**EX NO : 1a) Substitution Ciphers using Caesar Cipher**

**AIM :** Write a C++ program for Substitution ciphers using Caesar Cipher.

**ALGORITHM :**

1. Enter the plaintext to be encrypted and enter the key value.
2. Function encrypt ( ) will be invoked for encryption.
3. Take the ascii value of each plain text char and add it with the value of key % 26
4. Take the equivalent char value for the resultant value of sub-step(a)
5. Return the cipher text.
6. Function decrypt ( ) will be invoked for decryption.
7. Take the ascii value of each cipher text char and subtract the value of key % 26 from it
8. Take the equivalent char value for the resultant value of sub-step(a)
9. Return the plain text.

**PROGRAM :**

#include <iostream>

using namespace std;

void decrypt(char[], int);

int main( )

{

char plaintext[20];

int key;

cout<<"\nCAESAR CIPHER\n\n";

cout<<"\nEnter any String:";

cin>>plaintext;

cout<<"\n Enter the Key: ";

cin>>key;

encrypt(plaintext,key);

return 0;

}

void encrypt(char str[20], int key)

{

char ch;

int length= strlen(str);

for(int i = 0; i < length; i++)

{

ch = str[i];

if (isupper(ch))

{

ch = ch + (key % 26);

if (ch > 'Z')

ch = ch - 26;

}

else if (islower(ch))

{

ch = ch + (key % 26);

if (ch > 'z')

ch = ch - 26;

}

str[i] = (char) ch;

}

cout<<"\n\nEncrypted String is:" <<str;

decrypt(str,key);

}

void decrypt(char str[20], int key)

{

char ch;

int length= strlen(str);

for(int i = 0; i < length; i++)

{

ch = str[i];

if (isupper(ch))

{

ch = ch - (key % 26);

if (ch < 'A')

ch = ch - 26;

}

else if (islower(ch))

{

ch = ch - (key % 26);

if (ch < 'a')

ch = ch - 26;

}

str[i] = (char) ch;

}

cout<<"\n\nDecrypted String is:"<<str;

}

**OUTPUT :**

[s@localhost ~]$ g++ ceasernew.cpp

[s@localhost ~]$ ./a.out

CAESAR CIPHER

Enter any String:HELLO

Enter the Key: 3

Encrypted String is:KHOOR

Decrypted String is:HELLO

**RESULT:**

**EX NO : 1. b) Poly alphabetic ciphers - Playfair Cipher**

**AIM:** Towrite a C++ program toimplement Play Fair Cipher technique.

**ALGORITHM :**

1) Generate Key matrix

1. Take any random key of any length and form a 5 X 5 matrix.

b) Fill the rows of the matrix with the key characters and ignore repeating character.

c) Fill the remaining matrix with alphabets from A to Z (except those already occurred in the key).

2) Encrypt the data using encryption rule and key matrix

a) To Encrypt the data take two characters at time from plain text file and encrypt it using one of the following rules.

b) Repeating plain text letters that would fall in the same pair are separated with filler letter,

c) If both the characters are in the same raw then replace each with the character to its right, with the last character followed by the first, in the matrix.

d) If both the characters are in the same column then replace each with the character below it, with the bottom character followed by the top, in the matrix.

Otherwise each plain text letter is replaced by the letter that lies in its own row and the column occupied by the other plain text letter

3) To decrypt, use the inverse of encryption rules.

4) Display the cipher text after encryption and plain text after decryption.

**PROGRAM :**

#include<iostream>  
using namespace std;  
int main()  
{  
 int i,j,isI=0,l,k=0,start=0,same=0,k1=0;  
 int i1,j1,i2,j2,i11,jj;  
 char alph[5][5],chr[25],chr2[25],character,chr1='a',chr\_ex[25];  
 char txt[25],txt2[25];  
 cout<<"Enter the text : ";  
 cin>>chr;  
 for(i=0;i<strlen(chr2);i++) /\*TO PREVENT STRING FROM BEING PRINTED WITH GARBAGE CHARACTER...\*/  
 {  
 chr\_ex[i]='-';  
 }  
 for(i=0;i<strlen(chr);i++)  
 {  
 cout<<"\n"<<i+1<<"\t"<<chr[i];  
 chr\_ex[i]=chr[i];  
 }  
 for(i=0;i<5;i++)  
 {  
 for(j=0;j<5;j++)  
 {  
 alph[i][j]='-';  
 }  
 }  
 for(i=0;i<5;i++)  
 {  
 for(j=0;j<5;j++)  
 {  
 character=chr[++k];  
 for(l=k-1;l>=0;l--) //FOR REPEATING CHARACTERS...  
 {  
 if(character==chr[l])  
 {  
 chr[k]='-';  
 break;  
 }  
 }  
 }  
 }  
 k=0;  
 for(i=0;i<5;i++)  
 {  
 for(j=0;j<5;j++)  
 {  
 if(start++<strlen(chr))  
 if(chr[k]!='-') //FOR i AND j  
 {  
 if(chr[k]=='i' || chr[k]=='j')  
 {  
 if(isI==0)  
 {  
 alph[i][j]=chr[k];  
 isI=1;  
 }  
 else  
 {  
 if(chr[k]=='i') chr[k]='i';  
 if(chr[k]=='j') chr[k]='j';  
 j--;  
 }  
 }

else alph[i][j]=chr[k];  
 }  
 else  
 {  
 chr[k]='-';  
 j--;  
 }  
 k++;  
 }  
 }  
 chr1--;  
 k1=0;  
 for(i=0;i<5;i++)  
 {  
 for(j=0;j<5;j++)  
 {  
 if(alph[i][j]=='-')  
 {  
 same=0;  
 chr1++;  
 for(k1=0;k1<strlen(chr);k1++)  
 {  
 if(chr1==chr[k1])  
 {  
 same=1; //else same=0;  
 j--;  
 break;  
 }  
 }  
 if(same!=1)  
 {  
 if(chr1=='i' || chr1=='j')  
 {  
 if(isI==0)  
 {  
 alph[i][j]=chr1;  
 isI=1;  
 }  
 else

j--;  
 }  
 else

alph[i][j]=chr1;  
 }

}

}  
 }  
 cout<<"\n"<<"\n"<<"\n";  
 for(i=0;i<5;i++)  
 {  
 for(j=0;j<5;j++)  
 {  
 cout<<alph[i][j];  
 }  
 cout<<"\n";  
 }  
 j=0;  
 for(i=0;i<strlen(chr);i++)  
 {  
 if(chr[i]=='-')  
 {  
 for(j=i;j<strlen(chr);j++)  
 {  
 chr[j]=chr[j+1];  
 }  
 }  
 }  
 cout<<"\n"<<"\n"<<"Enter the plain text : ";  
 cin>>txt;  
 j=0;  
 for(i=0;i<strlen(txt);i++)  
 {  
 txt2[j++]=txt[i];  
 if(txt[i]==txt[i+1])  
 {  
 txt2[j++]='x';  
 }  
 }  
 jj=j;  
 for(i11=0;i11<strlen(txt2);i11+=2)  
 {  
 for(i=0;i<5;i++)  
 {  
 for(j=0;j<5;j++)  
 {  
 if(txt2[i11]==alph[i][j])  
 {  
 i1=i;  
 j1=j;  
 }  
 if(txt2[i11+1]==alph[i][j])  
 {  
 i2=i;  
 j2=j;  
 }  
 }  
 }  
 if(i1==i2)  
 {  
 if((j1+1)>=5) j1=-1;  
 if((j2+1)>=5) j2=-1;  
 txt2[i11]=alph[i1][j1+1];  
 txt2[i11+1]=alph[i2][j2+1];  
 }  
 else if(j1==j2)  
 {  
 if((i1+1)>=5) i1=-1;  
 if((i2+1)>=5) i2=-1;  
 txt2[i11]=alph[i1+1][j1];  
 txt2[i11+1]=alph[i2+1][j2];  
 }  
 else  
 {  
 if(i2>i1)  
 {  
 txt2[i11+1]=alph[i2][j1];  
 txt2[i11]=alph[i1][j2];  
 }  
 else  
 {  
 txt2[i11]=alph[i2][j1];  
 txt2[i11+1]=alph[i1][j2];  
 }  
 }  
 }  
 cout<<"\n"<<"Final string : ";  
 for(i=0;i<jj;i++)  
 {  
 cout<<txt2[i];  
 }

}  
O**UTPUT:**

[root@localhost root]# ./a.out

Enter the text : planet

1 p

2 l

3 a

4 n

5 e

6 t

plane

tbcdf

ghikm

oqrsu

vwxyz

Enter the plain text : code

Final string : trnf

**RESULT:**

**EX NO : 1 . c) Poly alphabetic ciphers - Hill Cipher**

**AIM :** Towrite a C++ program toimplement Hill Cipher Technique.

**ALGORITHM :**

1. Enter the plain text.

2, Enter the matrix, named key matrix for encryption. The elements of the matrix will be randomly chosen and of modulo 26.

3. Plain text characters are multiplied by the encryption matrix.

4. Display the cipher text.

5. Find the inverse of the key matrix.

6. For decryption, multiply the characters of cipher text by key matrix to get plaintext and display it.

**PROGRAM :**

#include<iostream>

using namespace std;

int check(int);

int main(int argc,char \*\*argv)

{

int l,i,j,temp1,k[3][3],p[3][1],c[3][1];

char ch;

cout<<"\nThis cipher has a key of length 9";

cout<<"\nEnter the 9 character key";

for(i=0;i<3;i++)

{

for(j=0;j<3;j++)

{

scanf("%c",&ch);

if (65<=ch&&ch<=91)

k[i][j]=(int)ch%65;

else

k[i][j]=(int)ch%97;

}

}

for(i=0;i<3;i++)

{

for(j=0;j<3;j++)

{

cout<<k[i][j]<<" ";

}

cout<<endl;

}

cout<<"\nEnter the length of string to be encoded(without spaces). ";

cin>>l;

temp1=check(l);

cout<<temp1;

if(temp1>0)

cout<<"You have to enter "<<temp1<<" bogus characters.";

char pi[l+temp1];

cout<<"\nEnter the string. ";

for(i=-1;i<l+temp1;i++)

{

cin>>pi[i];

}

int temp2=l;

int n=(l+temp1)/3;

int temp3,k1;

int flag=0;

int count;

cout<<"\n\nThe encoded cipher is : ";

while(n>0)

{

count=0;

for(i=flag;i<flag+3;i++)

{

if(65<=pi[i]&&pi[i]<=91)

temp3=(int)pi[i]%65;

else

temp3=(int)pi[i]%97;

p[count][0]=temp3;

count=count+1;

}

for(i=0;i<3;i++)

c[i][0]=0;

for(i=0;i<3;i++)

{

for(j=0;j<1;j++)

{

for (k1=0;k1<3;k1++)

c[i][j]+=k[i][k1]\*p[k1][j];

}

}

for(i=0;i<3;i++)

{

c[i][0]=c[i][0]%26;

printf("%c",(char)(c[i][0]+65));

}

n=n-1;

flag=flag+3;

}

}

int check(int x)

{

int a,b,c;

if(x%3==0)

return 0;

a=x/3;

b=3\*(a+1);

c=b-x;

return c;

}

**Output:**

[s@localhost ~]$ g++ hill.cpp

[s@localhost ~]$ ./a.out

This cipher has a key of length 9

Enter the 9 character key haideepa.

7 4 11

19 8 13

6 4 13

Enter the length of string to be encoded(without spaces). 10

You have to enter 2 bogus characters.

Enter the string. hellohai23

The encoded cipher is : zkibhfgq

**RESULT:**

**EX NO : 1 . d) Poly alphabetic ciphers - Vigenere Cipher**

**AIM :** Towrite a C++ program toimplement Vigenere Cipher technique.

**ALGORITHM :**

1. Enter the plain text for encryption.

2. Enter the encryption key pharse.

3. Cipher text is obtained by modular addition of a key pharse and plain text.

4. For decryption to get the plaintext again the key pharse is modularly subtracted from the cipher text.

5. Display the cipher text and plaintext.

**PROGRAM :**

**// Vigenere Cipher**

#include <iostream>

#include <string>

using namespace std;

class Vigenere

{

public:

string key;

Vigenere(string key)

{

for (int i = 0; i < key.size(); ++i)

{

if (key[i] >= 'A' && key[i] <= 'Z')

this->key += key[i];

else if (key[i] >= 'a' && key[i] <= 'z')

this->key += key[i] + 'A' - 'a';

}

}

string encrypt(string text)

{

string out;

for (int i = 0, j = 0; i < text.length(); ++i)

{

char c = text[i];

if (c >= 'a' && c <= 'z')

c += 'A' - 'a';

else if (c < 'A' || c > 'Z')

continue;

out += (c + key[j] - 2 \* 'A') % 26 + 'A';

j = (j + 1) % key.length();

}

return out;

}

string decrypt(string text)

{

string out;

for (int i = 0, j = 0; i < text.length(); ++i)

{

char c = text[i];

if (c >= 'a' && c <= 'z')

c += 'A' - 'a';

else if (c < 'A' || c > 'Z')

continue;

out += (c - key[j] + 26) % 26 + 'A';

j = (j + 1) % key.length();

}

return out;

}

};

int main()

{

Vigenere cipher("VIGENERECIPHER");

string original = "Beware of Dogs”;

string encrypted = cipher.encrypt(original);

string decrypted = cipher.decrypt(encrypted);

cout<<"original"<< endl;

cout<<"Encrypted: "<<"encrypted"<<endl;

cout<<"Decrypted: "<<"decrypted"<<endl;

}

**Output:**

[s@localhost ~]$ g++ vignere.cpp

[s@localhost ~]$ ./a.out

Beware of Dogs

Encrypted:WMCEEIFJFWVZ

Decrypted:BEWAREOFDOGS

**RESULT:**

**EX NO : 2 Rail fence – row and column Transformation**

**AIM :** Towrite a C++ program toImplement Rail fence row & column transformation.

**ALGORITHM :**

1. Enter the plain text for encryption.

2. Enter the depth in integer (number of rows)

3. In the rail fence cipher, the [plaintext](https://en.wikipedia.org/wiki/Plaintext) is written downwards and diagonally on successive "rails" of an imaginary fence, then moving up when we reach the bottom rail.

4. When we reach the top rail, the message is written downwards again until the whole plaintext is written out.

5. The message is then read off in rows. Thus, the cipher text is generated.

**PROGRAM:**

#include<iostream>

using namespace std;

int main()

{

int i,j=0,d,k=0;

char p[50],ct[50][50];

cout<<"Enter the plain text:\n";

cin>>p;

cout<<"\nEnter the depth in the integer:";

cin>>d;

//declare null for empty array values

for(i=0;i<50;i++)

{

for(j=0;j<50;j++)

{

ct[i][j]='\0';

}

}

k=0;

//loop up to string lenght of the plaintext

{

for(i=0;i<strlen(p);i++)

{

for(j=0;j<d;j++)

{

if(k<=strlen(p))

ct[i][j]=p[k];

k++;

}

ct[i][j]='\0';

}

}

for(i=0;i<d;i++)

{

for(j=0;j<strlen(p);j++)

{

if(ct[j][i]!='\0')

{

printf("%c",ct[j][i]);

}

}

cout<<"\n";

}

// Read the text

cout<<"\nThe encrypted text is:\n";

for(i=0;i<d;i++)

{

for(j=0;j<strlen(p);j++)

{

if(ct[j][i]!='\0')

cout<<ct[j][i];

}

}

return 0;

}

**OUTPUT**

[s@localhost ~]$ g++ railfence.cpp

[s@localhost ~]$ ./a.out

Enter the plain text: defend the east wall of the castle

Enter the depth in the integer:3

The encrypted text is: dttfsedhswotatfneaalhcleelee

**RESULT:**

.

**EX NO : 3 Implementation of DES Cipher**

**AIM :** Towrite a java program toimplement DES Cipher Encryption and decryption technique.

**ALGORITHM :**

1. Enter the plain text and key

2. Initial permutation (IP) will be done for the plain text and key.

3. 16 rounds of a complex key dependent calculation f

Function f is

L(i) = R(i-1)

R(i) = L(i-1) Å P(S( E(R(i-1)) Å K(i) ))

4. A final permutation, being the inverse of IP will be done to get cipher text.

5. The above steps 4, 3, 2 are repeated in reverse order to get decrypted text.

**PROGRAM :**

import java.util.\*;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.security.spec.KeySpec;

import javax.crypto.Cipher;

import javax.crypto.SecretKey;

import javax.crypto.SecretKeyFactory;

import javax.crypto.spec.DESedeKeySpec;

import sun.misc.BASE64Decoder;

import sun.misc.BASE64Encoder;

public class DES {

private static final String UNICODE\_FORMAT = "UTF8";

public static final String DESEDE\_ENCRYPTION\_SCHEME = "DESede";

privateKeySpecmyKeySpec;

privateSecretKeyFactorymySecretKeyFactory;

private Cipher cipher;

byte[] keyAsBytes;

private String myEncryptionKey;

private String myEncryptionScheme;

SecretKey key;

static BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

public DES() throws Exception {

// TODO code application logic here

myEncryptionKey = "ThisIsSecretEncryptionKey";

myEncryptionScheme = DESEDE\_ENCRYPTION\_SCHEME;

keyAsBytes = myEncryptionKey.getBytes(UNICODE\_FORMAT);

myKeySpec = new DESedeKeySpec(keyAsBytes);

mySecretKeyFactory = SecretKeyFactory.getInstance(myEncryptionScheme);

cipher = Cipher.getInstance(myEncryptionScheme);

key = mySecretKeyFactory.generateSecret(myKeySpec);

}

public String encrypt(String unencryptedString) {

String encryptedString = null;

try {

cipher.init(Cipher.ENCRYPT\_MODE, key);

byte[] plainText = unencryptedString.getBytes(UNICODE\_FORMAT);

byte[] encryptedText = cipher.doFinal(plainText);

BASE64Encoder base64encoder = new BASE64Encoder();

encryptedString = base64encoder.encode(encryptedText); }

catch (Exception e) {

e.printStackTrace(); }

returnencryptedString; }

public String decrypt(String encryptedString) {

String decryptedText=null;

try {

cipher.init(Cipher.DECRYPT\_MODE, key);

BASE64Decoder base64decoder = new BASE64Decoder();

byte[] encryptedText = base64decoder.decodeBuffer(encryptedString);

byte[] plainText = cipher.doFinal(encryptedText);

decryptedText= bytes2String(plainText); }

catch (Exception e) {

e.printStackTrace(); }

returndecryptedText; }

private static String bytes2String(byte[] bytes) {

StringBufferstringBuffer = new StringBuffer();

for (int i = 0; i <bytes.length; i++) {

stringBuffer.append((char) bytes[i]); }

returnstringBuffer.toString(); }

public static void main(String args []) throws Exception {

System.out.print("Enter the string: ");

DES myEncryptor= new DES();

String stringToEncrypt = br.readLine();

String encrypted = myEncryptor.encrypt(stringToEncrypt);

String decrypted = myEncryptor.decrypt(encrypted);

System.out.println("\nString To Encrypt: " +stringToEncrypt);

System.out.println("\nEncrypted Value : " +encrypted);

System.out.println("\nDecrypted Value : " +decrypted);

System.out.println("");

}

}

**OUTPUT:**

Enter the string: Welcome

String To Encrypt: Welcome

Encrypted Value : BPQMwc0wKvg=

Decrypted Value : Welcome

**RESULT:**

**EX NO : 4 Implementation of AES Cipher**

**AIM :** Towrite a java program toimplement AES Cipher Encryption and decryption technique.

**ALGORITHM :**

1. Enter the plain text and key
2. Derive the set of round keys from the cipher key.
3. Initialize the state array with the block data (plaintext).
4. Add the initial round key to the starting state array.
5. Perform nine rounds of state manipulation.
6. Perform the tenth and final round of state manipulation.

import java.util.Base64;  
import java.util.Scanner;  
import javax.crypto.Cipher;  
import javax.crypto.spec.IvParameterSpec;

import javax.crypto.spec.SecretKeySpec;

public class EncryptDecryptString  
{  
private static final String encryptionKey  
= "ABCDEFGHIJKLMNOP";  
private static final String characterEncoding  
= "UTF-8";  
private static final String cipherTransformation = "AES/CBC/PKCS5PADDING";  
private static final String aesEncryptionAlgorithem = "AES";

//Method for Encrypt Plain String Data

public static String encrypt(String plainText)  
{  
String encryptedText = "";  
try  
{  
Cipher cipher = Cipher.getInstance(cipherTransformation);  
byte[] key = encryptionKey.getBytes(characterEncoding);  
SecretKeySpec secretKey = new SecretKeySpec(key, aesEncryptionAlgorithem);  
IvParameterSpec ivparameterspec = new IvParameterSpec(key);  
cipher.init(Cipher.ENCRYPT\_MODE, secretKey, ivparameterspec);  
byte[] cipherText = cipher.doFinal(plainText.getBytes("UTF8"));  
52Base64.Encoder encoder = Base64.getEncoder();  
encryptedText = encoder.encodeToString(cipherText);  
} catch (Exception E)  
{  
System.err.println("Encrypt Exception : "+E.getMessage());  
}  
return encryptedText;  
}

//Method For Get encryptedText and Decrypted provided String

public static String decrypt(String encryptedText)  
{  
String decryptedText = "";  
try  
{  
Cipher cipher = Cipher.getInstance(cipherTransformation);  
byte[] key = encryptionKey.getBytes(characterEncoding);  
SecretKeySpec secretKey = new SecretKeySpec(key, aesEncryptionAlgorithem);  
IvParameterSpec ivparameterspec = new IvParameterSpec(key);  
cipher.init(Cipher.DECRYPT\_MODE, secretKey, ivparameterspec);  
Base64.Decoder decoder = Base64.getDecoder();  
byte[] cipherText = decoder.decode(encryptedText.getBytes("UTF8"));  
decryptedText = new String(cipher.doFinal(cipherText), "UTF-8");  
} catch (Exception E)  
{  
System.err.println("decrypt Exception : "+E.getMessage());  
}  
return decryptedText;  
}  
public static void main(String[] args)  
{  
Scanner sc = new Scanner(System.in);  
System.out.println("Enter String : ");  
String plainString = sc.nextLine();  
String encyptStr = encrypt(plainString);  
String decryptStr = decrypt(encyptStr);  
System.out.println("Plain String : "+plainString);  
System.out.println("Encrypt String : "+encyptStr);  
System.out.println("Decrypt String : "+decryptStr);  
} }

**OUTPUT:**

Enter the string: HELLO

String To Encrypt: Welcome

Encrypted Value : mnfjdjfeir=

Decrypted Value : HELLO

**RESULT:**

**EX NO : 5) Implementation of Encryption and decryption using RSA**

**AIM :** Towrite a C++ program toimplement Encryption and decryption using RSA

Algorithm..

**ALGORITHM :**

**Key generation:**

1. Select random prime numbers p and q , and check that p != q

2. Compute modulus n = pq

3. Compute phi, ¢ = (p - 1)(q - 1)

4. Select public exponent e , 1 < e < ¢ . such that gcd(e, ¢ ) = 1

5. Compute private exponent d =e-1 mod ¢

6. Public key is {n, e}, private key is d

**Encryption & Decryption**

7. Encryption: c = memod n ,

9. Decryption: m = cdmod n

10. Display the cipher text and plain text.

**PROGRAM :**

<html>

<head>

<title>RSA Encryption</title>

<meta name="viewport" content="width=device-width, initial-scale=1.0">

</head>

<body>

<center>

<h1>RSA Algorithm</h1>

<h2>Implemented Using HTML & Javascript</h2>

<hr>

<table>

<tr>

<td>Enter First Prime Number:</td>

<td><input type="number" value="53" id="p"></td>

</tr>

<tr>

<td>Enter Second Prime Number:</td>

<td><input type="number" value="59" id="q"></p>

</td>

</tr>

<tr>

<td>Enter the Message(cipher text):<br>[A=1, B=2,...]</td>

<td><input type="number" value="89" id="msg"></p>

</td>

</tr>

<tr>

<td>Public Key:</td>

<td>

<p id="publickey"></p>

</td>

</tr>

<tr>

<td>Exponent:</td>

<td>

<p id="exponent"></p>

</td>

</tr>

<tr>

<td>Private Key:</td>

<td>

<p id="privatekey"></p>

</td>

</tr>

<tr>

<td>Cipher Text:</td>

<td>

<p id="ciphertext"></p>

</td>

</tr>

<tr>

<td><button onclick="RSA();">Apply RSA</button></td>

</tr>

</table>

</center>

</body>

<script type="text/javascript">

function RSA() {

var gcd, p, q, no, n, t, e, i, x;

gcd = function (a, b) { return (!b) ? a : gcd(b, a % b); };

p = document.getElementById('p').value;

q = document.getElementById('q').value;

no = document.getElementById('msg').value;

n = p \* q;

t = (p - 1) \* (q - 1);

for (e = 2; e < t; e++) {

if (gcd(e, t) == 1) {

break;

}

}

for (i = 0; i < 10; i++) {

x = 1 + i \* t

if (x % e == 0) {

d = x / e;

break;

}

}

ctt = Math.pow(no, e).toFixed(0);

ct = ctt % n;

dtt = Math.pow(ct, d).toFixed(0);

dt = dtt % n;

document.getElementById('publickey').innerHTML = n;

document.getElementById('exponent').innerHTML = e;

document.getElementById('privatekey').innerHTML = d;

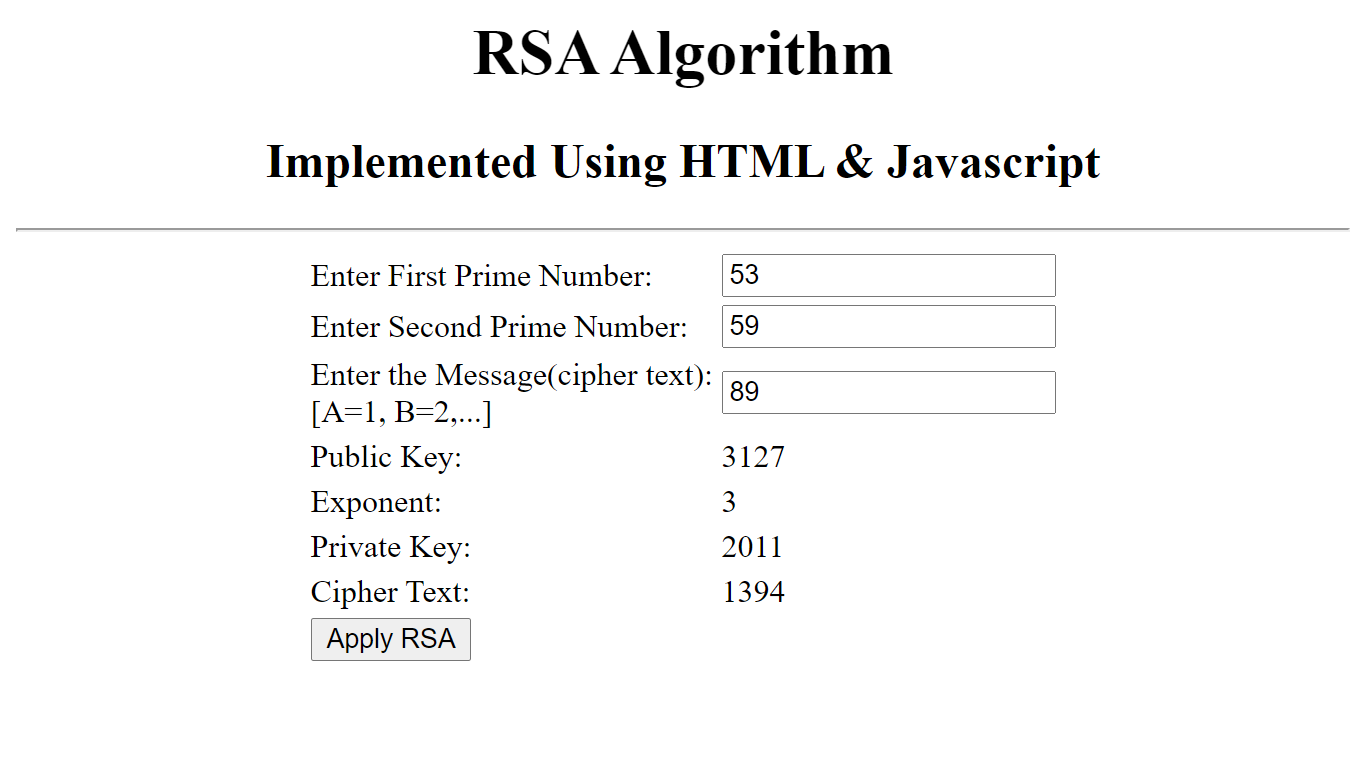
document.getElementById('ciphertext').innerHTML = ct;

}

</script>

</html>

**OUTPUT:**

****

**RESULT:**

.

**EX NO : 6) Implementation of Diffie Hellman key Exchange Algorithm**

**AIM :** Towrite a C++ program toImplement Diffie Hellman key exchange Algorithm.

**ALGORITHM :**

1. Enter the value of n and g, both are prime numbers.

2. Enter x and y and these are private chosen by the users alice and bob respectively.

3. Alice calculates g x mod n and Bob calculates g y mod n

4**.** Bob sends the value g y mod n to Alice and alice sends the value g x mod n to Bob.

5. Alice will calculate the shared secret by (g y mod n )x mod n and bob by

(g x mod n )y mod n.

6 . Display the shared secret key.

**PROGRAM :**

// Diffie Hellman key Exchange

#include<iostream>  
using namespace std;  
#include<math.h>  
int alice(int,double,double);  
int bob(int,double,double);  
int main( )  
{ long int a,b,k1,k2,n;  
double g,x,y;  
 cout<<"\n DIFFIE HELLMAN KEY EXCHANGE";  
 cout<<"\n\n\t Enter value of n & g";  
 cin>>n>>g;  
 cout<<"\n\n\t Enter value of x & y";  
 cin>>x>>y;  
 a=alice(n,g,x);  
 cout<<"\n\t alice end value:"<<a;  
 b=bob(n,g,y);  
 cout<<"\n\t bob end value:"<<b;  
 k1=alice(n,b,x);  
cout<<"\n\t value of k1 :"<<k1;  
 k2=bob(n,a,y);  
 cout<<"\n\t value of k2 :"<<k2;  
 }  
int alice(int n, double g, double x)  
{  
 long int a;  
double a1;  
 a1=pow(g,x);  
 a=(int)a1%n;  
 return(a);  
}  
int bob(int n, double g, double y)  
{  
 long int b;  
double b1;  
 b1=pow(g,y);  
 b=(int)b1%n;  
 return(b);  
}

**OUTPUT :**

[s@localhost ~]$ ./a.out  
DIFFIE HELLMAN KEY EXCHANGE  
Enter value of n & g 23 4  
Enter value of x & y 6 9

alice end value:2  
bob end value:13  
value of k1 :6  
value of k2 :6

**RESULT:**

.

**EX NO : 7) Implementation of SHA-1 Algorithm**

**AIM:** To write a java program to calculate the message digest of a text using the SHA-1 algorithm

**ALGORITHM :**

1. Input the plain text

2. Append padding bits

3. Append length

4. Initialize buffer.

5. Process message in 512-bit (16word)blocks

6. Encrypted in hexa decimal format

7. Display the encrypted text.

**PROGRAM:**

import java.security.\*;

public class SHA1 {

public static void main(String[] a) {

try {

MessageDigest md = MessageDigest.getInstance("SHA1");

System.out.println("Message digest object info: ");

System.out.println(" Algorithm = " +md.getAlgorithm());

System.out.println(" Provider = " +md.getProvider());

System.out.println(" ToString = " +md.toString());

String input = "";

md.update(input.getBytes());

byte[] output = md.digest();

System.out.println();

System.out.println("SHA1(\""+input+"\") = " +bytesToHex(output));

input = "abc";

md.update(input.getBytes());

output = md.digest();

System.out.println();

System.out.println("SHA1(\""+input+"\") = " +bytesToHex(output));

input = "abcdefghijklmnopqrstuvwxyz";

md.update(input.getBytes());

output = md.digest();

System.out.println();

System.out.println("SHA1(\"" +input+"\") = " +bytesToHex(output));

System.out.println(""); }

catch (Exception e) {

System.out.println("Exception: " +e);

}

}

public static String bytesToHex(byte[] b) {

char hexDigit[] = {'0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'A', 'B', 'C', 'D', 'E', 'F'};

StringBufferbuf = new StringBuffer();

for (int j=0; j<b.length; j++) {

buf.append(hexDigit[(b[j] >> 4) & 0x0f]);

buf.append(hexDigit[b[j] & 0x0f]); }

returnbuf.toString(); }}

**OUTPUT:**

Message digest object info:

Algorithm = SHA1

Provider = SUN version 1.6

ToString = SHA1 Message Digest from SUN, <initialized>

SHA1("") = DA39A3EE5E6B4B0D3255BFEF95601890AFD80709

SHA1("abc") = A9993E364706816ABA3E25717850C26C9CD0D89D

SHA1("abcdefghijklmnopqrstuvwxyz")=32D10C7B8CF96570CA04CE37F2A19D8424

0D3A89

**RESULT:**

**EX NO 8 Implement the SIGNATURE SCHEME - Digital Signature Standard**

**AIM:**

To implement Digital Signature Scheme using java.

**Algorithm:**

Input the plain text

Get the Claimed Signatory’s Identifier.

Generate the Domain Parameters and Public Key

Generate a Message Digest

Verify the Digital Signature

Digital Signature Validation Complete

**Coding:**

import java.security.KeyPair;

import java.security.KeyPairGenerator;

import java.security.Signature;

import sun.misc.BASE64Encoder;

public class DigSign {

public static void main(String[] args) throws Exception {

// TODO code application logic here

KeyPairGeneratorkpg = KeyPairGenerator.getInstance("RSA");

kpg.initialize(1024);

KeyPairkeyPair = kpg.genKeyPair();

byte[] data = "Sample Text".getBytes("UTF8");

Signature sig = Signature.getInstance("MD5WithRSA");

sig.initSign(keyPair.getPrivate());

sig.update(data);

byte[] signatureBytes = sig.sign();

System.out.println("Signature: \n" + new BASE64Encoder().encode(signatureBytes));

sig.initVerify(keyPair.getPublic());

sig.update(data);

System.out.println(sig.verify(signatureBytes));

}

}

**OUTPUT:**

Signature:

imwaKe99tkM6H6hiiP0rubmb/MrYJZLiwLdRSjslF2KlA5B23az5M2LKftQFCB+NH

Ce5F5/YfN8OsNSNLtucrrZTah0SrdWSzdGCOfYLdUZmPQ72j1SkLhYspsTsUb/U6

FPSYT4QebNSYobDtjKujkHdRimHI9TO4lLuqVQRdWU= true

**RESULT:**

**EX NO 9 DEMONSTRATE INTRUSION DETECTION SYSTEM (IDS)**

**AIM:**

To demonstrate Intrusion Detection System (IDS) using Snort software tool.

**STEPS ON CONFIGURING AND INTRUSION DETECTION:**

**1**. Download Snort from the Snort.org website. (http://www.snort.org/snort-downloads)

**2**. Download Rules(https://www.snort.org/snort-rules). You must register to get the rules. (You should download these often)

**3**. Double click on the .exe to install snort. This will install snort in the “C:\Snort” folder.It is important to have WinPcap (https://www.winpcap.org/install/) installed

**4**. Extract the Rules file. You will need WinRAR for the .gz file.

**5**. Copy all files from the “rules” folder of the extracted folder. Now paste the rules into *“C:\Snort\rules”* folder.

**6**. Copy “snort.conf” file from the “etc” folder of the extracted folder. You must paste it into “C:\Snort\etc” folder. Overwrite any existing file. Remember if you modify your snort.conf file and download a new file, you must modify it for Snort to work.

7. Open a command prompt (cmd.exe) and navigate to folder “C:\Snort\bin” folder. ( at the Prompt, type cd\snort\bin)

8. To start (execute) snort in sniffer mode use following command:

snort -dev -i 3

-i indicates the interface number. You must pick the correct interface number. In my case, it is 3.

-dev is used to run snort to capture packets on your network.

To check the interface list, use following command:

snort -W



**Finding an interface**

You can tell which interface to use by looking at the Index number and finding Microsoft. As you can see in the above example, the other interfaces are for VMWare. My interface is 3.

9. To run snort in IDS mode, you will need to configure the file “snort.conf” according to your network environment.

10. To specify the network address that you want to protect in snort.conf file, look for the following line.

var HOME\_NET 192.168.1.0/24 (You will normally see any here)

11. You may also want to set the addresses of DNS\_SERVERS, if you have some on your network.

**Example:**

example snort

12. Change the RULE\_PATH variable to the path of rules folder.

var RULE\_PATH c:\snort\rules

path to rules

13. Change the path of all library files with the name and path on your system. and you must change the path of snort\_dynamicpreprocessorvariable.

C:\Snort\lib\snort\_dynamiccpreprocessor

You need to do this to all library files in the “C:\Snort\lib” folder. The old path might be: “/usr/local/lib/…”. you will need to replace that path with your system path. Using C:\Snort\lib

14. Change the path of the “dynamicengine” variable value in the “snort.conf” file..

Example:

dynamicengine C:\Snort\lib\snort\_dynamicengine\sf\_engine.dll

15 Add the paths for “include classification.config” and “include reference.config” files.

include c:\snort\etc\classification.config

include c:\snort\etc\reference.config

16. Remove the comment (#) on the line to allow ICMP rules, if it is commented with a #.

include $RULE\_PATH/icmp.rules

17. You can also remove the comment of ICMP-info rules comment, if it is commented.

include $RULE\_PATH/icmp-info.rules

18. To add log files to store alerts generated by snort, search for the “output log” test in snort.conf and add the following line:

output alert\_fast: snort-alerts.ids

19. Comment (add a #) the whitelist $WHITE\_LIST\_PATH/white\_list.rules and the blacklist

Change the nested\_ip inner , \ to nested\_ip inner #, \

20. Comment out (#) following lines:

#preprocessor normalize\_ip4

#preprocessor normalize\_tcp: ips ecn stream

#preprocessor normalize\_icmp4

#preprocessor normalize\_ip6

#preprocessor normalize\_icmp6

21. Save the “snort.conf” file.

22. To start snort in IDS mode, run the following command:

snort -c c:\snort\etc\snort.conf -l c:\snort\log -i 3

(Note: 3 is used for my interface card)

If a log is created, select the appropriate program to open it. You can use WordPard or NotePad++ to read the file.

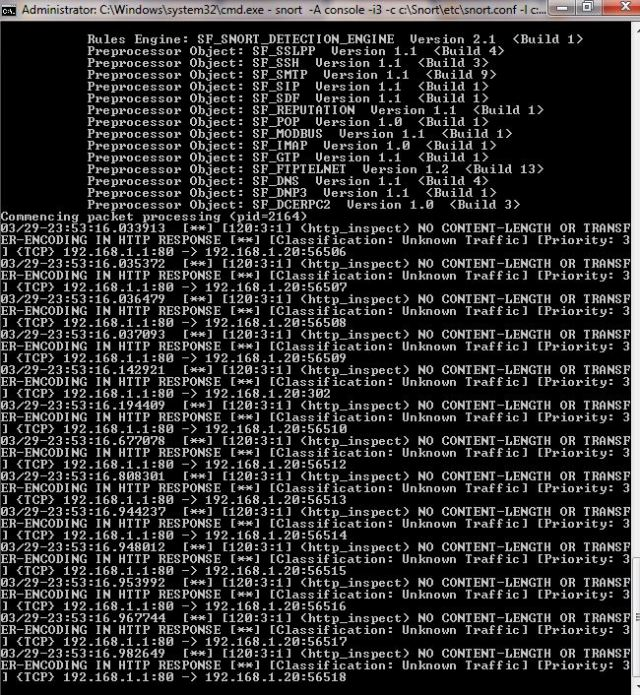
To generate Log files in ASCII mode, you can use following command while running snort in IDS mode:

snort -A console -i3 -c c:\Snort\etc\snort.conf -l c:\Snort\log -K ascii

23. Scan the computer that is running snort from another computer by using PING or NMap (ZenMap).

After scanning or during the scan you can check the snort-alerts.ids file in the log folder to insure it is logging properly. You will see IP address folders appear.

Snort monitoring traffic –



**RESULT:**

**EX 10 Exploring N-Stalker, a Vulnerability Assessment Tool**

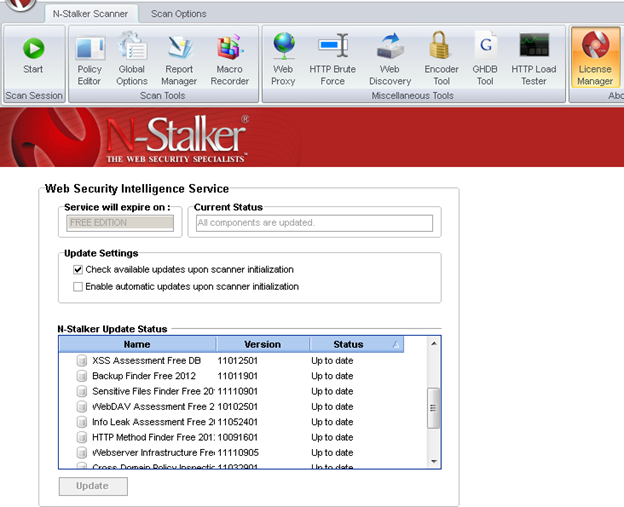
**AIM:**

To download the N-Stalker Vulnerability Assessment Tool and exploring the features.

**EXPLORING N-STALKER:**

* N-Stalker Web Application Security Scanner is a Web security assessment tool.
* It incorporates with a well-known N-Stealth HTTP Security Scanner and 35,000 Web attack signature database.
* This tool also comes in both free and paid version.
* Before scanning the target, go to “License Manager” tab, perform the update.
* Once update, you will note the status as up to date.
* You need to download and install N-Stalker from [www.nstalker.com](http://www.nstalker.com).

1. Start N-Stalker from a Windows computer. The program is installed under Start ➪ Programs ➪ N-Stalker ➪ N-Stalker Free Edition.
2. Enter a host address or a range of addresses to scan.
3. Click Start Scan.
4. After the scan completes, the N-Stalker Report Manager will prompt
5. you to select a format for the resulting report as choose Generate HTML.
6. Review the HTML report for vulnerabilities.



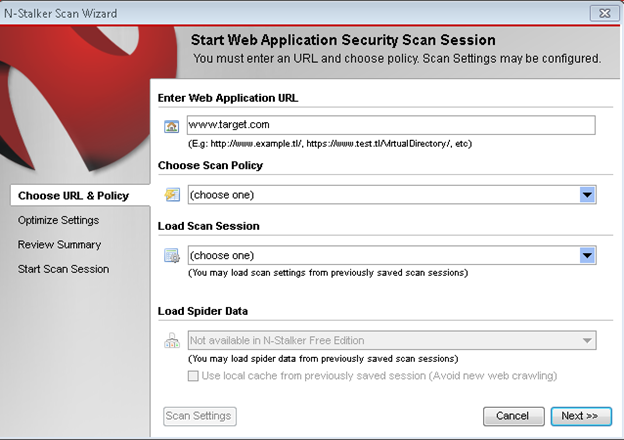
Now goto “Scan Session”, enter the target URL.

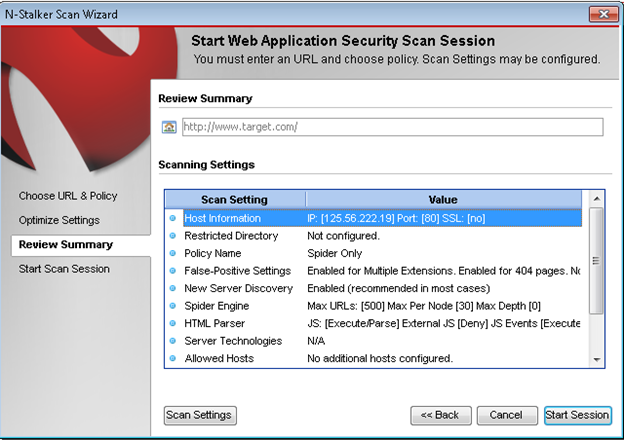
In scan policy, you can select from the four options,

* Manual test which will crawl the website and will be waiting for manual attacks.
* full xss assessment
* owasp policy
* Web server infrastructure analysis.

Once, the option has been selected, next step is “Optimize settings” which will crawl the whole website for further analysis.

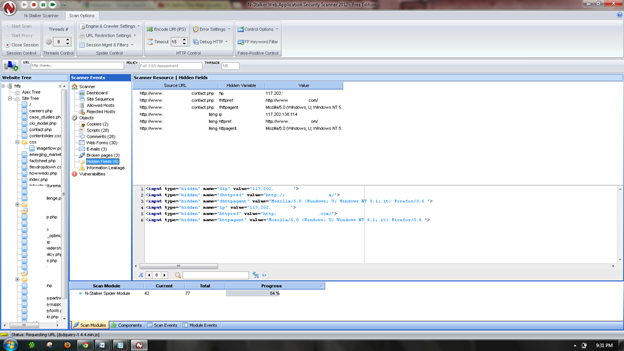
In review option, you can get all the information like host information, technologies used, policy name, etc.



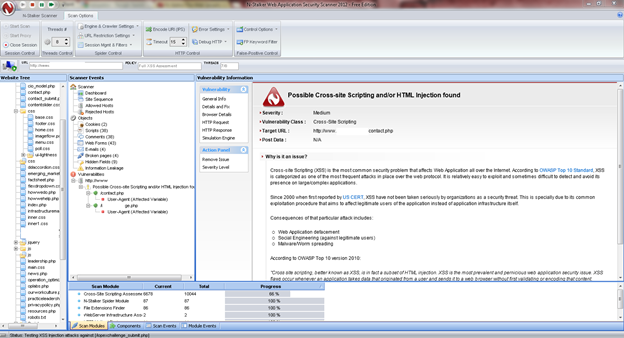


Once done, start the session and start the scan.

The scanner will crawl the whole website and will show the scripts, broken pages, hidden fields, information leakage, web forms related information which helps to analyze further.



Once the scan is completed, the NStalker scanner will show details like severity level, vulnerability class, why is it an issue, the fix for the issue and the URL which is vulnerable to the particular vulnerability?



**RESULT:**

**EX 11. DEFEATING MALWARE - BUILDING TROJANS**

**AIM:**

To build a Trojan and know the harmness of the trojan malwares in a computer system.

**PROCEDURE:**

1. Create a simple trojan by using Windows Batch File (***.bat***)
2. Type these below code in notepad and save it as **Trojan.bat**
3. Double click on ***Trojan.bat***file.
4. When the trojan code executes, it will open MS-Paint, Notepad, Command Prompt, Explorer, etc., infinitely.
5. Restart the computer to stop the execution of this trojan.

**TROJAN:**

* In computing, a Trojan horse,or trojan, is any malware which misleads users of its true intent.
* Trojans are generally spread by some form of social engineering, for example where a user is duped into executing an email attachment disguised to appear not suspicious, (e.g., a routine form to be filled in), or by clicking on some fake advertisement on social media or anywhere else.
* Although their payload can be anything, many modern forms act as a backdoor, contacting a controller which can then have unauthorized access to the affected computer.
* Trojans may allow an attacker to access users' personal information such as banking information, passwords, or personal identity.
* ***Example:*** *Ransomware* attacks are often carried out using a *trojan*.

**CODE:**

***Trojan.bat***

@echo off

:x

start mspaint

start notepad

start cmd

start explorer

start control

start calc

goto x

**OUTPUT**

(MS-Paint, Notepad, Command Prompt, Explorer will open infinitely)

**RESULT:**

**EX NO. 11 DEFEATING MALWARE - ROOTKIT HUNTER**

**AIM:**

To install a rootkit hunter and find the malwares in a computer.

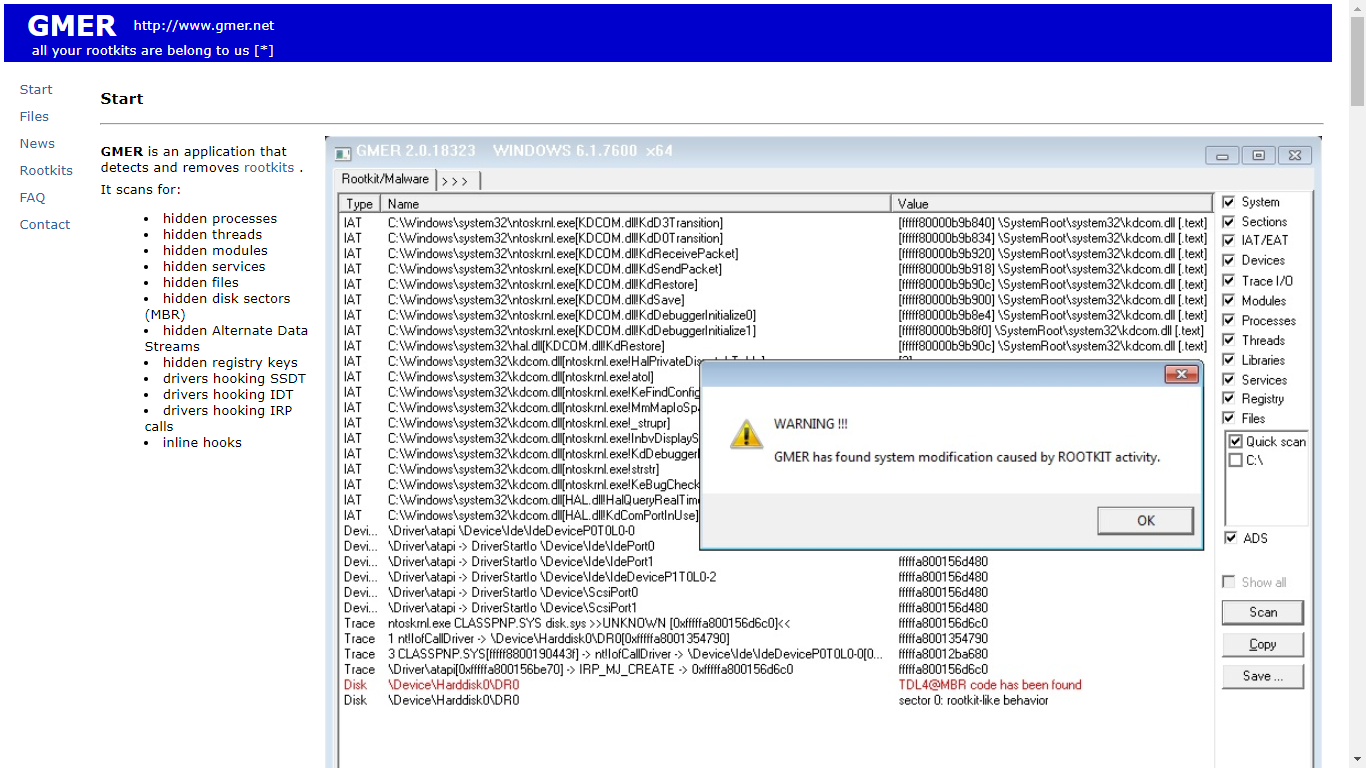
**ROOTKIT HUNTER:**

* rkhunter (Rootkit Hunter) is a Unix-based tool that scans for rootkits, backdoors and possible local exploits.
* It does this by comparing SHA-1 hashes of important files with known good ones in online databases, searching for default directories (of rootkits), wrong permissions, hidden files, suspicious strings in kernel modules, and special tests for Linux and FreeBSD.
* rkhunter is notable due to its inclusion in popular operating systems (Fedora, Debian, etc.)
* The tool has been written in Bourne shell, to allow for portability. It can run on almost all UNIX-derived systems.

**GMER ROOTKIT TOOL:**

* GMER is a software tool written by a Polish researcher Przemysław Gmerek, for detecting and removing rootkits.
* It runs on Microsoft Windows and has support for Windows NT, 2000, XP, Vista, 7, 8 and 10. With version 2.0.18327 full support for Windows x64 is added.

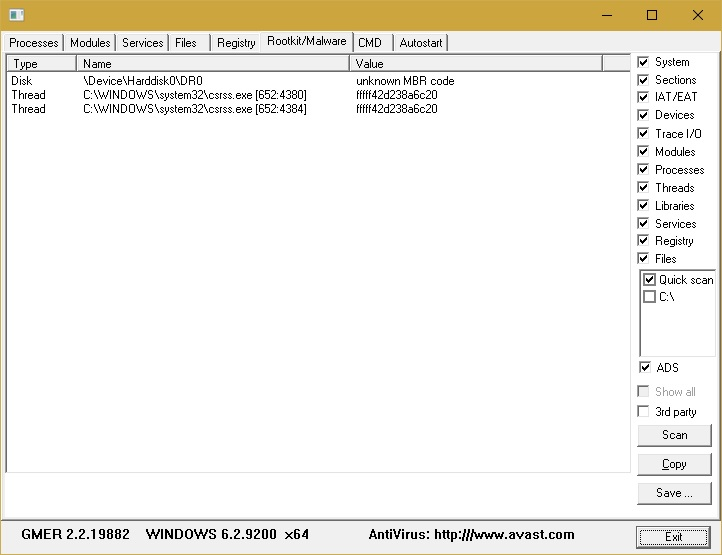
**Step 1**



Visit GMER's website (see Resources) and download the GMER executable.

Click the "Download EXE" button to download the program with a random file name, as some rootkits will close “gmer.exe” before you can open it.

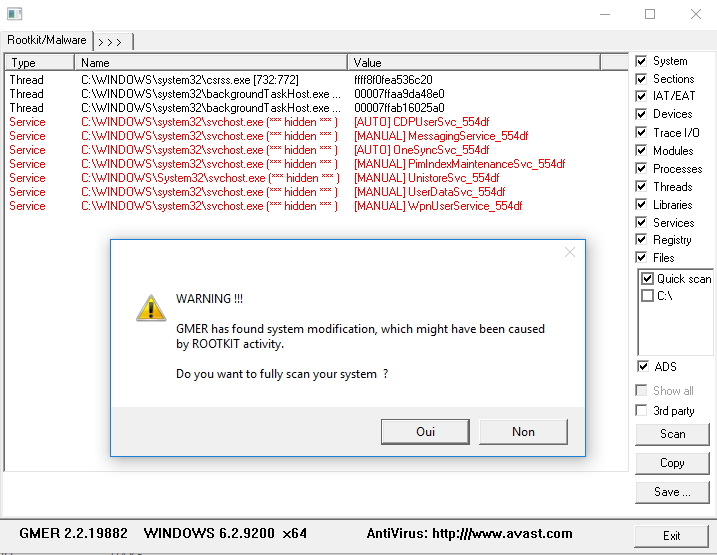
**Step 2**



Double-click the icon for the program.

Click the "Scan" button in the lower-right corner of the dialog box. Allow the program to scan your entire hard drive.

**Step 3**



When the program completes its scan, select any program or file listed in red. Right-click it and select "Delete."

If the red item is a service, it may be protected. Right-click the service and select "Disable." Reboot your computer and run the scan again, this time selecting "Delete" when that service is detected.

When your computer is free of Rootkits, close the program and restart your PC.

**RESULT:**