ST. JOSEPH'S COLLEGE OF ENGINEERING AND TECHNOLOGY

DEPARTMENT OF CSE

(REGULATION – 2017)

IT8761 - SECURITY LABORATORY

SEMESTER: VII

PREPARED BY

Ms.P.SUGANYA

AP / CSE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

ACADEMIC YEAR 2020-2021 (ODD SEMESTER)

SYLLABUS

CYCLE – I

Implementation of Ceaser Cipher

Implementation of play fair Cipher

Implementation of Hill Cipher

Implementation of Vigenere Cipher

2.a Implementation of Rail Fence - row & column transformation

3 Implementation of DES Algorithm for practical application

4 Implementation of AES Algorithm for practical application

5 Implementation of RSA Algorithm using HTML and Javascript

CYCLE – 2

` 6 Implementation of Diffie - Hellman algorithm

7 Calculate the message digest of a text using the SHA-1

8 Implementation of signature scheme - DSS

9. Demonstrate Intrusion Detection System (IDS) using any tool such as Snort or any other Software.

10 Automated Attack and Penetration Tools Exploring N-Stalker and a Vulnerability Assessment Tool

11. Defeating malware a. Building Trojans
b. Rootkit Hunter

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

LABORATORY COURSE PLAN

COURSE OBJECTIVE

1. Be exposed to the different cipher techniques
2. Learn to implement the algorithms DES,RSA, MD5, SHA-1
3. Learn to use network security tools and vulnerability assessment tools

LEARNING OUTCOME

Upon the completion of this lab, students should be able to

- Implement the cipher techniques
- Develop the various security algorithms
- Use different open source tools for network security and analysis

PRE-REQUISITE

Knowledge on java programming and open source tools for network security.

EQUIPMENTS / COMPONENTS / SOFTWARE REQUIRMENT

Hardware:

Standalone desktops – 30Nos. (or) Server supporting 30 terminals or more

Software:

C, C++, java
GnuPG, Snort and N stalker or Equivalent

Ex.No	Date	Title of the experiment	No. of Hrs. required	Cumulative No. of periods
CYCLE : I				
1a		Implementation of Ceaser Cipher	6	6
1b		Implementation of play fair Cipher		
1c		Implementation of Hill Cipher	6	12
1d		Implementation of Vigenere Cipher		
2a		Implementation of Rail Fence transposition techniques	3	15
2b		Implementation of row & column transformation	3	18
3		Implementation of DES Algorithm	6	24
4		Implementation of AES Algorithm	3	27
5		Implementation of RSA algorithm using HTML and Java script	3	30
CYCLE : II				
6		Implementation of Diffie - Hellman algorithm	6	36
7		Calculate the message digest of a text using the SHA1	3	39
8		Performs a digital signature on a given text using DSS	6	45
9		Demonstrate Intrusion Detection System (IDS) using any tool such as Snort or any other Software.	3	48
10		Exploring automated attack and Penetration tools(N-stalket)	6	54
11a		Building Trojans	3	57
11b		Rootkit Hunter	3	60

CONTENT BEYOND SYLLABUS

1. Study of assigning IP in Router

CONTENT

EX.NO: 1 A **IMPLEMENTATION OF CEASER CIPHER**
DATE:

AIM:

To write a Java program to perform encryption and decryption using the Ceaser Cipher algorithm.

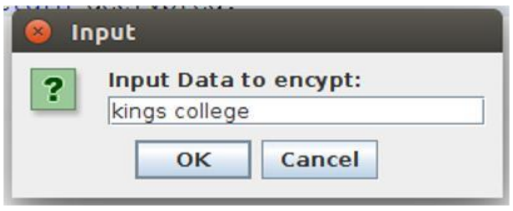
ALGORITHM:

```
Step1: Start the program with input plain text P and key as K.
Step2: For each plaintext letter P, substitute the cipher text letter C
            C=E(K,P)
              =(P+K) mod 26
            Where K=values from 1 to 25
Step 3: The decryption algorithm is simply
            P=D(K,C)
              =(C-K) mod 26
Step4: End the program.
```

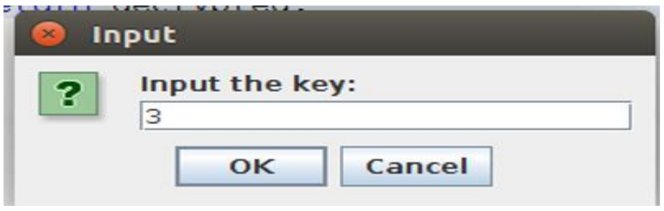
PROGRAM:

```
import javax.swing.JOptionPane;
public class CaesarCipher {
public static void main(String[] args)
        {
                //gets a string to encrypt
                String str = (JOptionPane.showInputDialog("Input Data to encypt:"));
                //gets a key
                String key = (JOptionPane.showInputDialog("Input the key:"));
                int keyLength=key.length();
                //prints encryption
                String encrypted = encrypt(str, keyLength);
                System.out.println("Encrypted:" + encrypted);
                //prints decryption
                String decrypted = decrypt(encrypted, keyLength);
                System.out.println("Decrypted:" + decrypted);
        }
        public static String encrypt(String str, int keyLength)
        {
                String encrypted = "";
        int key=keyLength;
                for(int i = 0; i < str.length(); i++)
                {
                        int c = str.charAt(i);
                        if (Character.isUpperCase(c))
                        {
                                //26 letters of the alphabet so mod by 26
                                c = c + (key % 26);
                                if (c > 'Z')
                                c = c - 26;
                        }
                        else if (Character.isLowerCase(c))
                        {
                                c = c + (key % 26);
                                if (c > 'z')
                                c = c - 26;
                        }
                        encrypted += (char) c;
                }
        return encrypted;
        }
        public static String decrypt(String str, int key)
        {
                String decrypted = "";
                for(int i = 0; i < str.length(); i++)
                {
                        int c = str.charAt(i);
                        if (Character.isUpperCase(c))
                        {
                                c = c - (key % 26);
                                if (c < 'A')
                                c = c + 26;
                        }
                        else if (Character.isLowerCase(c))
                        {
                                c = c - (key % 26);
                                if (c < 'a')
                                c = c + 26;
                        }
                        decrypted += (char) c;
                }
                return decrypted;
        }
}
```

OUTPUT:



Input Data to encypt:
kings college



Input the key:
3

```
Encrypted:ljoht dpmmfhf
Decrypted:kings college
BUILD SUCCESSFUL (total time: 50 seconds)
```

RESULT:

Thus the java program to implement Caesar Cipher was executed and verified successfully.

VIVA QUESTIONS:

1. Specify the four categories of security threads
2. Explain active and passive attack with example
3. Define integrity and non repudiation.
4. Differentiate symmetric and asymmetric encryption.
5. Define cryptanalysis

EX.NO: 1B **IMPLEMENTATION OF PLAY FAIR CIPHER**
DATE:

AIM:

To write a Java program to perform encryption and decryption using the play fair Cipher algorithm.

ALGORITHM:

Step1: Start the program.
Step2: The play fair algorithm is based on the use of a 5*5 matrix of letters constructed using a keyword.
Step3: The matrix is constructed by filling in the letters of the keyword from left to right and from top to bottom and then filling in the remainder of the matrix with the remaining letters in alphabetic order.
Step4: The letters I and J count as one letter.
Step5: Stop the program.

PROGRAM:

```
import java.util.Scanner;
public class PlayfairCipherEncryption
{
private String KeyWord        = new String();
private String Key           = new String();
private char  matrix_arr[][] = new char[5][5];

public void setKey(String k)
   {
       String K_adjust = new String();
boolean flag = false;
       K_adjust = K_adjust + k.charAt(0);
for (int i = 1; i < k.length(); i++)
       {
for (int j = 0; j < K_adjust.length(); j++)
           {
if (k.charAt(i) == K_adjust.charAt(j))
               {
flag = true;
               }
           }
if (flag == false)
               K_adjust = K_adjust + k.charAt(i);
flag = false;
       }
       KeyWord = K_adjust;
   }

public void KeyGen()
   {
boolean flag = true;
char current;
```

```
        Key = KeyWord;
for (int i = 0; i < 26; i++)
        {
current = (char) (i + 97);
if (current == 'j')
continue;
for (int j = 0; j < KeyWord.length(); j++)
            {
if (current == KeyWord.charAt(j))
                {
flag = false;
break;

                }
            }
if (flag)
                Key = Key + current;
flag = true;
        }
System.out.println(Key);
matrix();
    }

private void matrix()
    {
int counter = 0;
for (int i = 0; i < 5; i++)
        {
for (int j = 0; j < 5; j++)
            {
                matrix_arr[i][j] = Key.charAt(counter);
System.out.print(matrix_arr[i][j] + " ");
counter++;
            }
System.out.println();
        }
    }

private String format(String old_text)
    {
int i = 0;
int len = 0;
        String text = new String();
len = old_text.length();
for (int tmp = 0; tmp < len; tmp++)
        {
if (old_text.charAt(tmp) == 'j')
            {
text = text + 'i';
            }
else
text = text + old_text.charAt(tmp);
        }
len = text.length();
for (i = 0; i < len; i = i + 2)
        {
if (text.charAt(i + 1) == text.charAt(i))
            {
text = text.substring(0, i + 1) + 'x' + text.substring(i + 1);
            }
        }
return text;
    }

private String[] Divid2Pairs(String new_string)
    {
        String Original = format(new_string);
int size = Original.length();
if (size % 2 != 0)
        {
size++;
            Original = Original + 'x';
        }
        String x[] = new String[size / 2];
int counter = 0;
for (int i = 0; i < size / 2; i++)
        {
x[i] = Original.substring(counter, counter + 2);
counter = counter + 2;
        }
return x;
    }

public int[] GetDiminsions(char letter)
    {
int[] key = new int[2];
if (letter == 'j')
letter = 'i';
for (int i = 0; i < 5; i++)
        {
for (int j = 0; j < 5; j++)
            {
if (matrix_arr[i][j] == letter)
                {
key[0] = i;
key[1] = j;
break;
                }
            }
        }
```

```
return key;
   }

public String encryptMessage(String Source)
   {
       String src_arr[] = Divid2Pairs(Source);
       String Code = new String();
char one;
char two;
int part1[] = new int[2];
int part2[] = new int[2];
for (int i = 0; i < src_arr.length; i++)
       {
one = src_arr[i].charAt(0);
two = src_arr[i].charAt(1);
          part1 = GetDiminsions(one);
          part2 = GetDiminsions(two);
if (part1[0] == part2[0])
          {
if (part1[1] < 4)
part1[1]++;
else
part1[1] = 0;
if (part2[1] < 4)
part2[1]++;
else
part2[1] = 0;
          }
else if (part1[1] == part2[1])
          {
if (part1[0] < 4)
part1[0]++;
else
part1[0] = 0;
if (part2[0] < 4)
part2[0]++;
else
part2[0] = 0;
          }
else
          {
int temp = part1[1];
part1[1] = part2[1];
part2[1] = temp;
          }
          Code = Code + matrix_arr[part1[0]][part1[1]]
                 + matrix_arr[part2[0]][part2[1]];
       }
return Code;
   }
```

```
public static void main(String[] args)
   {
       PlayfairCipherEncryption x = new PlayfairCipherEncryption();
       Scanner sc = new Scanner(System.in);
System.out.println("Enter a keyword:");
       String keyword = sc.next();
x.setKey(keyword);
x.KeyGen();
       System.out
              .println("Enter word to encrypt: (Make sure length of message is even)");
       String key_input = sc.next();
if (key_input.length() % 2 == 0)
       {
System.out.println("Encryption: " + x.encryptMessage(key_input));
       }
else
       {
System.out.println("Message length should be even");
       }
sc.close();
   }
}
```

OUTPUT:

```
Enter a keyword:
Sanfoundry
Sanfoudrybceghiklmpqstvwxz
S a n f o
u d r y b
c e g h i
k l m p q
s t v w x
Enter word to encrypt: (Make sure length of message is even)
Learningcenter
Encryption: acndogrmegavgd
BUILD SUCCESSFUL (total time: 36 seconds)
```

RESULT:

Thus the java program to implement play fair cipher was executed and verified successfully.

VIVA QUESTIONS:

1. Compare stream cipher and block cipher with example.

2. Define security mechanism.

3. Define stegnography.

4. Why does the network need security?

5. Define Encryption.

EX.NO:1C **IMPLEMENTATION OF HILL CIPHER**
DATE:

AIM:

To write a Java program to perform encryption and decryption using the Hill Cipher algorithm.

ALGORITHM:

Step1: Start the program.
Step2: The encryption algorithm takes m successive plaintext letters and substitute for them m cipher text letters.
Step3: The substitution is determined by m linear equations in which each character is assigned a numerical value (a=0,b=1,c=2, ... z=25)the system can be described as follows:

$C_1=(K_{11}P_1+K_{12}P_2+K_{13}P_3) \text{ mod } 26$
$C_2=(K_{21}P_1+K_{22}P_2+K_{23}P_3) \text{ mod } 26$
$C_3=(K_{31}P_1+K_{32}P_2+K_{33}P_3) \text{ mod } 26$

Step4: This can be expressed in term of column vectors and matrices:

$C=KP \text{ mod } 26$

Step5: Decryption requires using the inverse of the matrix K.
Step6: Stop the program.

PROGRAM:

```
import java.io.*;
import java.util.*;
import java.io.*;
public class HillCipher {
static float[][] decrypt = new float[3][1];
static float[][] a = new float[3][3];
static float[][] b = new float[3][3];
static float[][] mes = new float[3][1];
static float[][] res = new float[3][1];
static BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
static Scanner sc = new Scanner(System.in);
public static void main(String[] args) throws IOException {
	// TODO code application logic here
getkeymes();
for(int i=0;i<3;i++)
for(int j=0;j<1;j++)
for(int k=0;k<3;k++) {
res[i][j]=res[i][j]+a[i][k]*mes[k][j]; }
System.out.print("\nEncrypted string is : ");
for(int i=0;i<3;i++) {
System.out.print((char)(res[i][0]%26+97));
res[i][0]=res[i][0];
	}
inverse();
for(int i=0;i<3;i++)
for(int j=0;j<1;j++)
for(int k=0;k<3;k++) {
decrypt[i][j] = decrypt[i][j]+b[i][k]*res[k][j]; }
System.out.print("\nDecrypted string is : ");
for(int i=0;i<3;i++){
System.out.print((char)(decrypt[i][0]%26+97));
        }
System.out.print("\n");
    }
public static void getkeymes() throws IOException {
System.out.println("Enter 3x3 matrix for key (It should be inversible): ");
for(int i=0;i<3;i++)
for(int j=0;j<3;j++)
a[i][j] = sc.nextFloat();
System.out.print("\nEnter a 3 letter string: ");
        String msg = br.readLine();
for(int i=0;i<3;i++)
mes[i][0] = msg.charAt(i)-97;
    }
public static void inverse() {
float p,q;
float[][] c = a;
for(int i=0;i<3;i++)
for(int j=0;j<3;j++) {
                //a[i][j]=sc.nextFloat();
if(i==j)
{  b[i][j]=1;
b[i][j]=1;
}
else b[i][j]=0;
            }
for(int k=0;k<3;k++) {
for(int i=0;i<3;i++) {
                p = c[i][k];
                q = c[k][k];
for(int 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-p*b[k][j];
                    } } } }
for(int i=0;i<3;i++)
for(int j=0;j<3;j++) {
b[i][j] = b[i][j]/c[i][i]; }
System.out.println("");
System.out.println("\nInverse Matrix is : ");
for(int i=0;i<3;i++) {
for(int j=0;j<3;j++)
System.out.print(b[i][j] + "  ");
System.out.print("\n"); }
        }
}
```

OUTPUT:

```
Enter 3x3 matrix for key (It should be inversible):
6 24 1
13 16 10
20 17 15

Enter a 3 letter string: ACT

Encrypted string is : RUN

Inverse Matrix is :
0.15873016   -0.7777778  0.50793654
0.011337869  0.15873016   -0.106575966
-0.2244898   0.85714287  -0.48979592

Decrypted string is : ACT
BUILD SUCCESSFUL (total time: 24 seconds)
```

RESULT:

Thus the java program to implement Hill cipher was executed and verified successfully.

VIVA QUESTIONS:

1. Compare Substitution and Transposition techniques.
2. What is meant by play fair cipher?
3. What is meant by Hill cipher?
4. Where do we use the Polyalphabetic Ciphers?
5. Explain Avalanche effect.

EX.NO: 1D **IMPLEMENTATION OF VIGENERE CIPHER**
DATE:

AIM:

To write a Java program to perform encryption and decryption using the Vigenere Cipher algorithm.

ALGORITHM:

Step1: Start the program.
Step2: The vigenere cipher is a poly alphabetic cipher. Thus the cipher can map an alphabetic character to several other characters.
Step3: In poly alphabetic cipher, each plaintext character may be replaced by more than one character. Since there are only 26 alphabets this process will require using a different representation than the alphabets. Alphabets „A‟ through „Z‟ are replaced by 00,01,02,... 25.We need two digits in this representation since we need to know how to reverse the process at the decryption side.
Step4: The most common method is vigenere cipher. Vigenere cipher starts with a 26*26 matrix of alphabets in sequence. First row starts with „A‟, Second row starts with „B‟,etc. This cipher also requires a keyword that the sender and receiver know ahead of time. Each character of the message is combined with the characters of the keyword to find the ciphertext character.
Step5: To decrypt, the receiver places the keyword characters below each ciphertext character. Using the table, choose the row corresponding to the keyword character and look for the ciphertext character in that row. Plaintext character is then at the top of that column.
Step6: Stop the program.

PROGRAM:

```
public class VigenereCipher {
public static void main(String[] args) {
      String key = "VIGENERECIPHER";
      String ori = "Beware the Jabberwock, my son! The jaws that bite, the claws that catch!";
      String enc = encrypt(ori, key);
System.out.println(enc);
System.out.println(decrypt(enc, key));
   }

static String encrypt(String text, final String key) {
      String res = "";
text = text.toUpperCase();
for (int i = 0, j = 0; i < text.length(); i++) {
char c = text.charAt(i);
if (c < 'A' || c > 'Z') continue;
res += (char)((c + key.charAt(j) - 2 * 'A') % 26 + 'A');
         j = ++j % key.length();
      }
return res;
   }

static String decrypt(String text, final String key) {
      String res = "";
text = text.toUpperCase();
for (int i = 0, j = 0; i < text.length(); i++) {
char c = text.charAt(i);
if (c < 'A' || c > 'Z') continue;
res += (char)((c - key.charAt(j) + 26) % 26 + 'A');
            j = ++j % key.length();
        }
return res;
    }
}
```

OUTPUT:

```
WMCEEIKLGRPIFVMEUGXQPWQVIOIAVEYXUEKFKBTALVXTGAFXYEVKPAGY
BEWARETHEJABBERWOCKMYSONTHEJAWSTHATBITETHECLAWSTHATCATCH
BUILD SUCCESSFUL (total time: 0 seconds)
```

RESULT:

Thus the java program to implement vigenere cipher was executed and verified successfully.

VIVA QUESTIONS:

1. Give the five modes of operation of Block cipher.
2. State the advantages of Counter mode.
3. What is traffic padding?
4. Define reversible mapping.
5. Specify the design criteria of block cipher.

EX.NO: 2A **IMPLEMENTATION OF RAIL FENCE TRANSFORMATION**
DATE:

AIM:

To write a C program to perform encryption and decryption using the Rail Fence transposition technique.

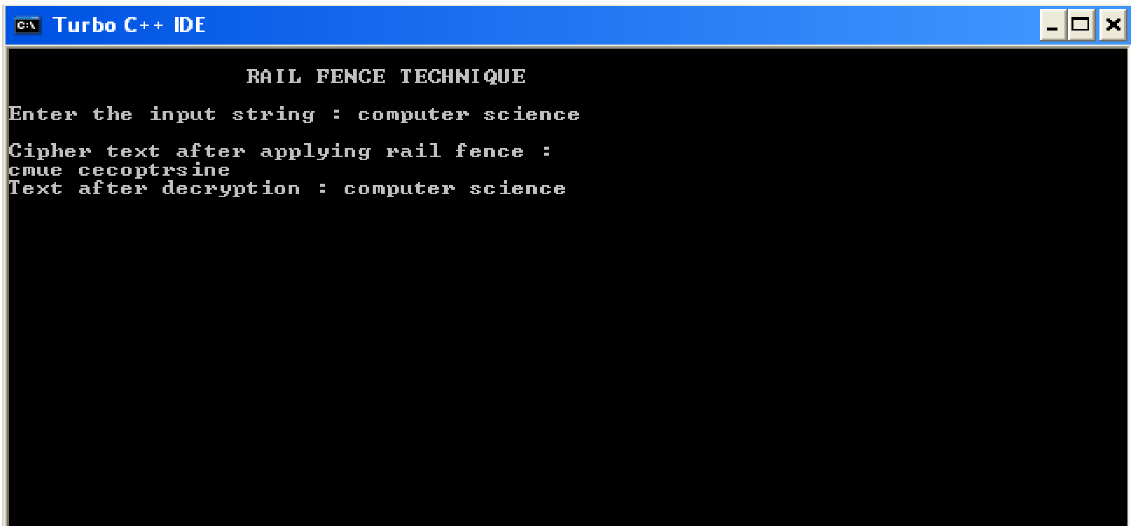
ALGORITHM:

Step1: Start the program.
Step2: The rail fence cipher is composed by writing the plaintext in two rows.
Step3: Proceeding down, then across and reading the cipher text across, then down.
Step4: It involves writing plain text as a sequence of diagonals and then reading it row by row to produce cipher text.
Step5: Stop the program.

PROGRAM:

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
int i,j,k,l;
char a[20],c[20],d[20];
clrscr();
printf("\n\t\t RAIL FENCE TECHNIQUE");
printf("\n\nEnter the input string : ");
gets(a);
l=strlen(a);
/*Ciphering*/
for(i=0,j=0;i<l;i++)
{
if(i%2==0)
c[j++]=a[i];
}
for(i=0;i<l;i++)
{
if(i%2==1)
c[j++]=a[i];
}
c[j]='\0';
printf("\nCipher text after applying rail fence :");
printf("\n%s",c);
/*Deciphering*/
if(l%2==0)
k=l/2;
else
k=(l/2)+1;
for(i=0,j=0;i<k;i++)
{
d[j]=c[i];
j=j+2;
}
for(i=k,j=1;i<l;i++)
{
d[j]=c[i];
j=j+2;
}
d[l]='\0';
printf("\nText after decryption : ");
printf("%s",d);
getch();
}
```

OUTPUT:



```
                  RAIL FENCE TECHNIQUE
Enter the input string : computer science

Cipher text after applying rail fence :
cmue cecoptrsine
Text after decryption : computer science
```

RESULT:

Thus the C program for encryption and decryption using rail fence algorithm was executed successfully.

VIVA QUESTIONS:

1. Define security attack, security mechanism and security services.

2. Mention the different types of security services.

3. What is meant by Brute force attack?

4. Mention the various types of cryptanalytic attacks.

5. Define the two basic building blocks of encryption techniques.

EX.NO: 2B **IMPLEMENTATION OF ROW & COLUMN TRANSFORMATION**
DATE:

AIM:

To write a C program to perform encryption and decryption using the row and column transposition technique.

ALGORITHM:

Step1: Start the program.
Step2: The message is written out in rows of a fixed length, and then read out again column by column, and the columns are chosen in some scrambled order.
Step3: Any spare spaces are filled with nulls or left blank or placed by a character (-).
Step4: Finally the message is read off in columns, in the order specified by the keyword.
Step5: To decipher, the recipient has to work out the column lengths by dividing the message length by the key length.
Step6: Then write the message out in columns again, and then reorder the columns by reforming the key word.
Step7: Stop the program.

PROGRAM:

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void cipher( int i, int c);
int findMin();
void makeArray(int,int);
char arr[22][22],darr[22][22],emessage[111],retmessage[111],key[55];
char temp[55],temp2[55];
int k=0;
int main() {
        char *message, *dmessage;
        int i,j,klen,emlen,flag=0;
        int r,c,index,min,rows;
        clrscr();       print(“Enter the key\n”);
        fflush(stdin);
        gets(key);
        printf(“\n Enter message to be ciphered\n”);
        fflush(stdin);  gets(message);
        strcpy(temp,key);
        klen=strlen(key);
        k=0;
        for(i=0; ;i++) {
                if(flag==1)
                        break;
                for(j=0;key[j]!=NULL;j++){
                        if(message[k]==NULL){
                                flag=1;
                                arr[i][j]=‟-„;
                        }else{
                                arr[i][j]=message[k++];
```

```
                    }
                }
            }
            r=i;    c=j;
            for(i=0;i<r;i++) {
                for(j=0;j<c;j++) {
                    printf("%c",arr[i][j]);
                }
                printf("\n");
            }
            k=0;
            for(i=0;i<klen;i++) {
                index=findMin();
                cipher(index,r);
            }
            emessage[k]='\0';
            printf("\n Encrypted message is\n");
            for ( i=o;emessage[i]!= NULL;i++)
                printf("%c",emessage[i]);
            printf("\n\n");
            //deciphering
            emlen=strlen(emessage);
            //emlen is length of encrypted message
            strcpy(temp,key);
            rows=emlen/klen;
            //rows is no of row of the array to made from ciphered message rows;
            j=0;
            for ( i=0; k=1; emessage[i]!=NULL;i++,k++){
                temp2[j++]=emessage[i];
                if((k%rows)==0) {
                    temp2[j]='\0';
                    index=findMin();
                    makeArray(index,rows);
                    j=0;
                }
            }
            Printf("\n Array Retrieved is \n");
    ``      k=0;
            for (i=0;i<r;i++) {
                for(j=0;j<c;j++) {
                    printf("%c", darr[i][j]);
                    retmessage[k++]=darr[i][j];
                }
                printf("\n");
            }
            retmessage[k]='\0';
            printf("\n Message retrieved is \n");
            for( i=0;retmessage[i]!=NULL;i++)
                printf("%c",retmessage[i]);
            getch();
```

```
return(0);
}
void cipher(int i, int r) {
        int j;
        for( j=0;j<r;j++) {{
                emessage[k+]=arr[j][i];                 }
}       }
void makeArray(int col, int row) {
        int i,j;
        for ( i=0;i<row;i++) {
                darr[i][col]=temp2[i];          }
}
int findMin() {
        int i,j,min,index;
        min=temp[0];
        index=0;
        for(j=0;temp[j]!=NULL;j++) {
                if(temp[j] <min) {
                        min=temp[j];
                        index=j;                }               }
temp[index]=123;
return(index);
}
```

OUTPUT:

```
Enter the key
hello

Enter message to be ciphered
how are you
h o w   a
r e   y  o
u - - - - -

Encrypted message is
oe-hruw – y-ao-

Array Retrieved is
how are you
h o w   a
r e   y  o
u - - - - -

Message retrieved is
how are you---
```

RESULT:

Thus the encryption and decryption for row and columnar transformation cipher algorithm had been executed successfully.

VIVA QUESTIONS:

1. What is a columnar transposition cipher?
2. How to break a columnar transposition cipher?
3. Who invented transposition cipher?
4. When was the transposition cipher used?
5. What is row transposition cipher?

EX.NO: 3 **IMPLEMENTATION OF DES ALGORITHM**
DATE:

AIM:

To write a Java program to implement the string encryption and decryption using DES algorithm.

ALGORITHM:

Step1: In the first step,the 64-bit plain text passes through an Initial Permutation(IP)
Step2: The Initial Permutation is performed on plain text.
Step3: Initial permutation produces two halves of the permuted block; say left plain Text and Right plain text.
Step4: Now each LPT and RPT go through 16 rounds of encryption process.
Step5: The output of last rounds are swapped to produce the pre output.
Step6: Finally the pre output is passed through a permutation(IP^{-1}) that is the inverse of the initial permutation function to produce the 64-bit cipher text.
Step7: DES decryption uses the same algorithm as encryption except that the application of the sub keys is reversed.

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";
private KeySpec myKeySpec;
private SecretKeyFactory mySecretKeyFactory;
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 {
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(); }
return encryptedString; }
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(); }
return decryptedText; }
private static String bytes2String(byte[] bytes) {
StringBuffer stringBuffer = new StringBuffer();
for (int i = 0; i <bytes.length; i++) {
stringBuffer.append((char) bytes[i]); }
return stringBuffer.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: Kings College of Engineering
String To Encrypt: Kings College of Engineering
Encrypted Value : M6HL2tcCtP0/uTVYnEBE9ugu93lL2yhNxw6sgyA0B/Q=
Decrypted Value : Kings College of Engineering
BUILD SUCCESSFUL (total time: 11 seconds)
```

RESULT:

Thus the java program to implement DES was executed and verified successfully.

VIVA QUESTIONS:

1. Mention the functions involved in simplified DES.

2. What are the strengths of triple DES?

3. Define symmetric key cryptography and public key cryptography.

4. List out the applications of the public key cryptosystems.

5. Define Euler‟s totient function.

EX.NO: 4 **IMPLEMENTATION OF AES ALGORITHM**
DATE:

AIM:

To write a java program to encrypt a password using AES algorithm.

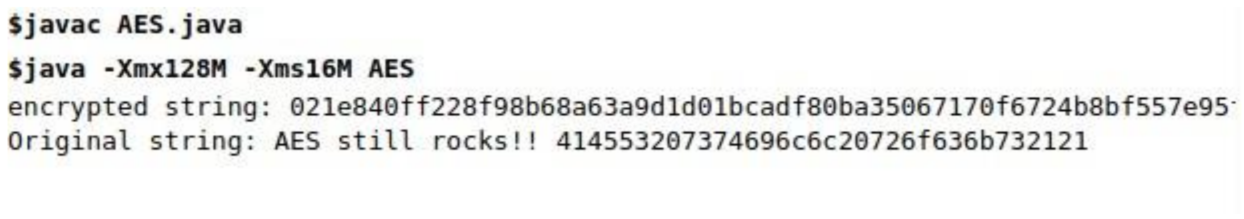
ALGORITHM:

Step1: In the first step, derive the set of round keys from the cipher key.
Step2: Initialize the state array with the block data.
Step3: Add the initial round key to the starting state array..
Step4: Perform the nine rounds of state manipulation.
Step5: Again perform tenth and final round of state manipulation.
Step6: Finally the state array produced the output as the encrypted data(ciphertext).

PROGRAM:

```
import java.security.*;
import javax.crypto.*;
import javax.crypto.spec.*;
import java.io.*;
public class AES {
public static String asHex (byte buf[]) {
StringBuffer strbuf = new StringBuffer(buf.length *2);
int i;
for (i = 0; i < buf.length; i++) {
if (((int) buf[i] & 0xff) < 0x10)
strbuf.append("0");
strbuf.append(Long.toString((int) buf[i] & 0xff, 16)); }
return strbuf.toString(); }
public static void main(String[] args) throws Exception
{ String message="AES still rocks!!";
// Get the KeyGenerator
KeyGenerator kgen = KeyGenerator.getInstance("AES");
kgen.init(128); // 192 and 256 bits may not be available
// Generate the secret key specs.
SecretKey skey = kgen.generateKey();
byte[] raw = skey.getEncoded();
SecretKeySpec skeySpec = new SecretKeySpec(raw, "AES");
// Instantiate the cipher
Cipher cipher = Cipher.getInstance("AES");
cipher.init(Cipher.ENCRYPT_MODE, skeySpec);
byte[] encrypted = cipher.doFinal((args.length == 0 ? message:
args[0]).getBytes()); System.out.println("encrypted string: " +
asHex(encrypted)); cipher.init(Cipher.DECRYPT_MODE, skeySpec);
byte[] original = cipher.doFinal(encrypted);
String originalString = new String(original);
System.out.println("Original string: " + originalString + " " + asHex(original));
}
}
```

OUTPUT



```
$javac AES.java
$java -Xmx128M -Xms16M AES
encrypted string: 021e840ff228f98b68a63a9d1d01bcadf80ba35067170f6724b8bf557e95
Original string: AES still rocks!! 414553207374696c6c20726f636b732121
```

RESULT:

Thus the java program to implement AES was executed and verified successfully.

VIVA QUESTIONS:

1. How is AES better than DES?

2. What operations are used in the AES?

3. How many rounds does the AES 192 perform?

4. What is meant by running key cipher?

5. What is meant by Advanced Encryption Standard (AES)?

EX.NO: 5 **IMPLEMENTATION OF RSA ALGORITHM**
DATE:

AIM:

To write a HTML with Javascript program to implement RSA Algorithm.

ALGORITHM:

Step1: The private and public keys in RSA are based on very large prime numbers.

Step2: Let us now understand how the public and private keys are generated, Using them, how we can perform encryption and decryption in RSA.

Step3: The whole process is as follows:

a. Choose two large prime numbers P and Q
b. Calculate N=P*Q
c. Select the public key E such that it is not a factor of (P-1) and (Q-1)
d. Select the private key D such that the following equation is true:
   1. (D*E)mod(P-1)(Q-1)=1
e. For encryption, calculate the cipher text CT from the plain text PT as follows:
   1. $CT=PT^E \bmod N$
f. Send CT as the cipher text to the receiver.
g. For decryption, calculate the plain text PT from the cipher text CT as follows:
   1. $PT=CT^D \bmod N$.

PROGRAM:

```
<html>
<head>
<title>Input</title>
<script language="JavaScript">
<!-- hide from old browsers
function gcd (a, b)
{
  var r;
  while (b>0)
  {
    r=a%b;
    a=b;
    b=r;
  }
  return a;
}

function rel_prime(phi)
{
  var rel=5;
  while (gcd(phi,rel)!=1)
    rel++;
  return rel;
}
```

```
function power(a, b)
{
  var temp=1, i;
  for(i=1;i<=b;i++)
    temp*=a;
   return temp;
}

function encrypt(N, e, M)
{
  var r,i=0,prod=1,rem_mod=0;
  while (e>0)
  {
    r=e % 2;
    if (i++==0)
      rem_mod=M % N;
    else
      rem_mod=power(rem_mod,2) % N;
    if (r==1)
    {
      prod*=rem_mod;
      prod=prod % N;
    }
    e=parseInt(e/2);
  }
  return prod;
}

function calculate_d(phi,e)
{
  var x,y,x1,x2,y1,y2,temp,r,orig_phi;
  orig_phi=phi;
  x2=1;x1=0;y2=0;y1=1;
  while (e>0)
  {
    temp=parseInt(phi/e);
    r=phi-temp*e;
    x=x2-temp*x1;
    y=y2-temp*y1;
    phi=e;e=r;
    x2=x1;x1=x;
    y2=y1;y1=y;
    if (phi==1)
    {
      y2+=orig_phi;
```

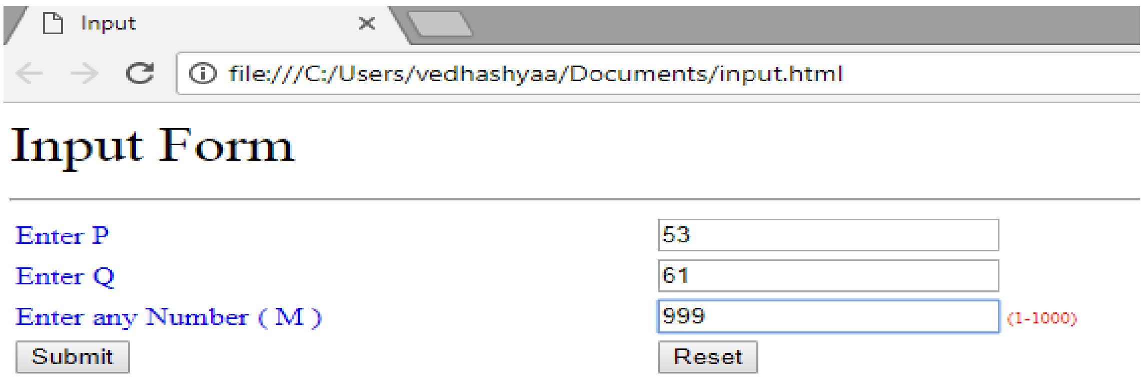
```
        break;
      }
    }
    return y2;
  }

  function decrypt(c, d, N)
  {
    var r,i=0,prod=1,rem_mod=0;
    while (d>0)
    {
      r=d % 2;
      if (i++==0)
        rem_mod=c % N;
      else
        rem_mod=power(rem_mod,2) % N;
      if (r==1)
      {
        prod*=rem_mod;
        prod=prod % N;
      }
      d=parseInt(d/2);
    }
    return prod;
  }


  function openNew()
  {
    var subWindow=window.open(
  "Output.htm", "Obj","HEIGHT=400,WIDTH=600,SCROLLBARS=YES");
    var p=parseInt(document.Input.p.value);
    var q=parseInt(document.Input.q.value);
    var M=parseInt(document.Input.M.value);
    var N=p * q;
    var phi=(p-1)*(q-1);
    var e=rel_prime(phi);
    var c=encrypt(N,e,M);
    var d=calculate_d(phi,e);
    subWindow.document.Output.N.value=N;
    subWindow.document.Output.phi.value=phi;
    subWindow.document.Output.e.value=e;
    subWindow.document.Output.c.value=c;
    subWindow.document.Output.d.value=d;
    subWindow.document.Output.M.value=decrypt(c,d,N);
  }
```

```
// end scripting here -->
</script>

</head>
<body>
<p><font size="6">Input Form</font></p>
<hr>
<form name="Input">
<table border="0" width="100%" height="109">
  <tr>
    <td width="24%" height="23">
        <font color="#0000FF">Enter P</font></td>
    <td width="76%" height="23">
         <input type="text" name="p" size="20"></td>
  </tr>
  <tr>
    <td width="24%" height="23"><font color="#0000FF">
            Enter Q</font></td>
    <td width="76%" height="23">
          <input type="text" name="q" size="20"></td>
  </tr>
  <tr>
    <td width="24%" height="20">
          <font color="#0000FF">Enter any Number ( M )</font></td>
    <td width="76%" height="20"><input type="text" name="M" size="20">
        <font size="1" color="#FF0000">(1-1000)</font></td>
  </tr>
  <tr>
    <td width="24%" height="19"><input type="button"
         value="Submit" name="Submit" onClick="openNew()"></td>
    <td width="76%" height="19"><input type="reset"
          value="Reset" name="Reset"></td>
  </tr>
</table>
</form>
<p>&nbsp;</p>
</body>
</html>

<html>
<head>
<title>Output</title>
</head>
<body>
<p><font size="6">Output Form</font></p>
```

```
<hr>
<p><font color="#FF0000">1.&nbsp;&nbsp;&nbsp; N = p * q
</font></p>
<p><font color="#FF0000">2.&nbsp;&nbsp;&nbsp;
phi = ( p - 1 ) * ( q - 1)&nbsp;&nbsp;&nbsp;&nbsp;</font></p>
<p><font color="#FF0000">3.&nbsp;&nbsp;&nbsp; GCD( phi , e ) = 1</font></p>
<p><font color="#FF0000">4.&nbsp;&nbsp;&nbsp; Encrypted Text ( c ) = M<sup>e</sup>
* ( mod N )</font></p>
<p><font color="#FF0000">5.&nbsp;&nbsp;&nbsp;e * d =&nbsp; 1 * ( mod phi )</font></p>
<p><font color="#FF0000">6.&nbsp;&nbsp;&nbsp; Decrypted Text = c<sup>d</sup> * (
mod N )</font></p>
<form name="Output">
<table border="0" width="100%">
  <tr>
    <td width="22%"><font color="#0000FF">N
      </font></td>
    <td width="78%"><input type="text" name="N"  size="20"></td>
  </tr>
  <tr>
    <td width="22%"><font color="#0000FF">Phi</font></td>
    <td width="78%"><input type="text"  name="phi" size="20"></td>
  </tr>
  <tr>
    <td width="22%"><font color="#0000FF">e
        </font></td>
    <td width="78%">
   <input type="text" name="e" size="20"></td>
  </tr>
  <tr>
    <td width="22%"><font color="#0000FF">Encrypted Text
     </font></td>
    <td width="78%">
    <input type="text" name="c" size="20"></td>
  </tr>
  <tr>
    <td width="22%"><font color="#0000FF">d
        </font></td>
    <td width="78%"><input type="text" name="d" size="20">
        </td>
  </tr>
  <tr>
    <td width="22%"><font color="#0000FF">
        Decrypted Text</font></td>
    <td width="78%"><input type="text" name="M" size="20"></td>
  </tr>
  <tr>
```

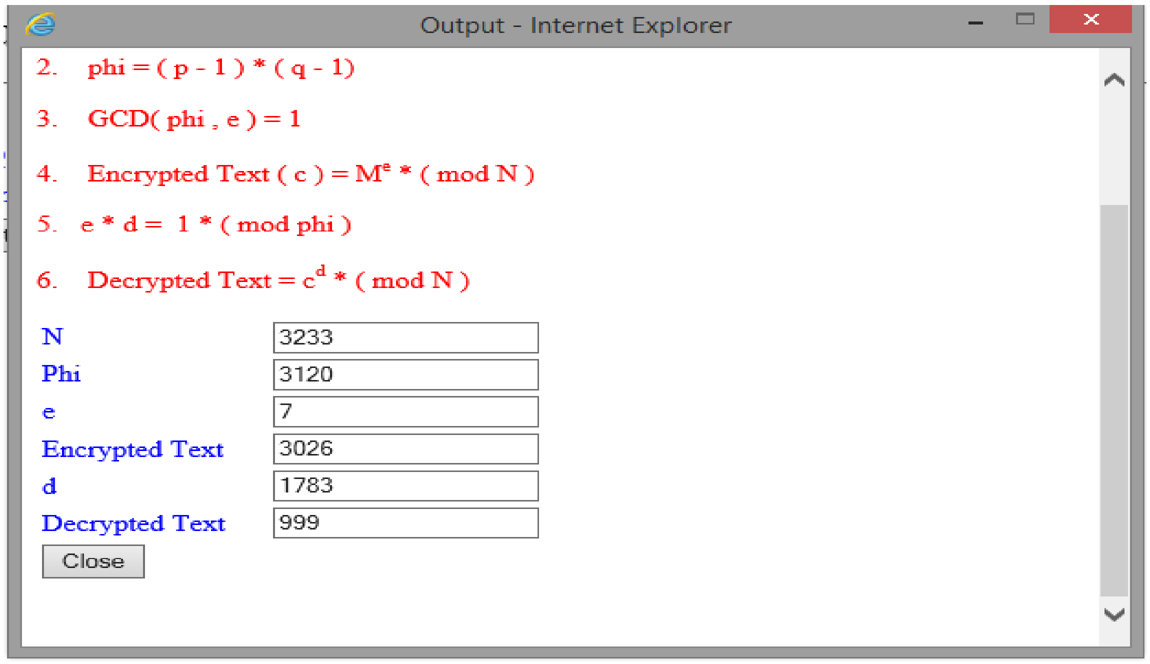
```
    <td width="22%">
      <input type="button" value="Close" name="Close"

    onClick="window.close()"></td>
    <td width="78%">&nbsp;</td>
  </tr>
</table>
</form>
<p>&nbsp;</p>

</body>
</html>
```

OUTPUT:



Input Form

Enter P	53
Enter Q	61
Enter any Number (M)	999 (1-1000)
Submit	Reset



2. phi = (p - 1) * (q - 1)
3. GCD(phi , e) = 1
4. Encrypted Text (c) = M^e * (mod N)
5. e * d = 1 * (mod phi)
6. Decrypted Text = c^d * (mod N)

N	3233
Phi	3120
e	7
Encrypted Text	3026
d	1783
Decrypted Text	999

RESULT:

Thus the HTML program to implement RSA was executed and verified successfully.

VIVA QUESTIONS:

1. Write a short note on RSA algorithm.
2. What is meant by quantum cryptogaphy?
3. Can users of RSA run out of distinct primes?
4. How fast is RSA algorithm?
5. What would it take to break RSA?

EX.NO: 6 **IMPLEMENTATION OF DIFFIE HELLMAN ALGORITHM**
DATE:

AIM:

To implement the Diffie Hellman Key Exchange mechanism using java.

ALGORITHM:

Step1: Firstly Alice and bob agree on two large prime numbers n and g. These two integers need not be kept secret. Alice and bob can use an insecure channel to agree on them.
Step2: Alice chooses another large random number X and calculates A such that
$A=g^X \bmod n$
Step3: Alice sends the number A to bob.
Step4: Bob independently chooses another large random integer y and calculates B such that
$B=g^Y \bmod n$
Step5: Bob sends the number B to Alice.
Step6: A now computes the secret key K1 as follows:
$K1=B^X \bmod n$
Step7: B now computes the secret key K2 as follows
$K2 =A^Y \bmod n$
Here K1=K2=K is the symmetric key, which alice and bob must keep secret.

PROGRAM:

```
import java.math.BigInteger;
import java.security.KeyFactory;
import java.security.KeyPair;
import java.security.KeyPairGenerator;
import java.security.SecureRandom;
import javax.crypto.spec.DHParameterSpec;
import  javax.crypto.spec.DHPublicKeySpec;
public class DiffeHellman {
public final static int pValue = 47;
public final static int gValue = 71;
public final static int XaValue = 9;
public final static int XbValue = 14;
public static void main(String[] args) throws Exception {
      // TODO code application logic here
BigInteger p = new BigInteger(Integer.toString(pValue));
BigInteger g = new BigInteger(Integer.toString(gValue));
BigInteger Xa = new BigInteger(Integer.toString(XaValue));
BigInteger Xb = new BigInteger(Integer.toString(XbValue));
createKey();
int bitLength = 512; // 512 bits
SecureRandom rnd = new SecureRandom();
      p = BigInteger.probablePrime(bitLength, rnd);
      g = BigInteger.probablePrime(bitLength, rnd);
createSpecificKey(p, g);
   }
public static void createKey() throws Exception {
KeyPairGenerator kpg = KeyPairGenerator.getInstance("DiffieHellman");
kpg.initialize(512);
KeyPair kp = kpg.generateKeyPair();
KeyFactory kfactory = KeyFactory
.getInstance("DiffieHellman");
DHPublicKeySpec kspec = (DHPublicKeySpec) kfactory.getKeySpec(kp.getPublic(),
DHPublicKeySpec.class);
System.out.println("Public key is: " +kspec);
   }
public static void createSpecificKey(BigInteger p, BigInteger g) throws Exception {
KeyPairGenerator kpg = KeyPairGenerator.getInstance("DiffieHellman");
DHParameterSpec param = new DHParameterSpec(p, g);
kpg.initialize(param);
KeyPair kp = kpg.generateKeyPair();
KeyFactory kfactory = KeyFactory.getInstance("DiffieHellman");
DHPublicKeySpec kspec = (DHPublicKeySpec) kfactory.getKeySpec(kp.getPublic(),
DHPublicKeySpec.class);
System.out.println("\nPublic key is : " +kspec);
   }
}
```

OUTPUT:

```
Public key is: javax.crypto.spec.DHPublicKeySpec@2f92e0f4

Public key is : javax.crypto.spec.DHPublicKeySpec@28a418fc
BUILD SUCCESSFUL (total time: 0 seconds)
```

RESULT:

Thus the java program to implement Diffie Hellman Key exchange algorithm was executed and verified successfully.

VIVA QUESTIONS:

1. What is elliptic curve cryptography?

2. What is the primitive root of a number?

3. What is meant by a one way function?

4. Describe in general terms an efficient procedure for picking a prime number

5. Define Fermat Theorem

EX.NO: 7 **IMPLEMENTATION OF SHA-1 ALGORITHM**
DATE:

AIM:

To calculate the message digest of a text using the SHA-1 algorithm in JAVA.

ALGORITHM:

Step 1: Append Padding Bits….
Message is "padded" with a 1 and as many 0‟s as necessary to bring the message length to 64 bits fewer than an even multiple of 512.

Step 2: Append Length....
64 bits are appended to the end of the padded message. These bits hold the binary format of 64 bits indicating the length of the original message.

Step 3: Prepare Processing Functions….
SHA1 requires 80 processing functions defined as:
f(t;B,C,D) = (B AND C) OR ((NOT B) AND D) (0 <= t <= 19)
f(t;B,C,D) = B XOR C XOR D (20 <= t <= 39)
f(t;B,C,D) = (B AND C) OR (B AND D) OR (C AND D) (40 <= t <=59)
f(t;B,C,D) = B XOR C XOR D (60 <= t <= 79)

Step 4: Prepare Processing Constants....
SHA1 requires 80 processing constant words defined as:

K(t) = 0x5A827999	(0 <= t <= 19)
K(t) = 0x6ED9EBA1	(20 <= t <= 39)
K(t) = 0x8F1BBCDC	(40 <= t <= 59)
K(t) = 0xCA62C1D6	(60 <= t <= 79)

Step 5: Initialize Buffers….
SHA1 requires 160 bits or 5 buffers of words (32 bits):
H0 = 0x67452301
H1 = 0xEFCDAB89
H2 = 0x98BADCFE
H3 = 0x10325476
H4 = 0xC3D2E1F0

Step 6: Processing Message in 512-bit blocks (L blocks in total message)….
This is the main task of SHA1 algorithm which loops through the padded and appended message in 512-bit blocks.
Input and predefined functions:
M[1, 2, ..., L]: Blocks of the padded and appended message f(0;B,C,D), f(1,B,C,D),...., f(79,B,C,D): 80 Processing Functions K(0), K(1),... , K(79): 80 Processing Constant Words
H0, H1, H2, H3, H4, H5: 5 Word buffers with initial values

PROGRAM:

```
import java.security.*;
public class SHA1 {
public static void main(String[] a) {
try {
```

```
MessageDigest md = MessageDigest.getInstance("SHA1");
System.out.println("Messagedigest 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'};
StringBuffer buf = new StringBuffer();
for (int j=0; j<b.length; j++) {
buf.append(hexDigit[(b[j] >> 4) & 0x0f]);
buf.append(hexDigit[b[j] & 0x0f]); }
return buf.toString(); }
}
```

OUTPUT:

```
Messagedigest object info:
  Algorithm = SHA1
  Provider = SUN version 1.8
  ToString = SHA1 Message Digest from SUN, <initialized>


SHA1("") = DA39A3EE5E6B4B0D3255BFEF95601890AFD80709
SHA1("abc") = A9993E364706816ABA3E25717850C26C9CD0D89D
SHA1("abcdefghijklmnopqrstuvwxyz") = 32D10C7B8CF96570CA04CE37F2A19D84240D3A89
BUILD SUCCESSFUL (total time: 0 seconds)
```

RESULT:

Thus the java program to implement SHA1 algorithm was executed and verified successfully.

VIVA QUESTIONS:

1. What is message authentication?

2. Specify the various types of authentication protocol..

3. What are the requirements for message authentication?

4. What is meant by hash function?

5. Differentiate MAC and Hash function

EX.NO: 8 **IMPLEMENTATION OF THE SIGNATURE SCHEME –**
DATE: **DIGITAL SIGNATURE STANDARD**

AIM:

To implement the Signature Scheme for Digital Signature Standard.

ALGORITHM:

Step1: A digital signature may be formed by encrypting the entire message with the senders private key or by encrypting a hash code of the message with the sender‟s private key.
Step2: Confidentiality can be provided by further encrypting the entire message plus signature with either the receiver‟s public key or shared secret key.
Step3: If the signature is calculated on an encrypted message, then the third party also needs access to the decryption key to read the original message.
Step4: The validity of this scheme depends on the security of the sender‟s private key.

PROGRAM:

```
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
KeyPairGenerator kpg = KeyPairGenerator.getInstance("RSA");
kpg.initialize(1024);
KeyPair keyPair = 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:
NhNPYYeZooeSv49Ia745oH59yTPWyOga5sHiaY9Dsx7GDbO89ci8N/vnAkVhxxXClCJM247M+gLc
qglJVjnWEai3sIJct4KQYZ9KlhzaTaa+OVziRRS8rRQYN3OwxH4pDIbih1EnSILqT1iMih64ofNy
JpBx4I2v8+YEHiiPPoY=
true
BUILD SUCCESSFUL (total time: 0 seconds)
```

RESULT:

Thus the java program to implement Digital Signature was performed and verified successfully.

VIVA QUESTIONS:

1. Distinguish between direct and arbitrated digital signature.
2. What are the two approaches of digital signature?
3. What requirements should a digital signature scheme should satisfy?
4. What is the need of Dual signature?
5. In the content of Kerberos , What is realm?

EX.NO:9 DEMONSTRATE INTRUSION DETECTION SYSTEM (IDS) USING ANY TOOL (SNORT OR ANY OTHER SOFTWARE.)

DATE:

AIM :

To demonstrate Intrusion Detection System (IDS) using any tool such as Snort or any other Software.

DESCRIPTION:

Building a Snort Windows System

Step1: Download a copy of Winpcap.exe from www.winpcap.org . This low-level packet driver will be needed to get Snort to work. After you install Win-Pcap, reboot if prompted.
Step2: Download the latest version of Snort from www.snort.org/dl/binaries/win32/ . At the time of this writing, the version is 2.80. After starting the download, start the Snort install.
Step3: Agree to accept the license agreement.
Step4: Check Support for Flexibility Response, and then click Next.
Step5: Verify that all components are checked, and then click Next to continue the installation.
Step6: Accept the defaults for location, and then click Install. The folder C:\Snort will be used.
Step7: Click Close to finish the Snort installation. During the actual installation, Snort creates a directory structure under C:\Snort that looks like this:

```
C:\snort\bin
C:\snort\contrib
C:\snort\doc
C:\snort\etc
C:\snort\log
C:\snort\rules
```

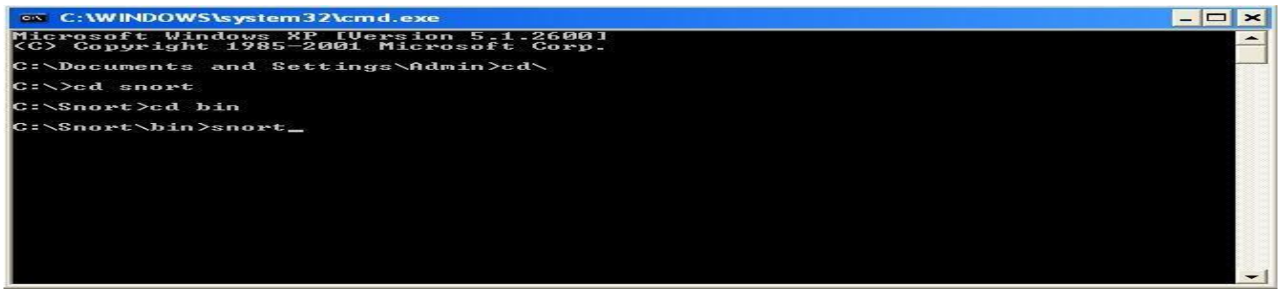
Step8: If necessary, click OK to close the Snort Setup information box.
Step9: In the snort.conf file, search for the variable statement that begins with var rule_path . If necessary, change the statement to refer to the path of your Snort rules folders, which is the var RULE_PATH c:\snort\rules .
Step10: Search for the variable statement var HOME_NET Any . Change it to the setting for your network (e.g., var HOME_NET 172.16.0.0/24).
Step11: Search for the statement include classification.config and change it to include c:\snort\etc\classification.config
Step12: Search for the statement include reference.config and change it to include c:\snort\etc\reference.config
Step13: Save and close the file.
Step14: Reboot your machine and log back on to Windows. To check that Snort was properly configured, open two command prompts.
Step15: At one of the command prompts, navigate to the C:\snort\bin folder , and enter snort –W. You should see a list of possible adapters on which you can install the sensor. The adapters are numbered 1, 2, 3, and so forth.
Step16: At the c:\snort\bin› prompt, enter snort –v –ix, where x is the number of the NIC to place your Snort sensor on.
Step17: Switch to the second command prompt you opened, and ping another computer, such as the gateway. When the ping is complete, switch back to the first command-prompt window running Snort, and press Ctrl+C to stop Snort.

A sample capture is shown here:

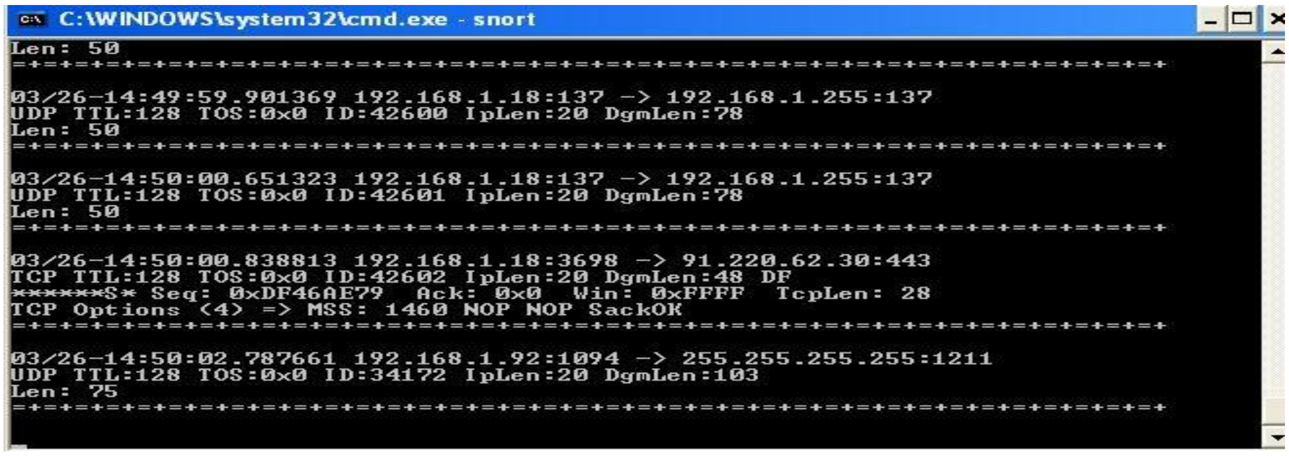
```
11/01-23:09:51.398772 192.168.13.10 -› 192.168.13.254
ICMP TTL:64 TOS:0x0 ID:38
ID:1039 Seq:0 ECHO
9E 85 00 3B 84 15 06 00 08 09 0A 0B 0C 0D 0E 0F ...:...........
10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F ................
20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F !"#$%&‟()*+,-./
30 31 32 33 34 35 36 37
01234567
```

This demonstrates the basic capabilities of Snort, but not everyone has the time or ability to constantly monitor the console. Therefore what is needed is a way to log the activity for later review. We can do this as follows:

1. If you are not already there, change to the directory where you installed Snort. Then at the command prompt, enter snort –ix –dev –l\snort\log. This command will start Snort and instruct it to record headers in the \snort\log folder.
2. Now ping some other device, such as the gateway. If you have a second computer on the network, you can use it to ping that computer, or you can even scan it with Nmap. The idea here is to generate some traffic to be logged in the Snort\log folder for review.
3. After you have generated some ping traffic or run some scans against the local machine, press Ctrl+C to stop the packet capture.
4. Use Windows Explorer to navigate to the snortlog folder.
5. You should see some files there. Use Notepad to examine the contents of the capture. (This is a great feature because now you can go back and review activity.)



```
Microsoft Windows XP [Version 5.1.2600]
<C> Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Admin>cd\

C:\>cd snort

C:\Snort>cd bin

C:\Snort\bin>snort_
```



```
Len: 50
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

03/26-14:49:59.901369 192.168.1.18:137 -> 192.168.1.255:137
UDP TTL:128 TOS:0x0 ID:42600 IpLen:20 DgmLen:78
Len: 50
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

03/26-14:50:00.651323 192.168.1.18:137 -> 192.168.1.255:137
UDP TTL:128 TOS:0x0 ID:42601 IpLen:20 DgmLen:78
Len: 50
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

03/26-14:50:00.838813 192.168.1.18:3698 -> 91.220.62.30:443
TCP TTL:128 TOS:0x0 ID:42602 IpLen:20 DgmLen:48 DF
******S* Seq: 0xDF46AE79  Ack: 0x0  Win: 0xFFFF  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

03/26-14:50:02.787661 192.168.1.92:1094 -> 255.255.255.255:1211
UDP TTL:128 TOS:0x0 ID:34172 IpLen:20 DgmLen:103
Len: 75
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

RESULT:

Thus the demonstration of Intrusion Detection System (IDS) using Snort is performed.

VIVA QUESTIONS:

1. Name the protocols that provide security in IPSec.
2. What is meant by security association?
3. Define transport and tunnel mode.
4. What is the need of an anti-replay service?
5. How can the security associations be combined?

EX.NO: 10 **AUTOMATED ATTACK AND PENETRATION TOOLS.**
DATE:

AIM:

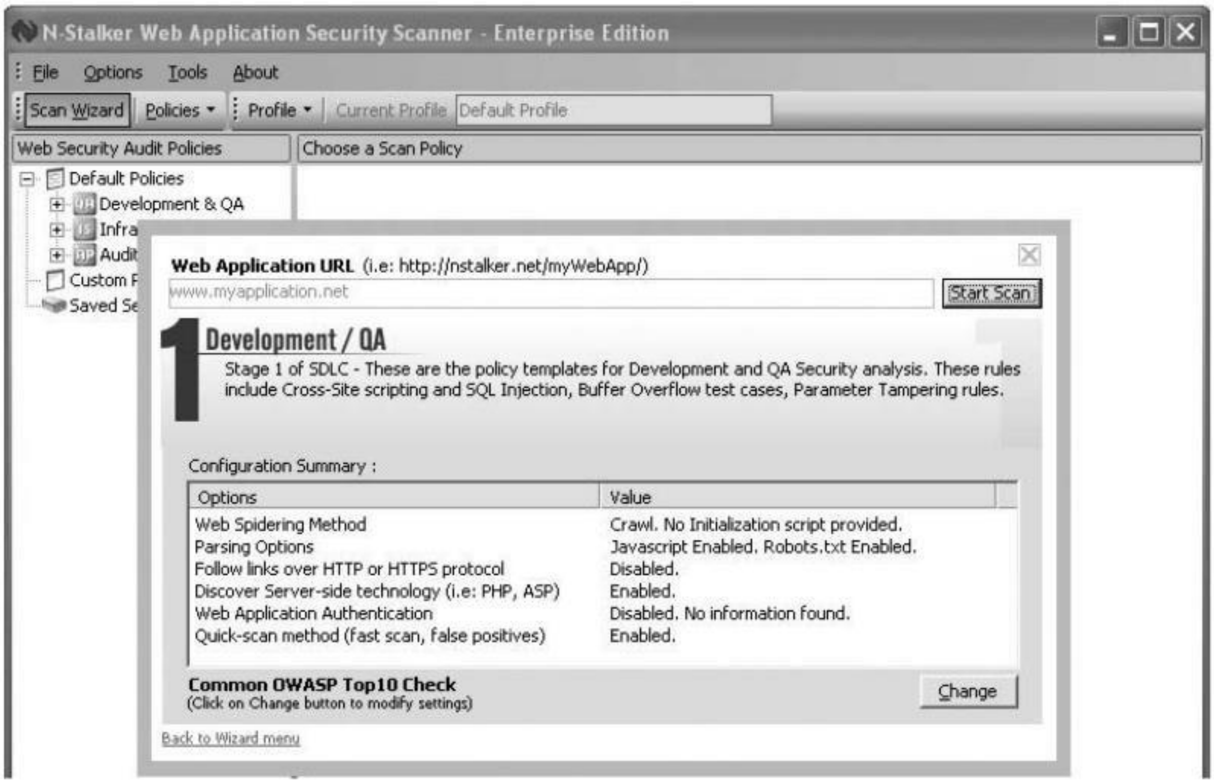
To explore N-Stalker for automated attack detection and study variety of options.

DESCRIPTION:

Exploring N-Stalker, a Vulnerability Assessment Tool
N-Stalker is a web server security-auditing tool that scans for more than 30,000 vulnerabilities. You need to download and install N-Stalker from www.nstalker.com.

Step1: Start N-Stalker from a Windows computer. The program is installed under Start➪Programs➪N-Stalker➪N-Stalker Free Edition. You will be presented with the startup screen shown in Figure 6-21.
Step2: Enter a host address or a range of addresses to scan.
Step3: Click Start Scan.
Step4: After the scan completes, the N-Stalker Report Manager will prompt you to select a format for the resulting report as choose Generate HTML.
Step5: Review the HTML report for vulnerabilities.

Armed with this report, your next step should be to set priorities on which services should be patched and hardened.



Web Application URL (i.e: http://nstalker.net/myWebApp/)
www.myapplication.net

Development / QA
Stage 1 of SDLC - These are the policy templates for Development and QA Security analysis. These rules include Cross-Site scripting and SQL Injection, Buffer Overflow test cases, Parameter Tampering rules.

Configuration Summary :

Options	Value
Web Spidering Method	Crawl. No Initialization script provided.
Parsing Options	Javascript Enabled. Robots.txt Enabled.
Follow links over HTTP or HTTPS protocol	Disabled.
Discover Server-side technology (i.e: PHP, ASP)	Enabled.
Web Application Authentication	Disabled. No information found.
Quick-scan method (fast scan, false positives)	Enabled.

Common OWASP Top10 Check
(Click on Change button to modify settings)

Figure 6-21 N-Stalker.

Exploring the SecurityForest.com Web Site

SecurityForest.com is a collaboratively edited forest consisting of trees that anyone can contribute to. These exploit trees break out in an orderly fashion so that they display the tools and exploits available for each step of a penetration test and for the exploits available for specific networks, systems, and applications:

Step1: Open your browser and go to www.securityforest.com.
Step2: Notice on the left of the screen that several trees are listed.
Step3: Click the Exploit Tree online interface.
Step4: This page will have links for applications, systems, and networks. Click the Applications link.
Step5: On this page, you will see links for all the applications that have been listed in the database. Find the link for web servers, and click the link for the IIS application.
Step6: Under IIS, locate theJill-win32.cexploit code. After you have found the code, you can view it by clicking the View button. If you have identified an IIS server susceptible to the IPP printer buffer overflow, this tool could be compiled and executed to take advantage of that vulnerability.
Step7: Continue to explore the SecurityForest web site. If you return to the main page, you will see that there is also a database of tools under the Tooltree link that lists all tools by category.
Step8: Finally, click the Exploitation Framework link.

The Exploitation Frame-work is similar to the Metasploit database except that it leverages the huge amount of exploits in the exploit tree. An available AVI movie demonstrates the tool at the www.securityforest.com/wiki/index.php/ExploitationFrameworkScreenshots page.
You can download the actual browser-based Windows tool from www.securityforest.com/wiki/index.php/ExploitationFrameworkDownload.

RESULT:

Thus the N-stalker is explored and studied variety of options.

VIVA QUESTIONS:

1. What is meant by SET?
2. What are the features of SET?
3. List the 3 classes of intruder
4. Define virus.
5. Specify the types of viruses

EX.NO:11a

DEFEATING MALWARE
(BUILDING TROJANS)

DATE:

AIM :

To simulate, how to build Trojans and malware technique to distribute.

DESCRIPTION:

Building Trojans

By default, most Windows systems automatically start a CD when it is inserted in the CD tray. You use this technique to distribute simulated malicious code. You need a blank CD and a CD burner for this exercise.

Step1: Create a text file named autorun.ini. Inside this text file, add the following contents:

```
[autorun]
Open paint.exe
Icon=paint.exe
```

Step2: Place the autorun.ini file and a copy of paint.exe into a folder to be burned to a CD.

Step3: After you have completed making the CD, reinsert it in the CD-ROM drive and observe the results. It should auto start and automatically start the Paint program.

Step4: Think about the results. While this exercise was benign, you could have just as easily used a Trojan program that had been wrapped with a legitimate piece of software. Just leaving the CD lying around or giving it an attractive title, such as „„pending 2006 bonuses,"" might lead some-one to pick it up to see exactly what it is. Anyone running the CD would then become infected. Even with Auto Run turned off, all it would take is for the user to double-click the CD-ROM icon and the program would still run.

RESULT:

Thus the demonstration of Intrusion Detection System (IDS) using Snort is performed.

VIVA QUESTIONS:

1. What is meant by VeriSign certificate?
2. What are the function areas of IP security?
3. Give the applications of IP security.
4. Give the benefits of IP security
5. Specify the IP security services.

EX.NO:11b **ROOTKIT HUNTER**

DATE:

AIM:

To perform Installation of rootkits and study variety of options.

DESCRIPTION:

Rootkits

Rootkit Hunter is an open source tool that checks Linux-based systems for the presence of rootkits and other unwanted tools. You can down-load and run this program on any Linux system. Rootkit Hunter can be downloaded from www.rootkit.nl/projects/rootkithunter.html.

Step1: Once you have started your Linux system, open a root terminal and download Rootkit Hunter. Enter the following at the command-line shell:

```
wget http://downloads.rootkit.nl/rkhunter-<version>.tar.gz
```

The <version> syntax will require you to enter the current version of the software. At the time of this writing, version 1.3.0 is the most current version.

Step2: When the download has completed, unpack the archived file. You can do so by entering the following command:

```
tar zxf rkhunter-<version>.tar.gz
```

Step3:The preceding command extracts Rootkit Hunter. Next, you want to install Rootkit Hunter. You need to change directories to the RootkitHunter folder:

```
cd rkhunter
```

Step4: After you are in the proper directory, run the installer. This will complete the installation. To accomplish this, enter the following:./installer.sh

Step5: If everything has gone correctly, the installation should have finished successfully. The code listed here shows the syntax of a successful installation:

```
Rootkit Hunter installer 1.2.4 (Copyright 2003-2005, Michael Boelen)
--------------
Starting installation/update

Checking /usr/local... OK
Checking file retrieval tools... /usr/bin/wget
Checking installation directories...
- Checking /usr/local/rkhunter...Exists
- Checking /usr/local/rkhunter/etc...Exists
- Checking /usr/local/rkhunter/bin...Exists
- Checking /usr/local/rkhunter/lib/rkhunter/db...Exists
- Checking /usr/local/rkhunter/lib/rkhunter/docs...Exists
- Checking /usr/local/rkhunter/lib/rkhunter/scripts...Exists
- Checking /usr/local/rkhunter/lib/rkhunter/tmp...Exists
- Checking /usr/local/etc...Exists
- Checking /usr/local/bin...Exists
Checking system settings...
- Perl... OK
Installing files...
Installing Perl module checker... OK
Installing Database updater... OK
Installing Portscanner... OK
Installing MD5 Digest generator... OK
Installing SHA1 Digest generator... OK
Installing Directory viewer... OK
Installing Database Backdoor ports... OK
Installing Database Update mirrors... OK
Installing Database Operating Systems... OK
Installing Database Program versions... OK
Installing Database Program versions... OK
Installing Database Default file hashes... OK
Installing Database MD5 blacklisted files... OK
Installing Changelog... OK
Installing Readme and FAQ... OK
Installing Wishlist and TODO... OK
Installing RK Hunter configuration file... Skipped (no overwrite)
Installing RK Hunter binary... OK
Configuration already updated.

Installation ready.
See /usr/local/rkhunter/lib/rkhunter/docs for more information.
Run "rkhunter" (/usr/local/bin/rkhunter)
```

Step6: With Rootkit Hunter installed, you can now run the program. There is avariety of options that can be used. To perform a complete check of thesystem, run this:

```
Rkhunter –checkall
```

Step7: Rootkit Hunter can search for many different types of rootkits. A partial list is shown here:

55808 Trojan - Variant A
ADM W0rm
AjaKit
aPa Kit
Apache Worm
Ambient (ark) Rootkit
Balaur Rootkit
BeastKit
beX2
BOBKit
CiNIK Worm (Slapper.B variant)
Danny-Boy‟s Abuse Kit
Devil RootKit
Dica
Dreams Rootkit
Duarawkz Rootkit
Flea Linux Rootkit
FreeBSD Rootkit
TeLeKiTT0rn Rootkit
Trojanit KitURK (Universal RootKit)
VcKit
Volc Rootkit
X-Org SunOS Rootkit

zaRwT.KiT Rootkit

Step8: When the scan is completed, you should receive a message similar to the following:

```
------------------------------Scan results ------------------------------
MD5
MD5 compared: 0
Incorrect MD5 checksums: 0
File scan Scanned files: 399
Possible infected files: 0

Application scan
Vulnerable applications: 9
Scanning took 15748 seconds
-------------------------------------------------------------------
Do you have some problems, undetected rootkits, false positives, ideas or suggestions? Please
email me by filling in the contact form (@http://www.rootkit.nl)
-------------------------------------------------------------------
```

In this exercise, we were fortunate to find that the system had not been infected. But had it been, you would have been faced with many challenges. This is primarily because it‟s almost impossible to clean up a rootkit. Because hiding is its main purpose, it is difficult to tell whether all remnants of the infection have been removed. You should always rebuild from known good media.

RESULT:

Thus the Rootkit is installed and studied variety of options.

VIVA QUESTIONS:

1. What is meant by data transmission?
2. Give the significance of exporting key.
3. What is meant by trust level?
4. State about signing file.
5. What is the use of generate key?