

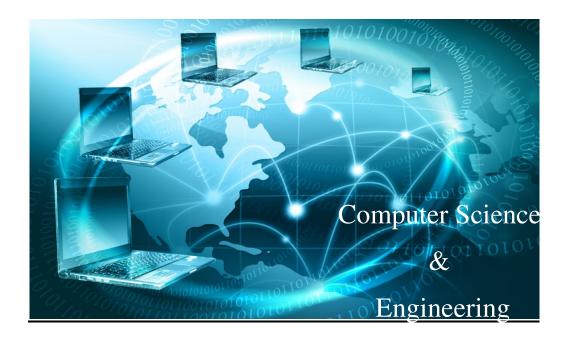
# Varuvan Vadivelan Institute of Technology

# Dharmapuri – 636 703

# LABORATORY MANUAL

Regulation	:	2013
Branch	•	B.E – CSE
Year / Semester	•	IV/ VII

# CS6711 – SECURITY LABORATORY



PREPARED BY B.THENMOZHI AP/CSE

# ANNA UNIVERSITY CHENNAI

# **REGULATION-2013**

# **CS6711 SECURITY LABORATORY**

# **OBJECTIVES:**

# The student should be made to:

- ▶ Learn to implement the algorithms DES, RSA, MD5, SHA-1
- > Learn to use network security tools like GnuPG, KF sensor, Net Strumbler

# LIST OF EXPERIMENTS:

- 1. Implement the following SUBSTITUTION & TRANSPOSITION TECHNIQUES concepts:
  - a) Caesar Cipher
  - b) Playfair Cipher
  - c) Hill Cipher
  - d) Vigenere Cipher
  - e) Rail fence row & Column Transformation
- **2.** Implement the following algorithms
  - a) DES
  - b) RSA Algorithm
  - c) Diffiee-Hellman
  - d) MD5
  - e) SHA-1
- 3. Implement the Signature Scheme Digital Signature Standard
- **4.** Demonstrate how to provide secure data storage, secure data transmission and for creating digital signatures (GnuPG)
- 5. Setup a honey pot and monitor the honeypot on network (KF Sensor)
- 6. Installation of rootkits and study about the variety of options
- 7. Perform wireless audit on an access point or a router and decrypt WEP and WPA. (Net Stumbler)
- 8. Demonstrate intrusion detection system (ids) using any tool (snort or any other s/w)

# **TOTAL: 45 PERIODS**

# **OUTCOMES:**

### At the end of the course, the student should be able to:

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

# LIST OF HARDWARE REQUIREMENTS & SOFTWARE REQUIREMENTS

# SOFTWARE REQUIREMENTS

- ▶ C
- ≻ C++
- Java or equivalent compiler GnuPG
- ➢ KF Sensor or Equivalent
- Snort
- Net Stumbler or Equivalent

# HARDWARE REQUIREMENTS

Standalone desktops (or) Server supporting 30 terminals or more

1(A)Caesar CipherImage: CipherImage: Cipher1(B)Playfair CipherImage: CipherImage: CipherImage: Cipher1(C)Hill CipherImage: CipherImage: CipherImage: Cipher1(D)Vigenere CipherImage: CipherImage: CipherImage: Cipher1(E)Rail fence – row & Column TransformationImage: CipherImage: Cipher2(A)Data Encryption Standard(DES)Image: CipherImage: Cipher2(B)RSA AlgorithmImage: CipherImage: Cipher2(C)Diffiee-Hellman AlgorithmImage: CipherImage: Cipher2(D)MD5Image: CipherImage: CipherImage: Cipher3Implement the Signature Scheme for Digital Signature StandardImage: CipherImage: Cipher4Demonstrate how to provide secure data storage, secure data transmission and for creating digital signatures (GnuPG)Image: CipherImage: Cipher5Setup a honey pot and monitor the honeypot on network (KF Sensor)Image: CipherImage: CipherImage: Cipher6Installation of rootkits and study about the variety of optionsImage: CipherImage: CipherImage: CipherImage: Cipher7Perform wireless audit on an access point or a router and decrypt WEP and WPA (Net Stumbler)Image: CipherImage: CipherImage: Cipher	S.No.	Name of the Program	Page No.	Staff Signature	Remarks
1C.Hill CipherImage: Cipher1(D)Vigenere CipherImage: Cipher1(E)Rail fence – row & Column TransformationImage: Cipher2(A)Data Encryption Standard(DES)Image: Cipher2(B)RSA AlgorithmImage: Cipher2(C)Diffiee-Hellman AlgorithmImage: Cipher2(D)MD5Image: Cipher2(E)SHA-1Image: Cipher3Implement the Signature Scheme for Digital Signature StandardImage: Cipher4Demonstrate how to provide secure data storage, secure data transmission and for creating digital signatures (GnuPG)Image: Cipher5Setup a honey pot and monitor the honeypot on network (KF Sensor)Image: Cipher6Installation of rootkits and study about the variety of optionsImage: Cipher7Perform wireless audit on an access point or a router and decrypt WEP and WPA (Net Stumbler)Image: Cipher	1(A)	Caesar Cipher			
1(D)Vigenere CipherI1(E)Rail fence - row & Column TransformationII2(A)Data Encryption Standard(DES)II2(B)RSA AlgorithmII2(C)Diffiee-Hellman AlgorithmII2(D)MD5III2(E)SHA-1III3Implement the Signature Scheme for Digital Signature StandardII4Demonstrate how to provide secure data storage, secure data transmission and for creating digital signatures (GnuPG)II5Setup a honey pot and monitor the honeypot on network (KF Sensor)III6Installation of rootkits and study adout the variety of optionsIII7Perform wireless audit on an access point or a router and decrypt WEP and WPA (Net Stumbler)III	1(B)	Playfair Cipher			
1(E)Rail fence – row & Column TransformationIII2(A)Data Encryption Standard(DES)III2(B)RSA AlgorithmIII2(C)Diffiee-Hellman AlgorithmIII2(D)MD5IIII2(E)SHA-1IIII3Implement the Signature Scheme for Digital Signature StandardIII4Demonstrate how to provide secure data storage, secure data transmission and for creating digital signatures (GnuPG)III5Setup a honey pot and monitor the honeypot on network (KF Sensor)IIII6Installation of rootkits and study about the variety of optionsIIII7Perform wireless audit on an access point or a router and decrypt WEP and WPA (Net Stumbler)IIII	1(C)	Hill Cipher			
TransformationImage: Constraint of the second s	1(D)	Vigenere Cipher			
2(B)RSA AlgorithmImage: Constraint of the second sec	1(E)				
2(C)Diffiee-Hellman AlgorithmImage: Constraint of the second of th	2(A)	Data Encryption Standard(DES)			
2(D)MD5Implement the Signature Scheme for Digital Signature StandardImplement the Signature Scheme for Digital Signature Standard4Demonstrate how to provide secure data storage, secure data transmission and for creating digital signatures (GnuPG)Implement the Signature Standard5Setup a honey pot and monitor the honeypot on network (KF Sensor)Implement the variety of options6Installation of rootkits and study about the variety of optionsImplement the variety of options7Perform wireless audit on an access point or a router and decrypt WEP and WPA (Net Stumbler)Implement the variety of options	2(B)	RSA Algorithm			
2(E)SHA-1Implement the Signature Scheme for Digital Signature StandardImplement the Signature Scheme for Digital Signature Standard4Demonstrate how to provide secure data storage, secure data transmission and for creating digital signatures (GnuPG)Implement the Signature Standard5Setup a honey pot and monitor the honeypot on network (KF Sensor)Implement the variety of options6Installation of rootkits and study about the variety of optionsImplement the variety of options7Perform wireless audit on an access point or a router and decrypt WEP and WPA (Net Stumbler)Implement the variety of options	2(C)	Diffiee-Hellman Algorithm			
3       Implement the Signature Scheme for Digital Signature Standard       Implement the Signature Scheme for Digital Signature Standard         4       Demonstrate how to provide secure data storage, secure data transmission and for creating digital signatures (GnuPG)       Implement the Signature S(GnuPG)         5       Setup a honey pot and monitor the honeypot on network (KF Sensor)       Implement the variety of options         6       Installation of rootkits and study about the variety of options       Implement the variety of options         7       Perform wireless audit on an access point or a router and decrypt WEP and WPA (Net Stumbler)       Implement the secure secure secure and secure data transmission and for creating digital signatures (GnuPG)	2(D)	MD5			
Image: A strange of the secure of the secure data storage, secure data transmission and for creating digital signatures (GnuPG)       Image: A secure data transmission and for creating digital signatures (GnuPG)         5       Setup a honey pot and monitor the honeypot on network (KF Sensor)       Image: A secure data study about the variety of options         6       Installation of rootkits and study about the variety of options       Image: A secure data study and WPA (Net Stumbler)	2(E)	SHA-1			
data storage, secure data transmission and for creating digital signatures (GnuPG)5Setup a honey pot and monitor the honeypot on network (KF Sensor)6Installation of rootkits and study about the variety of options7Perform wireless audit on an access point or a router and decrypt WEP and WPA (Net Stumbler)	3				
honeypot on network (KF Sensor)	4	data storage, secure data transmission and for			
about the variety of options	5				
point or a router and decrypt WEP and WPA ( Net Stumbler)	6				
	7	point or a router and decrypt WEP			
		( Net Stumbler)			
8 Demonstrate intrusion detection system (ids) using any tool	8	Demonstrate intrusion detection system (ids) using any tool			

# EX. NO: 1(A)

# **IMPLEMENTATION OF CAESAR CIPHER**

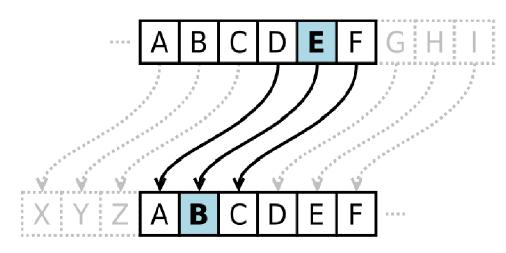
### AIM:

To implement the simple substitution technique named Caesar cipher using C language.

# **DESCRIPTION:**

To encrypt a message with a Caesar cipher, each letter in the message is changed using a simple rule: shift by three. Each letter is replaced by the letter three letters ahead in the alphabet. A becomes D, B becomes E, and so on. For the last letters, we can think of the alphabet as a circle and "wrap around". W becomes Z, X becomes A, Y becomes B, and Z becomes C. To change a message back, each letter is replaced by the one three before it.

### EXAMPLE:



#### **ALGORITHM:**

**STEP-1:** Read the plain text from the user.

**STEP-2:** Read the key value from the user.

**STEP-3:** If the key is positive then encrypt the text by adding the key with each character in the plain text.

**STEP-4:** Else subtract the key from the plain text.

**STEP-5:** Display the cipher text obtained above.

### **PROGRAM:** (Caesar Cipher)

```
#include <stdio.h>
#include <string.h>
#include<conio.h>
#include <ctype.h>
void main()
```

```
{
 char plain[10], cipher[10];
 int key, i, length;
 int result;
 clrscr();
printf("\n Enter the plain text:");
 scanf("%s", plain);
printf("\n Enter the key value:");
 scanf("%d", &key);
printf("\n \n \t PLAIN TEXt: %s",plain);
printf("\n \n \t ENCRYPTED TEXT: ");
 for(i = 0, length = strlen(plain); i < length; i++)</pre>
     ſ
     cipher[i]=plain[i] + key;
     if (isupper(plain[i]) && (cipher[i] > 'Z'))
     cipher[i] = cipher[i] - 26;
     if (islower(plain[i]) && (cipher[i] > 'z'))
     cipher[i] = cipher[i] - 26;
     printf("%c", cipher[i]);
     }
     printf("\n \n \t AFTER DECRYPTION : ");
 for(i=0;i<length;i++)</pre>
     {
      plain[i]=cipher[i]-key;
     if(isupper(cipher[i])&&(plain[i]<'A'))</pre>
     plain[i]=plain[i]+26;
     if(islower(cipher[i])&&(plain[i]<'a'))</pre>
     plain[i]=plain[i]+26;
     printf("%c",plain[i]);
 }
getch();
}
```

🚥 Turbo C++ IDE

Enter the plain text:hello Enter the key value:3

> PLAIN TEXt: hello ENCRYPTED TEXT: khoor AFTER DECRYPTION : hello

# **RESULT:**

Thus the implementation of Caesar cipher had been executed successfully.

VVIT DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

- 🗆 ×

### EX. NO: 1(B)

# IMPLEMENTATION OF PLAYFAIR CIPHER

AIM:

To write a C program to implement the Playfair Substitution technique.

### **DESCRIPTION:**

The Playfair cipher starts with creating a key table. The key table is a  $5\times5$  grid of letters that will act as the key for encrypting your plaintext. Each of the 25 letters must be unique and one letter of the alphabet is omitted from the table (as there are 25 spots and 26 letters in the alphabet).

To encrypt a message, one would break the message into digrams (groups of 2 letters) such that, for example, "HelloWorld" becomes "HE LL OW OR LD", and map them out on the key table. The two letters of the diagram are considered as the opposite corners of a rectangle in the key table. Note the relative position of the corners of this rectangle. Then apply the following 4 rules, in order, to each pair of letters in the plaintext:

- 1. If both letters are the same (or only one letter is left), add an "X" after the first letter
- 2. If the letters appear on the same row of your table, replace them with the letters to their immediate right respectively
- 3. If the letters appear on the same column of your table, replace them with the letters immediately below respectively
- 4. If the letters are not on the same row or column, replace them with the letters on the same row respectively but at the other pair of corners of the rectangle defined by the original pair.

### EXAMPLE:

D. Playfair Cipher

Example1: Plaintext: CRYPTO IS TOO EASY Key = INFOSEC Ciphertext: ?? Grouped text: CR YP TO IS TO XO EA SY Ciphertext: AQ TV YB NI YB YF CB OZ

1/J	N	F	0	S
E	С	Α	В	D
G	Н	К	L	M
Ρ	Q	R	Т	U
۷	W	X	Y	Z

# **ALGORITHM:**

**STEP-1:** Read the plain text from the user.

**STEP-2:** Read the keyword from the user.

- **STEP-3:** Arrange the keyword without duplicates in a 5\*5 matrix in the row order and fill the remaining cells with missed out letters in alphabetical order. Note that 'i' and 'j' takes the same cell.
- **STEP-4:** Group the plain text in pairs and match the corresponding corner letters by forming a rectangular grid.

**STEP-5:** Display the obtained cipher text.

# **PROGRAM:** (Playfair Cipher)

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
#include<ctype.h>
#define MX 5
void playfair(char ch1, char ch2, char key[MX][MX])
{
     int i,j,w,x,y,z;
     FILE *out;
if((out=fopen("cipher.txt", "a+"))==NULL)
{
     printf("File Currupted.");
}
for(i=0;i<MX;i++)</pre>
{
     for(j=0; j<MX; j++)</pre>
     if(ch1==key[i][j])
     {
     w=i;
     x=j;
     }
     else if(ch2==key[i][j])
     {
     y=i;
     z=j;
}}
//printf("%d%d %d%d",w,x,y,z);
if(w==y)
{
     x=(x+1) %5; z=(z+1) %5;
     printf("%c%c", key[w][x], key[y][z]);
     fprintf(out, "%c%c",key[w][x],key[y][z]);
}
else if(x==z)
{
```

```
w = (w+1) %5; y = (y+1) %5;
     printf("%c%c", key[w][x], key[y][z]);
     fprintf(out, "%c%c", key[w][x], key[y][z]);
}
else
{
     printf("%c%c", key[w][z], key[y][x]);
     fprintf(out, "%c%c", key[w][z], key[y][x]);
}
fclose(out);
}
void main()
{
     int i, j, k=0, 1, m=0, n;
     char key[MX][MX],keyminus[25],keystr[10],str[25]={0};
     char
     alpa[26]={'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L'
     , 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z'}
     clrscr();
     printf("\nEnter key:");
     gets(keystr);
     printf("\nEnter the plain text:");
     gets(str);
     n=strlen(keystr);
     //convert the characters to uppertext
     for (i=0; i<n; i++)</pre>
     {
           if(keystr[i]=='j')keystr[i]='i';
           else if(keystr[i]=='J')keystr[i]='I';
           keystr[i] = toupper(keystr[i]);
     }
     //convert all the characters of plaintext to uppertext
     for (i=0; i<strlen(str); i++)</pre>
     {
           if(str[i]=='j')str[i]='i';
           else if(str[i]=='J')str[i]='I';
           str[i] = toupper(str[i]);
     }
     j=0;
     for(i=0;i<26;i++)</pre>
     {
           for (k=0; k<n; k++)</pre>
           {
           if(keystr[k]==alpa[i])
           break;
           else if(alpa[i]=='J')
           break;
           }
           if(k==n)
           ł
           keyminus[j]=alpa[i];j++;
           }
     }
```

```
//construct key keymatrix
     k=0;
     for(i=0;i<MX;i++)</pre>
     {
           for(j=0;j<MX;j++)</pre>
           {
           if(k<n)
           {
           key[i][j]=keystr[k];
           k++; }
           else
           {
           key[i][j]=keyminus[m];m++;
           ł
          printf("%c ",key[i][j]);
           }
          printf("\n");
     }
     printf("\n\nEntered text :%s\nCipher Text :", str);
     for(i=0;i<strlen(str);i++)</pre>
     {
     if(str[i]=='J')str[i]='I';
     if(str[i+1]=='\0')
     playfair(str[i], 'X', key);
     else
     {
           if(str[i+1]=='J')str[i+1]='I';
           if(str[i]==str[i+1])
           playfair(str[i], 'X', key);
           else
           {
          playfair(str[i], str[i+1], key); i++;
           }}
     }
getch();
}
```

🚥 Turbo C++ IDE

Enter key:hello

Enter the plain text:cse H E L L O A B C D F G I K M N P Q R S T U U W X Y

Entered text :CSE Cipher Text :DRLV\_

# **RESULT:**

Thus the Playfair cipher substitution technique had been implemented successfully.

- 🗆 ×

### EX. NO: 1(C)

# **IMPLEMENTATION OF HILL CIPHER**

### AIM:

To write a C program to implement the hill cipher substitution techniques.

### **DESCRIPTION:**

Each letter is represented by a number modulo 26. Often the simple scheme A = 0, B = 1... Z = 25, is used, but this is not an essential feature of the cipher. To encrypt a message, each block of *n* letters is multiplied by an invertible  $n \times n$  matrix, against modulus 26. To decrypt the message, each block is multiplied by the inverse of the matrix used for encryption. The matrix used for encryption is the cipher key, and it should be chosen randomly from the set of invertible  $n \times n$  matrices (modulo 26).

#### EXAMPLE:

2	4	5	$\begin{bmatrix} 0 \end{bmatrix}$		[ 171 ]		[ 15 ]	
9	2	-	19	=	57	$\pmod{26} =$	5	= 'PFO'
3	17	7	19		456		14	

### **ALGORITHM:**

**STEP-1:** Read the plain text and key from the user.

**STEP-2:** Split the plain text into groups of length three.

**STEP-3:** Arrange the keyword in a 3\*3 matrix.

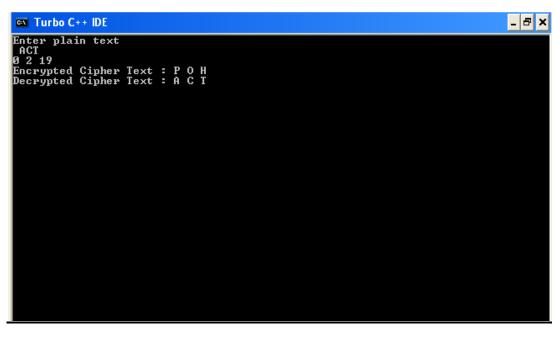
STEP-4: Multiply the two matrices to obtain the cipher text of length three.

**STEP-5:** Combine all these groups to get the complete cipher text.

### **<u>PROGRAM</u>**: (Hill Cipher)

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
int main() {
    unsigned int a[3][3]={{6,24,1}, {13,16,10}, {20,17,15}};
    unsigned int b[3][3]={{6,24,1}, {13,16,10}, {20,17,15}};
    unsigned int b[3][3]={{8,5,10}, {21,8,21}, {21,12,8}};
    int i, j, t=0;
    unsigned int c[20],d[20];
    char msg[20];
    char msg[20];
    clrscr();
    printf("Enter plain text\n ");
    scanf("%s",msg);
    for(i=0;i<strlen(msg);i++)
    {        c[i]=msg[i]-65;
    }
}</pre>
```

```
printf("%d ",c[i]);
      }
      for(i=0;i<3;i++)</pre>
      {
          t=0;
      for(j=0;j<3;j++)</pre>
      {
           t=t+(a[i][j]*c[j]);
      }
           d[i]=t%26;
printf("\nEncrypted Cipher Text :");
for(i=0;i<3;i++)</pre>
printf(" %c",d[i]+65);
for(i=0;i<3;i++)</pre>
{
     t=0;
     for(j=0;j<3;j++)</pre>
      {
           t=t+(b[i][j]*d[j]);
      }
     c[i]=t%26;
}
printf("\nDecrypted Cipher Text :");
for(i=0;i<3;i++)</pre>
printf(" %c",c[i]+65);
getch();
return 0;
}
```



# **RESULT:**

Thus the hill cipher substitution technique had been implemented successfully in C.

# EX. NO: 1(D) <u>IMPLEMENTATION OF VIGENERE CIPHER</u>

# AIM:

To implement the Vigenere Cipher substitution technique using C program.

# **DESCRIPTION:**

To encrypt, a table of alphabets can be used, termed a tabula recta, Vigenère square, or Vigenère table. It consists of the alphabet written out 26 times in different rows, each alphabet shifted cyclically to the left compared to the previous alphabet, corresponding to the 26 possible Caesar ciphers. At different points in the encryption process, the cipher uses a different alphabet from one of the rows. The alphabet used at each point depends on a repeating keyword.

Each row starts with a key letter. The remainder of the row holds the letters A to Z. Although there are 26 key rows shown, you will only use as many keys as there are unique letters in the key string, here just 5 keys, {L, E, M, O, N}. For successive letters of the message, we are going to take successive letters of the key string, and encipher each message letter using its corresponding key row. Choose the next letter of the key, go along that row to find the column heading that matches the message character; the letter at the intersection of [key-row, msg-col] is the enciphered letter.

### **EXAMPLE:**

CORPERATION AND NO NO NUTRI LIGATION FFGH: LKLMNOFGRSTU>WXYZABCDE **GOULSKINNOPORSTUVNXYZABOOES** HUIJKIMNOPORSTUVWXYZAECDEF II: KIMNOP QHSTUVWXYZARCDEFG JJMLMNOPORSTUVWXYZABODEFGHI KKLMNOPSRSTUVWXYZABODEF6HI: LIMNOP QUE TUV WXYZA ECDE FOHI I K MMNORGES UV WXYZAEODEF GHIIFK L Z X O P Q F S T U V V X X X A P O C E F O F I L X I OOF GRETUV WXYZABCDEFGHIJKIM O Q H S T U V W X Y Z A B C D E F C H I J K L M RISTUVWXYZABUDEFCHIJKLMZCPO S S T T V W X Y X A H O D E S O H I I X V W Y T P O P O P **ProrstuvWXYZABODEFGHI**5KLMNO TTUVWXYZA BCDEFCHIJKLMNOPQRS UUVWXYZABCIESGHIJMEMNOSQRST XXYZAGOOMIXIIIGFEOCHYW YYZABODEFCHIJKIMNOPGRSTUVW NNABCREFG -VWXYZABC WXYZABCDEFGHIJKLMNOPQHSTUV D C D E F C H DEFORMAXINGOROUSXXXXXX ABCDEFG ABCDEFUE JKLMNOPQRSTUVWXYZA IJMLMNOPQRSTUVWXYZ DEFCH IJKLMNOPQRSTUVWXYZ HIJKL M N O PQESTUVWXYZAB MNOPQESTUV I HK LMX O A CAR O D N O P w X Y Ġ N H M

### **ALGORITHM:**

**STEP-1:** Arrange the alphabets in row and column of a 26\*26 matrix.

- **STEP-2:** Circulate the alphabets in each row to position left such that the first letter is attached to last.
- STEP-3: Repeat this process for all 26 rows and construct the final key matrix.
- **STEP-4:** The keyword and the plain text is read from the user.
- **STEP-5:** The characters in the keyword are repeated sequentially so as to match with that of the plain text.
- **STEP-6:** Pick the first letter of the plain text and that of the keyword as the row indices and column indices respectively.
- STEP-7: The junction character where these two meet forms the cipher character.

STEP-8: Repeat the above steps to generate the entire cipher text.

### **<u>PROGRAM</u>**: (Vigenere Cipher)

```
#include <stdio.h>
#include<conio.h>
#include <ctype.h>
#include <string.h>
void encipher();
void decipher();
void main()
{
     int choice;
     clrscr();
     while(1)
     {
          printf("\n1. Encrypt Text");
          printf("\t2. Decrypt Text");
          printf("\t3. Exit");
          printf("\n\nEnter Your Choice : ");
          scanf("%d",&choice);
          if(choice == 3)
          exit(0);
          else if(choice == 1)
          encipher();
          else if(choice == 2)
          decipher();
          else
          printf("Please Enter Valid Option.");
     }
}
void encipher()
{
     unsigned int i, j;
     char input[50], key[10];
     printf("\n\nEnter Plain Text: ");
```

```
scanf("%s",input);
     printf("\nEnter Key Value: ");
     scanf("%s",key);
     printf("\nResultant Cipher Text: ");
     for(i=0, j=0; i<strlen(input); i++, j++)</pre>
     {
          if(j>=strlen(key))
                {
                      j=0;
                }
     printf("%c",65+(((toupper(input[i])-65)+(toupper(key[j])-
     65))%26));
     }}
void decipher()
{
     unsigned int i, j;
     char input[50], key[10];
     int value;
     printf("\n\nEnter Cipher Text: ");
     scanf("%s",input);
     printf("\n\nEnter the key value: ");
     scanf("%s",key);
for(i=0, j=0; i<strlen(input); i++, j++)</pre>
{
     if(j>=strlen(key))
            j=0;
     {
     value = (toupper(input[i])-64)-(toupper(key[j])-64);
     if (value < 0)
     { value = value * -1;
     }
     printf("%c",65 + (value % 26));
}}
```

#### 🚥 Turbo C++ IDE

1. Encrypt Text 2. Decrypt Text 3. Exit Enter Your Choice : 1

Enter Plain Text: hai

Enter Key Value: hello

Resultant Cipher Text: OET 1. Encrypt Text 2. Decrypt Text 3. Exit Enter Your Choice : 2

Enter Cipher Text: OET

Enter the key value: hello HAI 1. Encrypt Text 2. Decrypt Text 3. Exit Enter Your Choice : 3

# **RESULT:**

Thus the Vigenere Cipher substitution technique had been implemented successfully.

- 🗆 🗙

EX. NO: 1(E)

# <u>IMPLEMENTATION OF RAIL FENCE – ROW & COLUMN</u> <u>TRANSFORMATION TECHNIQUE</u>

# AIM:

To write a C program to implement the rail fence transposition technique.

# **DESCRIPTION:**

In the rail fence cipher, the plain text is written downwards and diagonally on successive "rails" of an imaginary fence, then moving up when we reach the bottom rail. When we reach the top rail, the message is written downwards again until the whole plaintext is written out. The message is then read off in rows.

# EXAMPLE:

			Н 2		
I R E	S E Y	CDO	R O S U A	V A R	E V S

yields the cipher

WIREEROSUAEVARBDEVSCACDOFESEYL.

# **ALGORITHM:**

**STEP-1:** Read the Plain text.

**STEP-2:** Arrange the plain text in row columnar matrix format.

- STEP-3: Now read the keyword depending on the number of columns of the plain text.
- **STEP-4:** Arrange the characters of the keyword in sorted order and the corresponding columns of the plain text.
- **STEP-5:** Read the characters row wise or column wise in the former order to get the cipher text.

```
PROGRAM: (Rail Fence)
```

```
#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<1;i++)</pre>
{
     if(i%2==0)
     c[j++]=a[i];
}
for(i=0;i<1;i++)</pre>
{
     if(i%2==1)
     c[j++]=a[i];
}
c[j]='\0';
printf("\nCipher text after applying rail fence :");
printf("\n%s",c);
/*Deciphering*/
if(1%2==0)
     k=1/2;
else
     k=(1/2)+1;
for(i=0, j=0; i<k; i++)</pre>
{
    d[j]=c[i];
    j=j+2;
}
for(i=k,j=1;i<1;i++)</pre>
{
    d[j]=c[i];
    j=j+2;
}
d[1] = ' \ 0';
printf("\nText after decryption : ");
printf("%s",d);
getch();
}
```

🔤 Turbo C++ IDE

#### 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 rail fence algorithm had been executed successfully.

- 🗆 🗙

### EX. NO: 2(A)

# **IMPLEMENTATION OF DES**

### AIM:

To write a C program to implement Data Encryption Standard (DES) using C Language.

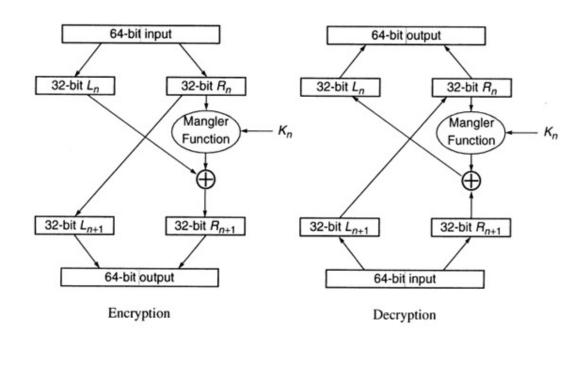
### **DESCRIPTION:**

DES is a symmetric encryption system that uses 64-bit blocks, 8 bits of which are used for parity checks. The key therefore has a "useful" length of 56 bits, which means that only 56 bits are actually used in the algorithm. The algorithm involves carrying out combinations, substitutions and permutations between the text to be encrypted and the key, while making sure the operations can be performed in both directions. The key is ciphered on 64 bits and made of 16 blocks of 4 bits, generally denoted  $k_1$  to  $k_{16}$ . Given that "only" 56 bits are actually used for encrypting, there can be  $2^{56}$  different keys.

### The main parts of the algorithm are as follows:

- Fractioning of the text into 64-bit blocks
- Initial permutation of blocks
- > Breakdown of the blocks into two parts: left and right, named L and R
- Permutation and substitution steps repeated 16 times
- > Re-joining of the left and right parts then inverse initial permutation

# **EXAMPLE:**



### **ALGORITHM:**

**STEP-1:** Read the 64-bit plain text.

STEP-2: Split it into two 32-bit blocks and store it in two different arrays.

STEP-3: Perform XOR operation between these two arrays.

- **STEP-4:** The output obtained is stored as the second 32-bit sequence and the original second 32-bit sequence forms the first part.
- **STEP-5:** Thus the encrypted 64-bit cipher text is obtained in this way. Repeat the same process for the remaining plain text characters.

### **PROGRAM:**

```
DES.java
```

```
import javax.swing.*;
import java.security.SecureRandom;
import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.SecretKey;
import javax.crypto.spec.SecretKeySpec;
import java.util.Random ;
class DES {
     byte[] skey = new byte[1000];
     String skeyString;
     static byte[] raw;
     String inputMessage, encryptedData, decryptedMessage;
public DES()
{
try
{
     generateSymmetricKey();
     inputMessage=JOptionPane.showInputDialog(null, "Enter
     message to encrypt");
     byte[] ibyte = inputMessage.getBytes();
     byte[] ebyte=encrypt(raw, ibyte);
     String encryptedData = new String(ebyte);
     System.out.println("Encrypted message "+encryptedData);
     JOptionPane.showMessageDialog(null, "Encrypted Data
     "+"\n"+encryptedData);
     byte[] dbyte= decrypt(raw,ebyte);
     String decryptedMessage = new String(dbyte);
     System.out.println("Decrypted message
     "+decryptedMessage);
     JOptionPane.showMessageDialog(null, "Decrypted Data
     "+"\n"+decryptedMessage);
}
catch (Exception e)
{
     System.out.println(e);
}
}
```

```
void generateSymmetricKey() {
try {
     Random r = new Random();
     int num = r.nextInt(10000);
     String knum = String.valueOf(num);
     byte[] knumb = knum.getBytes();
     skey=getRawKey(knumb);
     skeyString = new String(skey);
     System.out.println("DES Symmetric key = "+skeyString);
}
catch (Exception e)
{
     System.out.println(e);
}
}
private static byte[] getRawKey(byte[] seed) throws Exception
ł
     KeyGenerator kgen = KeyGenerator.getInstance("DES");
     SecureRandom sr = SecureRandom.getInstance("SHA1PRNG");
     sr.setSeed(seed);
     kgen.init(56, sr);
     SecretKey skey = kgen.generateKey();
     raw = skey.getEncoded();
     return raw;
}
private static byte[] encrypt(byte[] raw, byte[] clear) throws
Exception {
          SecretKeySpec skeySpec = new SecretKeySpec(raw,
          "DES");
          Cipher cipher = Cipher.getInstance("DES");
          cipher.init(Cipher.ENCRYPT_MODE, skeySpec);
          byte[] encrypted = cipher.doFinal(clear);
          return encrypted;
}
private static byte[] decrypt(byte[] raw, byte[] encrypted)
throws Exception
{
          SecretKeySpec skeySpec = new SecretKeySpec(raw,
          "DES");
          Cipher cipher = Cipher.getInstance("DES");
          cipher.init(Cipher.DECRYPT_MODE, skeySpec);
          byte[] decrypted = cipher.doFinal(encrypted);
          return decrypted;
}
public static void main(String args[]) {
     DES des = new DES();
}
}
```

C:\WINDOWS\system32\cmd.exe - java	DES	- 🗆
ncrypted message Ω <sup>μ</sup> ûæR <sub>I</sub> r‡¦∓∞♦¼Æ¦ ecrypted message Computer scienc	¦∲?î•\$→xo e	-
:\>		
:\>javac DES.java :\>java DES		
ES Symmetric key = #4>8		
	Input 🔀	
	<b>?</b> Enter message to encrypt	
	computer	
	OK Cancel	
▼ C:\WINDOWS\system32\cmd.exe - java	DES	- 🗆
		<u>-</u> []
ncrypted message Ջ <sup>լլ</sup> մա⊱րբ‡¦∓∞♦¼Æ  ecrypted message Computer scienc :\>		<u>-</u> []
crypted message Ω <sup>μ</sup> ûæξηr‡¦∓∞♦¼Æ¦ crypted message Computer scienc \\> \\>javac DES.java		<mark>-</mark> 🗆
icrypted message Ω <sup>μ</sup> ûæξ <sub>I</sub> rr‡¦∓∞♦¼Æ¦ crypted message Computer scienc \\} \\}javac DES.java S Symmetric key = #4>8	¦ <b>∲</b> ?î•S→xo e	<mark>_ []</mark>
crypted message Ω <sup>μ</sup> ûæξ <sub>I</sub> rr‡¦∓∞♦¼Æ¦ crypted message Computer scienc ∖> \>javac DES.java \>java DES S Symmetric key = #4>8	¦ <b>∲</b> ?î•S→xo e	
icrypted message Ω <sup>μ</sup> ûæξ <sub>I</sub> rr‡¦∓∞♦¼Æ¦ crypted message Computer scienc \\} \\}javac DES.java S Symmetric key = #4>8	¦ <b>∲</b> ?î•S→xo e	
icrypted message Ω <sup>u</sup> ûæt <sub>f</sub> m +¦∓∞♦¼Æ  crypted message Computer scienc :\> :\>javac DES.java :\>java DES S Symmetric key = #4>8	¦ <b>∲</b> ?î•S→xo e	
icrypted message Ω <sup>u</sup> ûæt <sub>f</sub> m +¦∓∞♦¼Æ  crypted message Computer scienc :\> :\>javac DES.java :\>java DES S Symmetric key = #4>8	ł <b>*</b> ?î •S→xo e > Message Encrypted Data	
ncrypted message Ω <sup>μ</sup> ûæξ <sub>I</sub> rr‡¦∓∞♦¼Æ¦ ecrypted message Computer scienc :\> :\>javac DES.java :\>java DES ES Symmetric key = #4>8	¦ ∲?î •S→xo e > Message	
▼ C:\WINDOWS\system32\cmd.exe - java ncrypted message Ω <sup>u</sup> üæt <sub>f</sub> r+;∓∞◆%ff ecrypted message Computer scienc :\> :\>javac DES.java :\>java DES ES Symmetric key = #4>8 ncrypted message %#áJ«[σ% 46∎4<]\$	ł <b>*</b> ?î •S→xo e > Message Encrypted Data	

ncrypted message Ω <sup>u</sup> ûæR <sub>ff</sub> r‡¦∓∞♦½Æ¦¦♠?ź ecrypted message Computer science	ì•S→xo			-
:\> :\>javac DES.java				
:\>java DES ES Symmetric key = #4>8 ncrypted message ¼#áJ«[σ% 46 <b>8</b> 4( \$> ecrypted message computer				
	Messag	e		
	Messag i	e Decrypted Data computer	×	

# **RESULT:**

Thus the data encryption standard algorithm had been implemented successfully using C language.

# EX. NO: 2(B)

# **IMPLEMENTATION OF RSA**

# AIM:

To write a C program to implement the RSA encryption algorithm.

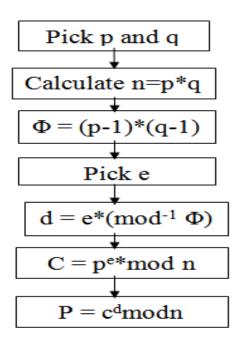
# **DESCRIPTION:**

RSA is an algorithm used by modern computers to encrypt and decrypt messages. It is an asymmetric cryptographic algorithm. Asymmetric means that there are two different keys. This is also called public key cryptography, because one of them can be given to everyone. A basic principle behind RSA is the observation that it is practical to find three very large positive integers e, d and n such that with modular exponentiation for all integer m:

# $(\mathbf{m}^{\mathbf{e}})^{\mathbf{d}} = \mathbf{m} \; (\mathbf{mod} \; \mathbf{n})$

The public key is represented by the integers n and e; and, the private key, by the integer d. m represents the message. RSA involves a public key and a private key. The public key can be known by everyone and is used for encrypting messages. The intention is that messages encrypted with the public key can only be decrypted in a reasonable amount of time using the private key.

# EXAMPLE:



# **ALGORITHM:**

STEP-1: Select two co-prime numbers as p and q.

STEP-2: Compute n as the product of p and q.

**STEP-3:** Compute (p-1)\*(q-1) and store it in z.

STEP-4: Select a random prime number e that is less than that of z.

**STEP-5:** Compute the private key, d as  $e * mod^{-1}(z)$ .

**STEP-6:** The cipher text is computed as  $message^{e} * mod n$ .

**STEP-7:** Decryption is done as cipher<sup>d</sup> mod n.

# PROGRAM: (RSA)

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#include<math.h>
#include<string.h>
     long int
     p,q,n,t,flag,e[100],d[100],temp[100],j,m[100],en[100],i;
     char msg[100];
     int prime(long int);
     void ce();
     long int cd(long int);
     void encrypt();
     void decrypt();
void main()
{
     clrscr();
     printf("\nENTER FIRST PRIME NUMBER\n");
     scanf("%d",&p);
     flag=prime(p);
     if(flag==0)
      {
          printf("\nWRONG INPUT\n");
          getch();
      }
      printf("\nENTER ANOTHER PRIME NUMBER\n");
      scanf("%d",&q);
      flag=prime(q);
      if(flag==0||p==q)
      {
       printf("\nWRONG INPUT\n");
       getch();
      }
      printf("\nENTER MESSAGE\n");
      fflush(stdin);
      scanf("%s",msq);
      for(i=0;msg[i]!=NULL;i++)
      m[i]=msg[i];
      n=p*q;
```

```
t=(p-1)*(q-1);
       ce();
      printf("\nPOSSIBLE VALUES OF e AND d ARE\n");
       for(i=0;i<j-1;i++)</pre>
      printf("\n%ld\t%ld",e[i],d[i]);
       encrypt();
       decrypt();
      getch();
}
int prime(long int pr)
{
int i;
j=sqrt(pr);
for(i=2;i<=j;i++)</pre>
{
if(pr%i==0)
return 0;
}
return 1;
}
void ce()
{
int k;
k=0;
for(i=2;i<t;i++)</pre>
{
 if(t%i==0)
 continue;
 flag=prime(i);
 if(flag==1&&i!=p&&i!=q)
 {
 e[k]=i;
 flag=cd(e[k]);
 if(flag>0)
 {
 d[k]=flag;
 k++;
 }
 if(k==99)
 break;
 long int cd(long int x)
 ł
 long int k=1;
 while(1)
 {
 k=k+t;
 if(k%x==0)
 return (k/x);
 } }
 void encrypt() {
 long int pt,ct,key=e[0],k,len;
 i=0;
 len=strlen(msg);
```

```
while(i!=len) {
pt=m[i];
pt=pt-96;
k=1;
for(j=0; j<key; j++)</pre>
{ k=k*pt;
k=k%n;
}
temp[i]=k;
ct=k+96;
en[i]=ct;
i++;
}
en[i]=-1;
printf("\nTHE ENCRYPTED MESSAGE IS\n");
for(i=0;en[i]!=-1;i++)
printf("%c",en[i]);
}
void decrypt()
{
long int pt,ct,key=d[0],k;
i=0;
while(en[i]!=-1)
{
ct=temp[i];
k=1;
for(j=0;j<key;j++)</pre>
{
k=k*ct;
k=k%n;
}
pt=k+96;
m[i]=pt;
i++;
}
m[i] = -1;
printf("\nTHE DECRYPTED MESSAGE IS\n");
for(i=0;m[i]!=-1;i++)
printf("%c",m[i]);
}
```

🚥 Turbo C++ IDE

ENTER FIRST PRIME NUMBER 7 ENTER ANOTHER PRIME NUMBER 13 ENTER MESSAGE hello POSSIBLE VALUES OF e AND d ARE 5 29 11 59 17 17 19 19 23 47 29 5 31 7 THE ENCRYPTED MESSAGE IS haåå<sup>Q</sup> THE DECRYPTED MESSAGE IS haå

# **RESULT:**

Thus the C program to implement RSA encryption technique had been implemented successfully

- 🗆 🗙

# EX. NO: 2(C)

# IMPLEMENTATION OF DIFFIE HELLMAN KEY EXCHANGE ALGORITHM

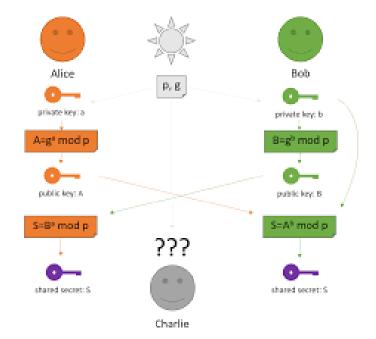
# AIM:

To implement the Diffie-Hellman Key Exchange algorithm using C language.

# **DESCRIPTION:**

Diffie–Hellman Key Exchange establishes a shared secret between two parties that can be used for secret communication for exchanging data over a public network. It is primarily used as a method of exchanging cryptography keys for use in symmetric encryption algorithms like AES. The algorithm in itself is very simple. The process begins by having the two parties, Alice and Bob. Let's assume that Alice wants to establish a shared secret with Bob.

# EXAMPLE:



# **ALGORITHM:**

**STEP-1:** Both Alice and Bob shares the same public keys g and p.

STEP-2: Alice selects a random public key a.

**STEP-3:** Alice computes his secret key A as g<sup>a</sup> mod p.

**STEP-4:** Then Alice sends A to Bob.

- **STEP-5:** Similarly Bob also selects a public key b and computes his secret key as B and sends the same back to Alice.
- **STEP-6:** Now both of them compute their common secret key as the other one's secret key power of a mod p.

### **PROGRAM:** (Diffie Hellman Key Exchange)

```
#include<stdio.h>
#include<conio.h>
long long int power(int a, int b, int mod)
{
     long long int t;
     if(b==1)
     return a;
     t=power(a, b/2, mod);
     if(b%2==0)
     return (t*t)%mod;
     else
     return (((t*t)%mod)*a)%mod;
}
long int calculateKey(int a, int x, int n)
{
     return power(a, x, n);
}
void main()
{
     int n,g,x,a,y,b;
     clrscr();
     printf("Enter the value of n and g : ");
     scanf("%d%d",&n,&g);
     printf("Enter the value of x for the first person : ");
     scanf("%d",&x);
     a=power(g, x, n);
     printf("Enter the value of y for the second person : ");
     scanf("%d", &y);
     b=power(g,y,n);
     printf("key for the first person is :
     lld(n", power(b, x, n));
     printf("key for the second person is :
     %lld\n",power(a,y,n));
     getch();
}
```

CV Turbo C++ IDE

- 🗆 🗙

```
Enter the value of n and g : 7
9
Enter the value of x for the first person : 6
Enter the value of y for the second person : 15
key for the first person is : 1
key for the second person is : 1
```

# **RESULT:**

Thus the Diffie-Hellman key exchange algorithm had been successfully implemented using C.

### EX. NO: 2(D)

# **IMPLEMENTATION OF MD5**

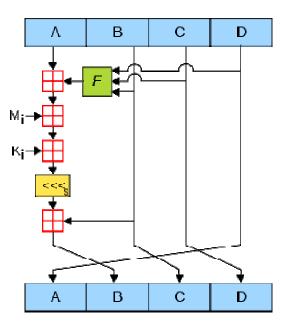
# AIM:

To write a C program to implement the MD5 hashing technique.

### **DESCRIPTION:**

MD5 processes a variable-length message into a fixed-length output of 128 bits. The input message is broken up into chunks of 512-bit blocks. The message is padded so that its length is divisible by 512. The padding works as follows: first a single bit, 1, is appended to the end of the message. This is followed by as many zeros as are required to bring the length of the message up to 64 bits less than a multiple of 512. The remaining bits are filled up with 64 bits representing the length of the original message, modulo 2<sup>64</sup>. The main MD5 algorithm operates on a 128-bit state, divided into four 32-bit words, denoted A, B, C, and D. These are initialized to certain fixed constants. The main algorithm then uses each 512-bit message block in turn to modify the state.

### EXAMPLE:



### **ALGORITHM:**

STEP-1: Read the 128-bit plain text.

STEP-2: Divide into four blocks of 32-bits named as A, B, C and D.

- **STEP-3:** Compute the functions f, g, h and i with operations such as, rotations, permutations, etc,.
- **STEP-4:** The output of these functions are combined together as F and performed circular shifting and then given to key round.
- **STEP-5:** Finally, right shift of 's' times are performed and the results are combined together to produce the final output.

# PROGRAM:( MD5)

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <math.h>
#include<conio.h>
typedef union uwb
{
    unsigned w;
    unsigned char b[4];
} MD5union;
     typedef unsigned DigestArray[4];
     unsigned func0( unsigned abcd[] ) {
     return ( abcd[1] & abcd[2]) | (~abcd[1] & abcd[3]);}
     unsigned func1( unsigned abcd[] ) {
     return ( abcd[3] & abcd[1]) | (~abcd[3] & abcd[2]);}
     unsigned func2( unsigned abcd[] ) {
     return abcd[1] ^ abcd[2] ^ abcd[3];}
     unsigned func3( unsigned abcd[] ){
     return abcd[2] ^ (abcd[1] |~ abcd[3]);}
     typedef unsigned (*DgstFctn) (unsigned a[]);
unsigned *calctable( unsigned *k)
{
     double s, pwr;
     int i;
     pwr = pow(2, 32);
for (i=0; i<64; i++)</pre>
{
     s = fabs(sin(1+i));
     k[i] = (unsigned) ( s * pwr );
}
     return k;
}
unsigned rol ( unsigned r, short N )
{
    unsigned mask1 = (1 < N) -1;
    return ((r>>(32-N)) & mask1) | ((r<<N) & ~mask1);
}
```

```
unsigned *md5( const char *msg, int mlen)
{
    static DigestArray h0 = { 0x67452301, 0xEFCDAB89,
    0x98BADCFE, 0x10325476 };
    static DgstFctn ff[] = { &func0, &func1, &func2, &func3};
    static short M[] = { 1, 5, 3, 7 };
    static short O[] = { 0, 1, 5, 0 };
    static short rot0[] = { 7,12,17,22};
    static short rot1[] = { 5, 9,14,20};
    static short rot2[] = { 4,11,16,23};
    static short rot3[] = { 6,10,15,21};
    static short *rots[] = {rot0, rot1, rot2, rot3 };
    static unsigned kspace[64];
    static unsigned *k;
    static DigestArray h;
    DigestArray abcd;
    DgstFctn fctn;
    short m, o, g;
    unsigned f;
    short *rotn;
    union
    {
         unsigned w[16];
         char
                 b[64];
    } mm ;
         int os = 0;
         int grp, grps, q, p;
         unsigned char *msg2;
         if (k==NULL) k= calctable(kspace);
    for (q=0; q<4; q++) h[q] = h0[q]; // initialize
    {
          grps = 1 + (mlen+8)/64;
          msg2 = malloc( 64*grps);
          memcpy( msg2, msg, mlen);
          msg2[mlen] = (unsigned char)0x80;
          q = mlen + 1;
     while (q < 64*grps) { msg2[q] = 0; q++ ; }</pre>
     {
          MD5union u;
          u.w = 8*mlen;
          q = 8;
          memcpy(msg2+q, &u.w, 4);
     }
     }
     for (grp=0; grp<grps; grp++)</pre>
     {
          memcpy( mm.b, msg2+os, 64);
```

```
for (q=0; q<4; q++) abcd [q] = h[q];
      for (p = 0; p<4; p++)
      {
           fctn = ff[p];
           rotn = rots[p];
           m = M[p]; o = O[p];
      for (q=0; q<16; q++)
      ł
           q = (m*q + o) \% 16;
          f = abcd[1] + rol(abcd[0] + fctn(abcd) + k[q+16*p]
          + mm.w[g], rotn[q%4]);
          abcd[0] = abcd[3];
          abcd[3] = abcd[2];
          abcd[2] = abcd[1];
         abcd[1] = f;
      }}
 for (p=0; p<4; p++)
 h[p] += abcd[p];
 os += 64;
 }
return h;}
void main()
{
     int j,k;
     const char *msg = "The quick brown fox jumps over
     the lazy dog";
     unsigned *d = md5(msg, strlen(msg));
     MD5union u;
     clrscr();
     printf("\t MD5 ENCRYPTION ALGORITHM IN C \n\n");
     printf("Input String to be Encrypted using MD5 :
     \n\t%s",msg);
     printf("\n\nThe MD5 code for input string is: \n");
     printf("\t= 0x");
     for (j=0;j<4; j++) {</pre>
     u.w = d[j];
     for (k=0;k<4;k++) printf("%02x",u.b[k]);</pre>
}
printf("\n");
printf("\n\t MD5 Encyption Successfully
Completed!!!\n\n");
getch();
system("pause");
getch();}
```

#### **OUTPUT:**

🚥 Turbo C++ IDE

#### MD5 ENCRYPTION ALGORITHM IN C

Input String to be Encrypted using MD5 : The quick brown fox jumps over the lazy dog

The MD5 code for input string is : = Øxf87f8c8f97408c8ff5718c8f6de98c8f MD5 Encyption Successfully Completed!!!

**RESULT:** 

Thus the implementation of MD5 hashing algorithm had been implemented successfully using C.

- 🗆 🗙

### EX. NO: 2(E)

## **IMPLEMENTATION OF SHA-I**

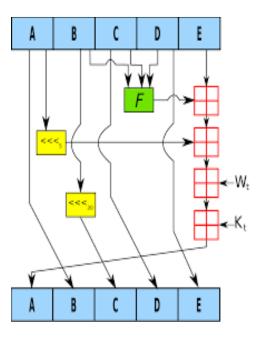
## AIM:

To implement the SHA-I hashing technique using C program.

#### **DESCRIPTION:**

In cryptography, SHA-1 (Secure Hash Algorithm 1) is a cryptographic hash function. SHA-1 produces a 160-bit hash value known as a message digest. The way this algorithm works is that for a message of size < 264 bits it computes a 160-bit condensed output called a message digest. The SHA-1 algorithm is designed so that it is practically infeasible to find two input messages that hash to the same output message. A hash function such as SHA-1 is used to calculate an alphanumeric string that serves as the cryptographic representation of a file or a piece of data. This is called a digest and can serve as a digital signature. It is supposed to be unique and non-reversible.

### EXAMPLE:



## **ALGORITHM:**

**STEP-1:** Read the 256-bit key values.

STEP-2: Divide into five equal-sized blocks named A, B, C, D and E.

STEP-3: The blocks B, C and D are passed to the function F.

**STEP-4:** The resultant value is permuted with block E.

STEP-5: The block A is shifted right by 's' times and permuted with the result of step-4.

- **STEP-6:** Then it is permuted with a weight value and then with some other key pair and taken as the first block.
- **STEP-7:** Block A is taken as the second block and the block B is shifted by 's' times and taken as the third block.

**STEP-8:** The blocks C and D are taken as the block D and E for the final output.

## **PROGRAM:** (Secure Hash Algorithm)

```
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'};
     StringBuffer buf = new StringBuffer();
     for (int j=0; j<b.length; j++) {</pre>
```

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

}

## **OUTPUT:**

C:\WINDOWS\system32\cmd.exe	×
@ <filename> Read options and filenames from file</filename>	
F:\>javac SHA1.java	
F:\>java SHA1 Message digest object info: Algorithm = SHA1 Provider = SUN version 1.7 ToString = SHA1 Message Digest from SUN, <initialized></initialized>	
Tobering onni nessage pryest from bon, (interatized)	
SHA1<''''> = DA39A3EE5E6B4B0D3255BFEF95601890AFD80709	
SHA1("abc") = A9993E364706816ABA3E25717850C26C9CD0D89D	
SHA1<"abcdefghijklmnopqrstuvwxyz"> = 32D10C7B8CF96570CA04CE37F2A19D84240D3A89	
F:\>_	
<u>RESULT:</u>	
Thus the SHA-1 hashing technique had been implemented successfully.	

### EX. NO: 3

# **IMPLEMENTATION OF DIGITAL SIGNATURE STANDARD**

### AIM:

To write a C program to implement the signature scheme named digital signature standard (Euclidean Algorithm).

## **ALGORITHM:**

**STEP-1:** Alice and Bob are investigating a forgery case of x and y.

- **STEP-2:** X had document signed by him but he says he did not sign that document digitally.
- **STEP-3:** Alice reads the two prime numbers p and a.
- **STEP-4:** He chooses a random co-primes alpha and beta and the x's original signature x.
- **STEP-5:** With these values, he applies it to the elliptic curve cryptographic equation to obtain y.
- **STEP-6:** Comparing this 'y' with actual y's document, Alice concludes that y is a forgery.

#### **PROGRAM:** (Digital Signature Standard)

```
import java.util.*;
import java.math.BigInteger;
class dsaAlg {
     final static BigInteger one = new BigInteger("1");
     final static BigInteger zero = new BigInteger("0");
public static BigInteger getNextPrime(String ans)
{
     BigInteger test = new BigInteger(ans);
while (!test.isProbablePrime(99))
e:
{
     test = test.add(one);
}
     return test;
}
public static BigInteger findQ(BigInteger n)
{
     BigInteger start = new BigInteger("2");
while (!n.isProbablePrime(99))
{
     while (!((n.mod(start)).equals(zero)))
     {
          start = start.add(one);
```

```
}
     n = n.divide(start);
}
     return n;
}
public static BigInteger getGen (BigInteger p, BigInteger q,
Random r)
{
     BigInteger h = new BigInteger(p.bitLength(), r);
     h = h.mod(p);
     return h.modPow((p.subtract(one)).divide(q), p);
}
public static void main (String[] args) throws
java.lang.Exception
{
     Random randObj = new Random();
     BigInteger p = getNextPrime("10600"); /* approximate
     prime */
     BigInteger q = findQ(p.subtract(one));
     BigInteger q = qetGen(p,q,randObj);
     System.out.println(" \n simulation of Digital Signature
     Algorithm n'';
     System.out.println(" \n global public key components
     are:\n");
     System.out.println("\np is: " + p);
     System.out.println("\nq is: " + q);
     System.out.println("\ng is: " + g);
     BigInteger x = new BigInteger(q.bitLength(), randObj);
     x = x.mod(q);
     BigInteger y = g.modPow(x,p);
     BigInteger k = new BigInteger(q.bitLength(), randObj);
     k = k.mod(q);
     BigInteger r = (q.modPow(k, p)).mod(q);
     BigInteger hashVal = new BigInteger(p.bitLength(),
     randObj);
     BigInteger kInv = k.modInverse(q);
     BigInteger s = kInv.multiply(hashVal.add(x.multiply(r)));
     s = s.mod(q);
     System.out.println("\nsecret information are:\n");
     System.out.println("x (private) is:" + x);
     System.out.println("k (secret) is: " + k);
     System.out.println("y (public) is: " + y);
     System.out.println("h (rndhash) is: " + hashVal);
     System.out.println("\n generating digital signature:\n");
     System.out.println("r is : " + r);
     System.out.println("s is : " + s);
     BigInteger w = s.modInverse(q);
     BigInteger u1 = (hashVal.multiply(w)).mod(q);
     BigInteger u2 = (r.multiply(w)).mod(q);
     BigInteger v = (g.modPow(u1,p)).multiply(y.modPow(u2,p));
     \mathbf{v} = (\mathbf{v}. \operatorname{mod}(\mathbf{p})) . \operatorname{mod}(\mathbf{q});
     System.out.println("\nverifying digital signature
     (checkpoints) \n:");
     System.out.println("w is : " + w);
```

```
System.out.println("u1 is : " + u1);
System.out.println("u2 is : " + u2);
System.out.println("v is : " + v);
if (v.equals(r))
{
System.out.println("\nsuccess: digital signature is
verified!\n " + r);
}
else
{
System.out.println("\n error: incorrect digital
signature\n ");
}
}
```

#### **OUTPUT:**

C:\WINDOWS\system32\cmd.exe	- 8 ×
E:\>javac dsaAlg.java	<b>_</b>
E:\>java dsaAlg	
simulation of Digital Signature Algorithm	
global public key components are:	
p is: 10601	
q is: 53	
g is: 5559	
secret information are:	
x (private) is:6 k (secret) is: 7 y (public) is: 1992 h (rndhash) is: 10088	
generating digital signature:	
r is : 42 s is : 31	
verifying digital signature (checkpoints)	
w is : 12 u1 is : 4 u2 is : 27 v is : 42	
success: digital signature is verified! 42	
E:\>	

#### **RESULT:**

Thus the simple Code Optimization techniques had been implemented successfully.

#### EX. NO: 04

# SECURE DATA STORAGE, SECURE DATA TRANSMISSION AND FOR CREATING DIGITAL SIGNATURES (GNUPG)

## AIM:

Demonstrate how to provide secure data storage, secure data transmission and for creating digital signatures (GnuPG).

## **INTRODUCTION:**

- > Here's the final guide in my PGP basics series, this time focusing on Windows
- The OS in question will be Windows 7, but it should work for Win8 and Win8.1 as well
- Obviously it's not recommended to be using Windows to access the DNM, but I won't go into the reasons here.
- ➤ The tool well be using is GPG4Win

## **INSTALLING THE SOFTWARE:**

1. Visit <u>www.gpg4win.org</u>. Click on the "Gpg4win 2.3.0" button



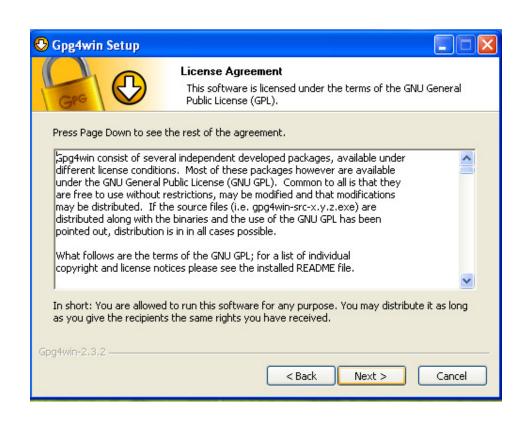
2. On the following screen, click the "Download Gpg4win" button.



3. When the "Welcome" screen is displayed, click the "Next" button



4. When the "License Agreement" page is displayed, click the "Next" button



5. Set the check box values as specified below, then click the "Next" button

	hoose Components		
Gre V	Choose which features of Gpg4win you want to install.		
Check the components you want to install and uncheck the components you don't want to install. Click Next to continue.			
install. Click Next to conditide.			
Select components to install:	<ul> <li>✓ GnuPG</li> <li>✓ Kleopatra</li> <li>✓ GPA</li> <li>✓ GpgOL</li> <li>✓ GpgEX</li> <li>✓ Gpg4win Compendium</li> </ul>	Description Position your mouse over a component to see its description.	
Space required: 100.3MB			
Gpg4win-2,3,2	< Back	Next > Cancel	

6. Set the location where you want the software to be installed. The default location is fine. Then, click the "Next" button.

O Care da da Catura	
Creation Setup	Choose Install Location Choose the folder in which to install Gpg4win.
Setup will install Gpg4win and select another folder	in the following folder. To install in a different folder, click Browse
and select another folder	
Destination Folder	
C:\Program Files\GNU	I\GnuPG Browse
Space required: 100.3MB	
Space available: 20.2GB	
Gpg4win-2,3,2	
	< Back Next > Cancel

7. Specify where you want shortcuts to the software placed, then click the "Next" button.

	Install Options Start links
Please select where Gpg4	win shall install links:
🗹 Start Menu	
Desktop	
Quick Launch Bar	
(Only programs will be link	ed into the quick launch bar.)
Gpg4win-2,3,2	
	< Back Next > Cancel

8. If you selected to have a GPG shortcut in your Start Menu, specify the folder in which it will be placed. The default "Gpg4win" is OK. Click the "Install" button to continue

😌 Gpg4win Setup	
Gre D	Choose Start Menu Folder Choose a Start Menu folder for the Gpg4win shortcuts.
Select the Start Menu fold can also enter a name to o Gpg4win	ler in which you would like to create the program's shortcuts. You create a new folder.
7-Zip Accessories Administrative Tools Android Studio ANSYS 10.0 ANSYS FLEXIm License Ma ArgoUML Autodesk Baidu Browser Casper XP CCleaner Cisco Packet Tracer	anager
Gpg4win-2,3,2	< Back Install Cancel

9. A warning will be displayed if you have Outlook or Explorer opened. If this occurs, click the "OK" button.

pinentry		×
	nter passphrase	
Pa	assphrase	
	Quality:	100%
	<u> </u>	Cancel

10. The installation process will tell you when it is complete. Click the "Next"

button

	Installation Complete Setup was completed successfully.
Completed	
Delete file: C:\Docume Delete file: C:\Docume Delete file: C:\Docume Create file: C:\Docume Create folder: C:\Docu Create folder: C:\Doc Create shortcut: C:\D Create shortcut: C:\D Create shortcut: C:\D	ents and Settings\All Users\Start Menu\Programs\Gpg4win\Doc uments and Settings\All Users\Start Menu\Programs\Gpg4win\Doc uments and Settings\All Users\Start Menu\Programs\Gpg4win\Doc ocuments and Settings\All Users\Start Menu\Programs\Gpg4win\Doc ents and Settings\All Users\Start Menu\Programs\Gpg4win\Doc ents and Settings\All Users\Start Menu\Programs\Gpg4win\Doc ents and Settings\All Users\Start Menu\Programs\Gpg4win\D ents and Settings\All Users\Start Menu\Programs\Gpg4wi ents and Settings\All Menu\Programs\Gpg4wi ents and Settings\All Menu\Programs\Gpg4wi ents and Settings\

11. Once the Gpg4win setup wizard is complete, the following screen will be displayed. Click the "Finish" button

Gpg4win Setup	Completing the Gpg4win Setup
Green	Wizard
$\bigcirc$	Your computer must be restarted in order to complete the installation of Gpg4win. Do you want to reboot now?
	• Reboot now
GPG	○ I want to manually reboot later
	< Back Finish Cancel

12. If you do not uncheck the "Show the README file" check box, the README file will be displayed. The window can be closed after you've reviewed it.

README.en - Notepad	0.0
File Edit Format View Help	
English README file for Gpg4win	
This is Gpg4win, version 2.3.0 (2015-11-24).	
content:	
1. Important Notes 2. Change Jugs (and Workarounds) 4. Interlatation 5. Version History 6. Version Numbers of Included Software 7. Legal Notices	
1. Important Notes	
The cpgSwin compendium describes the installation and use of CpgSwin. After installation it is available in the cpgswin start menu or online: http://www.spgwin.org/doc/en/pggVwin-compendium.html	
Please read the section "3. Known Bugs (and workarounds)" of this README before you start working with Gpg4win.	
Gpg4win supports these platforms:	
* Operating System: windows XP and later versions (for all: 32/64 bit)	
* MS Outlook: 2003, 2007, 2010, 2013, 2016 (for all: only 32bit!)	
2. changes	
Included Gpg4win components in version 2.3.0 are:	
GmuPG: 2.0.29 Kloopatra: 2.2.0-gitfb4ae3d Gmai: 0.9.9 GopOck: 1.3.3 Kompendium EN: 3.0.0 Kompendium EN: 3.0.0	
New in Gpg4win version 2.3.0 (2015-11-24)	
- GpgOL now supports reading PGP/MIME and S/MIME mails, the standard	

## **CREATING YOUR PUBLIC AND PRIVATE KEYS**

GPG encryption and decryption is based upon the keys of the person who will be receiving the encrypted file or message. Any individual who wants to send the person an encrypted file or message must possess the recipient's public key certificate to encrypt the message. The recipient must have the associated private key, which is different than the public key, to be able to decrypt the file. The public and private key pair for an individual is usually generated by the individual on his or her computer using the installed GPG program, called "Kleopatra" and the following procedure:

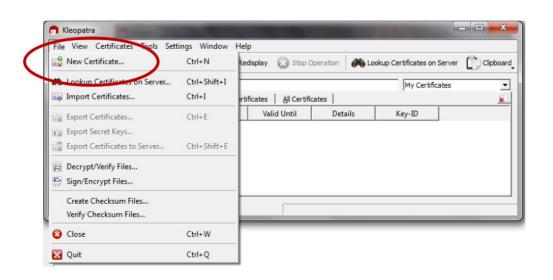
1. From your start bar, select the "Kleopatra" icon to start the Kleopatra certificate management software



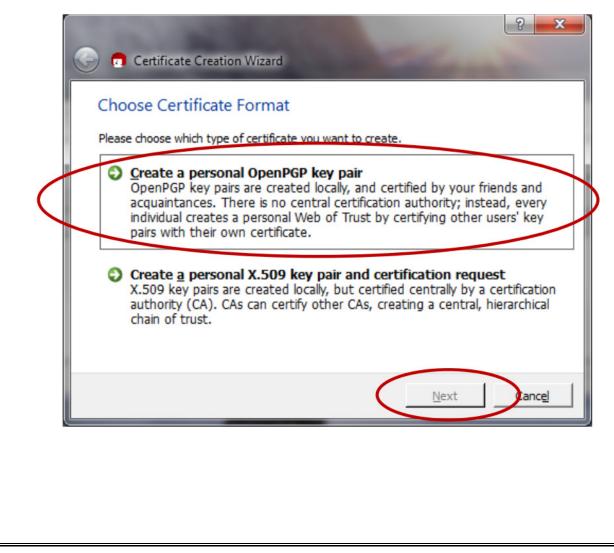
2. The following screen will be displayed

ile <u>V</u> iew <u>C</u> ertifica	tes <u>T</u> ools <u>S</u> ettin	ngs <u>W</u> indow <u>H</u> el	р	20		100 ADM	
Import Certificate	s 🙀 Export Ce	rtificates 🛛 😨 Re	display  Stop (	Operation	Lookup Certificates on	Server [	Clipboa
Search <alt+q></alt+q>					My Certific	ates	
My Certificates	Trusted Certific	cates Other Certi	ficates				×
Name 🗸	E-Mail	Valid From	Valid Until	Details	Key-ID		
				1			

3. From the "File" dropdown, click on the "New Certificate" option



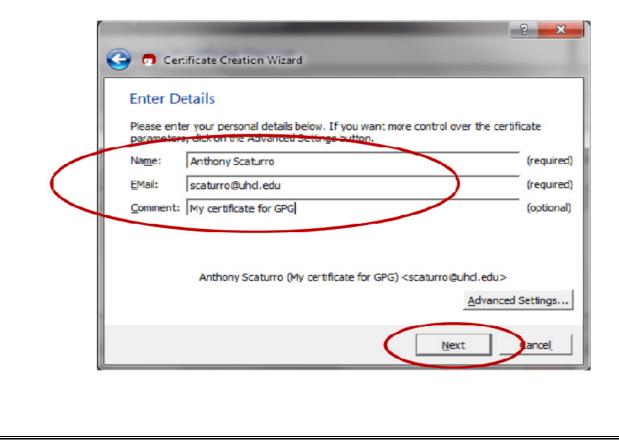
4. The following screen will be displayed. Click on "Create a personal OpenGPG key pair" and the "Next" button



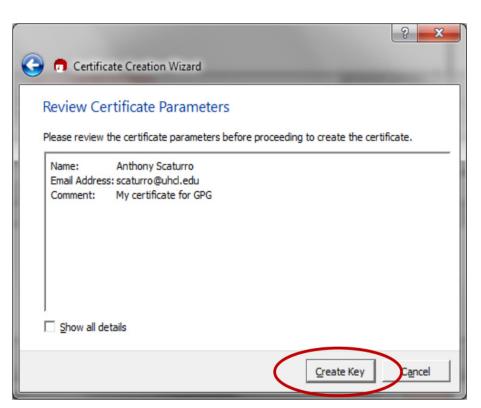
5. The Certificate Creation Wizard will start and display the following:

	? ×
🕒 👩 Certificate Creation Wizard	
Enter Details	
Please enter your personal details below. If you w parameters, dick on the Advanced Settings buttor	
Name:	(required)
EMail:	(required)
Comment:	(optional)
Name is required, but empty.	<u>A</u> dvanced Settings
	Next Cancel

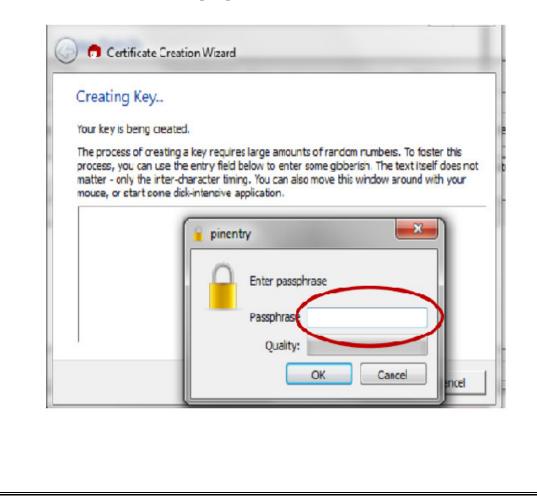
6. Enter your name and e-mail address. You may also enter an optional comment. Then, click the "Next" button



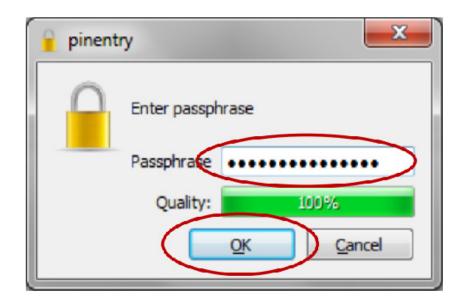
7. Review your entered values. If OK, click the "Create Key" button



8. You will be asked to enter a passphrase



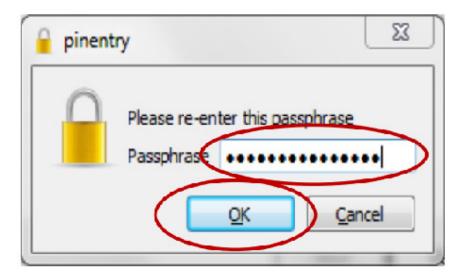
9. The passphrase should follow strong password standards. After you've entered your passphrase, click the "OK" button.



10. You will be asked to re-enter the passphrase

pinentry	23
Please re-enter this passphrase	e
<u>O</u> K <u>C</u>	ancel

11. Re-enter the passphrase value. Then click the "OK" button. If the passphrases match, the certificate will be created.



12. Once the certificate is created, the following screen will be displayed. You can save a backup of your public and private keys by clicking the "Make a backup Of Your Key Pair" button. This backup can be used to copy certificates onto other authorized computers.

ey Pair S	uccessfully Created	
	pair was created successfully. Please find details on the res xt steps below.	ult and some
Result		
	created successfully. t: 5CF546E0D74175F6978D248AC784D7802F328213	
Next Steps		
	Make a Backup Of Your Key Pair	
	Send Certificate By EMail	
	Upload Certificate To Directory Service	
		Finis

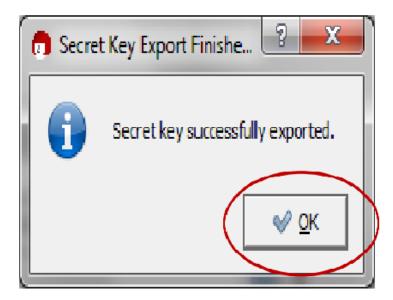
13. If you choose to backup your key pair, you will be presented with the following screen:

Export Secret Certific	ate 🦉 🗙
Please select export optio GPG) <scaturro@uhd.ed< th=""><th>ons for Anthony Scaturro (My certificate for u&gt; (2F328213):</th></scaturro@uhd.ed<>	ons for Anthony Scaturro (My certificate for u> (2F328213):
Qutput file:	
ASCII armor	
	OK Cancel

14. Specify the folder and name the file. Then click the "OK" button.

Export Secret Certificate	? ×
Please select export options for Anthony Scaturro (M GPG) <scaturro@uhcl.edu> (2F328213);</scaturro@uhcl.edu>	y certificate for
Qutput file: C:/Users/scaturro/Desktop/bkpcrt.gpg	0
SCII armor	
OK	<u>C</u> ancel

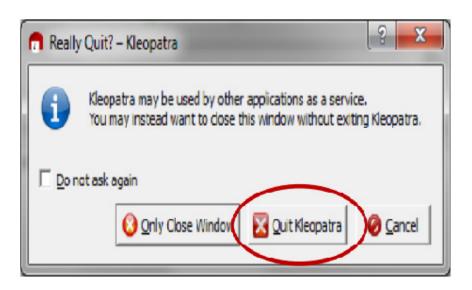
15. After the key is exported, the following will be displayed. Click the "OK" button.



16. You will be returned to the "Key Pair Successfully Created" screen. Click the "Finish" button.

ey Pair Si	uccessfully Created
	pair was created successfully. Please find details on the result and some ct steps below.
Result	
	created successfully. : 5CF546E0D74175F6978D248AC784D7802F328213
, Next Steps-	
<u>[</u>	Make a Badkup Of Your Key Pair
	Send Certificate By EMail
	Upload Certificate To Directory Service
	Finish

17. Before the program closes, you will need to confirm that you want to close the program by clicking on the "Quit Kleopatra" button



## DECRYPTING AN ENCRYPTED E-MAIL THAT HAS BEEN SENT TO YOU:

1. Open the e-mail message

Name	Better         Delete         Delete         Delete         Delete         Delete         Delete         Tasen final           Delete         Reply         Reply         Forward         Marcet         Delete         Reply         Marcet           Delete         Reply         Reply         Forward         Marcet         Deve         Reply         Marcet           Delete         Respend         Quick Streps         Quick Streps         Guick Streps         Marcet           Delete         Tur 2/23/08/4 8/35 AM         Lunch (, Kurt         Encrypted body message         Guick Streps	OneNote     Assign Mark Categorize Follow     Assign Mark Categorize Follow     Tarstate     Select -     Zoom	
All       Implicit of the second	Detete         Bergend         Quick Steps         C           Detete         Bergend         Quick Steps         C           Detete         Detete         Quick Steps         C           Detete         Detete         Quick Steps         C           Detete         Detetee         Quick Steps         C           Detete         Detected body message         Detected body message         Detected body message           Stature, Arditery		
Ver 2003/0044 & ABS MAK Land, Diczyptel do dy message           Fre * zotron.	tur 2/20/2014 4:35 AM     Lund, Kurt     Encrypted body message     Tor 2/20/2014 4:35 AM     Lund, Kurt     Encrypted body message     Tor 2/20/2014 2014 2014     Tor 2/2014     To		
Version: GnuPG v2 NoEMA9ACVI13CCEEXAQ_AvexglbfsRb6rvusRCINCIXFI9esMut Cleit.VDvr1v0KQr KSYXD07U/uexpSigLintLIFIPIIVydUYI8HXQFsvE6b3rg8Hx0e46PFLIFISHDTuyQ ZcdMazSw0rdDQLigsTTMTRibmlF7DpcrNARADomFaqlestegRIPPR20ywwDh8fk64g1+ S8976564dxV1LXvH1D78L2LLXTNRge6dVFLintMX5ABb2B2STvk69HKJ958EL IAL45g2rb0DeKg6e4vQLibr8revThicpR0HabdDAvela3Qvf9lCgEH/ZUU JR1r62Srvx6wVTholvL3L1WabddOvSc0EPFLipEXPK9FSZp00E4Kg958EL IAL45g2rtB2F498V45Ks0St010k43DV64b202FFLipEXPK9FSZp00E4Kg958EL IAL45g2rtB2F498V45Ks0St010k43DV64b202FFLipEXPK9FSZp00E4Kg958EL IAL45g2rtB2F498V45Ks0St010k43DV64b202FFLipEXPK9FSZp00E4Kg958EL IAL45g2rtB2F498V45Ks0St010k402V64BEg7VFLipEXPK9FSZp00E4Kg958EL IAL45g2rtB2F498V45Ks0St010k402V70FLipEXp00E4Kg90E4Kg970FLipEXp10E4Kg970FLipEXp10E SerCB1q0KR97U44-Jq001APylyfQ00E4Kg90E4Kg970V81y5yb53cqiqVL1AKCUNFK0V CV50U33dA8HbmHVEFK6833XXx6U4mJEb7HC9VFLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FLipEXp10FL	Version: GnuPG v2 NoEMAGACW13CXEEXAQQAwcxqibFsRb6rvusRCINCUXF9esMutDelt/Dvr1v0KQr KSYXQ07Ubuppdigt_mNLiPBIVydLIVI8kQExxEfu6rqBhXqq&6F5LifTS5D1vgQ ZcdMa23xxOr4DQg5vTM1EBinvirZ0pcr1NA8QmFAqbsBqrRB22ejyweDhB/66qis 5897653daArtL1X+HD7XIEZLK7/N6guK6IV+aAftyAVHS8QSOmg1EeAdQ3EV DwdToTDMbD0Fa/yQV0zV8BgYvTdmHTXpgDinaLwiA8b2HZ25tw6HK5958E1 IAL45pcH0DeKp6e4xQEbr8reyThicyfXs4w6IBW1UBDAHeltaQyf9CgEH/ZUU FXa1d52twrx6w0VUnjvLa1LiVM8dHd0vx6e3BPHyWISZDAAAB2HZ25tw6HK5958E1 IAL45pcH0DeKp6e4xQEbr8reyThicyfXs4w6IBW1UBDAHeltaQyf9CgEH/ZUU FXa1d52twrx6w0VUnjvLa1LiVK8dHd0vx6e3BPHyWISZDAAB2HC12FH7L00 EVCH95vH7221PBgX6/Ba2XK8UJsQpDPVQUmLbKNIFW5gDAULCKUmETFRolln CY0FU3aAA8HbmNWEFK6B3300xAftmWVSAAbd2Q2Y0Wa1y595b33qW1LXXCUHVICW0 C0dsg10/29MvG5etba0E2FVXaa6B2caFFCH6fVgsp2XXM595b314V48QUV3A BR1+VVLG6VD/W56Thtwa1gB85HT8Ce2Y740FD50G56Ak8B70x6X0+CP074bb4gLV3AB BR1+VVLG6VD/W56Thtwa1gB85HT8Ce2Y74FD50G56Ak8B70x6X0+CP074bb4gLV3AB		
KSYXD07JUlespEsjgLimHLIPBIVydUYJBIXDFxeEfuSrgHXogkEPFLIFTISt0TuyCL 2CdMatZSvOLROIgsFxTHTERSmit7DoprVNABGNFAgk+sRytPREPSolywoDhBrg/Keg1 5897555dkdr/LL/Xr-HID7XICZL/XTL/NKggUhauLikABASZkSTX-KehKIPSSEI JAL4352rHDDeks/GELVXGHTKFKggUhauLikABASZKSTX-KehKIPSSEI JAL4352rHD0ks/GELYXGHTKFKggUhauLikABASZKSTX-KehKIPSSEI JAL4352rHD0ks/GELYXGHTKFKggUhauLikABASZKSTX-KehKIPSSEI JAL4352rHD0ks/GELYXGHTKFKggUhauLikABASZKSTX-KehKIPSSEI JAL4352rHD0ks/GELYXGHTKFKggUhauLikABASZKSTX-KehKIPSSEI JAL4352rH21HgK/KBASXKSUJSgDPVQJUHLIKABUKTPSJULCKIHTFTKaULI CYDD13dA48HbemHVERGBI30XXLMmwVSAAdbO2gYOW14556813qi(WLIXCUHVICWO CXG5e]JQ/dPMC-srgbulCZLVKa66JGzHTKFGHF0GFKD42/sttWGA40+CP074ubSqLV/3el BRL+VLIGKPUWFSfHsvUSSgDPVQJESPRDFOGKAABASTROGKWABXTXKAC/CY5-BBLV8XIM L1V5SIHb0KyLn9XLnoT7qsJ/JGC33FA3y4UepumhITGaIEQMQN970T5zS04xeO DGYV -/ACn END PGP MESSAGE	KSYX007UU=sp5ajgLmNLIPBIY-diVY88kZFxxEfu5refIbXoq66PFLIPTISOTUvQ ZcdMaZ3xvCMOLQisFTMTIGEnieT7DpcrYNA8CmFAgeLsqvRefR82eJyxveDif8/64gi4 S89765ddAdYLL/X+HD7X6EZL/XT/N6gx66U*+aMYpXVH580xOmg1Fe840J8EV DwdToTDMbDDPaA/QV02V66EgYYTdmHTKppgUmaLwiiA8b2HZSTvxdHiKg5881 JAL33c:HD6Kp64vCkL8FseYHhyfXAsvdFBWIUBDANeia Zuf9/GgEHZ/UUU JKaTd5Zrxx5weYUnjvLstJivNkiHdH0tvs6LI99hcyWISZ/ponffg17Pfmc4sav2P 5+CDh0lpK67Ux451g00TAylq0DGX2KDKkDj76fLCqaEvptPxdd2wZMf DCP4Bj5+th7221fBigV6bs4VSSLU58epDFvqUmLb6NHW9bjULCKLmETFkiUn CYDfU3dA48HbmhWEH6i83XX0uMmwVSAAdb02gYOWa15y55130jQVLixZUhVICW0 cO5s6JI0/JdmvGc*gtbx0[Z2LVKap66]ccmTBcFuffepDGXk4A87KDxx3(CT5+BBLV8XDm BL+VULG6VJVWFaffhxu2gE87K8Te2QY7BFDF0GXk4A87KTxxx3(CT5+BBLV8XDm		
and and a	DGYV =/ACN END PGP MESSAGE		

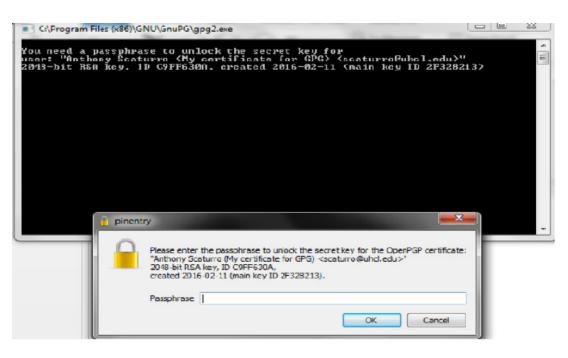
2. Select the GpgOL tab

Encrypted body message (HTML)	7 00 - 10
LE MESSACI GpgOL vlcAteeE-muil Scin ADOBE PDF	
Certificate Decryp	
Annayer General Textbody	
Tue 2/25/0056 0.35 AM	
Lund, Kurt	
Encrypted body message Solution Anthony	
avanue wienna	
ersion: GnuPG v2	
QEMA9ACW13CKEEXAQgAwcaglufsRb6rausRCINCI6R9esMuLDieLVDvr3v6KQr	
SYXQ07uUexpSqigLmNUPBIVydUYI8ck2FwEfuSrd8Nocpi6PFLIFTt5t0TuyQ	
CdMa25wOt <dqig5+tmtd8nrif7dp;yna8omfaqk+8qyr3f82ejywedhf8 64qi+<="" th=""><th></th></dqig5+tmtd8nrif7dp;yna8omfaqk+8qyr3f82ejywedhf8>	
Culvia23uCr+D2igt=TmTH3fmH7Dp;ThA6DmFAqfe45gR4762e(y+mBbH76/44q)+ 1976ScidadriL/X-iHD7XEZ/It7/A6gx661Vi=autr/p1XVH458CyOmg1484d;3EV vd7TDTMH5D9A/QV32v68EgtYv1dmhTEppgUmatux4k88222KSTv494HK95881	
89765/stdachtLL/k4-liht/DMERZL/M2T/M6Ga/k4/Wi-aa/MYg/XVM4/SS2b-C1-mg/SE&B4CJ/3EV wdToTDMbDDPbadyQCV2u2/68Cg/VVIdmhTKppgUmaLwikA8b212K2STwk9HKJ988E1 LL3pe:HCDepbadyQCV2u4/SB2FWVIdmhTKpBUMaLb10xf9CgBH22JU	
I976Scd4drLL/K+liHD7XIEZL/KT/N4gx46JUI+aMYpXVHHS82xOmg1Fe84Cj3EV wrdToTDMbDDPba/yQV3xV6BCgfVHilmhTKppgUmaLxxkk8b22KZTW49HK958EI L43pcHCDekp6e4vQ3xEF8reyThics(Xx4w6JIBMUBDANeIa1Qx/9jCgbH/2UU al 052Fixx8ix0vUmjAULLIXxMRIII004vx6e;JIS9ncyW152/201FE[21PffIud=au7P	
89765/stdachtLL/k4-liht/DMERZL/M2T/M6Ga/k4/Wi-aa/MYg/XVM4/SS2b-C1-mg/SE&B4CJ/3EV wdToTDMbDDPbadyQCV2u2/68Cg/VVIdmhTKppgUmaLwikA8b212K2STwk9HKJ988E1 LL3pe:HCDepbadyQCV2u4/SB2FWVIdmhTKpBUMaLb10xf9CgBH22JU	
IB97K5-KLeHrtLLK-ListOTXBEZL/AFL764G-Ak4Un-avMrgiXvurisSt2-to-mg15884LCJ8V wirdToTMb/DD91ay/OD24V88[grVridimTKEpgUmatkurk68E2K25Tv649ht9588] LL43pcHCDekp6a-k-QistEr8reyThics[XsAv65IBATUBDANels10q/9]CgBH/2:UU al.t052HvastivoTUBJAVLISIWR8EIH00+v68E.IP99-cyVI352(f)orfEgLPFfsub-au7r ~EShq4sEC5_243-[g00LA54]/q0OCa5Lc2008D3076HL04-respebulet_22K4f CH6j948F72E21PfgEC6684VX5LU5qe07VHqUmLb6HUMYSyLLCRUmETRioLD CH6j948F72E21PfgEC6684VX5LU5qe07VHqUmLb6HUMYSyLLCRUMETRioLD	
89765/delvh1L/K-IHOTMET2L/RT/ING_M64U/k-aMMPjKVn4i582bc0-mg15484CJ18V wdToTDMbDDPbayQCV2u/68EgYVridimhTKppgUmaLwikA8b2DEZSTw49HKj98611 LL3peritQDekp6a4-QlateFersyThecyDou/w61B9NUBDANeiL10y95/g_BH22JU al t052/knotiw07UnjvtXLIIJWR4IH00Hw61.P99ncyW152f2porffgL7Pfnu5eau7P ~cF6hig4KPC7-VAL3-Ig00L7A-glyQDD52CF0Bb3D70FL0_erspPulstRuzZkf CP4jBs+th7E22fPgZ/6b34VXSLuUSqeDPVqUmLb6HUFWSygLLCkUmETFiolLn TXDUJAA4SHomhwfLH64L30x0uHaevXSLut42Gg20Wa3z%58333gjaX1axCUhv1CWC hX62JQ/94mCxe3ptX01E2LVFap663jz2KTheftvdfyep2ZuXb9PU+blyQ052FU	
IB97K5-KLeHrtLLK-ListOTXBEZL/AFL764G-Ak4Un-avMrgiXvurisSt2-to-mg15884LCJ8V wirdToTMb/DD91ay/OD24V88[grVridimTKEpgUmatkurk68E2K25Tv649ht9588] LL43pcHCDekp6a-k-QistEr8reyThics[XsAv65IBATUBDANels10q/9]CgBH/2:UU al.t052HvastivoTUBJAVLISIWR8EIH00+v68E.IP99-cyVI352(f)orfEgLPFfsub-au7r ~EShq4sEC5_243-[g00LA54]/q0OCa5Lc2008D3076HL04-respebulet_22K4f CH6j948F72E21PfgEC6684VX5LU5qe07VHqUmLb6HUMYSyLLCRUmETRioLD CH6j948F72E21PfgEC6684VX5LU5qe07VHqUmLb6HUMYSyLLCRUMETRioLD	
897K6-ktd-ktrl,U-K-ii+OTXREZ/L/RT/AG_AKUVI-aAMP(2VVI-49K2)-O-mg 1548LC[3EV wirTOTMIADDFDHA/QOX294EgEYVIdim TKgpgUmatkuKABE2R25Y464H9(598EI L43ge-ktDakpde-kc3kEr4-ey/hicy12au-k5BRAUB2AH4is12A/99CgBH22U at032PVIS80-00000000000000000000000000000000000	
897K5-ktak-htt.JK-lisfOTXREZI/ATT/AkG-Akt/Un-axMrgiXnu4iSs2b-Or-ing 15884C[3EV wordTOTMIADDDbay/OD2V68EgYvridim TKippgUmaLxixdk82EXC5Tw64HK988E1 LL3pc+KDekp6e4v2kt8r8reyThicyX0x4v68IBANU8DANelis10y89(CgBH/2)JU al t052knoskw0TUnjvLSL3WR8E1H00vv68-IP9ncyW152f2porfEgUPftnu5eau7P ~-K5higkeRD-2x14/9001A-2ky4/QGDS12CF08bD30F/kt0-4x-psp-butR2-2x4H Chejbert72E12fPtgK/6bs47X5iJU5qeD7VgUmL6BKUMYSyJLLCKUmETRioLun PDU3dAR45HmMWLH64IS0x04Ma-WSAA4BC2gX0Wa35y548E3giqUX2-xxCH+CWC DX0a9JQPdmVx6-stgbm0dE2LVFap68jzz1ThEVdfyep22U8byPD+by8yo56szu neJAMIDIFC18ILhe=TX8ger/vsefbardx5ILV64F00C2X0ByPD+by8yo56szu NeJAMIDIFC18ILhe=TX8ger/vsefbardx5ILV64F00C2X0ByPD+by8yo56szu NeJAMIDIFC18ILhe=TX8ger/vsefbardx5ILV64F00C2X0ByPD+by8yo56szu NeJAMIDIFC18ILhe=TX8ger/vsefbardx5ILV64F00C2X0ByPD+by8yo56szu NeJAMIDIFC18ILhe=TX8ger/vsefbardx5ILV64F00C2X0ByPD+by8yo56szu NeJAMIDIFC18ILhe=TX8ger/vsefbardx5ILV64F00C2X0ByPD+by8yo56szu NeJAMIDIFC18ILhe=TX8ger/vsefbardx5ILV64F00C2X0ByPD+by8yo56szu NeJAMIDIFC18ILhe=TX8ger/vsefbardx5ILV64F00C2X0ByPD+by8yo56szu NeJAMIDIFC18ILhe=TX8ger/vsefbardx5ILV64F00C2X0ByPD+by8yo56szu NeJAMIDIFC18ILhe=TX8ger/vsefbardx5ILV64F00C2X0ByPD+by8yo56szu NeJAMIDIFC18ILhe=TX8ger/vsefbardx5ILV64F00C2X0ByPD+by8yo56szu NeJAMIDIFC18ILhe=TX8ger/vsefbardx5ILV64F00C2X0ByPD+by8yo56szu NeJAMIDIFC18ILhe=TX8ger/vsefbardx5ILV64F00C2X0ByPD+by8yo56szu NeJAMIDIFC18ILhe=TX8ger/vsefbardx5ILV64F00C2X0ByPD+by8yo56szu NeJAMIDIFC18ILhe=TX8ger/vsefbardx5ILV64F00C2X0ByPD+by8yo56szu NeJAMIDIFC18ILhe=TX8ger/vsefbardx5ILV64F00C2X0ByPD+by8yo56szu NeJAMIDIFC18ILhe=TX8ger/vsefbardx5ILV64F00C2X0ByPD+by8yo56szu NeJAMIDIFC18ILhe=TX8ger/vsefbardx5ILV64F00C2X0ByPD+by8yo56szu NeJAMIDIFC18ILhe=TX8ger/vsefbardx5ILV64F00C2X0ByPD+by8yo56szu NeJAMIDIFC18ILhe=TX8ger/vsefbardx5ILV64F00C2X0ByPD+by8yo56szu NeJAMIDIFC18ILhe=TX8ger/vsefbardx5ILV64F00C2X0ByPD+by8y0555804e0 GV	
897K6-ktd-ktrl,U-K-ii+OT2MEZ/LATI,MAG,MAKUN-aMMP(XVN-49KS2-OC-mg 1548LC)[31V wirTOTDML6DDFbJA/OD2V94EgYVridim TKpgpUmatkukA6822KS5Y464H9(5984) L4Jge-ktDavlpde-kcJate-fre-grhite/Stavu-SiBA/UBDAVIe120/g9(2;BH/2)U at032evnstvoru:OmjantziLinnetildenbursia.1999(gy)Us25/20FE[27FExit/aeu/P7 =< BhqliaKP6-7vA4-j0g00LAP-jy/cQDC3kCp0BibD)76FL0q+etypPukdtvaZMF EXDUAAA85H0-MAKUEAB000LAW-Sy3-Lat02g210W32/sy56813-0jqAVX.XCUNVICWC DKd5/UQ/dmVGc+gtbot(D2LLV/Rap66);zz1htEV0fyepzLat0xbyPD+kg%y056zu =>AMM010/C18HL=TXigg/~etaviolativ/SS3-Lat04487Kmasc/CY5-HBIL/48U =>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>	
BR7K6-ktdvtLLV-kiinOTXREZL/AFL7AkG_AK4Uki-aAMrgKVvaiKSC_VO-mg1E84LC[3EV WTOTDML6DDBAJQCO2VE8EgEVrvillam TKgpgUmatkukABSE2KSTV46Hyt6958E1 L43gerkDCApkgde4c3dkFefreyPhicytXukos5BR4UBDA4ka120_496_EBP42UU att02Pkr3K0vUTNjAKSL1XMR8H8UBO4V06L3926UV att02Pkr3K0vUTNjAKSL1XMR8H8UBO4V06L3926UV att02Pkr3K0vL7AL3QD01APajjxCD0528LC9DRAbJD76FUL0a+stypPukEFJ02UF ArcB8hgBxtb772221Pg5K6b24V58L456q20PvQuTL86HUTW5ybuL62KUmETH0L0 CF4Bj3xtb772221Pg5K6b24V58L456q20PvQuTL86HUTW5ybuL62KUmETH0L0 DC6JDQ74dmX62+gtb00L02LVFAap66jzz11184V64J02UV AUX0101C48L8L1=VSigg0+exekbar20111640245UV052UV0FV4bg40568zU AUX0101C48L8L1=VSigg0+exekbar20111640245UV052UV0FV4bg40568zU AUX0101C48L8L1=VSigg0+exekbar20111640245UV052UV0FV4bg40568zU AUX0101C48L8L1=VSigg0+exekbar20111640245UV052UV0F4bg40568zU AUX0101C48L8L1=VSigg0+exekbar20111640245UV052UV0F4bg40568zU AUX0101C48L8L1=VSigg0+exekbar20111640245UV052UV0F4bg40568zU AUX0101C48L8L1=VSigg0+exekbar20111640245UV052UV0F4bg40568zU AUX0101C48L8L1=VSigg0+exekbar20111640245UV054UV0F4bg4058zU AUX0101C48L8L1=VSigg0+exekbar20111640245UV054UV0F4bg4058zU AUX0101C48L8L1=VSigg0+exekbar20111640245UV054UV0F4bg4058zU AUX0101C48L8L1=VSigg0+exekbar20111640245UV054UV074bg40242d= AUX0101C48L8L1=VSigg0+exekbar20111640245UV054UV074bg4040242d= AUX0101C48L8L1=VSigg0+exekbar20111640245UV054UV074bg4040242d= AUX0101C48L8L1=VSigg0+exekbar20111640245UV054UV074bg4040242d= AUX0101C48L8L1=VSigg0+exekbar20111640245UV054UV040404 AUX0101C48L8L1=VSigg0+exekbar20111640245UV054UV040404 AUX0101C48L8L1=VSigg0+exekbar20111640245UV054UV0404 AUX0101C48L8L1=VSigg0+exekbar20111640245UV054UV0404 AUX0101C48L8L1=VSigg0+exekbar20111640245UV0404 AUX0101C48L8L8L1=VSigg0+exekbar20111640245UV054UV0404 AUX0101C48L8L8L40404 AUX0101C48L8L8L40404 AUX0101C48L8L40404 AUX01000404 AUX0100404 AUX0100404 AUX0100404 AUX0100404 AUX0100404 AUX0100404 AUX0100404 AUX0100404 AUX0100404 AUX0100404 AUX0100404 AUX0100404 AUX0100404 AUX0100404 AUX0100404 AUX0100404 AUX0100404 AUX0100404 AUX0100404 AUX0100404 AUX0100404 AUX010	
IBVRO-ELIAPTIL (AL-ELIYOTATIZ) (ATT (ALG)-AL-ALVIN-ALVIPSION) ISBN 0-95 (FBALC) [312] WITTOTIM (ADDBA) (XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	-
IBVRO-ELIAPTIL (AL-ELIYOTATIZ) (ATT (ALG)-AL-ALVIN-ALVIPSION) ISBN 0-95 (FBALC) [312] WITTOTIM (ADDBA) (XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	

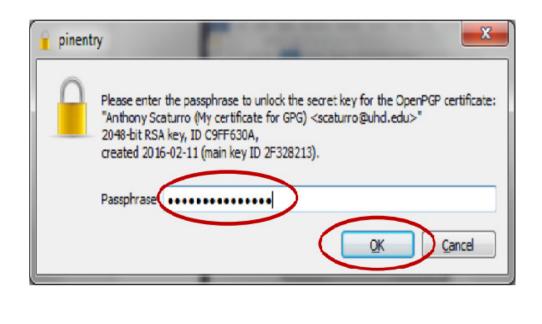
3. Click the "Decrypt" button

-	CAGE Geg0L McArce E-mill Scan ADDRE FDF	
Start Cetilitati Manager General	te Decrypt Wify	
Gereral	Tue 2/20/2016 8/35 AM	
	Lund, Kurt	
Te Scaturo, A	Encrypted body message	
in possing a	Annal	
	N PGP MESSAGE	
Version: G		
	ACW13CKEEXACgAvecrgibFsRb6rzusRCINCIXF9esMuLDIeLVDvr1x0K0r uUexp5qigLmNL/P9IVydUY/BK2ExvEfuSrq8XXoqK6PFLIFT5f0TuyQ	
ZCdMa25v	iwOs4DQ1g5+TMTd8nrtF7DprVNA8OmFAqk+8qyRRP82ejyweDHf8/64qi- iAdYLL/X+IHD7XEZL/KT/N6g-AsilVi+aMYpKVVH58QxOmg1Fe84Q(3EV	
DwdToTD	MbDDPba/yQV0zV68EgYVridmhTKppgUmaLwikA8b2fX25Twk9HKj958El	
	0DeKp6e4vQktBr8reyYhicyfXs4w5JIBWUBDANeIa1Qyf9jCgBH/2UU wx6woYUnjwtx111wNkdHid0+w5eJI9hcyWI52/ponFEg17Pfnu5+su7P	
5+rCBholo	oKP6f7uA1+j0g001APy;yfQD62kQF0RxkDj76fYLQq+ctypPukdZwZMf	
CYDfU3dA	th7211Pig1/6bu4VXSiuUSqeDPVqLmLb6NUfWSyjuLLCkUmETFkdLn A18HbmhWEHGkI300XuMewWSAAdbQ2gYOWa1ySySb13ojq/WL00CUhVICWO	
- C . C . C . D	/SdmvGc+gtbxIQEZLVKap6BjczcThEfvdfyepxZuXbyPD+ky0yo56rzu	
enoUAMID BKL+/VLIG	ElfCSI8LheTXSgwfwefbirdG3llh0d2/stLWGe4O+CP074ubSqKLv/Sel G5VDyIWFafHwaUge5818TceZrc/18EDCGKw4f81Knxsc/CY5+8BLVaXIm	
enoUAMID BKL+/VUG 1LV:SIJFH	ElfCSI8LheTXSg:wfweFblrdG9IFh0d2/stLWGe4O+CP074ubSqKLv/3e1	
enoUAMIE BKL+/VLIG 1LV35IiFIN DGW =/xCn	ШЕСЭВЬН=TXBgrofwelbardSMHb0d2,XaUWGe0+CP04ubleqEuV3+I GSVTQHW64PhuxUgo5F8FICe2CYTFFDGGKwaRF8TNnsc/CY5+B8LVa00m GKy1v9KUn+T7qdj/GC33FA3+4UspumhtTGalEQMQM97OT5s504we0	
enoUAMIE BKL+/VLIG 1LV:SII/FM DGYV =/xCn	ElfCSI8LheTXSgwfwefbirdG3llh0d2/stLWGe4O+CP074ubSqKLv/Sel G5VDyIWFafHwaUge5818TceZrc/18EDCGKw4f81Knxsc/CY5+8BLVaXIm	
enoUAMIE BKL+/VLIG 1LV:SII/FM DGYV =/xCn	ШЕСЭВЬН=TXBgrofwelbardSMHb0d2,XaUWGe0+CP04ubleqEuV3+I GSVTQHW64PhuxUgo5F8FICe2CYTFFDGGKwaRF8TNnsc/CY5+B8LVa00m GKy1v9KUn+T7qdj/GC33FA3+4UspumhtTGalEQMQM97OT5s504we0	
enoUAMIE BKL+/VLIG 1LV:SII/FM DGYV =/xCn	ШЕСЭВЬН=TXBgrofwelbardSMHb0d2,XaUWGe0+CP04ubleqEuV3+I GSVTQHW64PhuxUgo5F8FICe2CYTFFDGGKwaRF8TNnsc/CY5+B8LVa00m GKy1v9KUn+T7qdj/GC33FA3+4UspumhtTGalEQMQM97OT5s504we0	
enoUAMID BKL+//UG 1U/SIIFM DGW =/xCn END P	ШЕСЭВЬН=TXBgrofwelbardSMHb0d2,XaUWGe0+CP04ubleqEuV3+I GSVTQHW64PhuxUgo5F8FICe2CYTFFDGGKwaRF8TNnsc/CY5+B8LVa00m GKy1v9KUn+T7qdj/GC33FA3+4UspumhtTGalEQMQM97OT5s504we0	
enoUAMID BKL+//UG 1LV/SIIFM DGW =/xCn END P	IEIECIBILIETXIgwfwelburdSHIFbd2,XuLWGedO+CPO4ubleKuV3et GRVUMMATHANULgeStRIFC-ZEVYERDGGkundBirtnusc/CYVEHBU Naktm GRVLMATHANULgeStRIFC-ZEVYERDGGkundBirtnusc/CYVEHBU Naktm GRVLMATHANLgeStRIFC-ZEVYERDGkundhiTCalEQMQN97OTSe508we0	
enoUAMID BRL+/VLIG 1LV-SIIFM DGW =/xCn END P	IEIECIBILIETXIgwfwelburdSHIFbd2,XuLWGedO+CPO4ubleKuV3et GRVUMMATHANULgeStRIFC-ZEVYERDGGkundBirtnusc/CYVEHBU Naktm GRVLMATHANULgeStRIFC-ZEVYERDGGkundBirtnusc/CYVEHBU Naktm GRVLMATHANLgeStRIFC-ZEVYERDGkundhiTCalEQMQN97OTSe508we0	

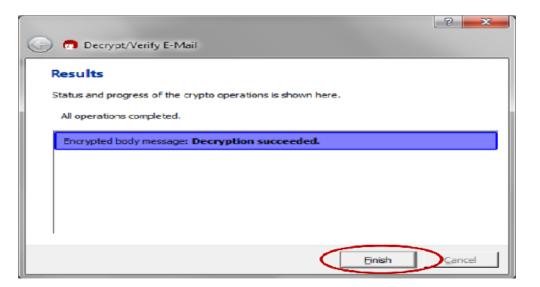
4. A command window will open along with a window that asks for the Passphrase to your private key that will be used to decrypt the incoming message.



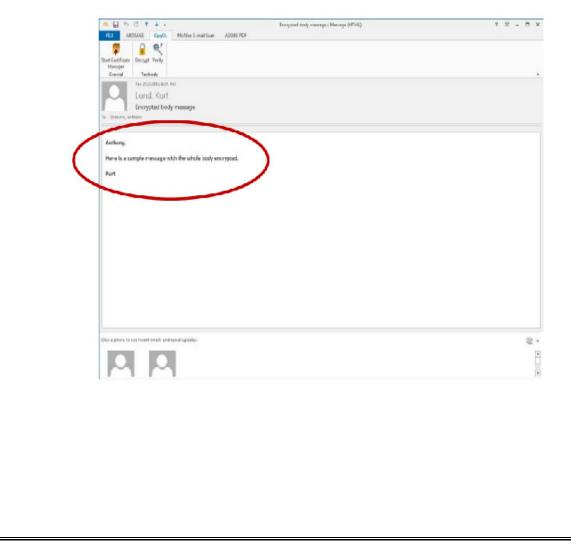
5. Enter your passphrase and click the "OK" button



6. The results window will tell you if the decryption succeeded. Click the "Finish" button top close the window



7. Your unencrypted e-mail message body will be displayed.



8. When you close the e-mail you will be asked if you want to save the e-mail message in its unencrypted form. For maximum security, click the "No" button. This will keep the message encrypted within the e-mail system and will require you to enter your passphrase each time you reopen the e-mail message



## **RESULT:**

Thus the secure data storage, secure data transmission and for creating digital signatures (GnuPG) was developed successfully.

#### EX. NO: 05

# <u>WORKING WITH KF SENSOR TOOL FOR CREATING AND</u> <u>MONITORING HONEYPOT</u>

## AIM:

Honey Pot is a device placed on Computer Network specifically designed to capture malicious network traffic. KF Sensor is the tool to setup as honeypot when KF Sensor is running it places a siren icon in the windows system tray in the bottom right of the screen. If there are no alerts then green icon is displayed.

#### **INTRODUCTION:**

#### **HONEY POT:**

A honeypot is a computer system that is set up to act as a decoy to lure cyber attackers, and to detect, deflect or study attempts to gain unauthorized access to information systems. Generally, it consists of a computer, applications, and data that simulate the behavior of a real system that appears to be part of a network but is actually isolated and closely monitored. All communications with a honeypot are considered hostile, as there's no reason for legitimate users to access a honeypot. Viewing and logging this activity can provide an insight into the level and types of threat a network infrastructure faces while distracting attackers away from assets of real value. Honeypots can be classified based on their deployment (use/action) and based on their level of involvement.

#### Based on deployment, honeypots may be classified as:

- 1. Production honeypots
- 2. Research honeypots

**Production honeypots** are easy to use, capture only limited information, and are used primarily by companies or corporations. Production honeypots are placed inside the production network with other production servers by an organization to improve their overall state of security. Normally, production honeypots are low-interaction honeypots, which are easier to deploy. They give less information about the attacks or attackers than research honeypots.

**Research honeypots** are run to gather information about the motives and tactics of the Black hat community targeting different networks. These honeypots do not add direct value to a specific organization; instead, they are used to research the threats that organizations face and to learn how to better protect against those threats.

#### KF SENSOR:

KFSensor is a Windows based honeypot Intrusion Detection System (IDS). It acts as a honeypot to attract and detect hackers and worms by simulating vulnerable system services and trojans. By acting as a decoy server it can divert attacks from critical systems and provide a higher level of information than can be achieved by using firewalls and NIDS alone. KFSensor is a system installed in a network in order to divert and study an attacker's behavior. This is a new technique that is very effective in detecting attacks.

The main feature of KFSensor is that every connection it receives is a suspect hence it results in very few false alerts. At the heart of KFSensor sits a powerful internet daemon service that is built to handle multiple ports and IP addresses. It is written to resist denial of service and buffer overflow attacks. Building on this flexibility KFSensor can respond to connections in a variety of ways, from simple port listening and basic services (such as echo), to complex simulations of standard system services. For the HTTP protocol KFSensor accurately simulates the way Microsoft's web server (IIS) responds to both valid and invalid requests. As well as being able to host a website it also handles complexities such as range requests and client side cache negotiations. This makes it extremely difficult for an attacker to fingerprint, or identify KFSensor as a honeypot.

#### **PROCEDURE:**

STEP-1: Download KF Sensor Evaluation Setup File from KF Sensor Website.

- **STEP-2:** Install with License Agreement and appropriate directory path.
- **STEP-3:** Reboot the Computer now. The KF Sensor automatically starts during windows boot.

**STEP-4:** Click Next to setup wizard.

- **STEP-5:** Select all port classes to include and Click Next.
- STEP-6: "Send the email and Send from email", enter the ID and Click Next.
- STEP-7: Select the options such as Denial of Service[DOS], Port Activity, Proxy Emulsion, Network Port Analyzer, Click Next.
- **STEP-8:** Select Install as System service and Click Next.

STEP-9: Click finish.

# **SCREENSHOTS:**

KESensor Professional - Evalu									- 5 ×
e view Scenario Sgnatures Sett									
3 🗟 🥃 🍕 🎖 🔠 낶		S 🖓 🖬 🖨 🏭 🗷	00 🐱	*	2				
💡 kisensor - localhost - M., 🛧		Start	Duration	Pr	Sins	Nane	Vsitor	5g. Message	Received
🖶 🚞 TCP	26	3J1J20129:54:40 AM 656	0.030	UD2	138	NB1 Detegrem	BTP5-PC		NBT DGRAM Packet: id:5
- 🕰 O Clesed TCP Ports	25 24 23	3JLJ2012 9:54:29 AM 015	0.030		60	115	COMF2	IIS view stript $s_{min}$	PROPERID JEanonLEP H
- 🧟 <del>2L FTP - Error</del>	🚨 24	3/L/2012 9:54:25 AM 015	0.010	тер		115	COMF2	IIS view script s	OPTIONS / HTTP/1 .: [DE
	<b>B</b> 23	SILI20129:54:11 AM 343	0.030	UD2	138	NBT Datagram 🕠	C0M15		NBT DGFAM Packet: id:0
	22 2. 20	311 J2012 9 :53:2E AM 968	01020	TOP		115	COMEZ	IIS view script s	PROPERIND JEaronLBP H
🗿 63 DHCP	🖉 2.	3/L/2012 9:53:2E AM 968	0.010	TOP	60	ns	COME2	IIS VIEW STIPLS	OPTIONS ( HTTP/LL.[OD
🧕 80 IIS - Recent	20	3/1/2012 9:53:22 AM 328	0.010		120	NBI Datagram	COMPLITER 14		DBT DGFAM Packet: id::
🙍 1LD POPS	目 19	SJ1/2012 9:53:05 AM 093	0.010		138	NB1 Detagram	FRONTOFFICEPC		NBT DGRAM Packet: id:0
- 🧟 119 MNT2	<b>2</b> 18	3/L/2012 9:52:56 AM 453	0.010		138	NBT Detagram 🕠	COMFL		NBT DGRAM Packet : id:0
135 HS RFC Error	817	3JLJ2012 9:52:54 AM 656	0.030			NB1 Datagram	C0M15		NBT DGFAM Packet: id:0
🧾 139 NBTSession 5	16	SILI2012 9:52:54 AM 609	0.030			NBT Datagram	comt5		NBT DGRAM Packet: id:0
319 LDAP		3/L/2012 9:52:42 AM 046		UD2	68	DHCP Clieni	192, 158, 1, 1		[02 01 05 01 D0(k[08] [)
- 🧟 443 HTTP5	<u>I</u> III	3/1/2012 9:52:10 AM 201	0.010		67	DHCP	com15		DHC": Dob %cque#[04
A15 MBTSMB Error	<b>A</b> 13	3/1/2012 9:52:41 AM 734	0.010		138	NBT Datagram	CIVEDEPT		NBT DGFAM Packet: id:1
<b>1</b> 533 CI5	圓12	3/1/2012 9:52:38 AM 750		ШР?	128	NBT Datagram	BTPS-PC		NBT DGRAM Packet: id:5
1128 MSCIS		3/L/2012 9:52:31 AM 078	0.030		67	DHCP	BTPS-PC		DHCP: Boot Request[0A
	10 ag 10	3/L/2012 9:52:28 AM 953	0.030		80		COMF2	IIS view stript s	PROPEIND (CaronL&P H
•	- <b>1</b>	3/L/2012 9:52:23 AM 000		τФ		115	COMF2	IIS view stript s	OPTIONS ( HTTP/1 .: [OE
📑 1133 SQ. Server	1 ja 1	3/L/2012 9:52:23 AM 015		TOP	80	115	COMF2	IIS view stript s	
2234 Directplay	1 <u></u> ]] <sup>7</sup>	3/L/2012 9:52:10 AM 562	0.010		100	NB1 Dotogram	CIVILIDEPT		NBT DGFAM Focket: id:0
- 🏭 3128 IISProxy	<b>B</b> <sup>6</sup>	3/1/2012 9:52:06 AM 78.	0.010		128	NBT Datagram	ocm15		DBT DGFAM Packet: id: 3
🗿 3268 Global Catal	85	3/1/2012 9:5.:55 AM 03.	0.010		138	NBT Datagram	COMFL		NBT DGRAM Packet: id::
🔒 3389 Terminal Ser	昌 4	3/1/2012 9:5.:49 AM 93?	0.010		138	NBT Datagram	COM15		NBT DGRAM Packet: id:0
	<u>특</u> 1	3/L/2012 9 :5 : :30 AM 500	0.010		138	NBT Detegrem	COMFL1		NBT DGRAM Packet: id:0
🔚 5357 Web Service	<b>B</b> 2	3JL/2012 9:5.:20 AM 974	0.010		68	DHCP dien/	192, 158, 1, 1		[02 01 05 00]L#[B0 03 0
	<b>B</b> 1	3/L/2012 9:5.:20 AM 968	0.010	UD2	67	DHCP	ccm15		DHCP: Boot Request[0A
🗄 🧰 WP									
- 🎣 O ClosedUDP Ports 🔚									
🗃 🕂 MONS UD?									
67 DHCP -Rece									
📕 68 CHCP Client									
- 👔 83 Kerberos									
138 NBT Datagr									
- 🛱 151. STUMP 🛛 😭									
	<								>
	,								

🛢 🗟 📢 👗 🎁 👯 🚏	10 E 9	📳 🔍 🖫 🔒 💕 🖴	10 M	*	2				
🐛 vistors	ID	Start	Duration	Pr	Sers	Name	Visitor	Sig. Ness age	Received
178.76.252.25 - Recen	@ 11	0/1/2012 9/56/17 AM-398	0.000	UIDP	1222	UDP Packet:	:222.107.67.174		[0E D1 15 00 02]9[2
122.45.101.467 - Recen	@ 43	3JL/2012 9/56117 AM.177	0.000	UDP	1224	UDP Packet:	178.76.252.26		Out[00 DID 4E E3]+3
176.73.49.60 - Recent	@P 42	3/L/2:012 9/56/L6 AM 895	0.000	UDP	1223	UDP Packet:	178.76.252.26		[CA CL]][00 F7 BB+
192.168.1.1 - Recent A	<b>3</b> 41	3/1/2012 9/57/08 AM /968	0.000	UDP	67	DHCP	IBTPS-PC		DHCPI Boot Reque
192.168.1.51 - BTP5-PC.	@ +0	3/L/2012 9:56:16 AM 21.1	0.000	UDP	1217	UDP Packet:	222.107.67.174		[D7 0D 00 D5 04 E
192.168.1.52 - CIVIL DE	<b>=</b> 39	3/1/2012 9:56:55 AN 375	000.0	LIDP	138	NBT Datagram	con15		NET DGRAM Packet
192.168.1.5-3 - FRONTO	@ 36	3/1/2012 9:56:16 AN.147	0.000	LIDP	1220	LEP Packet:	176.73.49.60		[E1 96 13 0D]W[04
192.168.1.58 - COMPUT	@ 37	3/L/2012 9:56:15 AN 821		LIDP	1219	LIDP Packet:	176.73.49.60		[6A 97 17 00 6A 88
192.168.1.62 - COMP11	<u>3</u> 3	3/1/2012 9:56±29 AM.125	0.000	TCP	80	115	COMP2	IIIS view script a	PROPERING JCanon
192.168.1.6.5 - ELEC TRL.	<b>25</b> 36	3/1/2:012 9:96::29 AM.125	000.0		80	115	COMP2	IIIS view script s	OPTIONS ( HTTP/1
192.168.1.70 - COMP2	@ >+	3/1/2012 9/56/15 AM-262	0.000			UIDP Packet:	122.45.101.67		[80][LF=00 D2:F5]
192.168.1.73 - COMP1	@ :::	3/L/2:012 9/56/L5 AM.049	0.000			UDP Packet:	122.45.101.67		6[1900]M[AD C9]
192,168,1,1 14 - com15	@ %	3/1/2012 9/56/11+AM-652	0.000	UDP		UBP Packet:	122.45.101.67		[02]K [00] F6 F7 0E
192.168.56.1 - COM1.5	<b>a</b> 31	3/L/2012 9/95/64 AM 406	0.000			NBT Datagrani	IELECTRICALDEP1		NET DGRUM Packe
222.107.67.174 - Rece	230 2329	3/L(2012 9:55:29 AM.093	0.016			115	COMP2	IIIS view script s	PROPERIND Conort
	<u>a</u> 29	3/1/2012 R 35:29 AN 046		TCP			CONF2	ILS VIEW INTER I	OPTIONS ( HTTP)
	8 <u>~</u>	3/1/2012 9:55-20 AN 090	0.000	LIDP	60 67	DHCP Clent	192.160.0.1		
	27	3/1/2012 9:55:26 AM.076	0.000		138	DescP	con15		DHCP: Boot Reque
	<b>2</b> 26	3/1/2012 9:54:40 AM //56	0.000	LIDP	138	NBT Datagram ItS	IBTPS-PC COMP2		NBT DGRAM Packet PR OPF3MD JCanoni
	25 29	3/1/2012 9:54:29 AM.01.5 3/1/2012 9:54:29 AM.01.5	0.000		80	115	COMP2 COMP2	IIIS view script s IIIS view script s	OPTIONS / HTTP/1
	23	3/L/2012 9/94/29 AM //L/5	0.000			NET Dataoran	CON15	THES WERE SCRIPT STORE	N8TDGRJM Packe
		3/L/2012 9/53/28 AM/968	0.000	TKP	80	IIS	COMP2	IIIS view script s	PROPERIND JCanon
	222 	3/L(2012 9/53/28 AM/968		TCP	80	115	COMP2	IIIS view script s	OPTIONS ( HTTP()
	21 20	3/L(2012 9:53:22 AN 325	0.000	LIDP	136	MBT Deteorem	CONFUTER14	THES MON SCRIPT SILL	NTDGRAM Packs
	819	3/L(2012 9:53:09 AM.093	0,000		138	NET Decegren	IRONTOPPICE PC		NET DGRAM Packet
	310	3/L(2012 9:52:56 AN /613		LIDP	135	NET Debegriere	CONF1		NET DGRAM Packet
	<b>1</b> 17	3/L(2012 9:52:54 AN 456		LIDP	138	NBT Datagram	CONLS		NET DGRAM Packet
	<b>E</b> 16	3/1/2/012 9:52-54 AM-60/9	0.000		138	NBT Datagram	cont5		NET DGRAM Packet
	<b>E</b> 15	3/L/2012 9:52:42 AM.046		LIDP	68	DHCP Clenit	192.168.1.1		T02 01 05 00 0014
	3	3/1/2012 9/52146 AM 234		LIDP	67	DHCP	contS		DHCPI Boot Reque
	13	3JL/2012 9/52141 AM.734	0.000	LIDP	138	NET Datagram	CIVILDEPT		NRTDGRAM Packet
	12	3/L(2012 9/52)/38 AM.750		UDP	138	NBT Datagram	INTEGER (		N8TDGRAM Packet
	Bu	3/L/2012 9/52/31 AM.078	0.000		67	DHCP	INTESPC		DHCPI Boot Reque
	310	3JL(2012 9:52:26 AM,953	0.000		80	115	COMP2	IIS view script s	PROPERIND JCanon
	3.	311/2012 0-92-23 IN 000	0.000			11.5	COM07	TER share entitle	OPTIONS / HTTP/S
	<								

erv Scenario Signatures Se								
🕯 🗟 剩 🚹 🖬 👭 🕷	' 📧 🕾 🖻 🖓	🐚 🔒	êl 🛛	BB 📈	* *			
Visitors	et .	Duration	Ph	Sens	Nane	Visitor	Sig. Miessage	Received
🔏 0.0.0.0 - com15 - R	2012 9 (58) 30 AM 217	0.000	UDP	1523	UDP Packet	MICROSOF-6566EA		8[D7][00]EM[C6 80]K#[A866 A6 8.
24.51.70.132 - Rece	2012 9 (58:29 AM.903	0.000	UIDP	1922	UDP Packet	MICROSOF-6566EA		H[15 1A 00 C9]V6[FE]][AO EF 16 06
🤾 31.121.81.158 - Rec.	2012 1 0 10 129 AM	0.016	TICP	90	IIS	COMP2	EIS view solot s	PROPEND J CaroniaP HTTP/L (10D)
16.43.203.12 - Rece	2012 1-0100129 AM	0.000	TICP	90	IIS	COMP2	EIS view soript so.	OPTIONS / HTTP/ L.L[00 DA]translat.
46.63.6.244 - MDRD	2012 9 :50:29 AM 501	D.000	UIDP	1520	UDP Pecket	70.97.100.133		(F+ IF 13 00 80) #(A5 03) (91) (10)
46.1632.77.223 - Ret	2012 9 :50:29 AM 207	D_000	LIDP	1518	LIDP Parchet	70.97.100.133		n#16(00 90 90 AD A4 00)[07]6~[07
46.100.169.132 - R.,	2012 9 :58:29 AM.020	D_000	LIDP	1516	LIDP Parchet	112.205.4.149.pidt.net		(D0]+(00 8*)/q*-(00)q(C9 93 0* A
46.241.118.5 - Recs.	2012 9 50:25 AM 525	D_000	LIDP	1515	LIDF Packat	112.205.4.149.pldt.net		[12:03 10:00 00 0A][00]05[A3]0[C
49.248.13.190 - smt	2012 9 58:28 AM.196	0.000	LIDP	1514	LIDP Parchat	78.111.104.187		[WF 82],[D0]0[E7 07 C7 89 AF]p[EF]
58.117.151.26 - Rec.	2012 9 58:27 AM.857	D_000	LIDP	1513	UDP Packet	78.111.104.187		VEGE OF OOT BUICS FOTOEBINGATUE.
59.23.151.204 - Rec.	2012 9 (58:27 AM.310	0.000	UICP	1510	UDP Packet	189-173-12-171.next		THE FAILT OF EACHTTLE CADA
59.126.122.92 - Rec.	2012 9 IS8:26 AM 667	0.000	UICP	1907	UDP Packet	CAMERAS		[#7]5[00][E819 0E84 % D11C 98
59.1+44.55.226 - AES	2012 9 (68)26 AM (695	0.000	UIDP	1909	UDP Packet	santp .ttml.co.in		[0884]%[00]8(970F05]2(840A 6
	2012 9 (58:26 AM-561	0.000	UDP	1908	UDP Packet	santp.ttml.co.in		sr[13.00]7s(F7.F0+94.AD]O[81.07][5]
	2012 9 :58:26 AM.329	0.000	UIDP	1506	UDP Pecket	CAMERAS		[84 9* 17 00 84 60]-[16 61 61 6*]-[
1.34.107.100 - Rec.	2012 9 58:25 AM.971	D_000	LIDP	1504	UDP Packat	bacd02d6, vitua.com.br		R{00.87]ggt(86)k(3PA6.89P9.A9
71.43.42.154 - #05	2012 9 :58:25 AM.380	D_000	LIDP	1503	LIDP Packat	bacd02d6, virtua.com.br		H[15 1A 00 C9]/6[P2]][AO EP 16 D6
77.238.93.227 - Rec.	2012 9 :58:24 AM.818	D_000	LIDP	1502	LIDP Parchat	customer144131.meg		[89]6"[00 F8]\$(A4]>[9A 02]{[EC C
2 78.60.251.50 - Rece	2012 9 58:24 AM.349	D_000	LIDP	1501	LIDP Parchatt	customer144131.mag		[LE B3 18 00 BD GIA][BD]BS[A3]Q[C
💈 78.97.105.133 - Rec	2012 9 :58:13 AM.601	D_000	LIDP	1.494	UDP Packet	CC_KOMP4		F.[1700F1]v[98]v,[E2]tvV,[[0189]
78.97.160.74 - Recs.	2012 9 ISBIL3 AM 212	0.000	UICP	1.493	UDP Packet	OC_KOMP4		[#5 15 15:00 CE]V[04]vi ~[F7]V[D8]
💈 78.111.104.187 - R.a.	2012 1 0 09 (29 AM	0.000	TICP	80	IIS	COMP2	EIS view script so	PROPEIND / CanonLBP HTTP/L, 1[00/
A 79.1 14.1 72.205 - 79	2012 1 0 09 129 AM	0.000	TICP	80	IIS	O0#P2	EIS view script s	OPTIONS / HTTP/ L.L[00 DA]translat
💈 79.125.89.51 - IP-0	2012 9 (58) L2 AM (932	0.000	UIDP	1.492	UDP Packet	1.1.2.206. 1.93.74.pkt		*[96]0[00 B8]T[A6]-[E9 00]efg[05](
2 79.170.188.185 - DC	2012 9 :50:L2 AM /473	0.000	UIDP	1491	UDP Pecket	LE2.206. E83.74.pidt		3[85 1400]_[05 D6 64]df[10]_[03])
2 82.80.130.160 - CA	2012 9 55:L1 AM.897	D_000	LIDP	1490	LIDP Parchat	178-168-14-215.mtar		a[12](00 PB]=(PD 85])(88)-(D2 8P
1 84.240.34.146 - ACE	2012 9 59:11 AM 595	D_000	LIDP	1.433	LIDP Parchat	170-160-14-215.etar		)[74 05 00 F5 C1]](C3 54 07].[F5 5
85.122.41.194 - Rec.	2012 9 :58:L0 AM.164	D_000_0	LIDP	1404	UDP Patchat	129-74-127-249.mga		7+[90]+[00];[02 EC E3 E5 E9];[A2 5
85.2D4.143.139 - R	2012 9 58:09 AM 676	D_000	LIDP	1483	LIDP Parchatt	189-74-127-249.erga		[C5]{[16 ID 96]R[F3 ED E-4 t5]q[F5]
87.69.221.227 - USE	2012 9 :58:08 AM.813	0.000	UDP	1490	UDP Packet	AGER		[1,F 84]\$[00 06 C1,F2]6[9F A4]\$[F8 .
88.80.107.148 - HOL.	2012 9 (58) 06 AM / 496	0.000	UDP	1479	UDP Packet	ACER		AW/[00:80:90]3[:19:6+ 69:01 6E][[0;
88.2D8.208.192 - se	2012 9 ISBIO7 AM-272	0.000	UICP	1472	UDP Packet	CHANGE//E1		r[Ec]_[00_8D_05_81].#v[8_1]eee[8E_
	2012 9 (98) 06 AM (999	0.000	UICP	1471	UDP Packet	CHANGE/1E1		[012]K[00 P6 F7 0E 04 C7 16]J[0+ A
	2012 9 (58:06 AM/557	0.000	UIDP	1.470	UDP Packet	78.97.160.74		[FO 82 18:00]/\$[18]!=[ED D1 A9185
2 92.114.113.151 - Rm	2012 9 55:06 AM 331	D.000	UIDP	1469	UDP Packet	78.97.160.74		[AS EC]N[00FE]>3[01 9E FD][[40-8.
2 93.118.250.195 - R 2	C 2012 0 49409 BM 201	n aaa	1810	1 ann	100 Dariat	1998 174 807 104		fore os tecnolofina noteñerena seño

## **RESULT:**

Thus the study of setup a hotspot and monitor the hotspot on network has been developed successfully.

#### EX. NO: 06

## **INSTALLATION OF ROOTKITS**

## AIM:

Rootkit is a stealth type of malicious software designed to hide the existence of certain process from normal methods of detection and enables continued privileged access to a computer.

#### **INTRODUCTION:**

Breaking the term rootkit into the two component words, root and kit, is a useful way to define it. Root is a UNIX/Linux term that's the equivalent of Administrator in Windows. The word kit denotes programs that allow someone to obtain root/admin-level access to the computer by executing the programs in the kit — all of which is done without end-user consent or knowledge.

A rootkit is a type of malicious software that is activated each time your system boots up. Rootkits are difficult to detect because they are activated before your system's Operating System has completely booted up. A rootkit often allows the installation of hidden files, processes, hidden user accounts, and more in the systems OS. Rootkits are able to intercept data from terminals, network connections, and the keyboard.

Rootkits have two primary functions: remote command/control (back door) and software eavesdropping. Rootkits allow someone, legitimate or otherwise, to administratively control a computer. This means executing files, accessing logs, monitoring user activity, and even changing the computer's configuration. Therefore, in the strictest sense, even versions of VNC are rootkits. This surprises most people, as they consider rootkits to be solely malware, but in of themselves they aren't malicious at all.

The presence of a rootkit on a network was first documented in the early 1990s. At that time, Sun and Linux operating systems were the primary targets for a hacker looking to install a rootkit. Today, rootkits are available for a number of operating systems, including Windows, and are increasingly difficult to detect on any network.

## **PROCEDURE:**

STEP-1: Download Rootkit Tool from GMER website www.gmer.net.

- STEP-2: This displays the Processes, Modules, Services, Files, Registry, RootKit / Malwares, Autostart, CMD of local host.
- STEP-3: Select Processes menu and kill any unwanted process if any.
- STEP-4: Modules menu displays the various system files like .sys, .dll
- **STEP-5:** Services menu displays the complete services running with Autostart, Enable, Disable, System, Boot.

**STEP-6:** Files menu displays full files on Hard-Disk volumes.

**STEP-7:** Registry displays Hkey\_Current\_user and Hkey\_Local\_Machine.

STEP-8: Rootkits / Malwares scans the local drives selected.

**STEP-9:** Autostart displays the registry base Autostart applications.

STEP-10:CMD allows the user to interact with command line utilities or Registry

#### **SCREENSHOTS:**

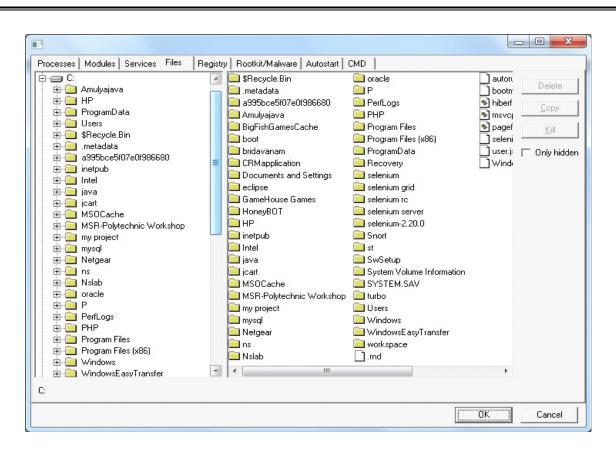
rocesses   Modules   Services   File:	1 1					11 e	LK LC	1520	
Process	Parameters	PID	Memory	Thr	Handles	User time	Kernel time	-	Kill process
ystem Idle vstem		0 4	24 1372	4 132	0 913	0.000	5038.925 70.730	=	
ystem mss.exe		4 336	900	2	35	0.000	0.062		Kill all
vchost.exe		352	19236	20	571	1.201	1.466		
siss.exe		468	4004	10	900	0.093	1.216		Restore SSD
ntsvr.exe		484	4296	4	87	0.000	0.000		
vchost.exe		496	115504	22	641	35.443	4.243		
vchost.exe		524	51824	52	1850	8.361	6.396		Restart
srss.exe		588	13384	10	718	0.546	5.397		-
vininit.exe		596	3632	3	81	0.000	0.187		
vinlogon.exe		632	5136	3	117	0.093	0.265		Libraries
ervices.exe		692	10364	8	342	0.889	1.466		I. Libialies
ass.exe		700	11756	10	1031	79.856	105.425		
m.exe		708	4096	10	229	0.078	0.078		
udiodg.exe		752	27900	21	431	205.000	2.433		
vchost.exe		804	9524	12	423	4.976	5.974		
vchost.exe		892	11792	10	517	0.951	0.920		
1sMpEng.exe		964	48132	27	454	70.668	4.664		Files
VINWORD.EXE		988	59612	8	989	26.894	5.428		
IPDrvMntSvc.exe		1012	3228	4	71	0.015	0.000	Ŧ	
ibraries Threads									
Name					Size	Addre	388		
rocesses: 146			Comm	and: [					R <u>u</u> n
								к	Cancel

## **CS6711 SECURITY LABORATORY**

Processes Modules	Services Files Registry Rootkit/Malware Autostart	CMD			
Name	File	Address	Size		*
ntoskrnl.exe	\SystemRoot\system32\ntoskrnl.exe	0305C000	6193152		
hal.dll	\SystemRoot\system32\hal.dll	03013000	299008		
kdcom.dll	\SystemRoot\system32\kdcom.dll	00BCC000	40960		E
mcupdate_Genui	\SystemRoot\system32\mcupdate_GenuineIntel.dll	0000000	323584		
PSHED.dll	\SystemRoot\system32\PSHED.dll	00C4F000	81920		
CLFS.SYS	\SystemRoot\system32\CLFS.SYS	00C63000	385024		
Ci.dli	\SystemRoot\system32\Cl.dll	00CC1000	786432		
Wdf01000.sys	\SystemRoot\system32\drivers\Wdf01000.sys	00EA3000	671744		
WDFLDR.SYS	\SystemRoot\system32\drivers\WDFLDR.SYS	00F47000	61440		
ACPI.sys	\SystemRoot\system32\drivers\ACPI.sys	00F56000	356352		
WMILIB.SYS	\SystemRoot\system32\drivers\WMILIB.SYS	00FAD 000	36864		
msisadrv.sys	\SystemRoot\system32\drivers\msisadrv.sys	00FB6000	40960		
pci.sys	\SystemRoot\system32\drivers\pci.sys	00FC0000	208896		
vdrvroot.sys	\SystemRoot\system32\drivers\vdrvroot.sys	00FF3000	53248		
partmgr.sys	\SystemRoot\System32\drivers\partmgr.sys	00E00000	86016		
compbatt.sys	\SystemRoot\system32\DRIVERS\compbatt.sys	00E15000	36864		
BATTC.SYS	\SystemRoot\system32\DRIVERS\BATTC.SYS	00E1E000	49152		
volmgr.sys	\SystemRoot\system32\drivers\volmgr.sys	00E2A000	86016		
volmgrx.sys	\SystemRoot\System32\drivers\volmgrx.sys	00E3F000	376832		
mountmgr.sys	\SystemRoot\System32\drivers\mountmgr.sys	00D81000	106496	-	
iaStor.sys	\SystemRoot\system32\DRIVERS\iaStor.sys	0103A000	2138112		
atapi.sys	\SystemRoot\system32\drivers\atapi.sys	01244000	36864		
ataport.SYS	\SystemRoot\system32\drivers\ataport.SYS	0124D000	172032		
msahci.sys	\SystemRoot\system32\drivers\msahci.sys	01277000	45056		
PCIIDEX.SYS	\SystemRoot\system32\drivers\PCIIDEX.SYS	01282000	65536		
amdxata.sys	\SystemRoot\system32\drivers\amdxata.sys	01292000	45056		
fltmgr.sys	\SystemRoot\system32\drivers\fltmgr.sys	0129D000	311296		
fileinfo.sys	\SystemRoot\system32\drivers\fileinfo.sys	012E9000	81920		
Ntfs.sys	\SystemRoot\System32\Drivers\Ntfs.sys	0142D000	1716224		
msrpc.sys	\SystemRoot\System32\Drivers\msrpc.sys	012FD000	385024		
ksecdd.sys	\SystemRoot\System32\Drivers\ksecdd.sys	015D0000	110592		-

Processes   Module		Files Registry Rootkit/Malware Autostart CM		
Name	Start	File name	Description	<u> </u>
.NET CLR Data .NET CLR Netwo .NET CLR Netwo .NET Data Provid .NET Data Provid				
NETFramework				
1394ohci	MANUAL	\SystemRoot\system32\drivers\1394ohci.sys	1394 OHCI Compliant Host Controller	
ACPI	BOOT	system32\drivers\ACPI.sys	Microsoft ACPI Driver	
AcpiPmi	MANUAL	\SystemRoot\system32\drivers\acpipmi.sys	ACPI Power Meter Driver	
adp94xx	MANUAL	\SystemRoot\system32\DRIVERS\adp94xx.sys		
adpahci	MANUAL	\SystemRoot\system32\DRIVERS\adpahci.sys		
adpu320	MANUAL	\SystemRoot\system32\DRIVERS\adpu320.sys		
adsi				
AeLookupSvc	MANUAL	%systemroot%\system32\svchost.exe -k netsvcs	@%SystemRoot%\system32\aelupsvc.dll,-2	
AERTFilters	AUTO	C:\Program Files\Realtek\Audio\HDA\AERTSr	Andrea RT Filters Service	
AFD	SYSTEM	\SystemRoot\system32\drivers\afd.sys	@%systemroot%\system32\drivers\afd.sys,-1000	
AgereSoftModem	MANUAL	system32\DRIVERS\agrsm64.sys	Agere Systems Soft Modem	
agp440	MANUAL	\SystemRoot\system32\drivers\agp440.sys	Intel AGP Bus Filter	
ALG	MANUAL	%SystemRoot%\System32\alg.exe	@%SystemRoot%\system32\Alg.exe,-113	
aliide	MANUAL	\SystemRoot\system32\drivers\aliide.sys		
amdide	MANUAL	\SystemRoot\system32\drivers\amdide.sys		
AmdK8	MANUAL	\SystemRoot\system32\DRIVERS\amdk8.sys	AMD K8 Processor Driver	
AmdPPM	MANUAL	\SystemRoot\system32\DRIVERS\amdppm.sys	AMD Processor Driver	
amdsata	MANUAL	\SystemRoot\system32\drivers\amdsata.sys		
amdsbs	MANUAL	\SystemRoot\system32\DRIVERS\amdsbs.sys		
amdxata	BOOT	system32\drivers\amdxata.sys		
AppHostSvc	AUTO	%windir%\system32\svchost.exe -k apphost	@%windir%\system32\inetsrv\iisres.dll,-30012	
AppID	MANUAL	\SystemRoot\system32\drivers\appid.sys	@%systemroot%\system32\appidsvc.dll,-103	
AppIDSvc	MANUAL	%SystemRoot%\system32\svchost.exe -k Local	@%systemroot%\system32\appidsvc.dll,-101	
Appinfo	MANUAL	%SystemRoot%\system32\svchost.exe -k netsvcs	@%systemroot%\system32\appinfo.dll,-101	
AppMgmt	MANUAL	%SystemRoot%\system32\svchost.exe -k netsvcs	@appmgmts.dll,-3251	
~~	MANUTAL	CustomDoon sustom 27/11DB//EDC) sto sus		
			OK Car	ncel

#### **CS6711 SECURITY LABORATORY**



#### **RESULT:**

Thus the study of installation of Rootkit software and its variety of options were developed successfully.

#### EX. NO: 07

# WORKING WITH NET STUMBLER TO PERFORM WIRELESS AUDIT ON A ROUTER

## AIM:

To perform wireless audit on an access point or a router and decrypt WEP and WPA (Net Stumbler).

### **INTRODUCTION:**

## **NET STUMBLER:**

NetStumbler (Network Stumbler) is one of the Wi-Fi hacking tool which only compatible with windows, this tool also a freeware. With this program, we can search for wireless network which open and infiltrate the network. Its having some compatibility and network adapter issues. NetStumbler is a tool for Windows that allows you to detect Wireless Local Area Networks (WLANs) using 802.11b, 802.11a and 802.11g. It runs on Microsoft Windows operating systems from Windows 2000 to Windows XP. A trimmed-down version called MiniStumbler is available for the handheld Windows CE operating system.

It has many uses:

- ✓ Verify that your network is set up the way you intended
- ✓ Find locations with poor coverage in your WLAN.
- $\checkmark$  Detect other networks that may be causing interference on your network
- ✓ Detect unauthorized "rogue" access points in your workplace
- ✓ Help aim directional antennas for long-haul WLAN links.
- ✓ Use it recreationally for WarDriving.

## **PROCEDURE:**

**STEP-1:** Download and install Netstumbler.

- **STEP-2:** It is highly recommended that the PC should have wireless network card in order to access wireless router.
- STEP-3: Now Run Netstumbler in record mode and configure wireless card.
- **STEP-4:** There are several indicators regarding the strength of the signal, such as GREEN indicates Strong, YELLOW and other color indicates a weaker signal, RED indicates a very weak and GREY indicates a signal loss.
- STEP-5: Lock symbol with GREEN bubble indicates the Access point has encryption enabled.

**STEP-6:** MAC assigned to Wireless Access Point is displayed on right hand pane.

- **STEP-7:** The next column displays the Access points Service Set Identifier[SSID] which is useful to crack the password.
- **STEP-8:** To decrypt use WireShark tool by selecting Edit  $\rightarrow$  preferences  $\rightarrow$  IEEE 802.11.

**STEP-9:** Enter the WEP keys as a string of hexadecimal numbers as A1B2C3D4E5.

## **SCREENSHOTS:**

🕼 File Edit View Device V	Window Help								-	. 8 X	
🗅 🚅 🖬 🕨 🍕 🚭 🖞	• 15 🖽 🛲 @ ()	2 ?									
- ∰ Channels SSIDs Filters 	MAC	SSID			Name		Chan	Speed	1   I	Vendor	
Encippion On     Soft (AP)     ESS (AP)     ESS (Peer)     CF Pollable     Short Preamble     PBCC     Short Slot Time (11g)     Default SSID											
Network Stumbler - [20060108	v Help										
<ul> <li>A File Edit View Device Window</li> <li>C 2 → C + C + C + C + C + C + C + C + C + C</li></ul>	r Heb ↓ 计注册包电										- [8]
Image: Second secon	+ Help 1 1- 日前 御 句 句 MAC 000F3D 38195E	SSID default	2	1	Vendor	Type AP	Enc	SNR 18	-82	Noise- -100	-   0   -   8     SNR 18
A       File       Edit       View       Device       Window         Image: Sime state	v Heb 1 迂田苗夜风 NAC	SSID	termination and success	54 Mbps 11 Mbps	Vendor Linksys Linksys	and some the same	Enc WEP	a second second	A country of the other the	Noise	_[#]   SNR
File       Edit       View       Device       Window         Image: State of the	<ul> <li>Heb</li> <li>T= 111 111 (R, R)</li> <li>MAC</li> <li>000F30 38195E</li> <li>000F6694AE99</li> </ul>	SSID default linkaya	2	54 Mbps 11 Mbps	Linkoys	AP AP		18 19	-82 -80	Noise- -100 -100	_ [#] SNR 18 20

#### CS6711 SECURITY LABORATORY

D-44JW	* IEEE 802.11 wireless L	AN		
H.245		eassemble fragmented 802.11 datagrams:		
H.501		eassemble fragmented ov211 datagrams		
H248		Ignore vendor-specific HT elements:		
H263P	Call sub	dissector for retransmitted 802.11 frames:	V	
H264	Select	Arrows and the barry 505	1273	
HCI_ACL		Assume packets have FCS:		
Hilscher		Ignore the Protection bit:	No	Yes - w
HTTP	E	Enable decryption:	V	
I2C	13			
IEEE 802.11 IEEE 802.15.4		Key examples: 01:02:03:04:05 (40/64-bit 01:02:03:04:05:06070809101111213 (1:04/12 wpa-pwd:MyPassword]:MyAP) (WPA- wpa-psk:01:02:03:04:056061626364 (WP Key #1:	28-bit WEP), + plainteat p A + 256-bit	cey). Invalid
IEEE 802.11 IEEE 802.15.4		010203040506070809101111213 (104/12 wpa-pwd:MyPassword]:MyAP] (WPA- wpa-psk:01020304056061626364 (WP	28-bit WEP), + plainteat p A + 256-bit	
ICMP IEEE 802.15.4 IEEE 802.1AH	chable decryption:	010203040506070809101111213 (104/12 wpa-pwd:MyPassword]:MyAP] (WPA- wpa-psk:01020304056061626364 (WP	28-bit WEP), + plainteat p A + 256-bit	cey). Invalid I
IEEE 802.13 IEEE 802.15.4 IEEE 802.1AH	Key examples: 01:02:	010203040506070809101111213 (104/12 wpa-pwd:MyPassword[:MyAP] (WPA- wpa-psk:01020304056061626364 (WP Key #1:	8-bit WEP), + plaintest p A + 256-bit i wpa-pwd	cey). Invalid I
IEEE 802.15.4 IEEE 802.1AH	Key examples: 01:02:	010203040506070809101111213 (104/12 wpa-pwd:MyPassword[:MyAP] (WPA- wpa-psk:01020304056061626364 (WP Key #1: 03:04:05 (40/64-bit WEP), 101111213 (104/128-bit WEF	8-bit WEP), + plaintext p A + 256-bit wpa-pwd	cey). Invalid I

## Adding Keys: Wireless Toolbar

- If the system is having the Windows version of Wireshark and have an AirPcap adapter, then we can add decryption keys using the wireless toolbar.
- ➤ If the toolbar isn't visible, you can show it by selecting View →Wireless Toolbar.
- Click on the Decryption Keys button on the toolbar:

#### CS6711 SECURITY LABORATORY

ew Go Cepture Analyze Satisfics Telephony Iools Help M M 🗁 📅 💥 😅 🖴 🔍 🗢 🍬 🥥 🐺 🛓 🗐 🗐 🔍 Q, Q, 🖸 🛛 🖉 🧏 🎉 🛱	
✓ Expression Clear Apply	

This will open the decryption key management window. As shown in the window you can select between three decryption modes: None, Wireshark and Driver:

Wireshark	Select Decryp	tion Mode	
None	Key	SSID	New
Wireshark	etwothree	FOUR	Edit
Driver	uction		Delete
			<u>U</u> р
			Down

## **RESULT:**

Thus the wireless audit on an access point or a router and decrypt WEP and WPA (Net Stumbler) was done successfully.

#### EX. NO: 08

# <u>WORKING WITH SNORT TOOL TO DEMONSTRATE INTRUSION</u> <u>DETECTION SYSTEM</u>

## AIM:

Snort is an open source network intrusion detection system (NIDS) and it is a packet sniffer that monitors network traffic in real time.

#### **INTRODUCTION:**

#### **INTRUSION DETECTION SYSTEM :**

Intrusion detection is a set of techniques and methods that are used to detect suspicious activity both at the network and host level. Intrusion detection systems fall into two basic categories:

- ✓ Signature-based intrusion detection systems
- $\checkmark$  Anomaly detection systems.

Intruders have signatures, like computer viruses, that can be detected using software. You try to find data packets that contain any known intrusion-related signatures or anomalies related to Internet protocols. Based upon a set of signatures and rules, the detection system is able to find and log suspicious activity and generate alerts.

Anomaly-based intrusion detection usually depends on packet anomalies present in protocol header parts. In some cases these methods produce better results compared to signature-based IDS. Usually an intrusion detection system captures data from the network and applies its rules to that data or detects anomalies in it. Snort is primarily a rule-based IDS, however input plug-ins are present to detect anomalies in protocol headers.

#### **SNORT TOOL:**

Snort is based on libpcap (for library packet capture), a tool that is widely used in TCP/IPtraffic sniffers and analyzers. Through protocolanalysis and content searching and matching, Snort detects attack methods, including denial of service, buffer overflow, CGI attacks, stealthport scans, and SMB probes. When suspicious behavior is detected, Snort sends a real-time alert to syslog, a separate 'alerts' file, or to apop-up window.

Snort is currently the most popular free network intrusion detection software. The advantages of Snort are numerous. According to the snort web site, "It can perform protocol

analysis, content searching/matching, and can be used to detect a variety of attacks and probes, such as buffer overflow, stealth port scans, CGI attacks, SMB probes, OS fingerprinting attempts, and much more" (Caswell).

One of the advantages of Snort is its ease of configuration. Rules are very flexible, easily written, and easily inserted into the rule base. If a new exploit or attack is found a rule for the attack can be added to the rule base in a matter of seconds. Another advantage of snort is that it allows for raw packet data analysis.

#### SNORT can be configured to run in three modes:

- 1. Sniffer mode
- 2. Packet Logger mode
- 3. Network Intrusion Detection System mode

### 1. Sniffer mode

- ✓ Snort –v Print out the TCP/IP packets header on the screen
- ✓ Snort –vd show the TCP/IP ICMP header with application data in transmit

## 2. Packet Logger mode

- ✓ snort -dev -l c:\log [create this directory in the C drive] and snort will automatically know to go into packet logger mode, it collects every packet it sees and places it in log directory.
- ✓ snort -dev -l c:\log -h ipaddress/24:This rule tells snort that you want to print out the data link and TCP/IP headers as well as application data into the log directory. snort -l c:\log -b This is binary mode logs everything into a single file.

#### 3. Network Intrusion Detection System mode

- ✓ snort –d c:\log –h ipaddress/24 –c snort.conf This is a configuration file applies rule to each packet to decide it an action based upon the rule type in the file.
- ✓ Snort –d –h ipaddress/24 –l c:\log –c snort.conf This will enfigure snort to run in its most basic NIDS form, logging packets that trigger rules specifies in the snort.conf.

## **PROCEDURE:**

**STEP-1:** Sniffer mode  $\rightarrow$  snort  $-v \rightarrow$  Print out the TCP/IP packets header on the screen. **STEP-2:** Snort  $-vd \rightarrow$  Show the TCP/IP ICMP header with application data in transit.

- **STEP-3:** Packet Logger mode  $\rightarrow$  snort –dev –l c:\log [create this directory in the C drive] and snort will automatically know to go into packet logger mode, it collects every packet it sees and places it in log directory.
- **STEP-4:** snort -dev -l c:\log -h ipaddress/24  $\rightarrow$  This rule tells snort that you want to print out the data link and TCP/IP headers as well as application data into the log directory.
- **STEP-5:** snort -1 c:\log  $-b \rightarrow$  this binary mode logs everything into a single file.
- STEP-6: Network Intrusion Detection System mode → snort -d c:\log -h ipaddress/24 -c snort.conf → This is a configuration file that applies rule to each packet to decide it an action based upon the rule type in the file.
- STEP-7: snort -d -h ip address/24 -l c:\log -c snort.conf → This will configure snort to run in its most basic NIDS form, logging packets that trigger rules specifies in the snort.conf.

STEP-8: Download SNORT from snort.org. Install snort with or without database support.

STEP-9: Select all the components and Click Next. Install and Close.

**STEP-10:** Skip the WinPcap driver installation.

- **STEP-11:** Add the path variable in windows environment variable by selecting new classpath.
- **STEP-12:** Create a path variable and point it at snort.exe variable name  $\rightarrow$  path and variable value  $\rightarrow$  c:\snort\bin.
- **STEP-13:** Click OK button and then close all dialog boxes. Open command prompt and type the following commands:

nort 2.8.0.2 Setup		
stallation Options elect which configurat	ion options you want installed	(
	f Snort already contain support for logging to MySQL and Ol ect any additional functionality that you desire.	DBC
I do not plan to log above.	to a database, or I am planning to log to one of the databa	ses lister
I need support for lo	ogging to Microsoft SQL Server. Note that the SQL Server of	dient
	dy be installed on this computer.	
I need support for lo	ogging to Orade. Note that the Orade dient software must	talready
<sup>J</sup> be installed on this o	computer.	
oft Install System v2.	09	
	< Back Next >	Cance
	< Back Next >	Cance
	< <u>Back</u> Next >	Cance
vironment Variat		Cance
	bles	
User variables for	bles	
User variables for	r Dad Value	
User variables for	bles	
User variables for Variable CLASSPATH	bles r Dad Value ;C:\PROGRA~1\JME21~1.1\ib\soun	
User variables for Variable CLASSPATH TEMP	bles r Dad Value ,,,;C:\PROGRA~1\JMF21~1.1\ib\soun %USERPROFILE%\AppData\Local\Temp	
User variables for Variable CLASSPATH TEMP	bles r Dad Value ValueC:\PROGRA~1\UMF21~1.1\ib\soun %USERPROFILE%\AppData\Local\Temp %USERPROFILE%\AppData\Local\Temp Mew Edit Delete	
User variables for Variable CLASSPATH TEMP TMP	bles r Dad Value ValueC:\PROGRA~1\UMF21~1.1\ib\soun %USERPROFILE%\AppData\Local\Temp %USERPROFILE%\AppData\Local\Temp Mew Edit Delete	
User variables for Variable CLASSPATH TEMP TMP	ples r Dad Value C:\PROGRA~1\UMF21~1.1\ib\soun %USERPROFILE%\AppData\Local\Temp %USERPROFILE%\AppData\Local\Temp Mew Edit Delete	
User variables for Variable CLASSPATH TEMP TMP System variables Variable CLASSPATH ComSpec	bles r Dad ValueC:\PROGRA~1\JMF21~1.1\ib\soun %USERPROFILE%\AppData\Local\Temp %USERPROFILE%\AppData\Local\Temp %USERPROFILE%\AppData\Local\Temp Value .;c:\Program Files\Java\jre1.6.0\lib\ext C:\Windows\system32\cmd.exe	
User variables for Variable CLASSPATH TEMP TMP System variables Variable CLASSPATH ComSpec FP_NO_HOST_0	bles r Dad Value	
User variables for Variable CLASSPATH TEMP TMP System variables Variable CLASSPATH ComSpec	bles r Dad Value	
User variables for Variable CLASSPATH TEMP TMP System variables Variable CLASSPATH ComSpec FP_NO_HOST_0	bles r Dad Value	
User variables for Variable CLASSPATH TEMP TMP System variables Variable CLASSPATH ComSpec FP_NO_HOST_0	value Value New Edit Value	

## **CS6711 SECURITY LABORATORY**

Administrator: C:\Windows\system32\cmd.exe - snort -v	
IDP TTL:128 TOS:0x0 ID:903 IpLen:20 DgmLen:78 .en: 50	
1611 • JØ :+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=	+=+
)3/20-12:13:57.248341 192.168.56.101:63650 -> 224.0.0.252:5355  DP TTL:1 TOS:0x0 ID:904 IpLen:20 DgmLen:50 <sub>/</sub> en: 22	
∶+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=	+=+
13/20-12:13:57.348568 192.168.56.101:63650 -> 224.0.0.252:5355 1DP TTL:1 TOS:0x0 ID:905 IpLen:20 DgmLen:50 .en: 22	
**=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*=*	+=+
03/20-12:13:57.548888 192.168.56.101:137 -> 192.168.56.255:137 JDP TTL:128 TOS:0x0 ID:906 IpLen:20 DgmLen:78 Jen: 50	
;+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=	+=+
)3/20-12:13:58.298907 192.168.56.101:137 -> 192.168.56.255:137 JDP TTL:128 TOS:0x0 ID:907 IpLen:20 DgmLen:78 Jen: 50	
;+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=	+=+ =

#### **CS6711 SECURITY LABORATORY**

Administrator: C:\Windows\system32\cmd.exe   Run time for packet processing was 703.909000 seconds Snort processed 1409 packets. Snort ran for 0 days 0 hours 11 minutes 43 seconds Pkts/min: 128 Pkts/min: 128 Pkts/sec: 2 Packet I/O Totals: Received: 1411 Analyzed: 1409 < 99.858% Dropped: 0 < 0.000% Outstanding: 2 < 0.142% Injected: 0 Eth: 1409 (100.000%) ULAN: 0 < 0.000% ILAN: 0
Run time for packet processing was 703.909000 seconds         Snort processed 1409 packets.         Snort ran for 0 days 0 hours 11 minutes 43 seconds         Pkts/min:       128         Pkts/sec:       2         Packet I/O Totals:         Received:       1411         Analyzed:       1409 < 99.858x)         Dropped:       0 < 0.000x)         Filtered:       0 < 0.000x)         Outstanding:       2 < 0.142x)         Injected:       0         Breakdown by protocol <includes packets="" rebuilt="">:         Eth:       1409 (100.000x)         ULAN:       0 &lt; 0.000x)         IP4:       927 ( 65.791x)         Frag:       0 &lt; 0.000x)</includes>
Snort processed 1409 packets.         Snort ran for Ø days Ø hours 11 minutes 43 seconds         Pkts/min:       128         Pkts/sec:       2         Packet I/O Totals:         Received:       1411         Analyzed:       1409 < 99.858%)         Dropped:       0 < 0.000%)         Filtered:       0 < 0.000%)         Outstanding:       2 < 0.142%)         Breakdown by protocol (includes rebuilt packets):         Eth:       1409 (100.000%)         ULAN:       0 < 0.000%)         IP4:       927 ( 65.791%)         Frag:       0 < 0.000%)
Pkts/min:       128         Pkts/sec:       2         Packet I/O Totals:       1411         Received:       1411         Analyzed:       1409 < 99.858×>         Dropped:       0 < 0.000×>         Filtered:       0 < 0.000×>         Outstanding:       2 < 0.142×>         Injected:       0         Breakdown by protocol (includes rebuilt packets):         Eth:       1409 (100.000×)         ULAN:       0 < 0.000×>         IP4:       927 ( 65.791×)         Frag:       0 < 0.000×>
Packet I/O Totals: Received: 1411 Analyzed: 1409 (99.858%) Dropped: 0 (0.000%) Filtered: 0 (0.000%) Outstanding: 2 (0.142%) Injected: 0 Breakdown by protocol (includes rebuilt packets): Eth: 1409 (100.000%) ULAN: 0 (0.000%) IP4: 927 (65.791%) Frag: 0 (0.000%)
Received:       1411         Analyzed:       1409 < 99.858x>         Dropped:       0 < 0.000x>         Filtered:       0 < 0.000x>         Outstanding:       2 < 0.142x>         Injected:       0         Breakdown by protocol <includes packets="" rebuilt="">:         Eth:       1409 (100.000x)         ULAN:       0 &lt; 0.000x&gt;         IP4:       927 ( 65.791x)         Frag:       0 &lt; 0.000x&gt;</includes>
Analyzed:       1409 < 99.858%)
Filtered:       0 < 0.000%>         Outstanding:       2 < 0.142%>         In jected:       0         Breakdown by protocol <includes packets="" rebuilt="">:         Eth:       1409 (100.000%)         ULAN:       0 &lt; 0.000%&gt;         IP4:       927 ( 65.791%)         Frag:       0 &lt; 0.000%&gt;</includes>
Outstanding:       2 < 0.142%
Breakdown by protocol (includes rebuilt packets): Eth: 1409 (100.000%) ULAN: 0 ( 0.000%) IP4: 927 ( 65.791%) Frag: 0 ( 0.000%)
Eth: 1409 (100.000%) ULAN: 0 < 0.000%) IP4: 927 (65.791%) Frag: 0 < 0.000%)
ULAN: 0 ( 0.000%) IP4: 927 ( 65.791%) Frag: 0 ( 0.000%)
IP4: 927 (65.791%) Frag: 0 (0.000%)
UDP: 892 (63.307%) TCP: 0 (0.000%)
IP6: 473 ( 33.570%)
IP6 Ext: 0 ( 0.000%) IP6 Opts: 0 ( 0.000%)
Frag6: 0 < 0.000%) ICMP6: 0 < 0.000%)
UDP6: 0 < 0_000%>
TCP6: 0 < 0.000x> Teredo: 0 < 0.000x>
ICMP-IP: 0 < 0.000x> EAPOL: 0 < 0.000x>
IP4/IP4: 0 < 0.000x>
IP4/IP6: 0 ( 0.000%) IP6/IP4: 0 ( 0.000%)
IP6/IP6: 0 < 0.000%>
GRE: 0 < 0.000%) GRE Eth: 0 < 0.000%)
GRE ULAN: 0 < 0_000%> GRE IP4: 0 < 0_000%>
GRE IP6: 0 < 0.000%>
GRE IP6 Ext: 0 < 0.000%) GRE PPTP: 0 < 0.000%)
GRE ARP: 0 < 0.000%>
GRE IPX: 0 ( 0.000%) GRE Loop: 0 ( 0.000%)
MPLŠ: 0 ( 0.000%) ARP: 9 ( 0.639%)
IPX: 0 ( 0_000%)
Eth Loop: 0 < 0.000%> Eth Disc: 0 < 0.000%>
IP4 Disc: 0 < 0.000x>
IP6 Disc: 0 ( 0.000%) TCP Disc: 0 ( 0.000%)
UDP Disc: 0 ( 0.000%) ICMP Disc: 0 ( 0.000%)
All Discard: 0 < 0.000%)
Other: 35 (2.484%) Bad Chk Sum: 0 ( 0.000%)
Bad TTL: 0 < 0.000%)
85 G 2: 0 < 0.000x>
Total: 1409
Snort exiting
C:\Snort\bin>

## **RESULT:**

Thus the demonstration of the instruction detection using Snort tool was done successfully.