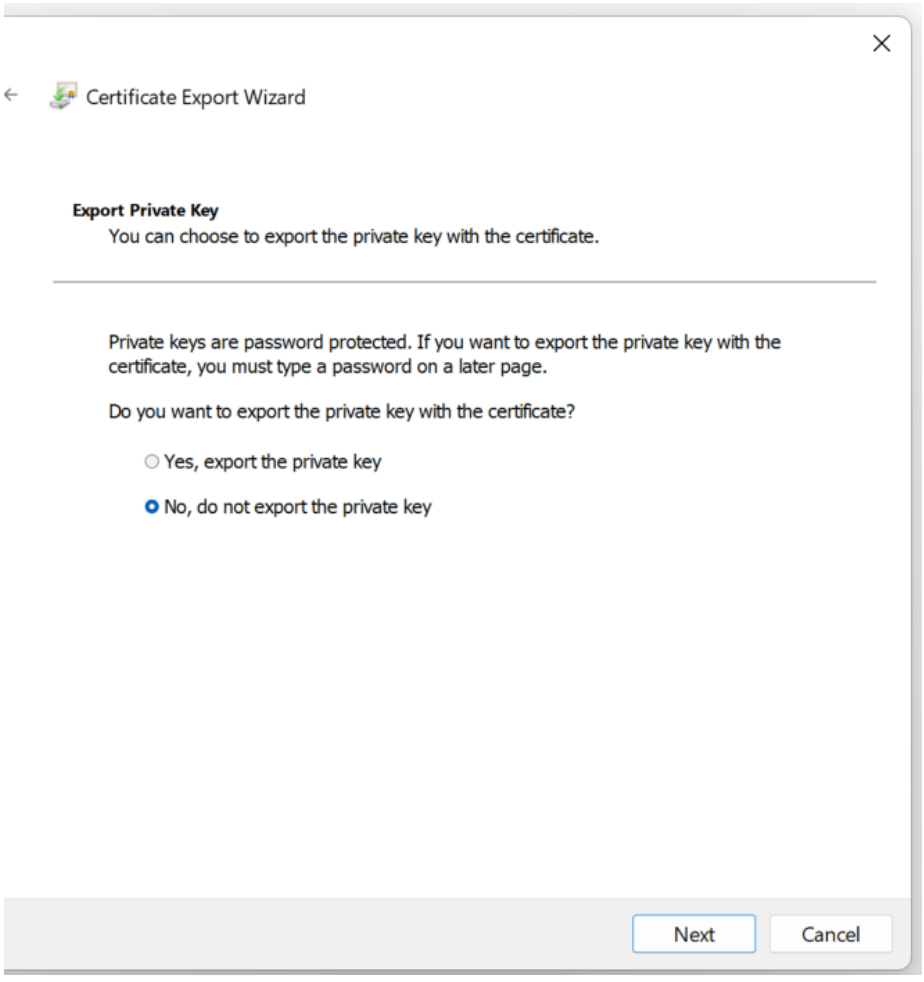
Step 3:-


Add or Remove Snap-ins
You can select snap-ins for this console from those availabl
For extensible snap-ins, you can configure which extension
Available snap-ins:
Snap-in
Vendor
ActiveX Control
Microsoft Co...
Authorization Man...
Microsoft Co...
Certificates
Microsoft Co...
Component Services
Microsoft Co...
Computer Manage...
Microsoft Co...
Device Manager
Microsoft Co...
Disk Management
Microsoft an...
Event Viewer
Microsoft Co...
Folder
Microsoft Co...
IP Security Monitor
Microsoft Co...
IP Security Policy M...
Microsoft Co...
Link to Web Address
Microsoft Co...
Local Users and Gro...
Microsoft Co...
Add >
Description:
The Certificates snap-in allows you to browse the contents
OK
Cancel
Select Computer
Select the computer you want this snap-in to manage.
This snap-in will always manage:
Local computer: (the computer this console is running on)
Another computer:
Browse...
Allow the selected computer to be changed when launching from the command line. This only applies if you save the console.
< Back
Finish
Cancel

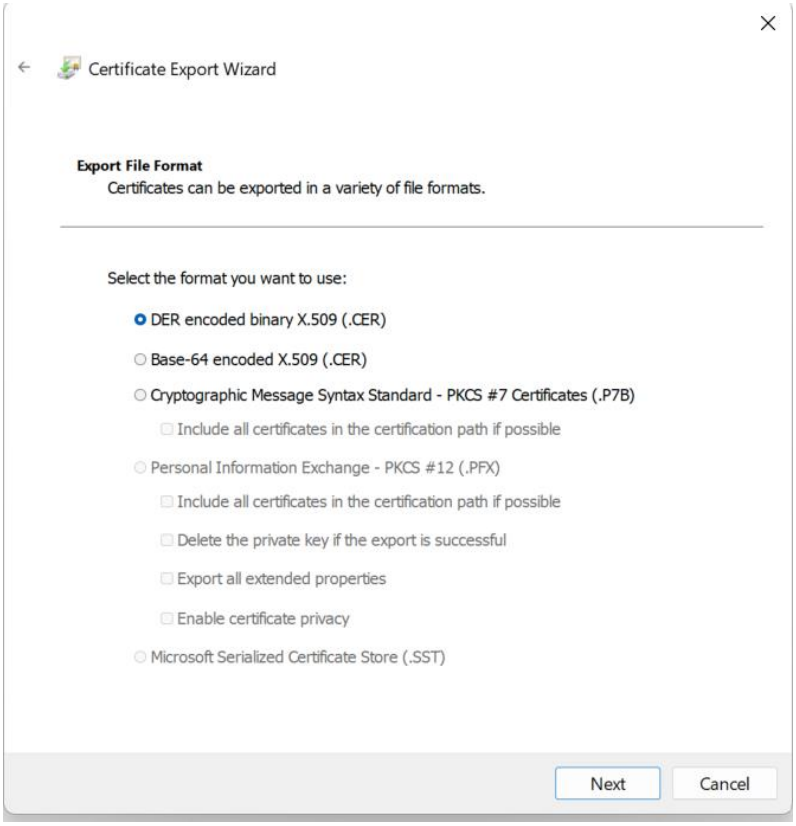
Step 4:-


Console1 - [Console Root\Certificates (Local Computer)\Personal\Certificates]
File Action View Favorites Window Help
Console Root
Certificates (Local Computer)
Personal
Certificates
Trusted Root Certification Autho
Enterprise Trust
Intermediate Certification Autho
Trusted Publishers
Untrusted Certificates
Third-Party Root Certification Au
Trusted People
Client Authentication Issuers
Preview Build Roots
Test Roots
AAD Token Issuer
eSIM Certification Authorities
OEM eSIM Certification Authorit
Passpoint Trusted Roots
Certificate Enrollment Requests
Smart Card Trusted Roots
Trusted Packaged App Installatic
Trusted Devices
Windows Live ID Token Issuer
WindowsServerUpdateServices
Issued To
Issued By
Expiration Date
Intended Purposes
Friendly Name
Status
Certificate Tem...
localhost
10-07-2028
Server Authentication
IIS Express Develop...
Open
All Tasks
Cut
Copy
Delete
Properties
Help
Open
Request Certificate with New Key...
Renew Certificate with New Key...
Manage Private Keys...
Advanced Operations
Export...

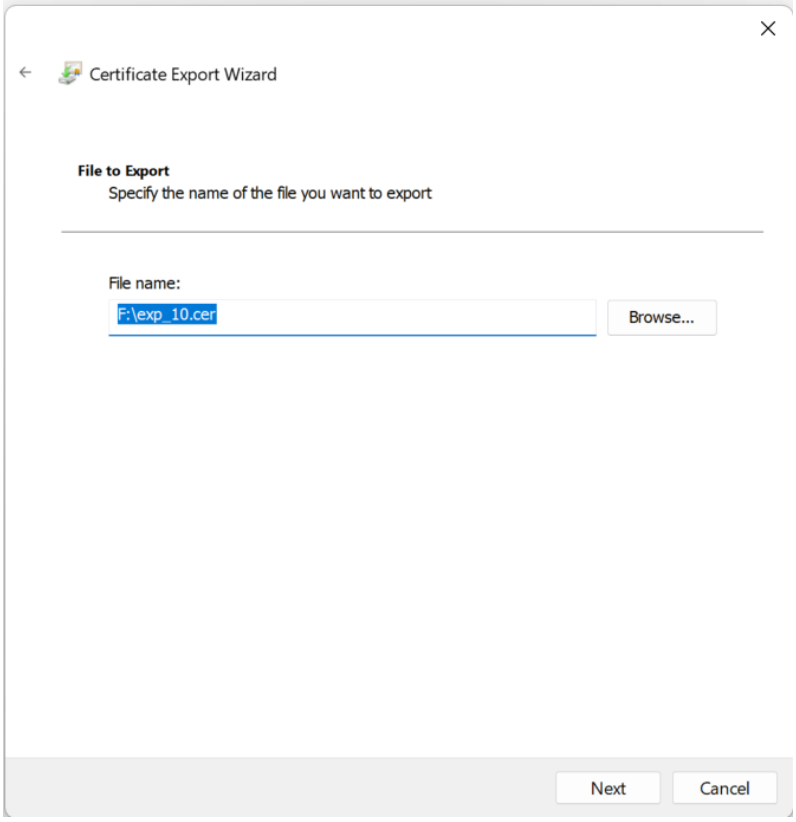
Step 5:-


Certificate Export Wizard
Export Private Key
You can choose to export the private key with the certificate.
Private keys are password protected. If you want to export the private key with the certificate, you must type a password on a later page.
Do you want to export the private key with the certificate?
Yes, export the private key
No, do not export the private key
Next
Cancel

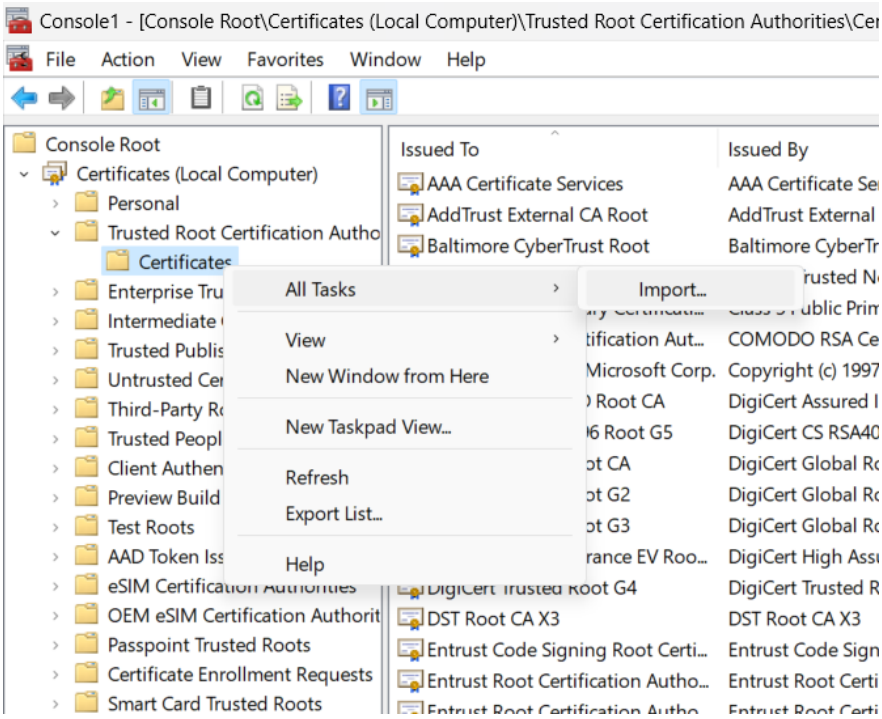
Step 6:-


Certificate Export Wizard
Export File Format
Certificates can be exported in a variety of file formats.
Select the format you want to use:
DER encoded binary X.509 (.CER)
Base-64 encoded X.509 (.CER)
Cryptographic Message Syntax Standard - PKCS #7 Certificates (.P7B)
Include all certificates in the certification path if possible
Personal Information Exchange - PKCS #12 (.PFX)
Include all certificates in the certification path if possible
Delete the private key if the export is successful
Export all extended properties
Enable certificate privacy
Microsoft Serialized Certificate Store (.SST)
Next
Cancel

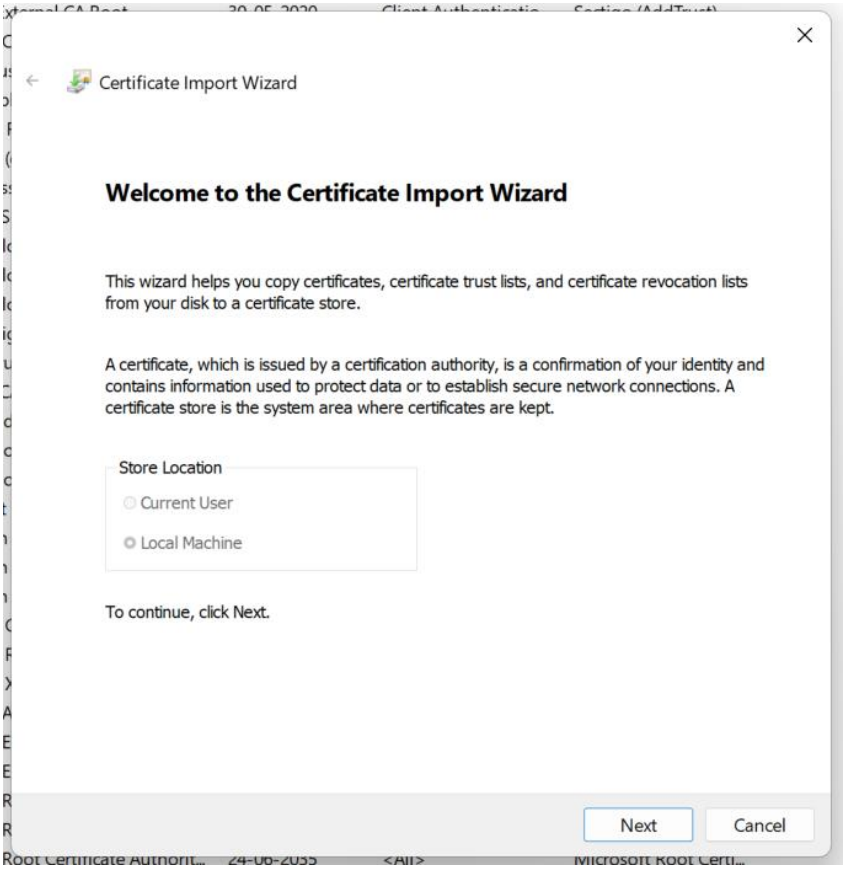
Step 7:-


Certificate Export Wizard
File to Export
Specify the name of the file you want to export
File name:
F:\exp_10.cer
Browse...
Next
Cancel

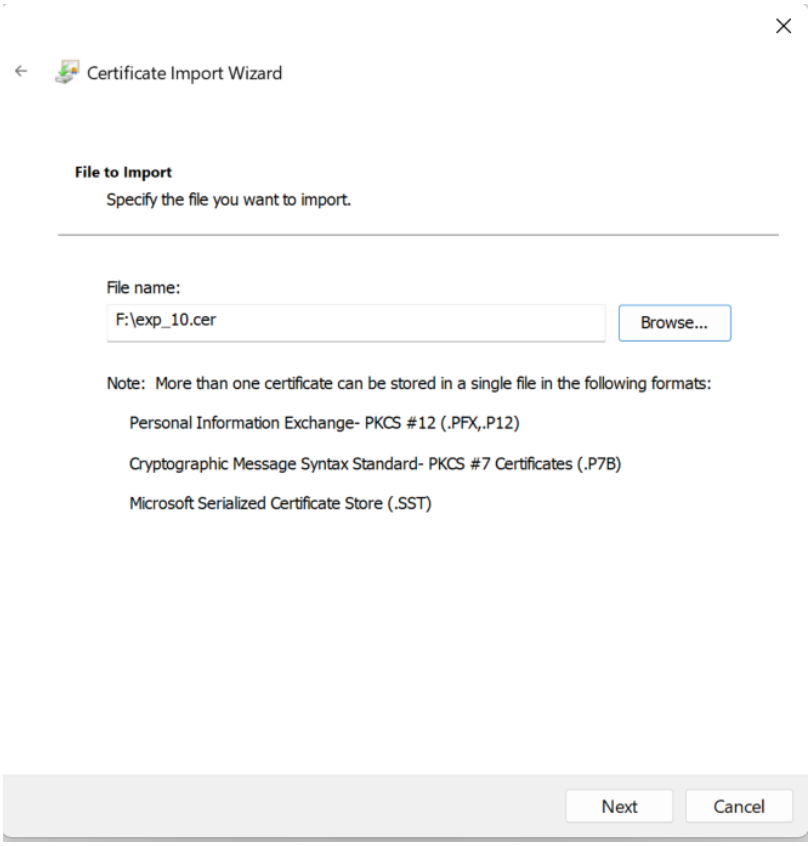
Step 8:-


Console1 - [Console Root\Certificates (Local Computer)\Trusted Root Certification Authorities\Cer
File Action View Favorites Window Help
Console Root
Certificates (Local Computer)
Personal
Trusted Root Certification Autho
Certificates
Enterprise Tru
Intermediate
Trusted Publis
Untrusted Cer
Third-Party R
Trusted Peopl
Client Authen
Preview Build
Test Roots
AAD Token Iss
eSIM Certificat
OEM eSIM Certification Authorit
Passpoint Trusted Roots
Certificate Enrollment Requests
Smart Card Trusted Roots
Issued To
Issued By
AAA Certificate Services
AAA Certificate Se
AddTrust External CA Root
AddTrust External
Baltimore CyberTrust Root
Baltimore CyberTr
All Tasks
Import...
View
New Window from Here
New Taskpad View...
Refresh
Export List...
Help
ification Aut...
COMODO RSA Ce
Microsoft Corp.
Copyright (c) 1997
Root CA
DigiCert Assured I
6 Root G5
DigiCert CS RSA40
ot CA
DigiCert Global R
ot G2
DigiCert Global R
ot G3
DigiCert Global R
rance EV Roo...
DigiCert High Assu
DigiCert Trusted Root G4
DigiCert Trusted R
DST Root CA X3
DST Root CA X3
Entrust Code Signing Root Certi...
Entrust Code Sign
Entrust Root Certification Autho...
Entrust Root Certi

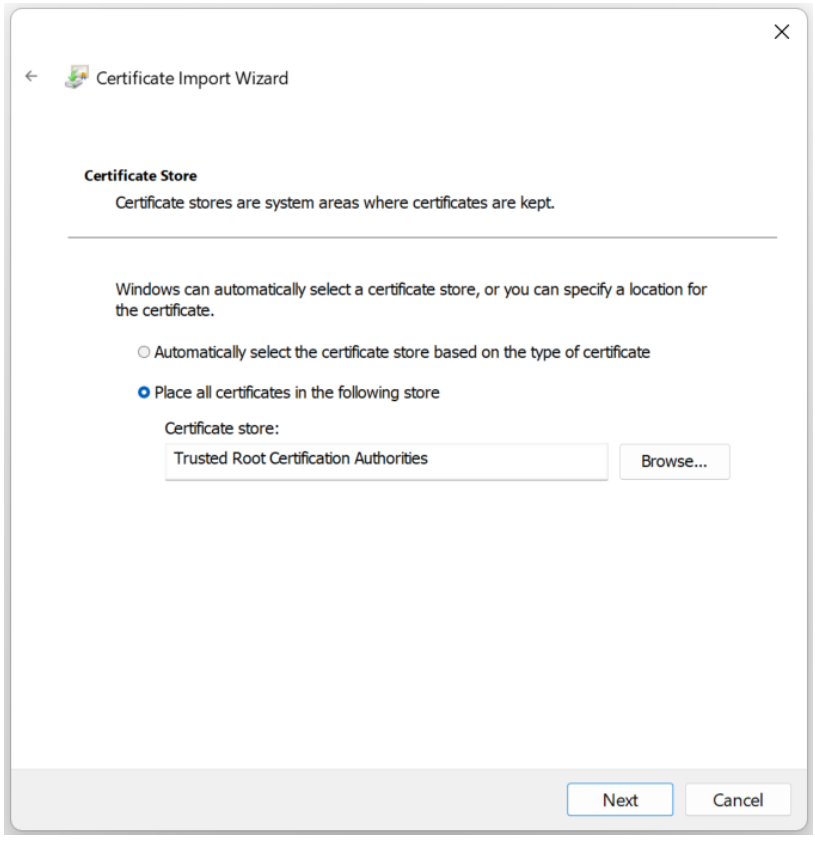
Step 9:-


Certificate Import Wizard
Welcome to the Certificate Import Wizard
This wizard helps you copy certificates, certificate trust lists, and certificate revocation lists from your disk to a certificate store.
A certificate, which is issued by a certification authority, is a confirmation of your identity and contains information used to protect data or to establish secure network connections. A certificate store is the system area where certificates are kept.
Store Location
Current User
Local Machine
To continue, click Next.
Next
Cancel

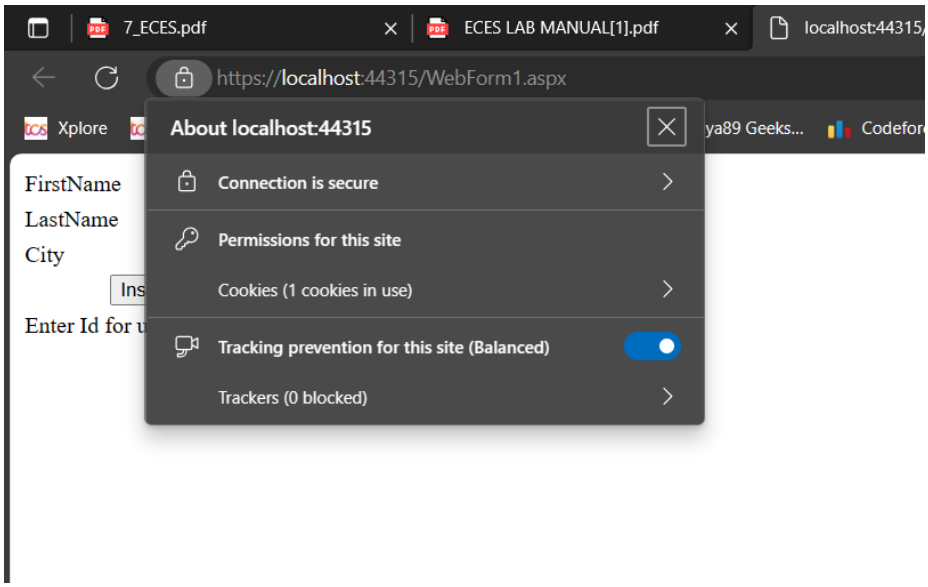
Step 10:-



Step 11:-



Output:-


7_ECES.pdf
ECES LAB MANUAL[1].pdf
localhost:44315/
https://localhost:44315/WebForm1.aspx
Xplore
About localhost:44315
Connection is secure
Permissions for this site
Cookies (1 cookies in use)
Tracking prevention for this site (Balanced)
Trackers (0 blocked)
FirstName
LastName
City
Ins
Enter Id for u