Vellalar College of Engineering & Technology, Erode

(An Autonomous Institution, Affiliated to Anna University, Chennai)

Department of Computer Science and Engineering

LABORATORY MANUAL

Regulation	**:**	**2018**
Branch	**:**	**B.E – CSE**
Year / Semester	**:**	**IV/ VII**

18CSL71 Network 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

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

- Java or equivalent compiler GnuPG
- KF Sensor
- Snort
- Net Stumbler
-

HARDWARE REQUIREMENTS

- Standalone desktops (or) Server supporting 30 terminals or more

INDEX

EX. NO: 1(A)

IMPLEMENTATION OF CAESAR CIPHER

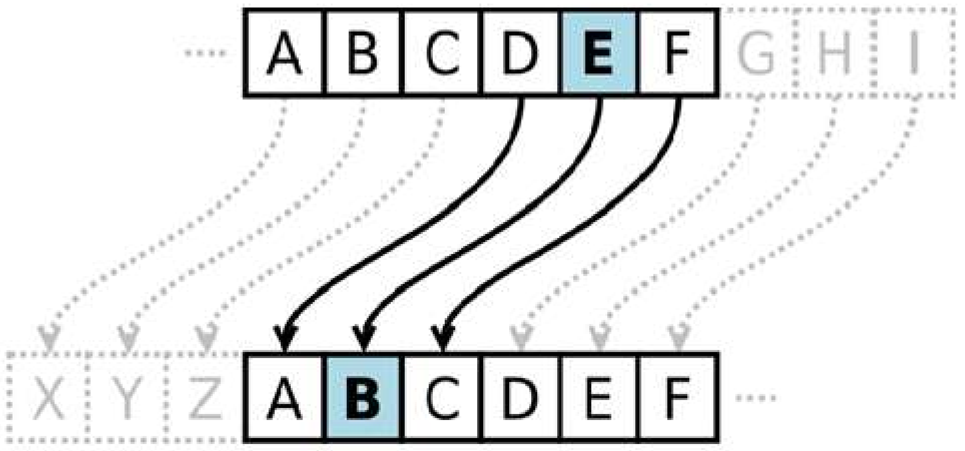
AIM:

To implement the simple substitution technique named Caesar cipher using java 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 bec omes 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.

<u>PROGRAM:</u> (Caesar Cipher)

```
//A Java Program to illustrate Caesar Cipher Technique
class CaesarCipher
{
   // Encrypts text using a shift od s
   public static StringBuffer encrypt(String text, int s)
   {
      StringBuffer result= new StringBuffer();

      for (int i=0; i<text.length(); i++)
      {
         if (Character.isUpperCase(text.charAt(i)))
         {
            char ch = (char)(((int)text.charAt(i) +
                              s - 65) % 26 + 65);
            result.append(ch);
         }
         else
         {
            char ch = (char)(((int)text.charAt(i) +
                              s - 97) % 26 + 97);
            result.append(ch);
         }
      }
      return result;
   }

   // Driver code
   public static void main(String[] args)
   {
      String text = "ATTACKATONCE";
      int s = 4;
      System.out.println("Text  : " + text);
      System.out.println("Shift : " + s);
      System.out.println("Cipher: " + encrypt(text, s));
   }
}
```

OUTPUT:

```
Text  : ATTACKATONCE
Shift : 4
Cipher: EXXEGOEXSRGI
```

RESULT:

Thus the implementation of Caesar cipher had been executed successfully.

EX. NO: 1(B)

IMPLEMENTATION OF PLAYFAIR CIPHER

AIM:

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

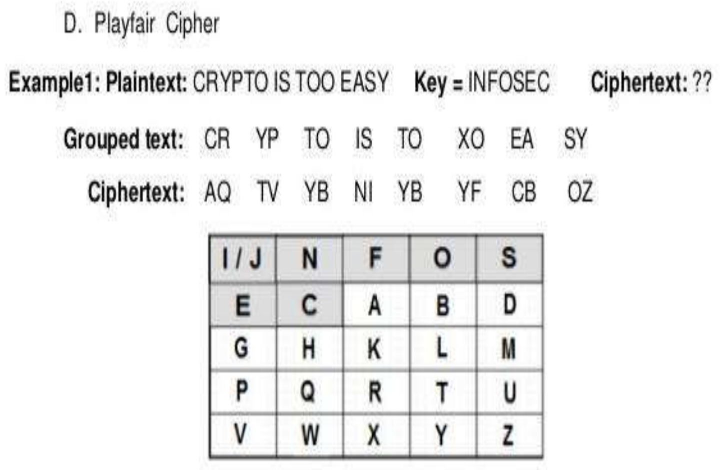
DESCRIPTION:

The Playfair cipher starts with creating a key table. The key table is a 5×5 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

I / J	N	F	O	S
E	C	A	B	D
G	H	K	L	M
P	Q	R	T	U
V	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)

```
// Java Program to Enode a Message Using Playfair Cipher

import java.io.*;
import java.util.*;

class Playfair {
    String key;
    String plainText;
    char[][] matrix = new char[5][5];

    public Playfair(String key, String plainText)
    {
        // convert all the characters to lowercase
        this.key = key.toLowerCase();

        this.plainText = plainText.toLowerCase();
    }

    // function to remove duplicate characters from the key
    public void cleanPlayFairKey()
    {
        LinkedHashSet<Character> set
            = new LinkedHashSet<Character>();

        String newKey = "";

        for (int i = 0; i < key.length(); i++)
            set.add(key.charAt(i));

        Iterator<Character> it = set.iterator();

        while (it.hasNext())
            newKey += (Character)it.next();
        key = newKey;
    }

    // function to generate playfair cipher key table
    public void generateCipherKey()
    {
        Set<Character> set = new HashSet<Character>();

        for (int i = 0; i < key.length(); i++)
        {
            if (key.charAt(i) == 'j')
                continue;
            set.add(key.charAt(i));
        }

        // remove repeated characters from the cipher key
        String tempKey = new String(key);

        for (int i = 0; i < 26; i++)
        {
            char ch = (char)(i + 97);
            if (ch == 'j')
                continue;

            if (!set.contains(ch))
                tempKey += ch;
        }

        // create cipher key table
        for (int i = 0, idx = 0; i < 5; i++)
            for (int j = 0; j < 5; j++)
                matrix[i][j] = tempKey.charAt(idx++);

        System.out.println("Playfair Cipher Key Matrix:");

        for (int i = 0; i < 5; i++)
            System.out.println(Arrays.toString(matrix[i]));
    }

    // function to preprocess plaintext
    public String formatPlainText()
    {
        String message = "";
        int len = plainText.length();

        for (int i = 0; i < len; i++)
        {
            // if plaintext contains the character 'j',
            // replace it with 'i'
            if (plainText.charAt(i) == 'j')
                message += 'i';
            else
            message += plainText.charAt(i);
        }

        // if two consecutive characters are same, then
        // insert character 'x' in between them
        for (int i = 0; i < message.length(); i += 2)
        {
            if (message.charAt(i) == message.charAt(i + 1))
                message = message.substring(0, i + 1) + 'x'
                          + message.substring(i + 1);
        }

        // make the plaintext of even length
        if (len % 2 == 1)
            message += 'x'; // dummy character

        return message;
    }

    // function to group every two characters
    public String[] formPairs(String message)
    {
        int len = message.length();
        String[] pairs = new String[len / 2];

        for (int i = 0, cnt = 0; i < len / 2; i++)
            pairs[i] = message.substring(cnt, cnt += 2);

        return pairs;
    }

    // function to get position of character in key table
    public int[] getCharPos(char ch)
    {
        int[] keyPos = new int[2];

        for (int i = 0; i < 5; i++)
        {
            for (int j = 0; j < 5; j++)
            {

                if (matrix[i][j] == ch)
                {
                    keyPos[0] = i;
                    keyPos[1] = j;
                    break;
                }
            }
        }
        return keyPos;
    }
    public String encryptMessage()
    {
        String message = formatPlainText();
        String[] msgPairs = formPairs(message);
        String encText = "";

        for (int i = 0; i < msgPairs.length; i++)
        {
            char ch1 = msgPairs[i].charAt(0);
            char ch2 = msgPairs[i].charAt(1);
            int[] ch1Pos = getCharPos(ch1);
            int[] ch2Pos = getCharPos(ch2);

            // if both the characters are in the same row
            if (ch1Pos[0] == ch2Pos[0]) {
                ch1Pos[1] = (ch1Pos[1] + 1) % 5;
                ch2Pos[1] = (ch2Pos[1] + 1) % 5;
            }

            // if both the characters are in the same column
            else if (ch1Pos[1] == ch2Pos[1])
            {
                ch1Pos[0] = (ch1Pos[0] + 1) % 5;
                ch2Pos[0] = (ch2Pos[0] + 1) % 5;
            }

            // if both the characters are in different rows
            // and columns
            else {
                int temp = ch1Pos[1];
                ch1Pos[1] = ch2Pos[1];
                ch2Pos[1] = temp;
            }

            // get the corresponding cipher characters from
            // the key matrix
            encText = encText + matrix[ch1Pos[0]][ch1Pos[1]]
                      + matrix[ch2Pos[0]][ch2Pos[1]];
        }

        return encText;
    }
}

public class GFG {
    public static void main(String[] args)
    {
        System.out.println("Example-1\n");

        String key1 = "Problem";
        String plainText1 = "Playfair";
        System.out.println("Key: " + key1);
        System.out.println("PlainText: " + plainText1);

        Playfair pfc1 = new Playfair(key1, plainText1);
        pfc1.cleanPlayFairKey();
        pfc1.generateCipherKey();

        String encText1 = pfc1.encryptMessage();
        System.out.println("Cipher Text is: " + encText1);

        System.out.println("\nExample-2\n");

        String key2 = "Problem";
        String plainText2 = "Hello";

        System.out.println("Key: " + key2);
        System.out.println("PlainText: " + plainText2);

        Playfair pfc2 = new Playfair(key2, plainText2);
        pfc2.cleanPlayFairKey();
        pfc2.generateCipherKey();

        String encText2 = pfc2.encryptMessage();
        System.out.println("Cipher Text is: " + encText2);
    }
}
    } }
```

OUTPUT:

```
Example-1

Key: Problem
PlainText: Playfair
Playfair Cipher Key Matrix:
[p, r, o, b, l]
[e, m, a, c, d]
[f, g, h, i, k]
[n, q, s, t, u]
[v, w, x, y, z]
Cipher Text is: rpcxhegb

Example-2

Key: Problem
PlainText: Hello
Playfair Cipher Key Matrix:
[p, r, o, b, l]
[e, m, a, c, d]
[f, g, h, i, k]
[n, q, s, t, u]
[v, w, x, y, z]
Cipher Text is: faozpb
```

RESULT:

Thus the Playfair cipher substitution technique had been implemented successfully.

EX. NO: 1(C)

IMPLEMENTATION OF HILL CIPHER

AIM:

To write a java 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* × *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* × *n* matrices (modulo 26).

EXAMPLE:

$$\begin{bmatrix} 2 & 4 & 5 \\ 9 & 2 & 1 \\ 3 & 17 & 7 \end{bmatrix} \begin{bmatrix} 0 \\ 19 \\ 19 \end{bmatrix} = \begin{bmatrix} 171 \\ 57 \\ 456 \end{bmatrix} \pmod{26} = \begin{bmatrix} 15 \\ 5 \\ 14 \end{bmatrix} = \text{'PFO'}$$

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.

PROGRAM: (Hill Cipher)

```
// Java code to implement Hill Cipher
public class GFG
{

// Following function generates the
// key matrix for the key string
static void getKeyMatrix(String key, int keyMatrix[][])
{
   int k = 0;
   for (int i = 0; i < 3; i++)
   {
      for (int j = 0; j < 3; j++)
      {
```

```
                keyMatrix[i][j] = (key.charAt(k)) % 65;
                k++;
            }
        }
    }

    // Following function encrypts the message
    static void encrypt(int cipherMatrix[][],
                int keyMatrix[][],
                int messageVector[][])
    {
        int x, i, j;
        for (i = 0; i < 3; i++)
        {
            for (j = 0; j < 1; j++)
            {
                cipherMatrix[i][j] = 0;

                for (x = 0; x < 3; x++)
                {
                    cipherMatrix[i][j] +=
                        keyMatrix[i][x] * messageVector[x][j];
                }

                cipherMatrix[i][j] = cipherMatrix[i][j] % 26;
            }
        }
    }

    // Function to implement Hill Cipher
    static void HillCipher(String message, String key)
    {
        // Get key matrix from the key string
        int [][]keyMatrix = new int[3][3];
        getKeyMatrix(key, keyMatrix);

        int [][]messageVector = new int[3][1];

        // Generate vector for the message
        for (int i = 0; i < 3; i++)
            messageVector[i][0] = (message.charAt(i)) % 65;

        int [][]cipherMatrix = new int[3][1];

        // Following function generates
        // the encrypted vector
        encrypt(cipherMatrix, keyMatrix, messageVector);

        String CipherText="";

        // Generate the encrypted text from
        // the encrypted vector
        for (int i = 0; i < 3; i++)
            CipherText += (char)(cipherMatrix[i][0] + 65);

        // Finally print the ciphertext
        System.out.print(" Ciphertext:" + CipherText);
    }

    // Driver code
    public static void main(String[] args)
    {
        // Get the message to be encrypted
        String message = "ACT";

        // Get the key
        String key = "GYBNQKURP";

        HillCipher(message, key);
        }
    }
```

OUTPUT:

```
Ciphertext:POH
```

RESULT:

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

EX. NO: 2(A)

IMPLEMENTATION OF DES

AIM:

To write a java program to implement Data Encryption Standard (DES).

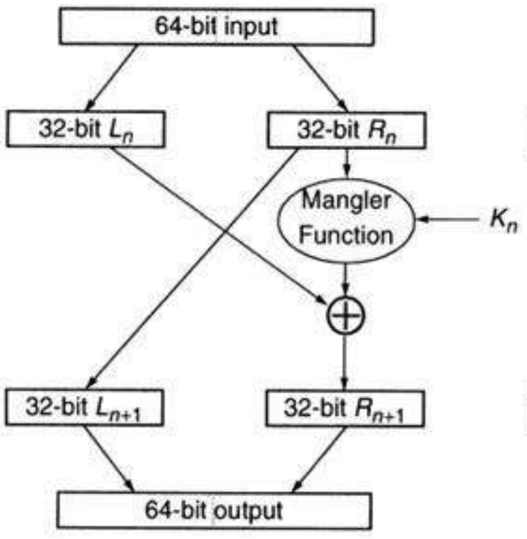
DESCRIPTION:

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.

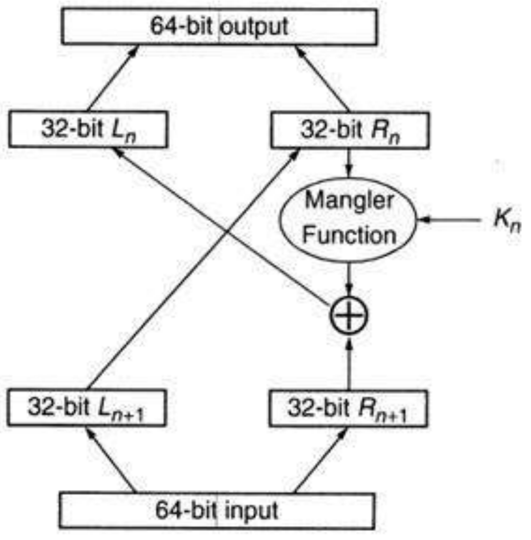
The main parts of the algorithm are as follows:

- 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:



Encryption



Decryption

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.

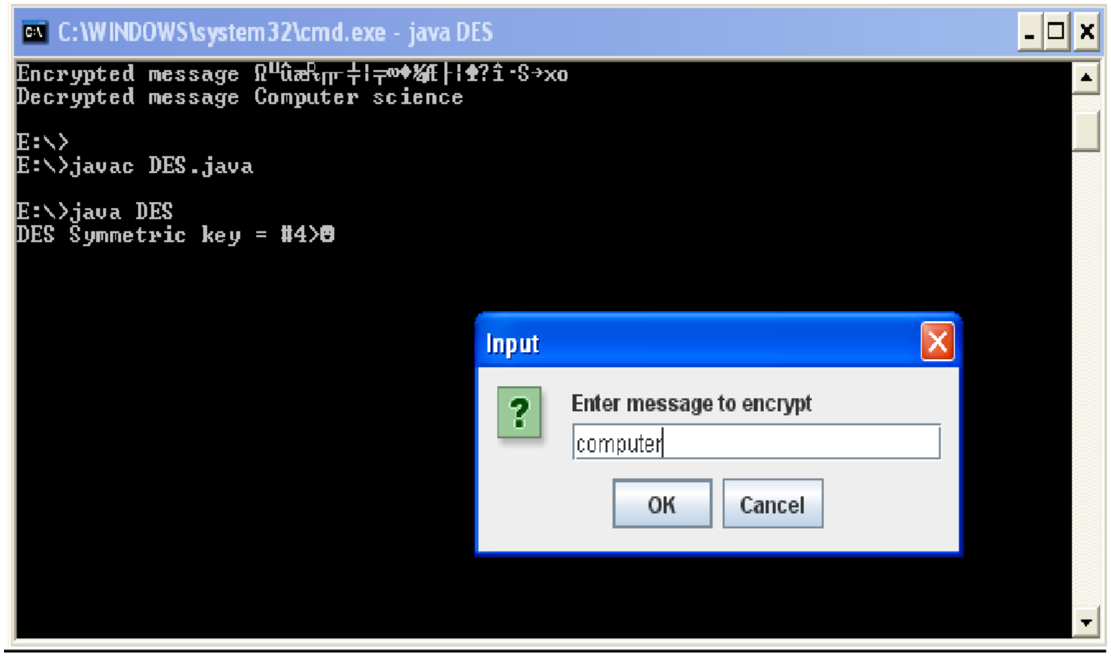
STEP-5: Thus the encrypted 64-bit cipher text is obtained in this way. Repeat the same process for the remaining plain text characters.

PROGRAM:

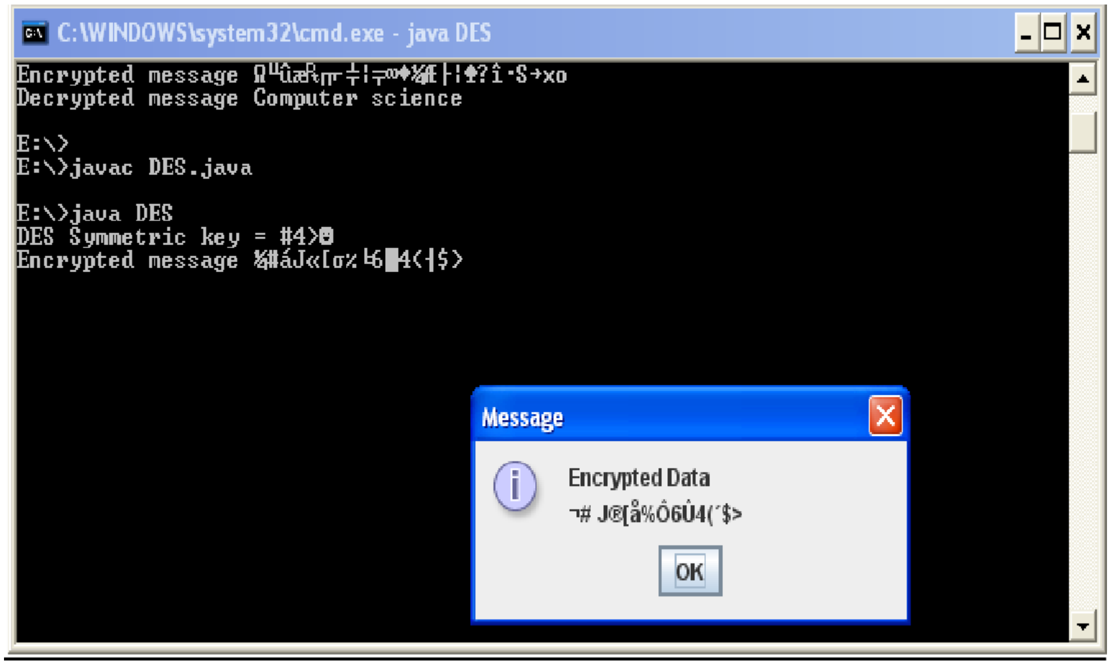
DES.java

```
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 ;
public 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
{
     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();
}
}
```

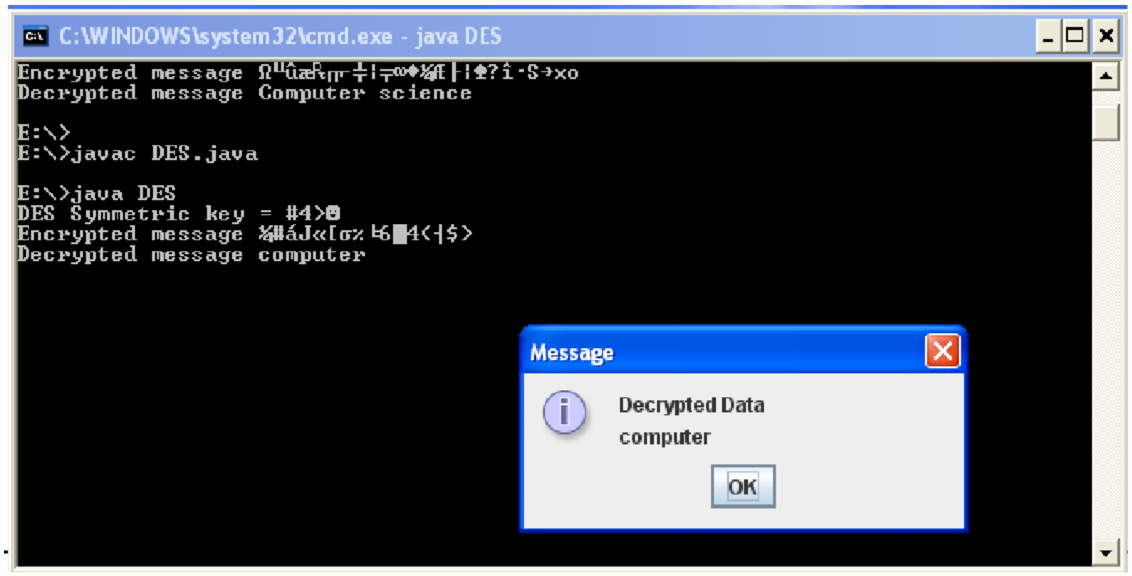
OUTPUT:



```
Encrypted message Ω╙ûæ₧╓╒╪¦╤∞♦¼£├¦♠?î·S→xo
Decrypted message Computer science

E:\>
E:\>javac DES.java

E:\>java DES
DES Symmetric key = #4>☻
```



```
Encrypted message Ω╙ûæ₧╓╒╪¦╤∞♦¼£├¦♠?î·S→xo
Decrypted message Computer science

E:\>
E:\>javac DES.java

E:\>java DES
DES Symmetric key = #4>☻
Encrypted message ¼#áJ«[σ%╘6█4(┤$>
```



```
Encrypted message Ω╙ûæ₧╓╓╪¦╤∞♦¼£├¦♠?î-S→xo
Decrypted message Computer science

E:\>
E:\>javac DES.java

E:\>java DES
DES Symmetric key = #4>☻
Encrypted message ¾#áJ«[σ%╘6█4<┤$>
Decrypted message computer
```

Decrypted Data
computer

RESULT:

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

EX. NO: 2(B) **IMPLEMENTATION OF RSA**

AIM:

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

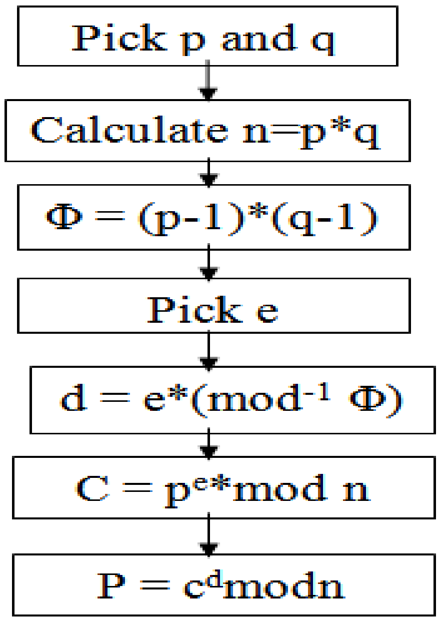
DESCRIPTION:

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:

$$(m^e)^d = m \pmod n$$

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^{d}mod n.

PROGRAM: (RSA)

```
// Java Program to Implement the RSA Algorithm
import java.math.*;
import java.util.*;

public class RSA {
    public static void main(String args[])
    {
        int p, q, n, z, d = 0, e, i;

        // The number to be encrypted and decrypted
        int msg = 12;
        double c;
        BigInteger msgback;

        // 1st prime number p
        p = 3;

        // 2nd prime number q
        q = 11;
        n = p * q;
        z = (p - 1) * (q - 1);
        System.out.println("the value of z = " + z);

        for (e = 2; e < z; e++) {
            // e is for public key exponent
            if (gcd(e, z) == 1) {
                break;
            }
        }
        System.out.println("the value of e = " + e);
        for (i = 0; i <= 9; i++) {
            int x = 1 + (i * z);

            // d is for private key exponent
            if (x % e == 0) {
                d = x / e;
                break;
            }
        }
        System.out.println("the value of d = " + d);
        c = (Math.pow(msg, e)) % n;
        System.out.println("Encrypted message is : " + c);

        // converting int value of n to BigInteger
        BigInteger N = BigInteger.valueOf(n);

        // converting float value of c to BigInteger
        BigInteger C = BigDecimal.valueOf(c).toBigInteger();
        msgback = (C.pow(d)).mod(N);
        System.out.println("Decrypted message is : "
                           + msgback);
    }

    static int gcd(int e, int z)
    {
        if (e == 0)
            return z;
        else
            return gcd(z % e, e);
    }
```

```
}
```

OUTPUT:

```
the value of z = 20
the value of e = 3
the value of d = 7
Encrypted message is : 12.0
Decrypted message is : 12
```

RESULT:

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

EX. NO: 2(C)

IMPLEMENTATION OF DIFFIE HELLMAN KEY EXCHANGE ALGORITHM

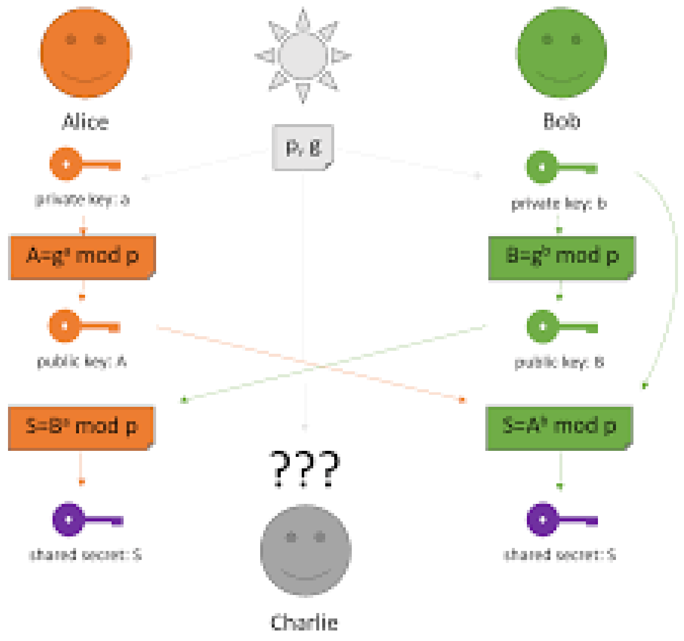
AIM:

To implement the Diffie-Hellman Key Exchange algorithm using java 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^a 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)

```
// This program calculates the Key for two persons
// using the Diffie-Hellman Key exchange algorithm
public class GFG{

// Power function to return value of a ^ b mod P
private static long power(long a, long b, long p)
{
   if (b == 1)
      return a;
   else
      return (((long)Math.pow(a, b)) % p);
}

// Driver code
public static void main(String[] args)
{
   long P, G, x, a, y, b, ka, kb;

   // Both the persons will be agreed upon the
   // public keys G and P

   // A prime number P is taken
   P = 23;
   System.out.println("The value of P:" + P);

   // A primitve root for P, G is taken
   G = 9;
   System.out.println("The value of G:" + G);
```

```
        // Alice will choose the private key a
        // a is the chosen private key
        a = 4;
        System.out.println("The private key a for Alice:" + a);

        // Gets the generated key
        x = power(G, a, P);

        // Bob will choose the private key b
        // b is the chosen private key
        b = 3;
        System.out.println("The private key b for Bob:" + b);

        // Gets the generated key
        y = power(G, b, P);

        // Generating the secret key after the exchange
        // of keys
        ka = power(y, a, P); // Secret key for Alice
        kb = power(x, b, P); // Secret key for Bob

        System.out.println("Secret key for the Alice is:" + ka);
        System.out.println("Secret key for the Bob is:" + kb);
    }
    }
```

OUTPUT:

```
The value of P:23
The value of G:9
The private key a for Alice:4
The private key b for Bob:3
Secret key for the Alice is:9
Secret key for the Bob is:9
```

RESULT:

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

EX. NO: 2(D)

IMPLEMENTATION OF MD5

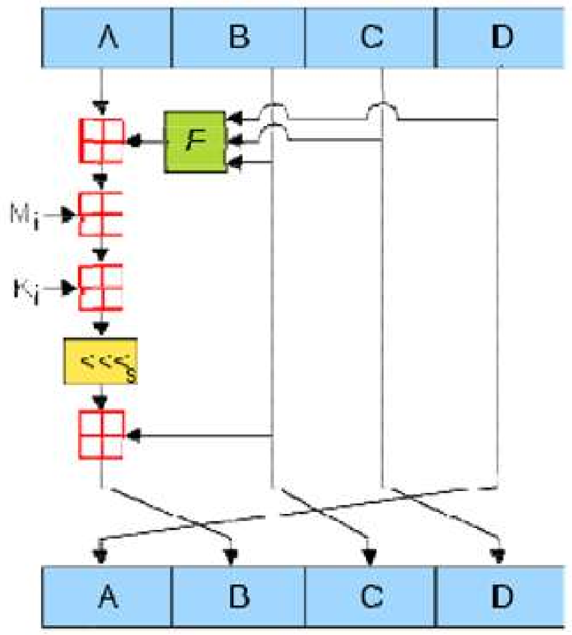
AIM:

To write a java 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^{64}.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)

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

// Java program to calculate MD5 hash value
public class MD5 {
   public static String getMd5(String input)
   {
      try {

         // Static getInstance method is called with hashing MD5
         MessageDigest md = MessageDigest.getInstance("MD5");

         // digest() method is called to calculate message digest
         //  of an input digest() return array of byte
         byte[] messageDigest = md.digest(input.getBytes());

         // Convert byte array into signum representation
         BigInteger no = new BigInteger(1, messageDigest);

         // Convert message digest into hex value
         String hashtext = no.toString(16);
         while (hashtext.length() < 32) {
            hashtext = "0" + hashtext;
         }
         return hashtext;
      }
```

```
        // For specifying wrong message digest algorithms
        catch (NoSuchAlgorithmException e) {
            throw new RuntimeException(e);
        }
    }

    // Driver code
    public static void main(String args[]) throws NoSuchAlgorithmException
    {
        String s = "welcome to java";
        System.out.println("Your HashCode Generated by MD5 is: " + getMd5(s));
    }
}
```

OUTPUT:

```
Your HashCode Generated by MD5 is: ef7bc0d9fd0ff2489218753e8216f8ce
```

RESULT:

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

EX. NO: 2(E)

IMPLEMENTATION OF SHA-I

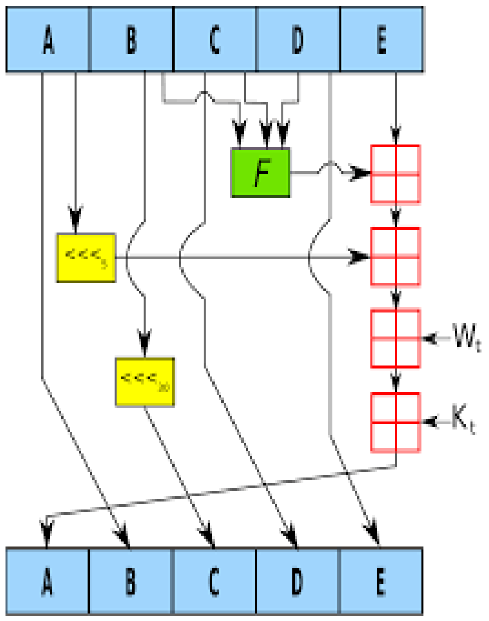
AIM:

To implement the SHA-I hashing technique using java 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.

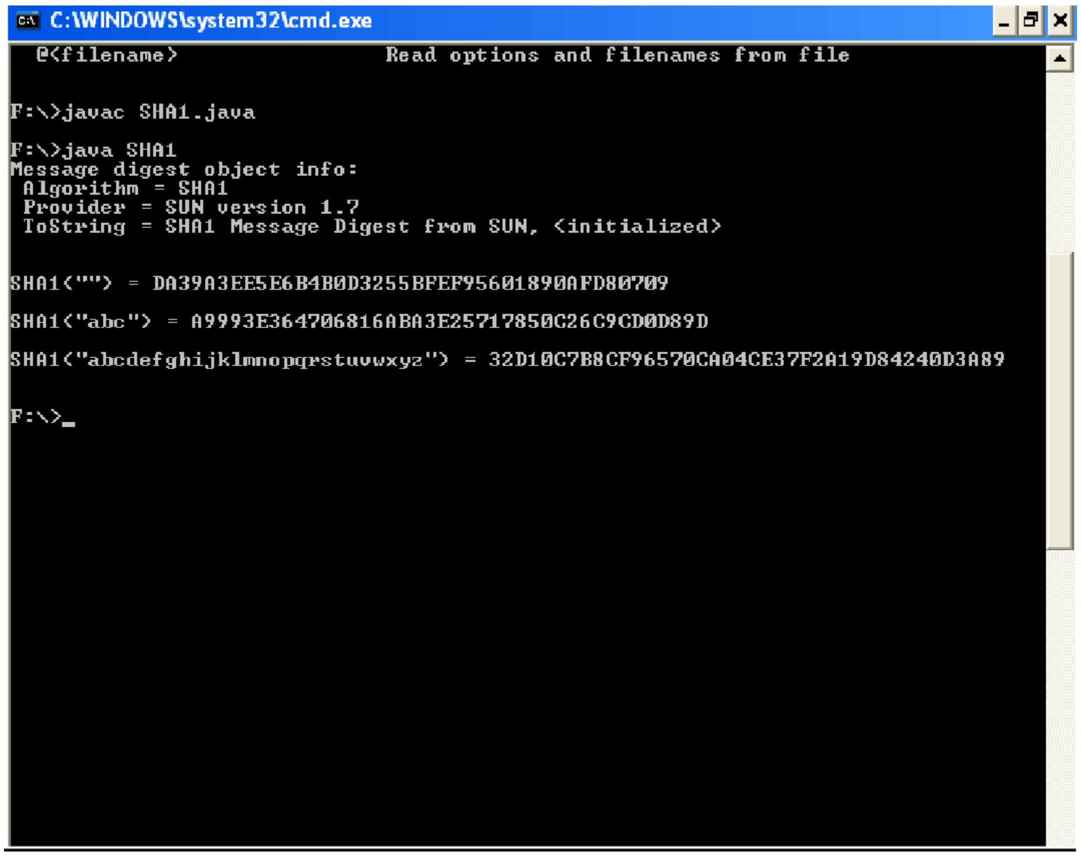
STEP-7: Block A is taken as the second block and the block B is shifted by 's' times and taken as the third block.

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

OUTPUT:



```
  @<filename>                  Read options and filenames from file

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>

SHA1("") = DA39A3EE5E6B4B0D3255BFEF95601890AFD80709

SHA1("abc") = A9993E364706816ABA3E25717850C26C9CD0D89D

SHA1("abcdefghijklmnopqrstuvwxyz") = 32D10C7B8CF96570CA04CE37F2A19D84240D3A89

F:\>_
```

RESULT:

Thus the SHA-1 hashing technique had been implemented successfully.

EX. NO: 3

IMPLEMENTATION OF DIGITAL SIGNATURE STANDARD

AIM:

To write a Java 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.

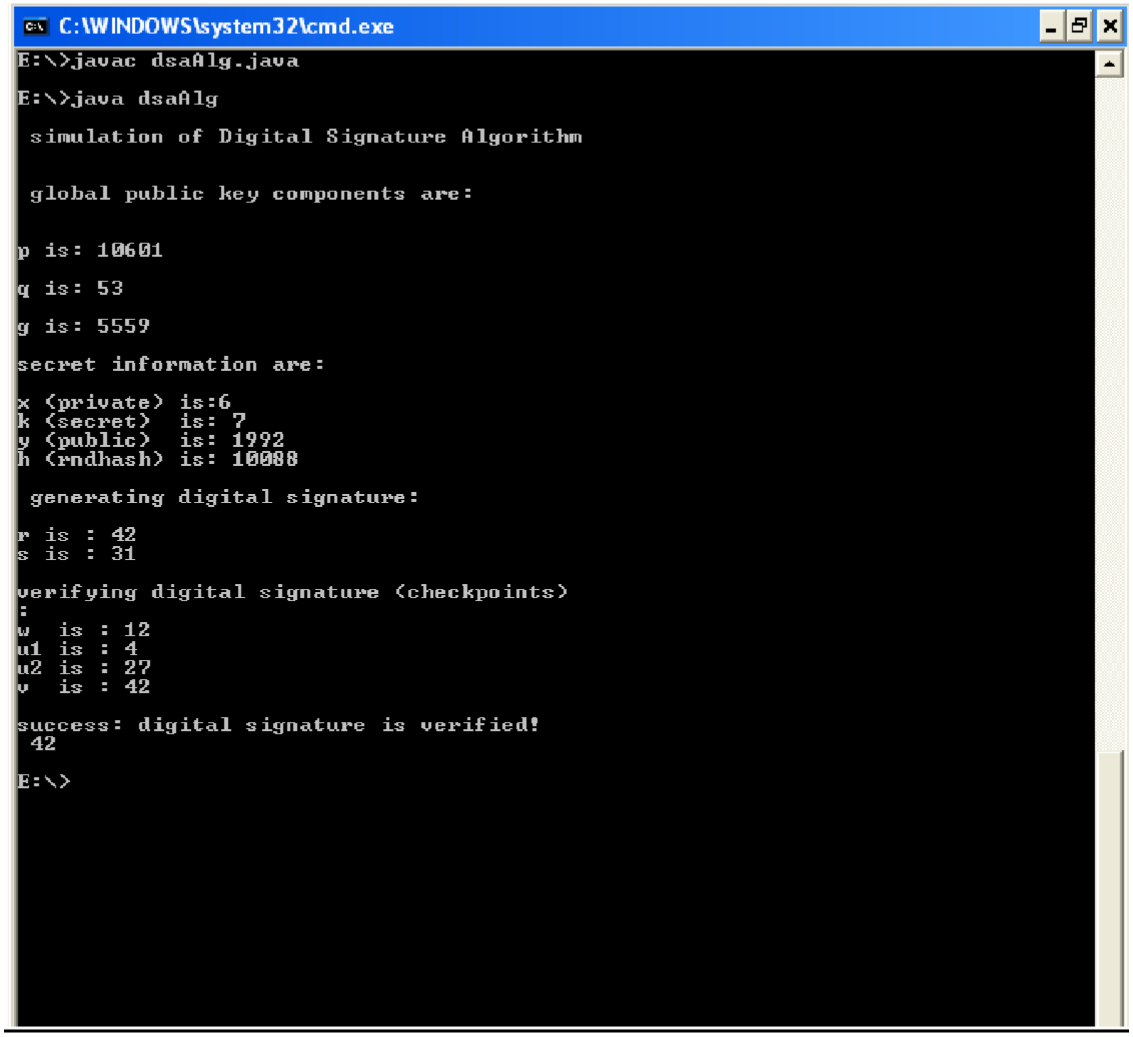
PROGRAM: (Digital Signature Standard)

```
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 g = getGen(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 = (g.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));
          v = (v.mod(p)).mod(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

```
E:\>javac dsaAlg.java

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:

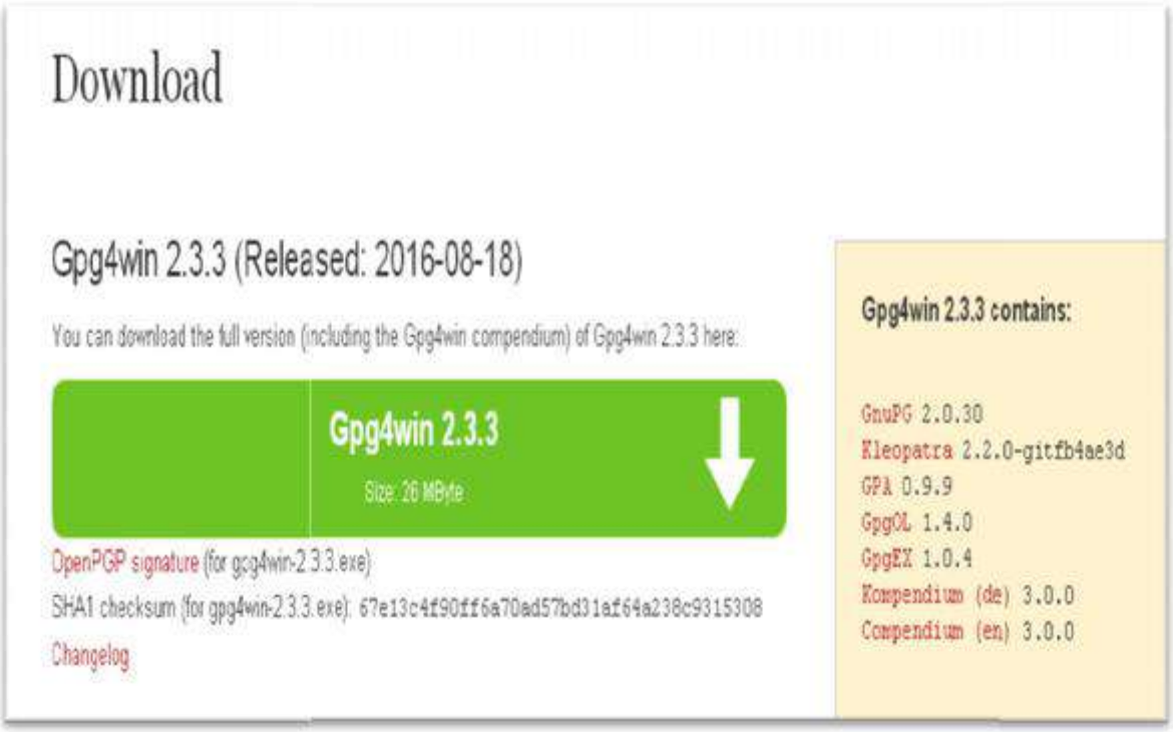
Demonstrate how to provide secure data storage, secure data transmission and for creating digital signatures (GnuPG).

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 www.gpg4win.org. Click on the "Gpg4win 2.3.0" button



Download

Gpg4win 2.3.3 (Released: 2016-08-18)

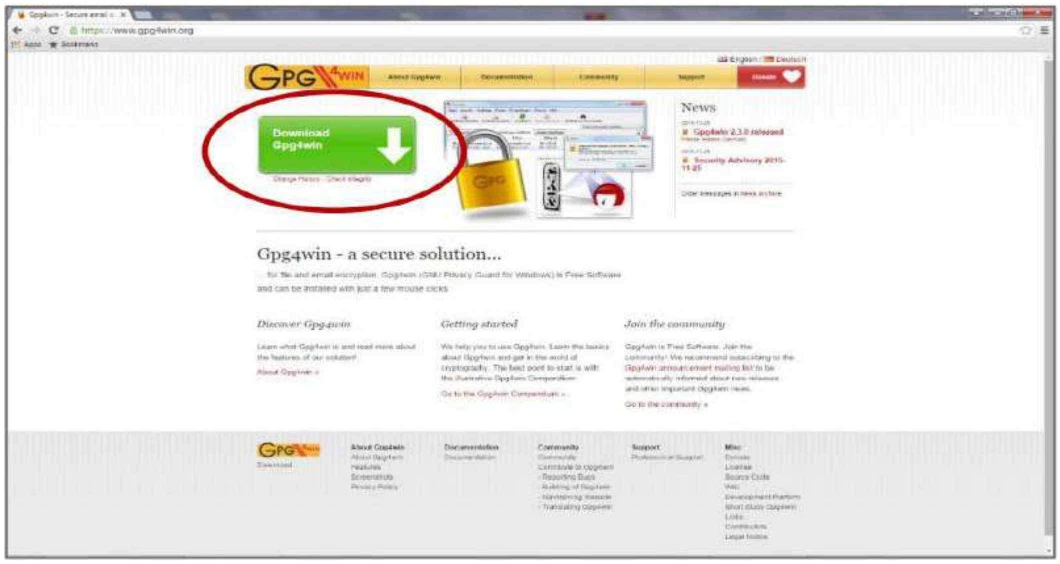
You can download the full version (including the Gpg4win compendium) of Gpg4win 2.3.3 here:

OpenPGP signature (for gpg4win-2.3.3.exe)
SHA1 checksum (for gpg4win-2.3.3.exe): 67e13c4f90ff6a70ad57bd31af64a238c9315308
Changelog

Gpg4win 2.3.3 contains:

GnuPG 2.0.30
Kleopatra 2.2.0-gitfb4ae3d
GPA 0.9.9
GpgOL 1.4.0
GpgEX 1.0.4
Kompendium (de) 3.0.0
Compendium (en) 3.0.0

2. On the following screen, click the “Download Gpg4win” button.



Gpg4win - a secure solution...

... for file and email encryption. Gpg4win (GNU Privacy Guard for Windows) is Free Software and can be installed with just a few mouse clicks.

Discover Gpg4win

Learn what Gpg4win is and read more about the features of our solution!

About Gpg4win »

Getting started

We help you to use Gpg4win. Learn the basics about Gpg4win and get in the world of cryptography. The best point to start is with the illustrative Gpg4win Compendium.

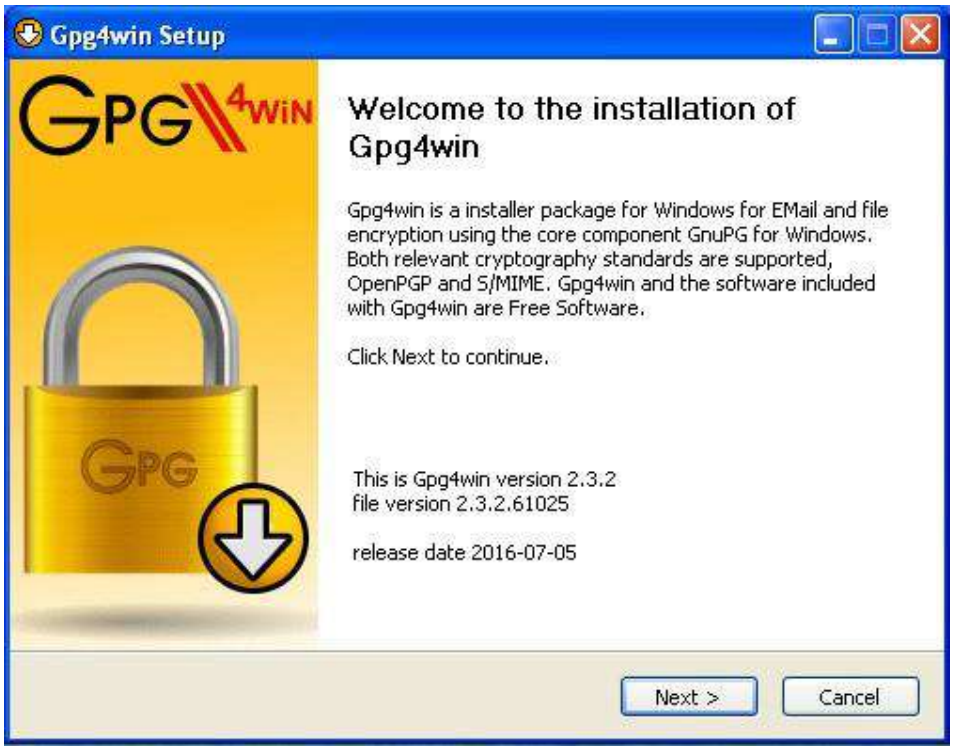
Go to the Gpg4win Compendium »

Join the community

Gpg4win is Free Software. Join the community! We recommend subscribing to the Gpg4win announcement mailing list to be automatically informed about new releases and other important Gpg4win news.

Go to the community »

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



Welcome to the installation of Gpg4win

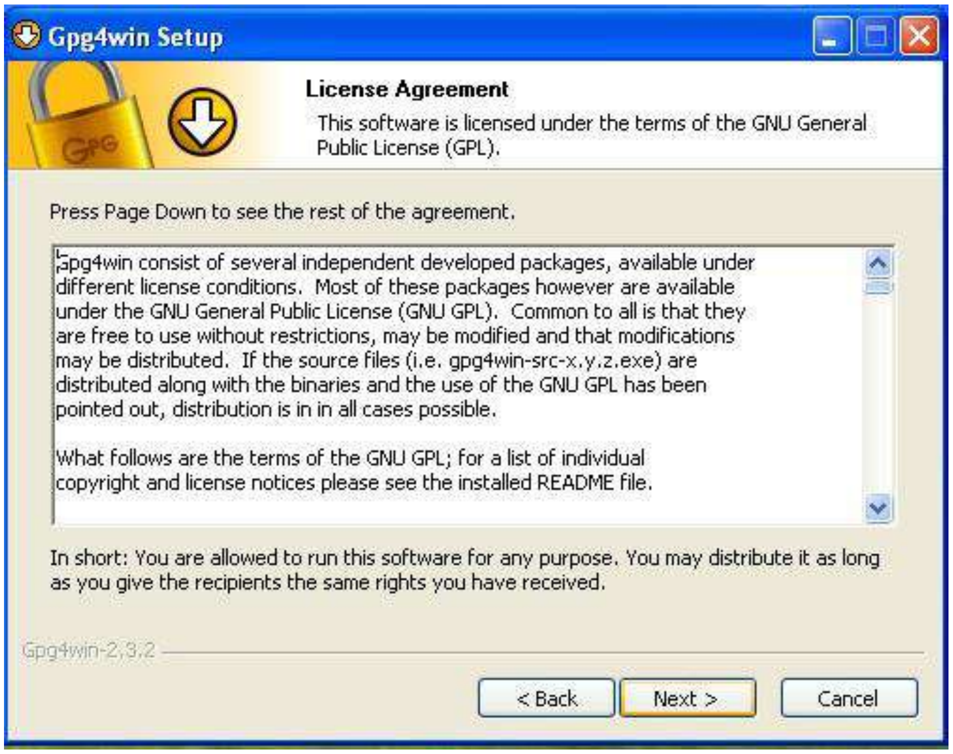
Gpg4win is a installer package for Windows for EMail and file encryption using the core component GnuPG for Windows. Both relevant cryptography standards are supported, OpenPGP and S/MIME. Gpg4win and the software included with Gpg4win are Free Software.

Click Next to continue.

This is Gpg4win version 2.3.2
file version 2.3.2.61025

release date 2016-07-05

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



Gpg4win Setup

License Agreement

This software is licensed under the terms of the GNU General Public License (GPL).

Press Page Down to see the rest of the agreement.

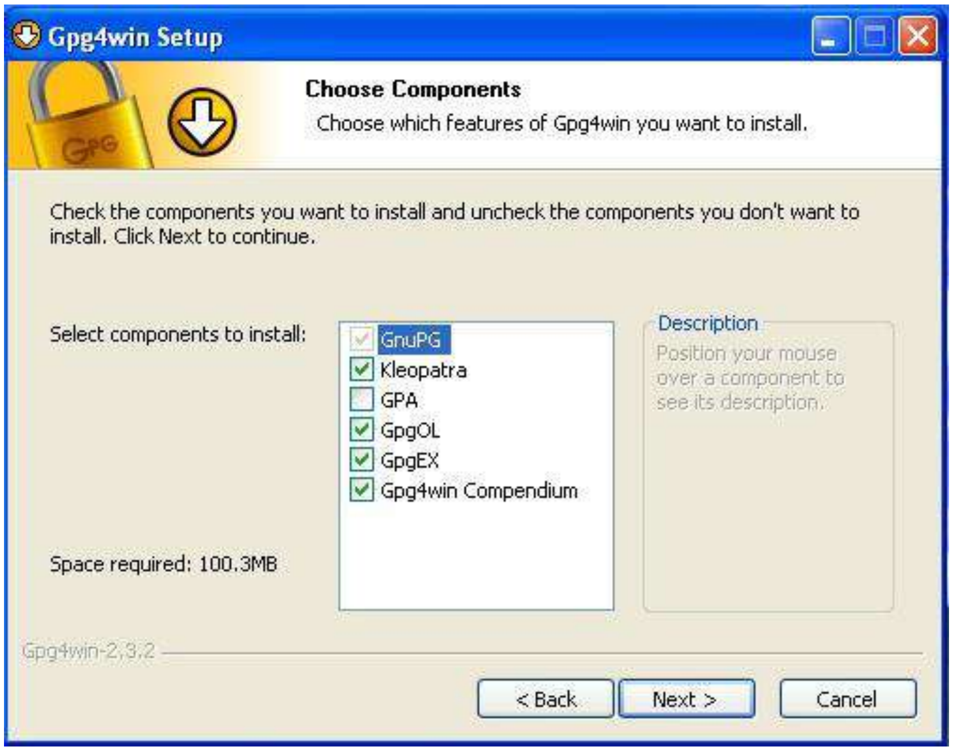
Gpg4win consist of several independent developed packages, available under different license conditions. Most of these packages however are available under the GNU General Public License (GNU GPL). Common to all is that they are free to use without restrictions, may be modified and that modifications may be distributed. If the source files (i.e. gpg4win-src-x.y.z.exe) are distributed along with the binaries and the use of the GNU GPL has been pointed out, distribution is in in all cases possible.

What follows are the terms of the GNU GPL; for a list of individual copyright and license notices please see the installed README file.

In short: You are allowed to run this software for any purpose. You may distribute it as long as you give the recipients the same rights you have received.

Gpg4win-2.3.2

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



Gpg4win Setup

Choose Components

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.

Select components to install:

- ✓ GnuPG
- ✓ Kleopatra
- ☐ GPA
- ✓ GpgOL
- ✓ GpgEX
- ✓ Gpg4win Compendium

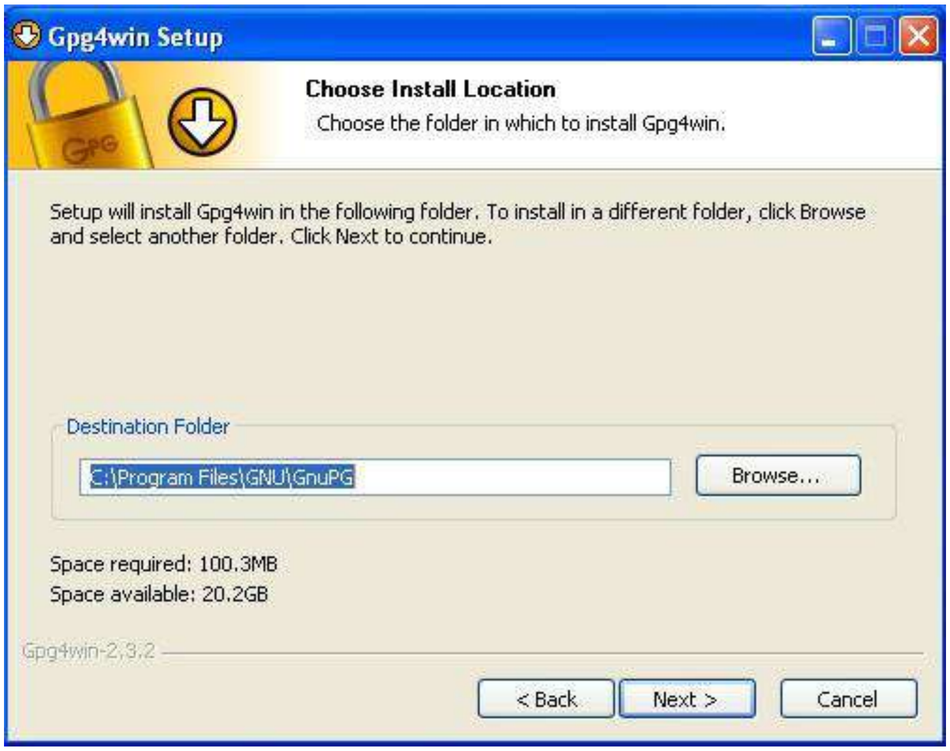
Description

Position your mouse over a component to see its description.

Space required: 100.3MB

Gpg4win-2.3.2

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

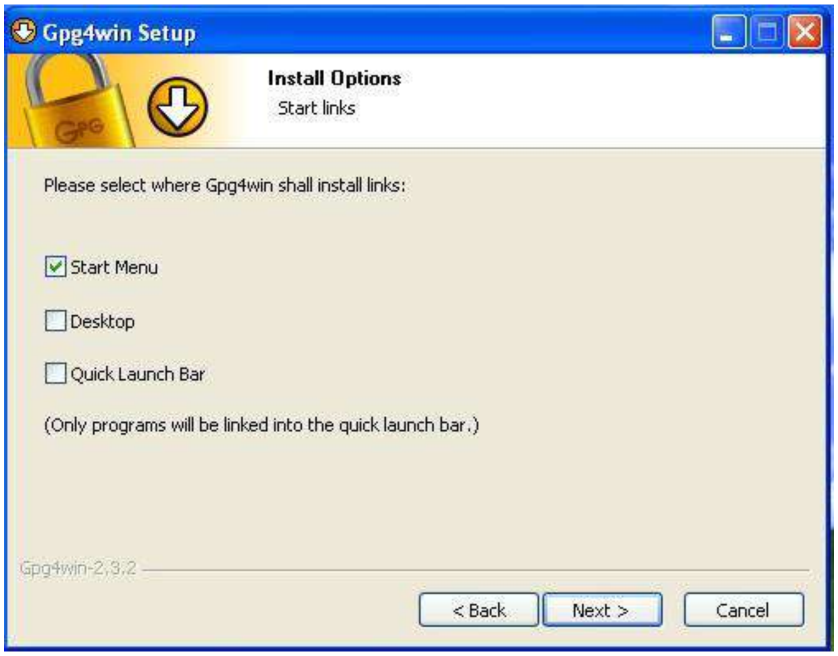


Setup will install Gpg4win in the following folder. To install in a different folder, click Browse and select another folder. Click Next to continue.

Destination Folder: C:\Program Files\GNU\GnuPG

Space required: 100.3MB
Space available: 20.2GB

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

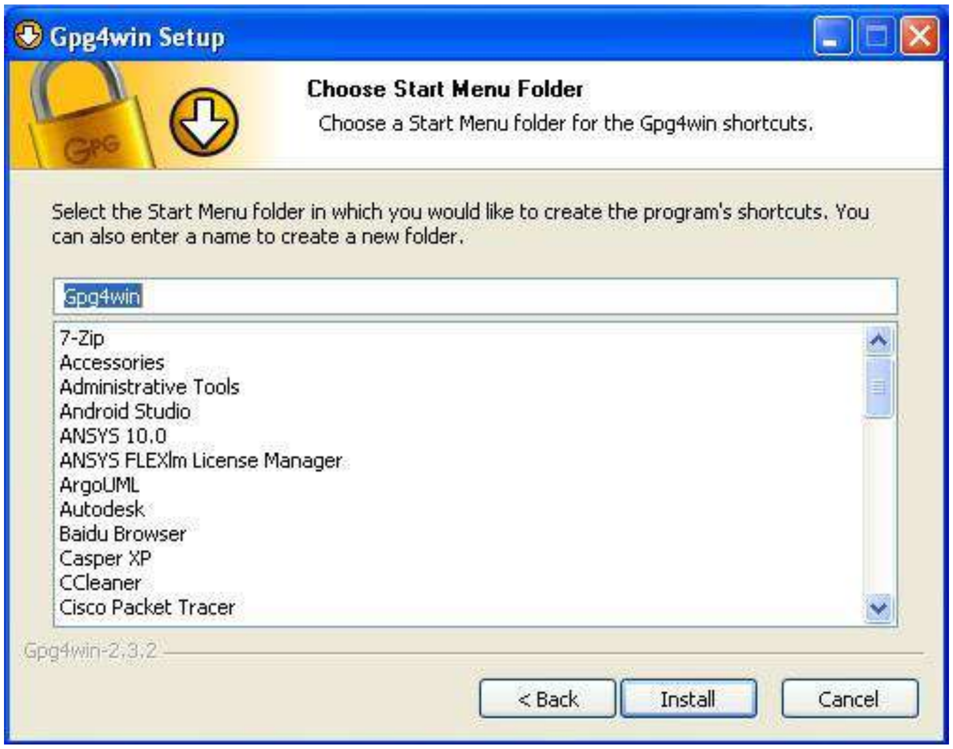


Please select where Gpg4win shall install links:

- [✓] Start Menu
- [ ] Desktop
- [ ] Quick Launch Bar

(Only programs will be linked into the quick launch bar.)

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



Choose Start Menu Folder

Choose a Start Menu folder for the Gpg4win shortcuts.

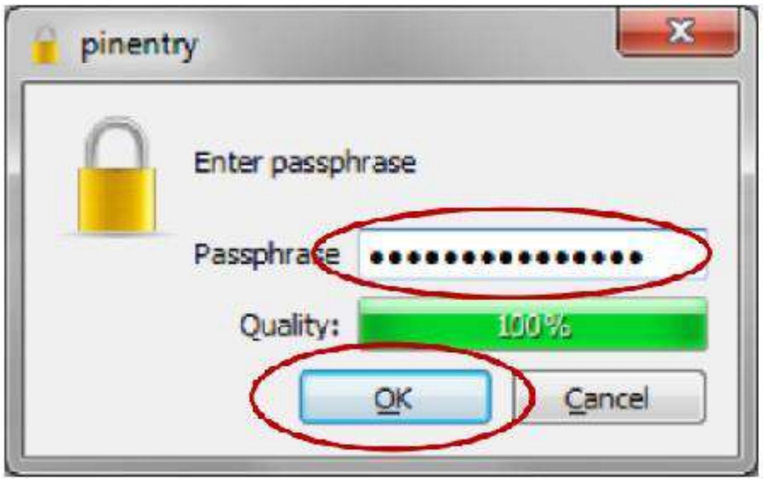
Select the Start Menu folder in which you would like to create the program's shortcuts. You can also enter a name to create a new folder.

Gpg4win

7-Zip
Accessories
Administrative Tools
Android Studio
ANSYS 10.0
ANSYS FLEXlm License Manager
ArgoUML
Autodesk
Baidu Browser
Casper XP
CCleaner
Cisco Packet Tracer

Gpg4win-2.3.2

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

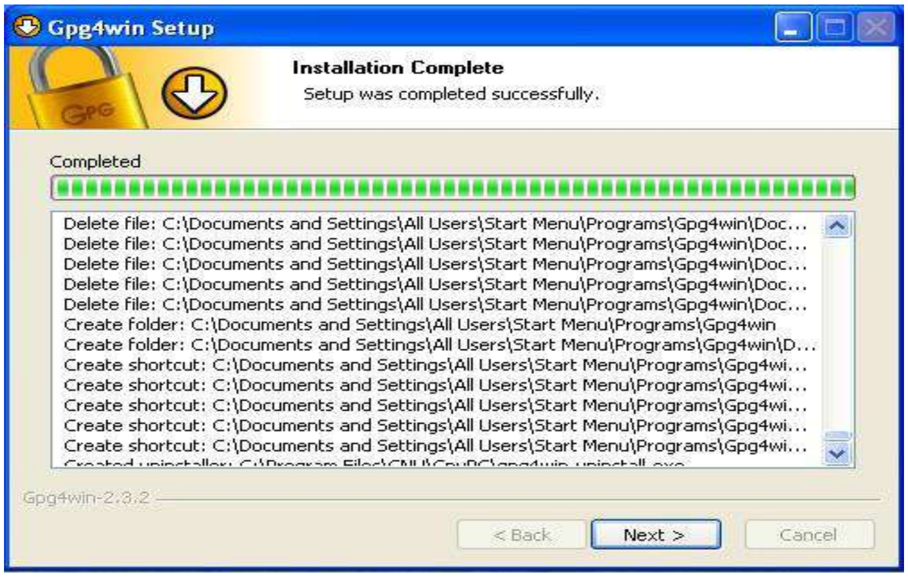


Enter passphrase

Passphrase ●●●●●●●●●●●●●●●●

Quality: 100%

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



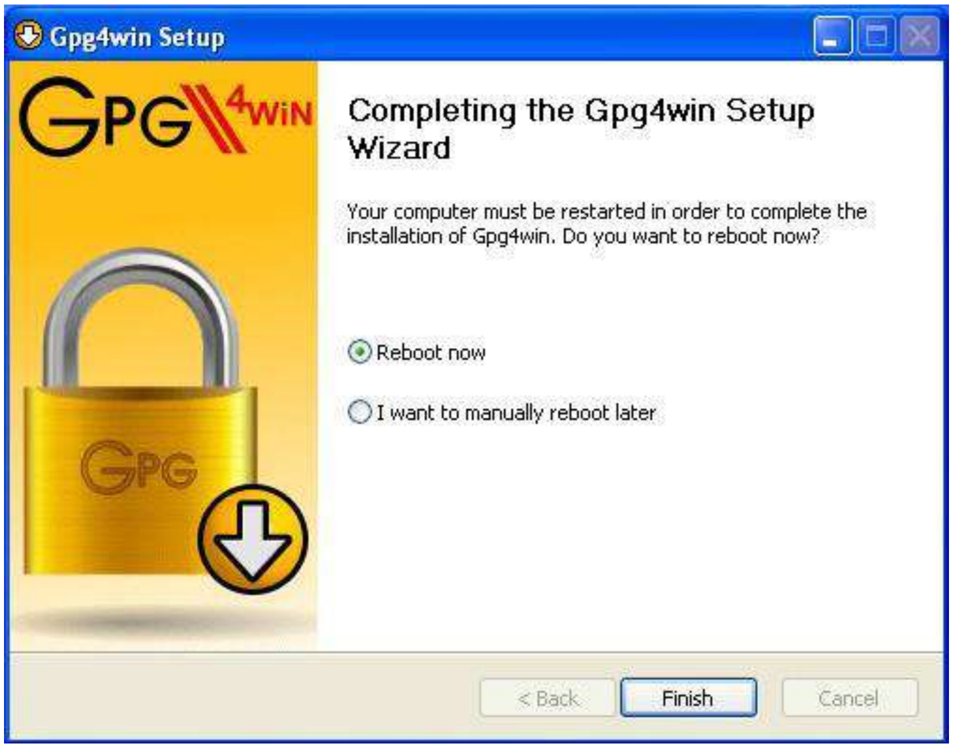
Installation Complete

Setup was completed successfully.

Completed

```
Delete file: C:\Documents and Settings\All Users\Start Menu\Programs\Gpg4win\Doc...
Delete file: C:\Documents and Settings\All Users\Start Menu\Programs\Gpg4win\Doc...
Delete file: C:\Documents and Settings\All Users\Start Menu\Programs\Gpg4win\Doc...
Delete file: C:\Documents and Settings\All Users\Start Menu\Programs\Gpg4win\Doc...
Delete file: C:\Documents and Settings\All Users\Start Menu\Programs\Gpg4win\Doc...
Create folder: C:\Documents and Settings\All Users\Start Menu\Programs\Gpg4win
Create folder: C:\Documents and Settings\All Users\Start Menu\Programs\Gpg4win\D...
Create shortcut: C:\Documents and Settings\All Users\Start Menu\Programs\Gpg4wi...
Create shortcut: C:\Documents and Settings\All Users\Start Menu\Programs\Gpg4wi...
Create shortcut: C:\Documents and Settings\All Users\Start Menu\Programs\Gpg4wi...
Create shortcut: C:\Documents and Settings\All Users\Start Menu\Programs\Gpg4wi...
Create shortcut: C:\Documents and Settings\All Users\Start Menu\Programs\Gpg4wi...
```

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



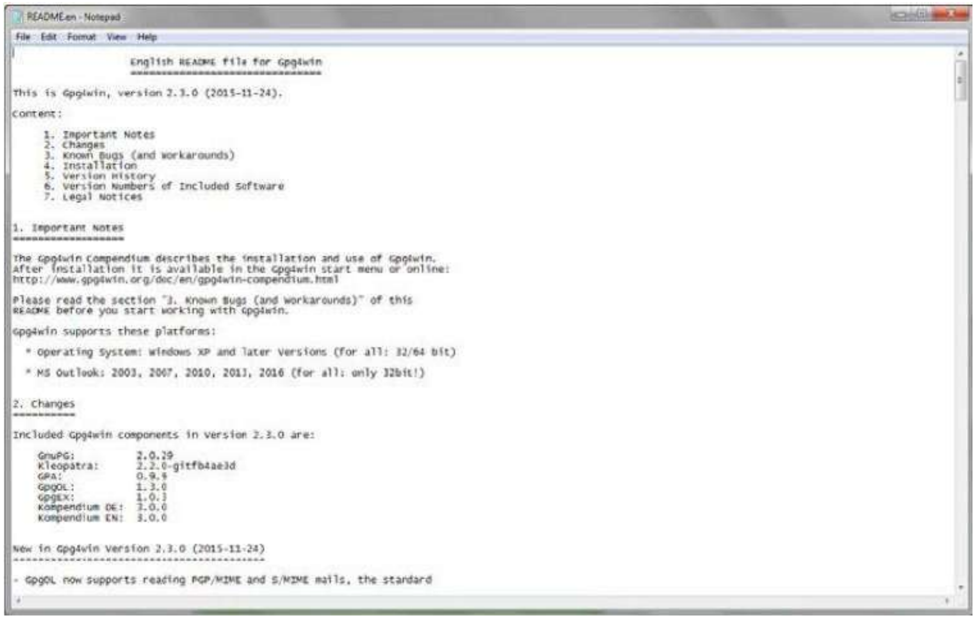
Completing the Gpg4win Setup Wizard

Your computer must be restarted in order to complete the installation of Gpg4win. Do you want to reboot now?

[✓ Reboot now]

I want to manually reboot later

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.

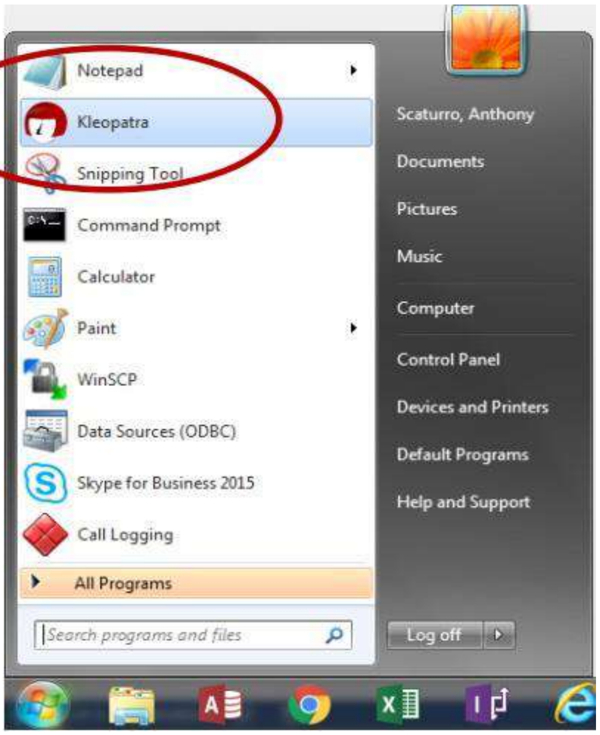


```
                English README file for Gpg4win
                ===============================

This is Gpg4win, version 2.3.0 (2015-11-24).

Content:

     1. Important Notes
     2. Changes
     3. Known Bugs (and Workarounds)
     4. Installation
     5. Version History
     6. Version Numbers of Included Software
     7. Legal Notices

1. Important Notes
==================

The Gpg4win Compendium describes the installation and use of Gpg4win.
After installation it is available in the Gpg4win start menu or online:
http://www.gpg4win.org/doc/en/gpg4win-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:

    GnuPG:           2.0.29
    Kleopatra:       2.2.0-gitfb4ae3d
    GPA:             0.9.9
    GpgOL:           1.3.0
    GpgEX:           1.0.3
    Kompendium DE:   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
```

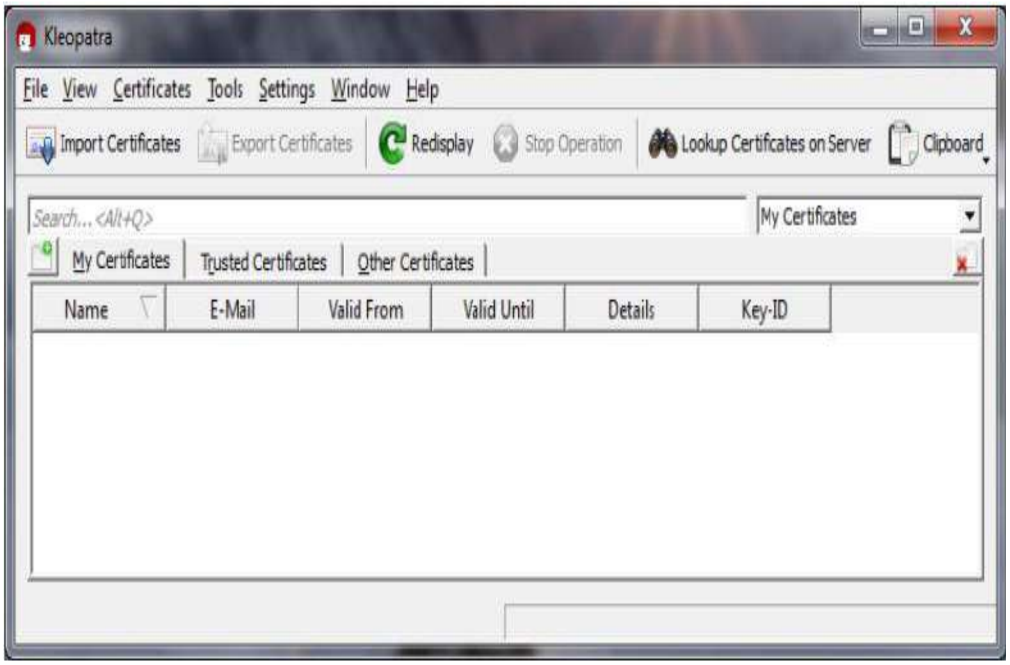
CREATING YOUR PUBLIC AND PRIVATE KEYS

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:

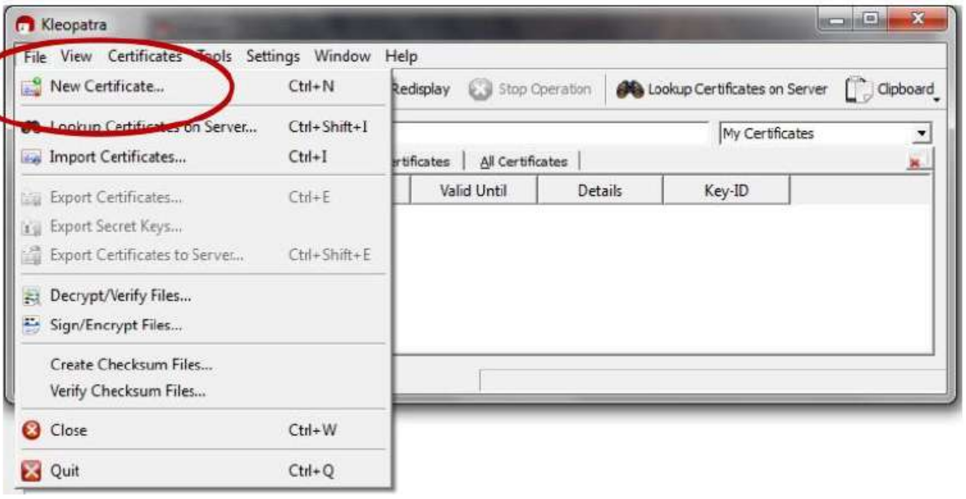
1. From your start bar, select the "Kleopatra" icon to start the Kleopatra certificate management software



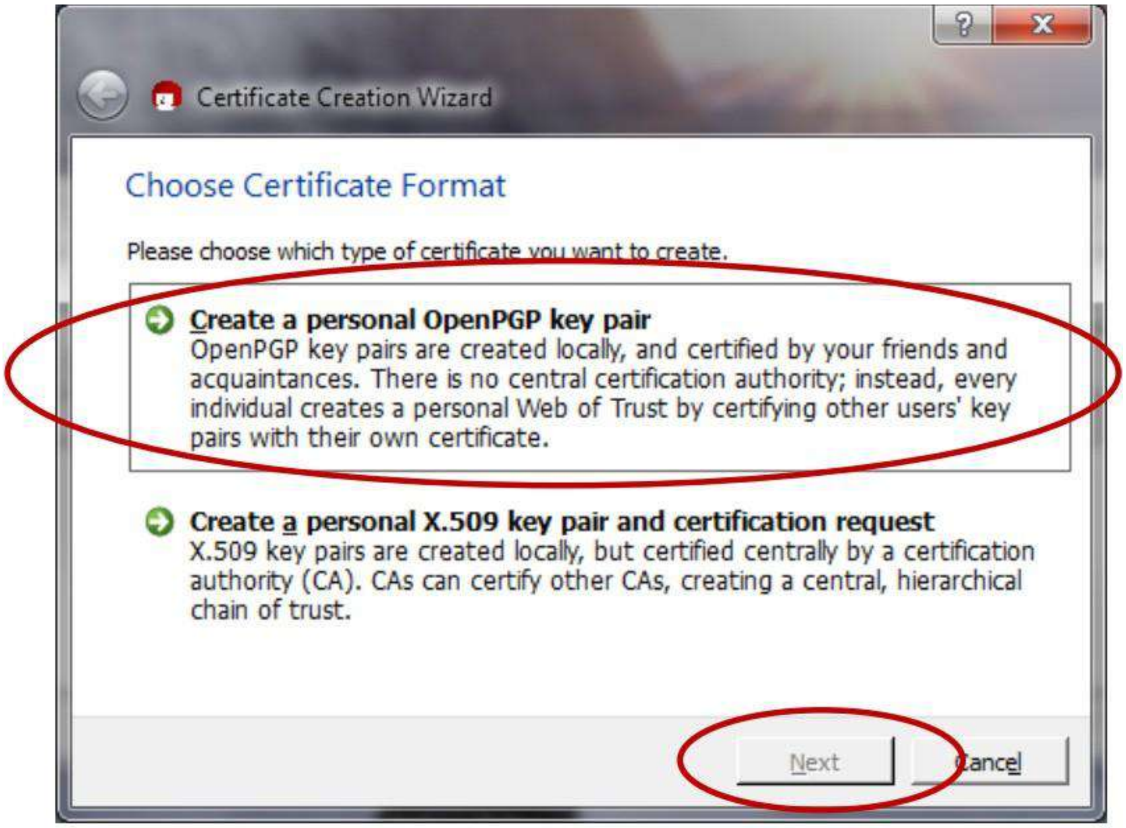
2. The following screen will be displayed



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


Kleopatra
File View Certificates Tools Settings Window Help
New Certificate... Ctrl+N
Lookup Certificates on Server... Ctrl+Shift+I
Import Certificates... Ctrl+I
Export Certificates... Ctrl+E
Export Secret Keys...
Export Certificates to Server... Ctrl+Shift+E
Decrypt/Verify Files...
Sign/Encrypt Files...
Create Checksum Files...
Verify Checksum Files...
Close Ctrl+W
Quit Ctrl+Q
Redisplay
Stop Operation
Lookup Certificates on Server
Clipboard
My Certificates
All Certificates
Valid Until
Details
Key-ID

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



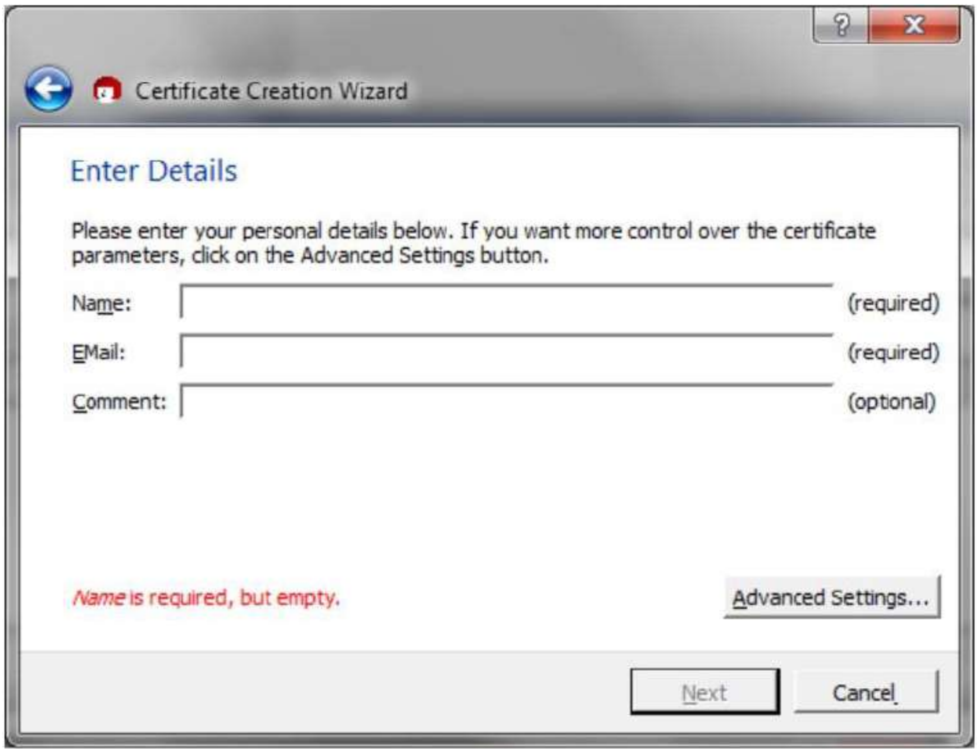
Choose Certificate Format

Please choose which type of certificate you want to create.

Create a personal OpenPGP key pair
OpenPGP key pairs are created locally, and certified by your friends and acquaintances. There is no central certification authority; instead, every individual creates a personal Web of Trust by certifying other users' key pairs with their own certificate.

Create a personal X.509 key pair and certification request
X.509 key pairs are created locally, but certified centrally by a certification authority (CA). CAs can certify other CAs, creating a central, hierarchical chain of trust.

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



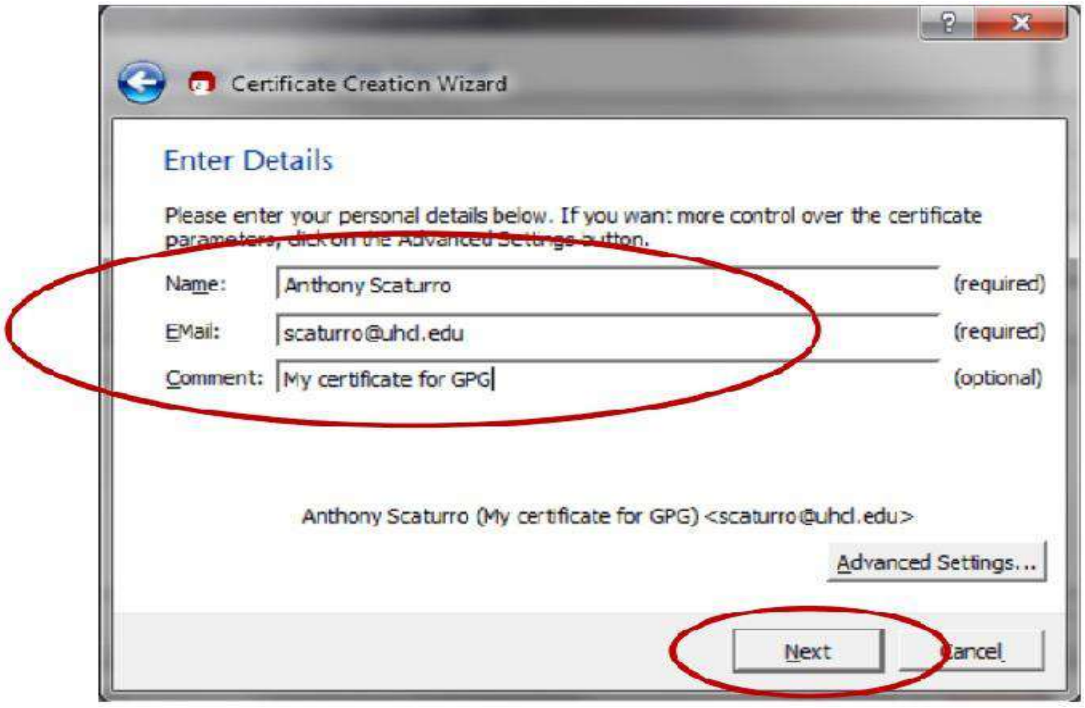
Enter Details

Please enter your personal details below. If you want more control over the certificate parameters, click on the Advanced Settings button.

Name: (required)
EMail: (required)
Comment: (optional)

Name is required, but empty.

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



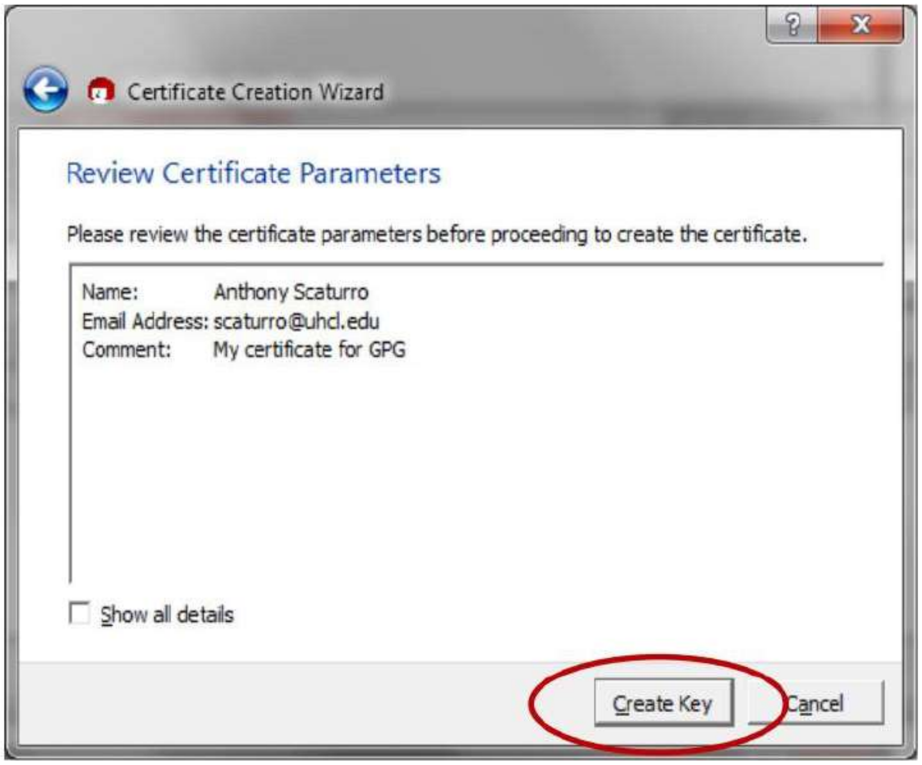
Enter Details

Please enter your personal details below. If you want more control over the certificate parameters, click on the Advanced Settings button.

Name: Anthony Scaturro (required)
EMail: scaturro@uhcl.edu (required)
Comment: My certificate for GPG (optional)

Anthony Scaturro (My certificate for GPG) <scaturro@uhcl.edu>

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

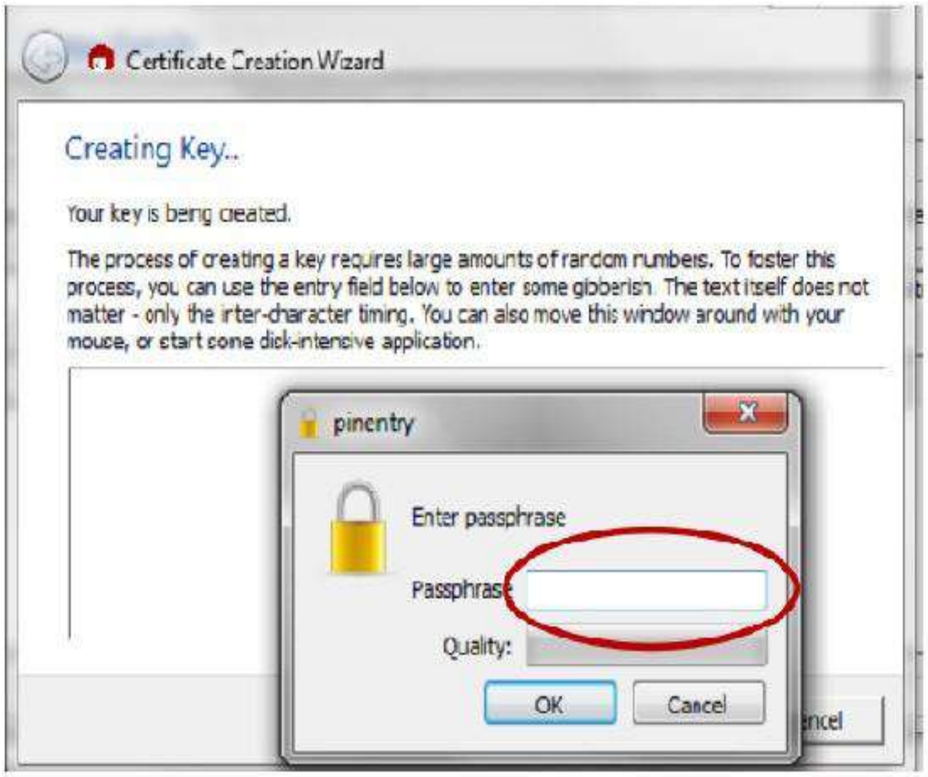

Certificate Creation Wizard

Review Certificate Parameters

Please review the certificate parameters before proceeding to create the certificate.

```
Name:          Anthony Scaturro
Email Address: scaturro@uhcl.edu
Comment:       My certificate for GPG
```

8. You will be asked to enter a passphrase

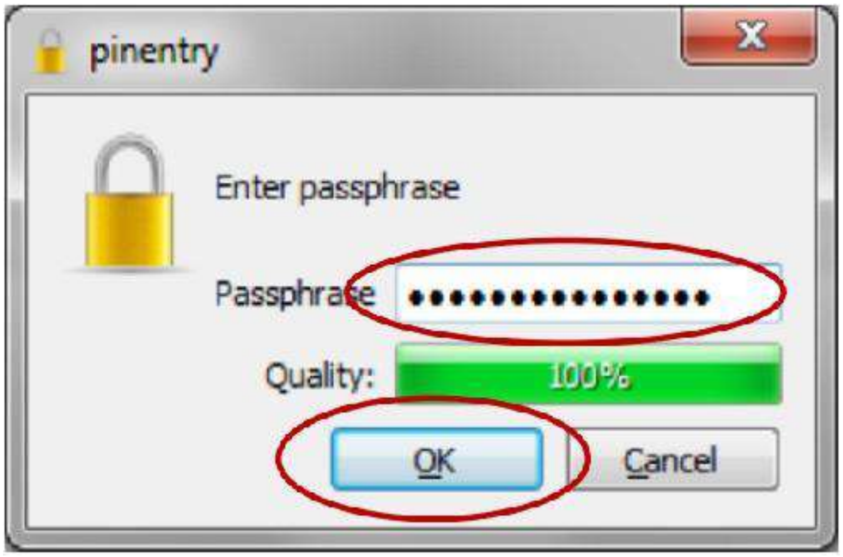

Certificate Creation Wizard

Creating Key...

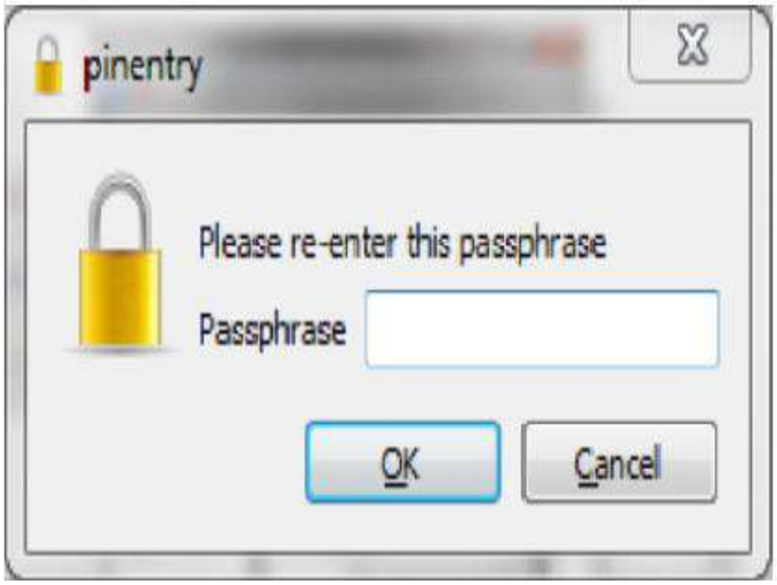
Your key is being created.

The process of creating a key requires large amounts of random numbers. To foster this process, you can use the entry field below to enter some gibberish. The text itself does not matter - only the inter-character timing. You can also move this window around with your mouse, or start some disk-intensive application.

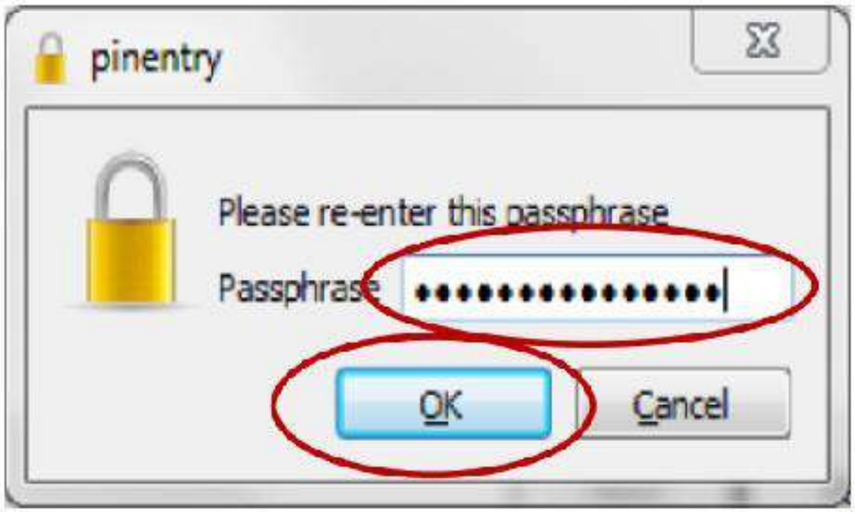
9. The passphrase should follow strong password standards. After you've entered your passphrase, click the "OK" button.



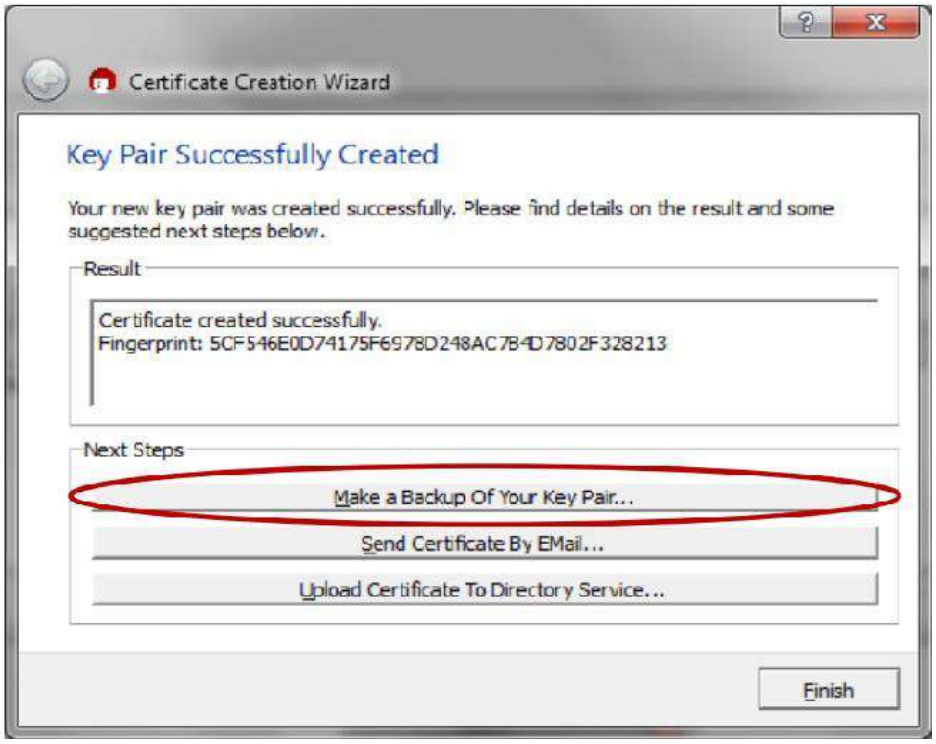
10. You will be asked to re-enter the passphrase



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.


Certificate Creation Wizard
Key Pair Successfully Created

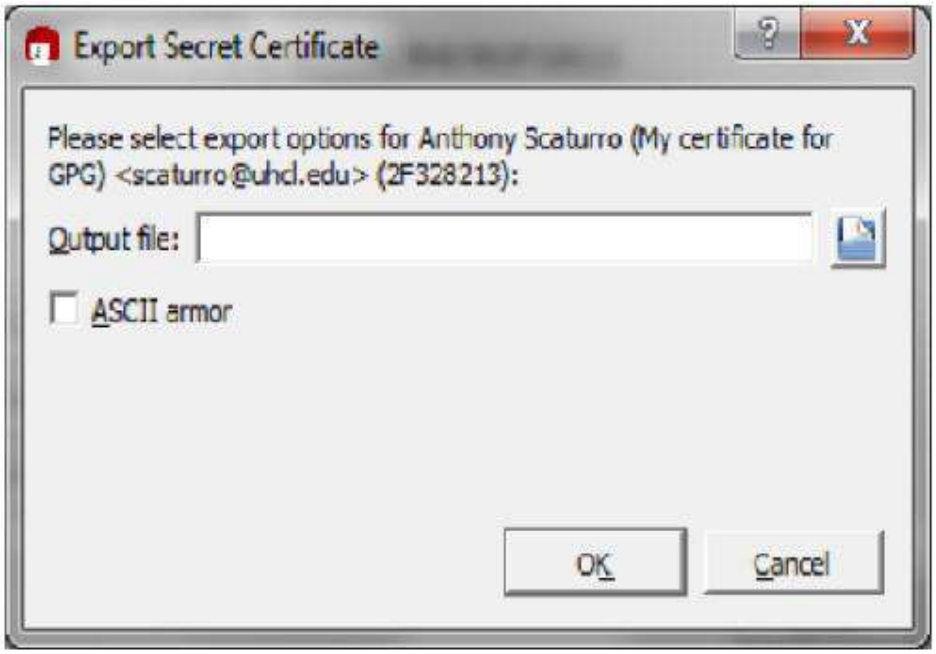
Your new key pair was created successfully. Please find details on the result and some suggested next steps below.

Result

```
Certificate created successfully.
Fingerprint: 5CF546E0D74175F6978D248AC7B4D7802F328213
```

Next Steps
Make a Backup Of Your Key Pair...
Send Certificate By EMail...
Upload Certificate To Directory Service...
Finish

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

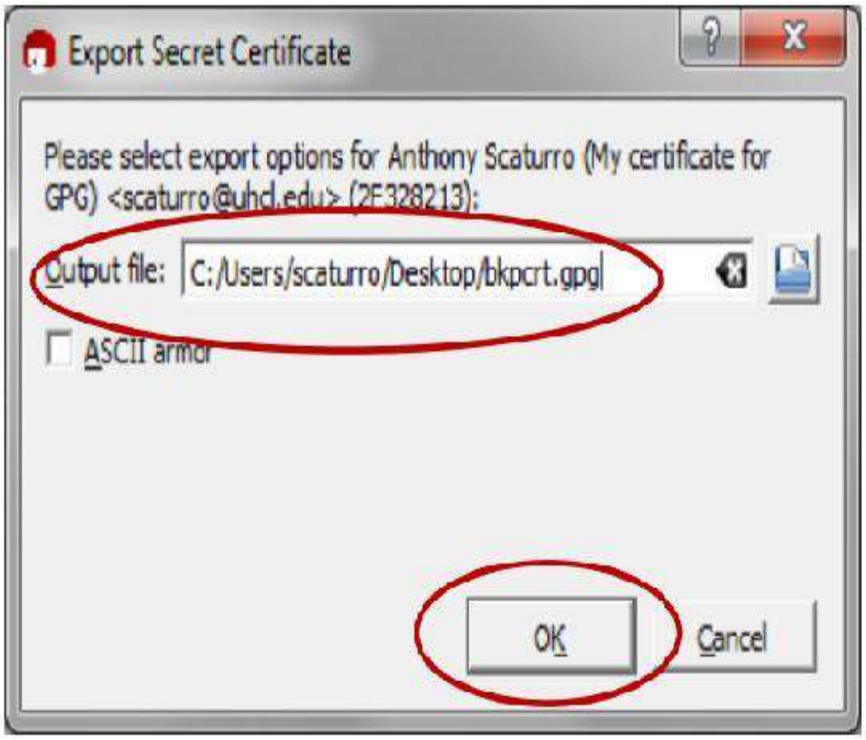

Export Secret Certificate

Please select export options for Anthony Scaturro (My certificate for GPG) <scaturro@uhcl.edu> (2F328213):

Output file:

ASCII armor

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

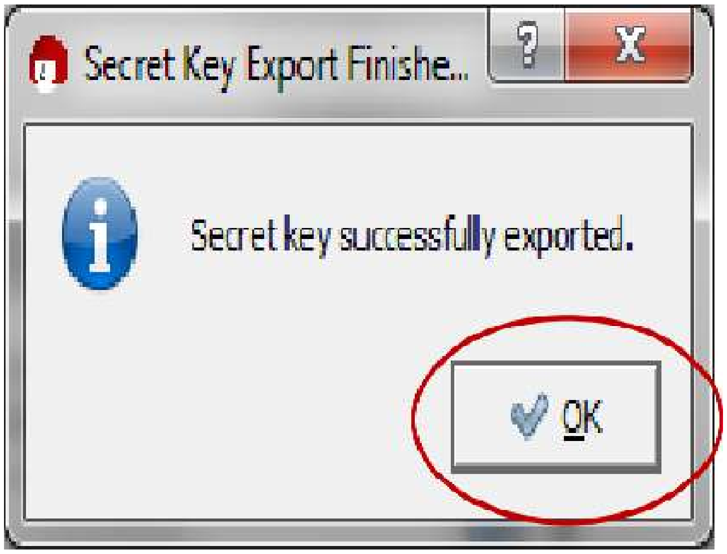

Export Secret Certificate

Please select export options for Anthony Scaturro (My certificate for GPG) <scaturro@uhcl.edu> (2F328213):

Output file: C:/Users/scaturro/Desktop/bkpcrt.gpg

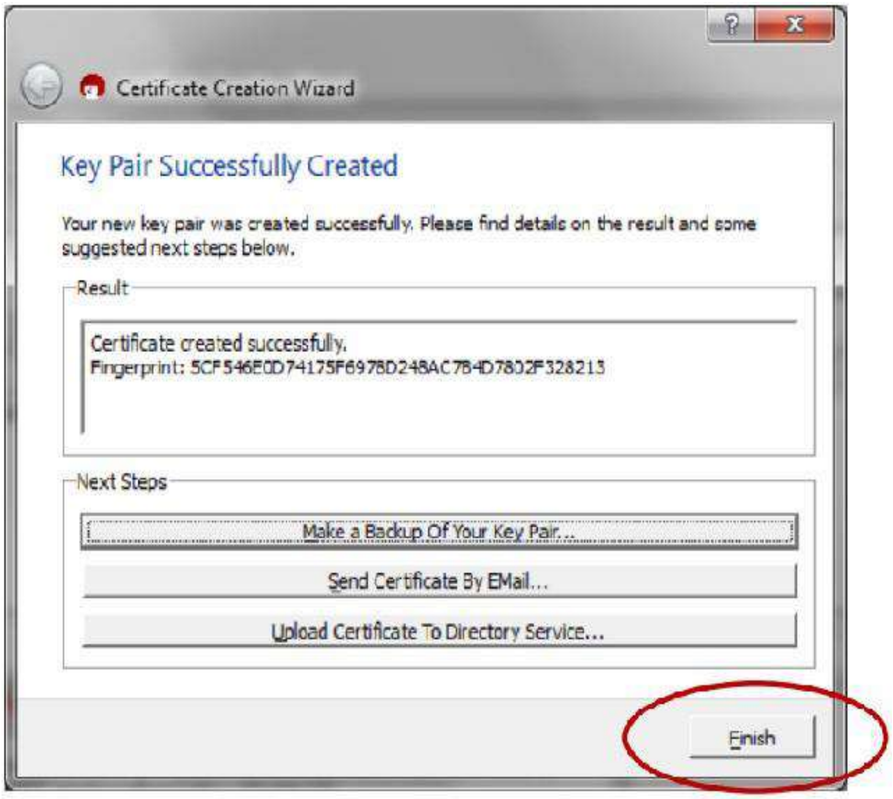
ASCII armor

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


Secret Key Export Finishe...

Secret key successfully exported.

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


Certificate Creation Wizard

Key Pair Successfully Created

Your new key pair was created successfully. Please find details on the result and some suggested next steps below.

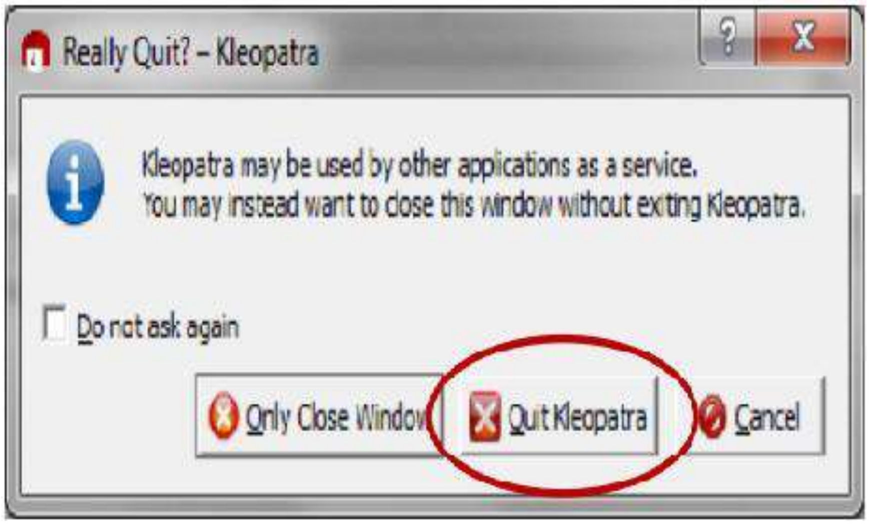
Result

Certificate created successfully.
Fingerprint: 5CF546E0D74175F6978D248AC7B4D7802F328213

Next Steps

Make a Backup Of Your Key Pair...
Send Certificate By EMail...
Upload Certificate To Directory Service...
Finish

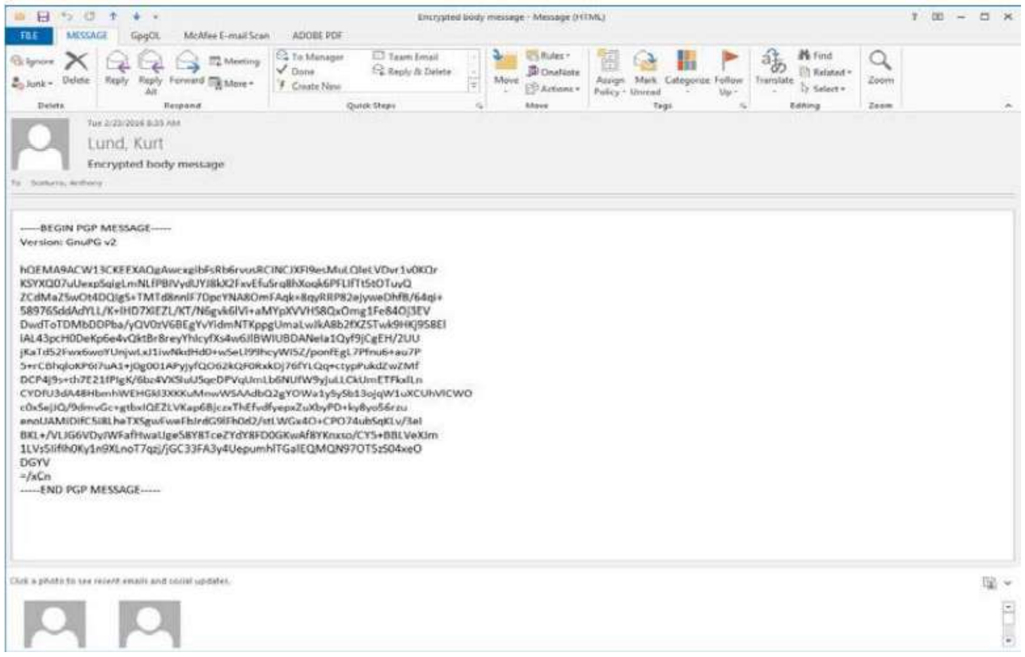
17. Before the program closes, you will need to confirm that you want to close the program by clicking on the “Quit Kleopatra” button



Kleopatra may be used by other applications as a service.
You may instead want to close this window without exiting Kleopatra.

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

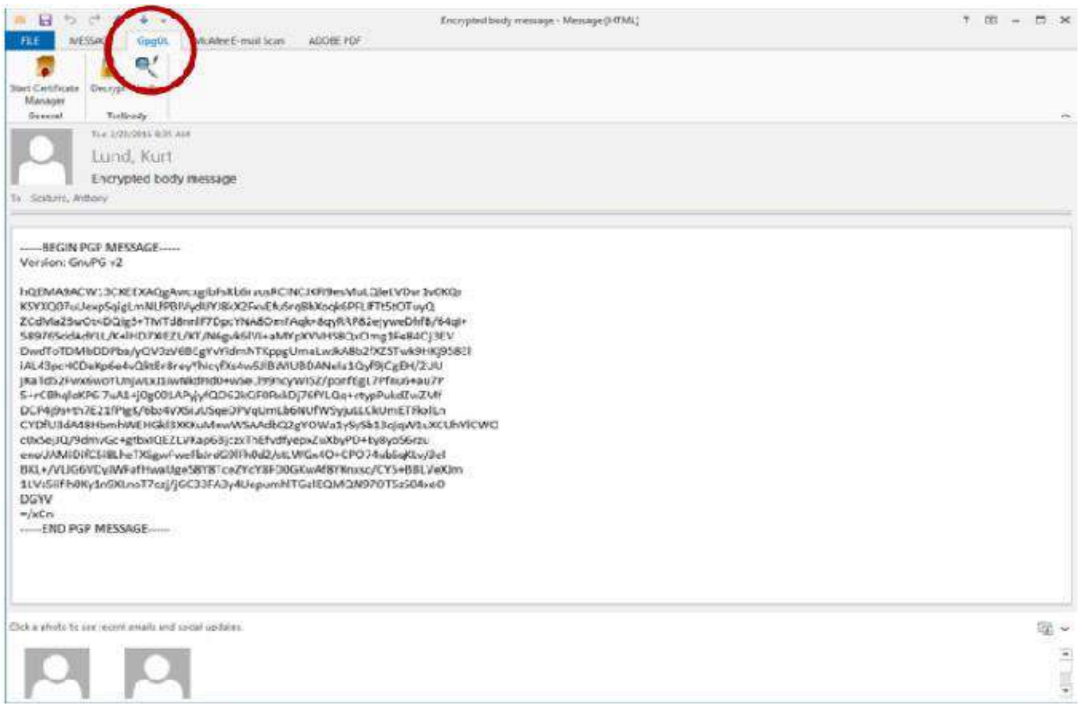
1. Open the e-mail message



Tue 2/23/2016 8:35 AM
Lund, Kurt
Encrypted body message
To Scaturro, Anthony

```
-----BEGIN PGP MESSAGE-----
Version: GnuPG v2

hQEMA9ACW13CKEEXAQgAwcxglbFsRb6rvusRCINCJXFI9esMuLQleLVDvr1v0KOr
KSYXQ07uUexpSqigLmNLfPBiVydUYJ8kX2FxvEfuSrq8hXoqk6PFLIfTt5tOTuvQ
ZCdMaZ5wOt4DQIgS+TMTd8nnlF7DpcYNA8OmFAqk+8qyRRP82ejyweDhfB/64qi+
58976SddAdYLL/K+IHD7XIEZL/KT/N6gvk6lVi+aMYpXVVH58QxOmg1Fe84OJ3EV
DwdToTDMbDOPba/yQV0zV6BEgYvYldmNTKppgUmaLwJkA8b2fXZSTwk9HKj9S8EI
iAL43pcH0DeKp6e4vQktBr8reyYhlcyfXs4w6JlBWIUBDANela1Qyf9jCgEH/2UU
jKaTd52Fwx6woYUnjwLxJ1iwNkdHd0+wSeLl99hcyWI5Z/ponfEgL7Pfnu6+au7P
5+rC8hqloKP6t7uA1+jOg001APyjyfQO62kQF0RxkDj76fYLQq+ctypPukdZwZMf
DCP4j9s+th7E21fPlgK/6bz4VXSluU5qeDPVqUm1b6NUfW9yjuLLCkUmETFkalLn
CYDfU3dA48HbmhWEHGkl3XKKuMnwWSAAdbQ2gYOWa1y5y5b13ojqW1uXCUhVlCWO
c0xSejJQ/9dmvGc+gtbxlQEZLVKap6BjczxThEfvdfyepxZuXbyPD+ky8yo56rzu
enoUAMiDlfC5i8LhaTX5guFweFbIrdG9lFh0d2/stLWGx4O+CPO74ubSqKLv/3el
BKL+/VLJG6VDyJWFafHwaUge58Y8TceZYdY8FDOGKwAf8YKnxso/CY5+BBLVeXlm
1LVsSlifh0Ky1n9XLnoT7qzj/jGC33FA3y4UepumhlTGaIEQMQN97OT5z504xeO
DGYV
=/xCn
-----END PGP MESSAGE-----
```

2. Select the GpgOL tab

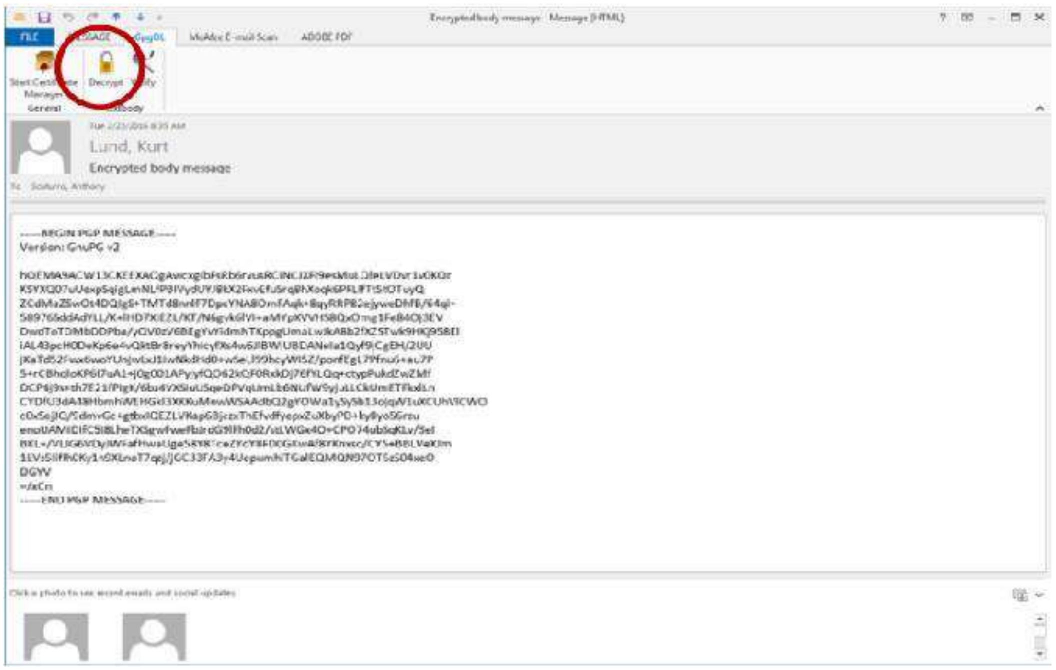


Tue 2/23/2016 8:35 AM
Lund, Kurt
Encrypted body message
To Sciturro, Anthony

```
-----BEGIN PGP MESSAGE-----
Version: GnuPG v2

hQEMA9ACW13CKEEXAQgAwcxgibFsRb6rzusRCINCJ2F9esMut3leLVDvr1vOKQr
KSYXQ07uUexpSqigLmNLfPBIVydUYJ8lX2FevEfuSrqBhXoqk6PFLfFTt5tOTuyQ
ZCdMaZ5wOt4DQIg5+TMTd8nnlF7DpcYNA8OmfAqk+8qyRRP82ejyweDhf8/64ql+
589765ddAdYLL/K+IHD7XEZL/KT/N6gvk6lVl+aMYpKVVH58QxOmg1FeB4OJ3EV
DwdToTDMbDDPba/yOV0zV6BEgYvYidmhTKppgUmaLwJkABb2fXZ5Twk9HKQ958EI
iAL43pcH0DeKp6e4vQkt8r8reyYhicyfXs4w5JlBWUBDANela1Qyf9jCgEH/2UU
jKaTd52Fwx6woYUnjwbJ1lwNkdHd0+wSeJ99hcyWI5Z/ponfEgL7Pfnu6+au7P
5+rCBhcloKP6I7uA1+jOgO01APyyfQO62kCF0RxkDj76fYLQq+ctypPukdZwZMf
DCP4j9s+th7E21fPlgK/6bu4VX5iuUSqeDPVqUmLb6NUfWSyjuLLCkUmETFkdLn
CYDfU3dA48HbmhWEHGkl3XKKuMewWSAAdbQ2gYOWa1ySySb13ojqWluXCUhVICWO
c0x5ejJQ/9dmvGc+gtbxlQEZLVKap63jczxThEfvdfyepxZuXbyPD+ky8yo56rzu
enoUAMiDIfC5l8LheTXSgwfwefbzrdG9lFh0d2/vLWGx4O+CPO74ub6qKLv/5el
BKL+/VUG6VDyIWFafHwaUge5KY8TceZYcY8FD0GKwAf8YKnxcc/CY5+B8LVeKlm
1LVt5ilFh0Ky1n9XLnoT7qzj/jGC33FA3y4UepumhTGalEQMQN97OT5z5O4xeO
DGYV
=/xCn
-----END PGP MESSAGE-----
```

Click a photo to see recent emails and social updates.

3. Click the “Decrypt” button

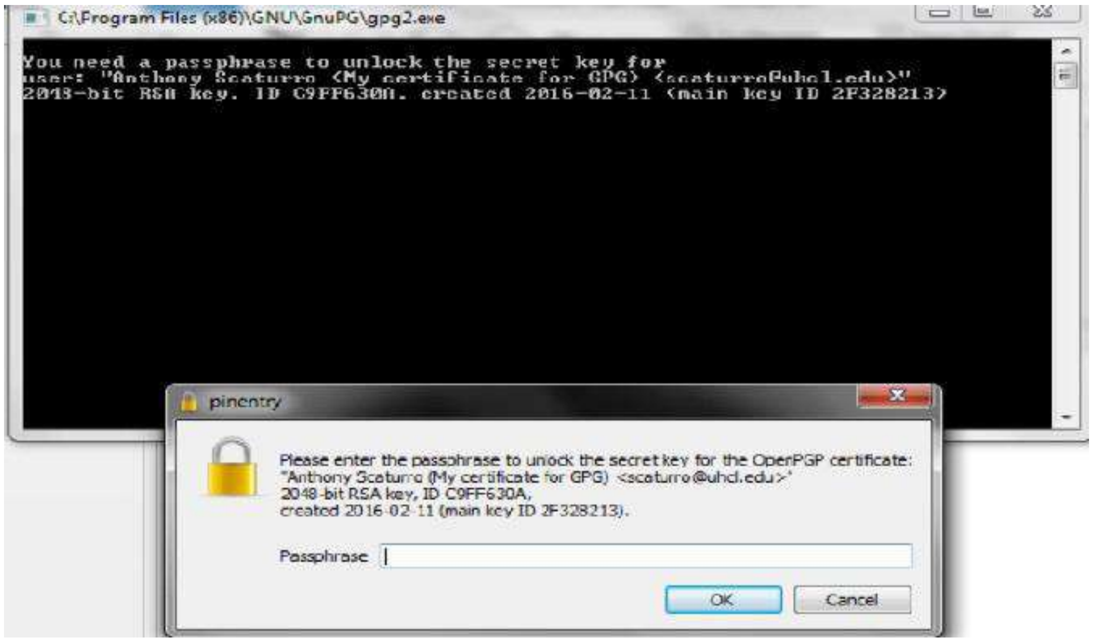


Tue 2/23/2016 8:35 AM
Lund, Kurt
Encrypted body message
To Sciturro, Anthony

```
-----BEGIN PGP MESSAGE-----
Version: GnuPG v2

hQEMA9ACW13CKEEXAQgAwcxgibFsRb6rzusRCINCJ2F9esMut3leLVDvr1vOKQr
KSYXQ07uUexpSqigLmNLfPBIVydUYJ8lX2FevEfuSrqBhXoqk6PFLfFTt5tOTuyQ
ZCdMaZ5wOt4DQIg5+TMTd8nnlF7DpcYNA8OmfAqk+8qyRRP82ejyweDhf8/64ql+
589765ddAdYLL/K+IHD7XEZL/KT/N6gvk6lVl+aMYpKVVH58QxOmg1FeB4OJ3EV
DwdToTDMbDDPba/yOV0zV6BEgYvYidmhTKppgUmaLwJkABb2fXZ5Twk9HKQ958EI
iAL43pcH0DeKp6e4vQkt8r8reyYhicyfXs4w5JlBWUBDANela1Qyf9jCgEH/2UU
jKaTd52Fwx6woYUnjwbJ1lwNkdHd0+wSeJ99hcyWI5Z/ponfEgL7Pfnu6+au7P
5+rCBhcloKP6I7uA1+jOgO01APyyfQO62kCF0RxkDj76fYLQq+ctypPukdZwZMf
DCP4j9s+th7E21fPlgK/6bu4VX5iuUSqeDPVqUmLb6NUfWSyjuLLCkUmETFkdLn
CYDfU3dA48HbmhWEHGkl3XKKuMewWSAAdbQ2gYOWa1ySySb13ojqWluXCUhVICWO
c0x5ejJQ/9dmvGc+gtbxlQEZLVKap63jczxThEfvdfyepxZuXbyPD+ky8yo56rzu
enoUAMiDIfC5l8LheTXSgwfwefbzrdG9lFh0d2/vLWGx4O+CPO74ub6qKLv/5el
BKL+/VUG6VDyIWFafHwaUge5KY8TceZYcY8FD0GKwAf8YKnxcc/CY5+B8LVeKlm
1LVt5ilFh0Ky1n9XLnoT7qzj/jGC33FA3y4UepumhTGalEQMQN97OT5z5O4xeO
DGYV
=/xCn
-----END PGP MESSAGE-----
```

Click a photo to see recent emails and social updates.

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.

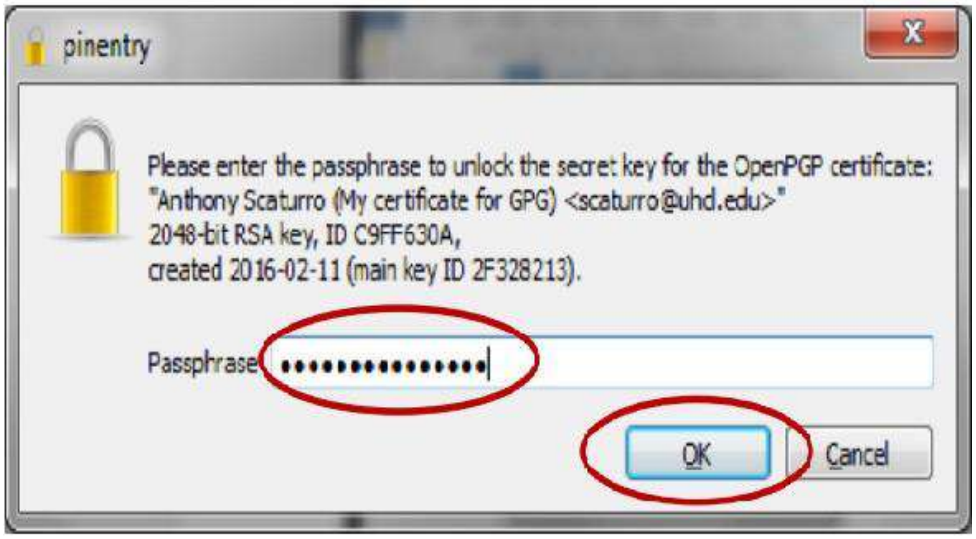


```
You need a passphrase to unlock the secret key for
user: "Anthony Scaturro (My certificate for GPG) <scaturro@uhcl.edu>"
2048-bit RSA key, ID C9FF630A, created 2016-02-11 (main key ID 2F328213)
```

Please enter the passphrase to unlock the secret key for the OpenPGP certificate:
"Anthony Scaturro (My certificate for GPG) <scaturro@uhcl.edu>"
2048-bit RSA key, ID C9FF630A,
created 2016-02-11 (main key ID 2F328213).

Passphrase

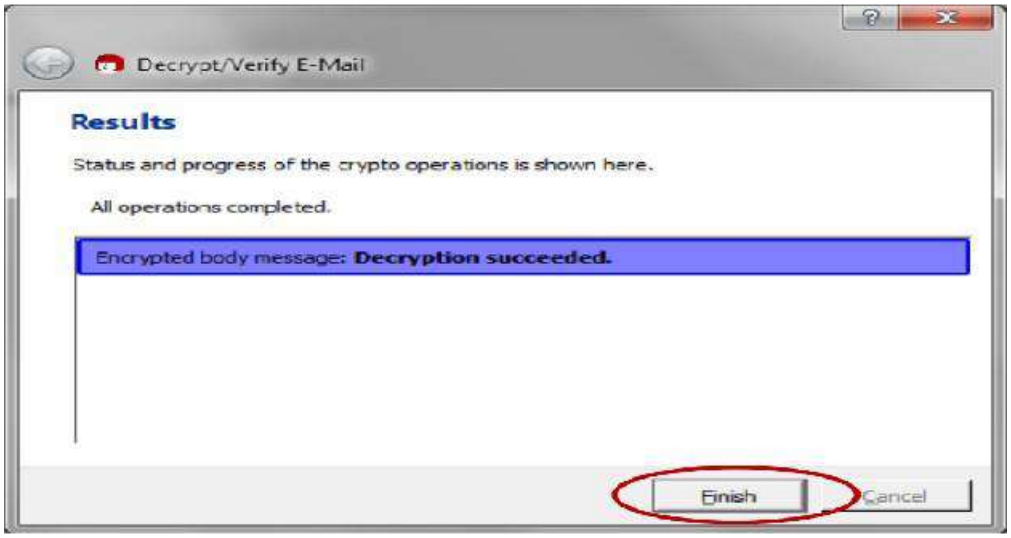
5. Enter your passphrase and click the “OK” button



Please enter the passphrase to unlock the secret key for the OpenPGP certificate:
"Anthony Scaturro (My certificate for GPG) <scaturro@uhd.edu>"
2048-bit RSA key, ID C9FF630A,
created 2016-02-11 (main key ID 2F328213).

Passphrase ••••••••••••••••

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



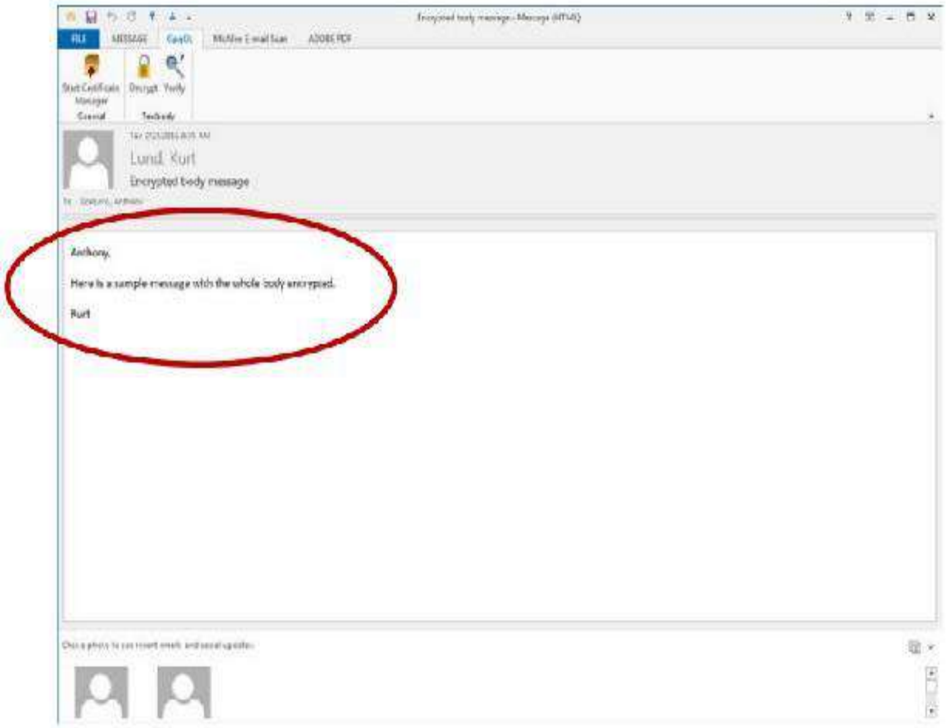
Results

Status and progress of the crypto operations is shown here.

All operations completed.

Encrypted body message: **Decryption succeeded.**

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



Lund, Kurt

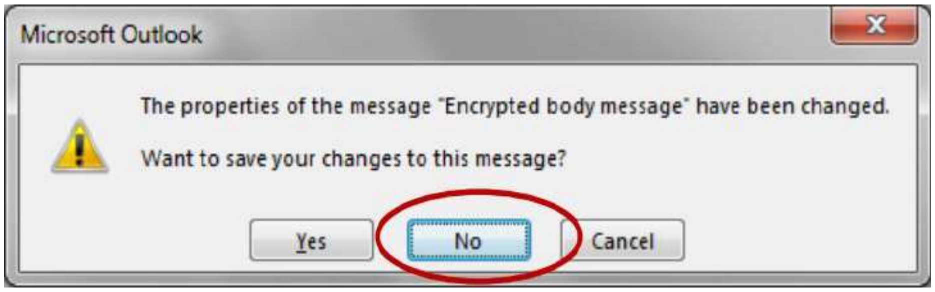
Encrypted body message

Anthony,

Here is a sample message with the whole body encrypted.

Kurt

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


Microsoft Outlook

The properties of the message "Encrypted body message" have been changed.

Want to save your changes to this message?

RESULT:

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

EX. NO: 05

WORKING WITH KF SENSOR TOOL FOR CREATING AND MONITORING HONEYPOT

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.

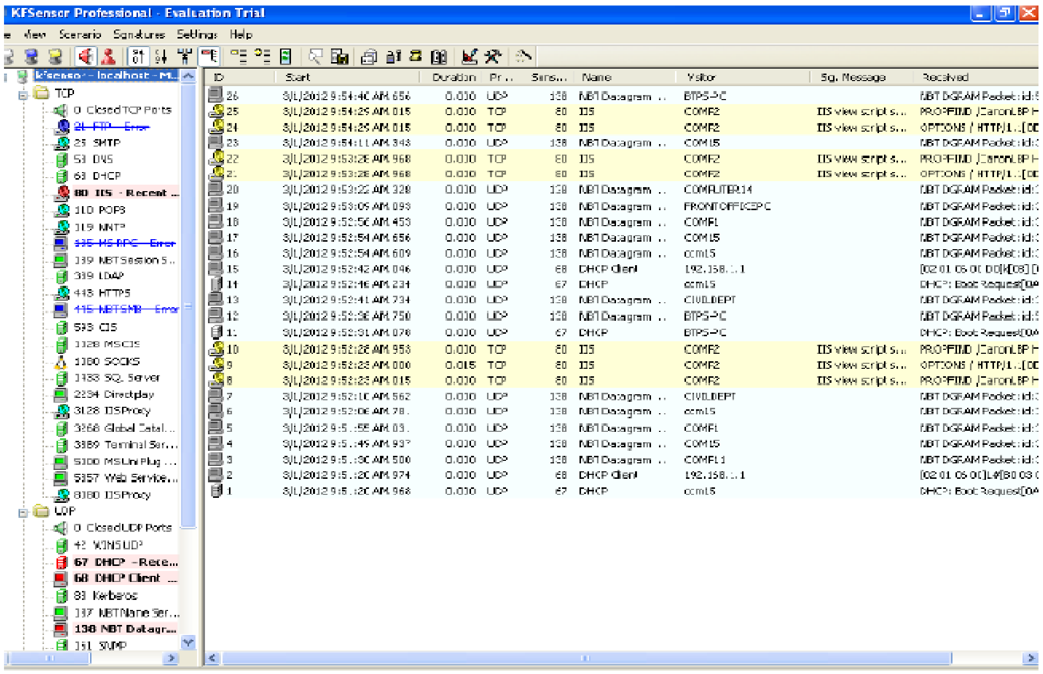
STEP-6: “Send the email and Send from email”, enter the ID and Click Next.

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.

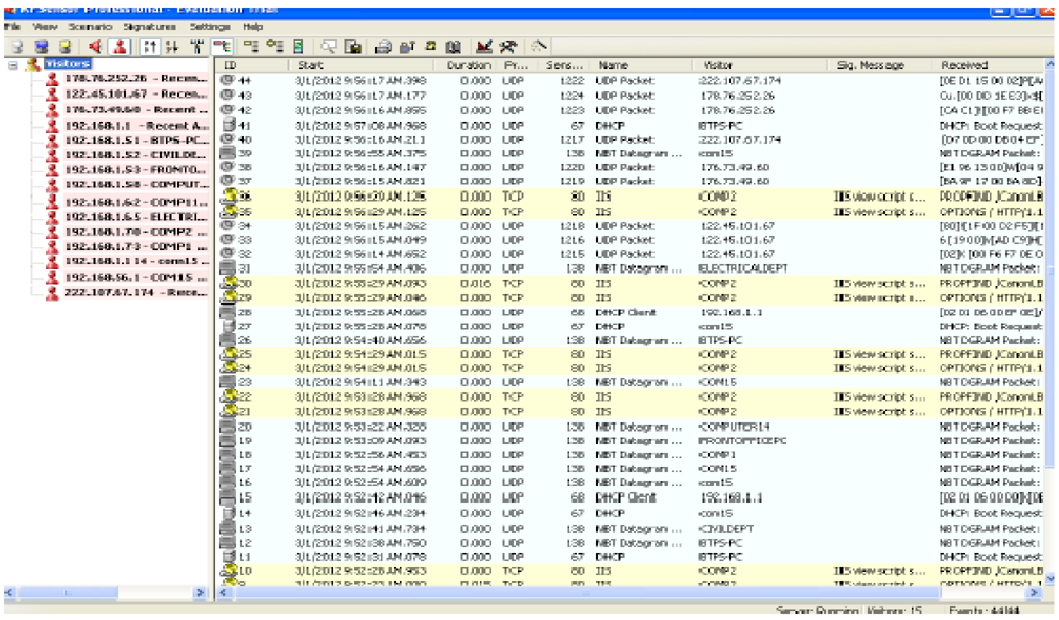
SCREENSHOTS:


KFSensor Professional - Evaluation Trial

e View Scenario Signatures Settings Help

kfsensor - localhost - M...
- TCP
  - 0 Closed TCP Ports
  - ~~21 FTP - Error~~
  - 25 SMTP
  - 53 DNS
  - 68 DHCP
  - 80 IIS - Recent ...
  - 110 POP3
  - 119 NNTP
  - ~~135 MS RPC - Error~~
  - 139 NBT Session S...
  - 389 LDAP
  - 443 HTTPS
  - ~~445 NBT SMB - Error~~
  - 593 CIS
  - 1028 MSCIS
  - 1080 SOCKS
  - 1433 SQL Server
  - 2234 Directplay
  - 3128 IISProxy
  - 3268 Global Catal...
  - 3389 Terminal Ser...
  - 5000 MS Uni Plug ...
  - 5357 Web Service...
  - 8080 IISProxy
- UDP
  - 0 Closed UDP Ports
  - 42 WINS UDP
  - 67 DHCP - Rece...
  - 68 DHCP Client ...
  - 88 Kerberos
  - 137 NBT Name Ser...
  - 138 NBT Datagr...
  - 161 SNMP

ID	Start	Duration	Pr...	Sens...	Name	Visitor	Sig. Message	Received
26	3/1/2012 9:54:40 AM 656	0.000	UDP	138	NBT Datagram ...	BTPS-PC		NBT DGRAM Packet: id:5
25	3/1/2012 9:54:29 AM 015	0.000	TCP	80	IIS	COMP2	IIS view script s...	PROPFIND /CanonLBP H
24	3/1/2012 9:54:29 AM 015	0.000	TCP	80	IIS	COMP2	IIS view script s...	OPTIONS / HTTP/1.1[0D
23	3/1/2012 9:54:11 AM 343	0.000	UDP	138	NBT Datagram ...	COM15		NBT DGRAM Packet: id:3
22	3/1/2012 9:53:28 AM 968	0.000	TCP	80	IIS	COMP2	IIS view script s...	PROPFIND /CanonLBP H
21	3/1/2012 9:53:28 AM 968	0.000	TCP	80	IIS	COMP2	IIS view script s...	OPTIONS / HTTP/1.1[0D
20	3/1/2012 9:53:22 AM 328	0.000	UDP	138	NBT Datagram ...	COMPUTER14		NBT DGRAM Packet: id:1
19	3/1/2012 9:53:09 AM 093	0.000	UDP	138	NBT Datagram ...	FRONTOFFICEPC		NBT DGRAM Packet: id:3
18	3/1/2012 9:52:56 AM 453	0.000	UDP	138	NBT Datagram ...	COMP1		NBT DGRAM Packet: id:3
17	3/1/2012 9:52:54 AM 656	0.000	UDP	138	NBT Datagram ...	COM15		NBT DGRAM Packet: id:3
16	3/1/2012 9:52:54 AM 609	0.000	UDP	138	NBT Datagram ...	com15		NBT DGRAM Packet: id:3
15	3/1/2012 9:52:42 AM 046	0.000	UDP	68	DHCP Client	192.168.1.1		[02 01 06 00 00]k[08] [0
14	3/1/2012 9:52:46 AM 234	0.000	UDP	67	DHCP	com15		DHCP: Boot Request[0A
13	3/1/2012 9:52:41 AM 734	0.000	UDP	138	NBT Datagram ...	CIVILDEPT		NBT DGRAM Packet: id:1
12	3/1/2012 9:52:38 AM 750	0.000	UDP	138	NBT Datagram ...	BTPS-PC		NBT DGRAM Packet: id:5
11	3/1/2012 9:52:31 AM 078	0.000	UDP	67	DHCP	BTPS-PC		DHCP: Boot Request[0A
10	3/1/2012 9:52:28 AM 953	0.000	TCP	80	IIS	COMP2	IIS view script s...	PROPFIND /CanonLBP H
9	3/1/2012 9:52:23 AM 000	0.015	TCP	80	IIS	COMP2	IIS view script s...	OPTIONS / HTTP/1.1[0D
8	3/1/2012 9:52:23 AM 015	0.000	TCP	80	IIS	COMP2	IIS view script s...	PROPFIND /CanonLBP H
7	3/1/2012 9:52:10 AM 562	0.000	UDP	138	NBT Datagram ...	CIVILDEPT		NBT DGRAM Packet: id:3
6	3/1/2012 9:52:06 AM 781	0.000	UDP	138	NBT Datagram ...	com15		NBT DGRAM Packet: id:1
5	3/1/2012 9:51:55 AM 031	0.000	UDP	138	NBT Datagram ...	COMP1		NBT DGRAM Packet: id:1
4	3/1/2012 9:51:49 AM 937	0.000	UDP	138	NBT Datagram ...	COM15		NBT DGRAM Packet: id:3
3	3/1/2012 9:51:30 AM 500	0.000	UDP	138	NBT Datagram ...	COMP11		NBT DGRAM Packet: id:3
2	3/1/2012 9:51:20 AM 974	0.000	UDP	68	DHCP Client	192.168.1.1		[02 01 06 00]L#[80 03 0
1	3/1/2012 9:51:20 AM 968	0.000	UDP	67	DHCP	com15		DHCP: Boot Request[0A

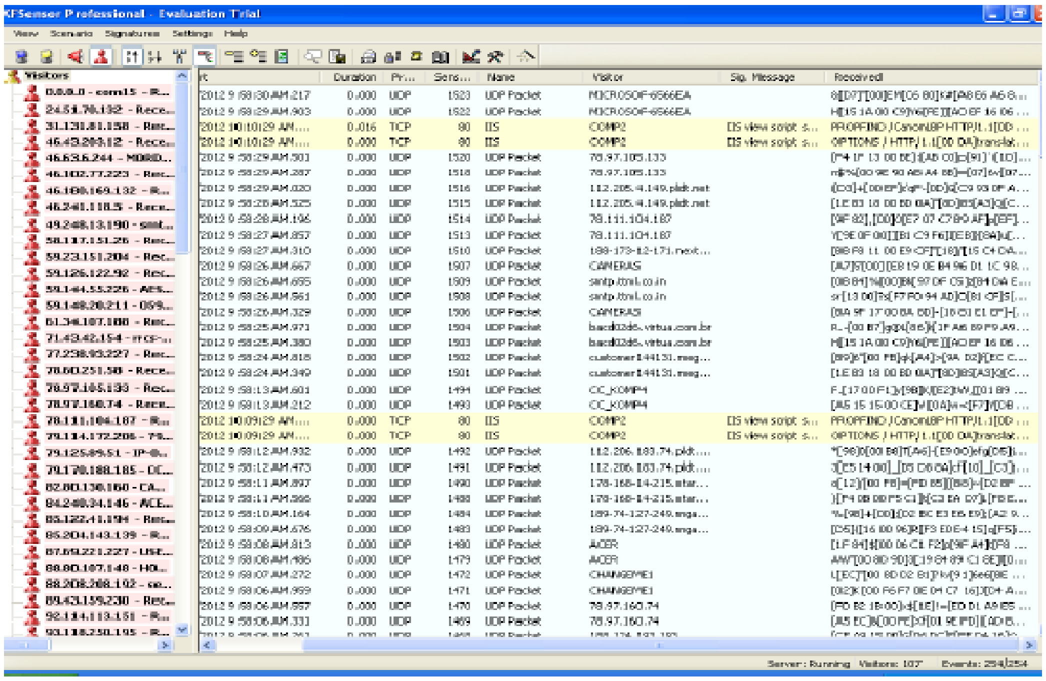

KFSensor Professional - Evaluation Trial

File View Scenario Signatures Settings Help

- Visitors
  - 178.76.252.26 - Recen...
  - 122.45.101.67 - Recen...
  - 176.73.49.60 - Recent ...
  - 192.168.1.1 - Recent A...
  - 192.168.1.51 - BTPS-PC...
  - 192.168.1.52 - CIVILDE...
  - 192.168.1.53 - FRONTO...
  - 192.168.1.58 - COMPUT...
  - 192.168.1.62 - COMP11...
  - 192.168.1.65 - ELECTRI...
  - 192.168.1.70 - COMP2 ...
  - 192.168.1.73 - COMP1 ...
  - 192.168.1.114 - com15 ...
  - 192.168.56.1 - COM15 ...
  - 222.107.67.174 - Rece...

ID	Start	Duration	Pr...	Sens...	Name	Visitor	Sig. Message	Received
44	3/1/2012 9:56:17 AM.398	0.000	UDP	1222	UDP Packet:	222.107.67.174		[0E 01 15 00 02]P[A
43	3/1/2012 9:56:17 AM.177	0.000	UDP	1224	UDP Packet:	178.76.252.26		Cu.[00 0D 1E E3]>$[
42	3/1/2012 9:56:16 AM.895	0.000	UDP	1223	UDP Packet:	178.76.252.26		[CA C1]}[00 F7 B8 E
41	3/1/2012 9:57:08 AM.968	0.000	UDP	67	DHCP	BTPS-PC		DHCP: Boot Request:
40	3/1/2012 9:56:16 AM.211	0.000	UDP	1217	UDP Packet:	222.107.67.174		[D7 0D 00 D6 04 EF]
39	3/1/2012 9:56:55 AM.375	0.000	UDP	138	NBT Datagram ...	com15		NBT DGRAM Packet:
38	3/1/2012 9:56:16 AM.147	0.000	UDP	1220	UDP Packet:	176.73.49.60		[E1 96 15 00]W[04 9
37	3/1/2012 9:56:15 AM.821	0.000	UDP	1219	UDP Packet:	176.73.49.60		[BA 9F 17 00 BA 8D]
36	3/1/2012 9:56:29 AM.125	0.000	TCP	80	IIS	COMP2	IIS view script s...	PROPFIND /CanonLB
35	3/1/2012 9:56:29 AM.125	0.000	TCP	80	IIS	COMP2	IIS view script s...	OPTIONS / HTTP/1.1
34	3/1/2012 9:56:15 AM.262	0.000	UDP	1218	UDP Packet:	122.45.101.67		[80]{[1F 00 02 F5][
33	3/1/2012 9:56:15 AM.049	0.000	UDP	1216	UDP Packet:	122.45.101.67		6[19 00]M[AD C9]H[
32	3/1/2012 9:56:14 AM.652	0.000	UDP	1215	UDP Packet:	122.45.101.67		[02]K [00 F6 F7 0E 0
31	3/1/2012 9:55:54 AM.406	0.000	UDP	138	NBT Datagram ...	ELECTRICALDEPT		NBT DGRAM Packet:
30	3/1/2012 9:55:29 AM.093	0.016	TCP	80	IIS	COMP2	IIS view script s...	PROPFIND /CanonLB
29	3/1/2012 9:55:29 AM.046	0.000	TCP	80	IIS	COMP2	IIS view script s...	OPTIONS / HTTP/1.1
28	3/1/2012 9:55:28 AM.068	0.000	UDP	68	DHCP Client	192.168.1.1		[02 01 06 00 EF 0E]/
27	3/1/2012 9:55:28 AM.078	0.000	UDP	67	DHCP	com15		DHCP: Boot Request:
26	3/1/2012 9:54:40 AM.656	0.000	UDP	138	NBT Datagram ...	BTPS-PC		NBT DGRAM Packet:
25	3/1/2012 9:54:29 AM.015	0.000	TCP	80	IIS	COMP2	IIS view script s...	PROPFIND /CanonLB
24	3/1/2012 9:54:29 AM.015	0.000	TCP	80	IIS	COMP2	IIS view script s...	OPTIONS / HTTP/1.1
23	3/1/2012 9:54:11 AM.343	0.000	UDP	138	NBT Datagram ...	COM15		NBT DGRAM Packet:
22	3/1/2012 9:53:28 AM.968	0.000	TCP	80	IIS	COMP2	IIS view script s...	PROPFIND /CanonLB
21	3/1/2012 9:53:28 AM.968	0.000	TCP	80	IIS	COMP2	IIS view script s...	OPTIONS / HTTP/1.1
20	3/1/2012 9:53:22 AM.328	0.000	UDP	138	NBT Datagram ...	COMPUTER14		NBT DGRAM Packet:
19	3/1/2012 9:53:09 AM.093	0.000	UDP	138	NBT Datagram ...	FRONTOFFICEPC		NBT DGRAM Packet:
18	3/1/2012 9:52:56 AM.453	0.000	UDP	138	NBT Datagram ...	COMP1		NBT DGRAM Packet:
17	3/1/2012 9:52:54 AM.656	0.000	UDP	138	NBT Datagram ...	COM15		NBT DGRAM Packet:
16	3/1/2012 9:52:54 AM.609	0.000	UDP	138	NBT Datagram ...	com15		NBT DGRAM Packet:
15	3/1/2012 9:52:42 AM.046	0.000	UDP	68	DHCP Client	192.168.1.1		[02 01 06 00 00]k[0
14	3/1/2012 9:52:46 AM.234	0.000	UDP	67	DHCP	com15		DHCP: Boot Request:
13	3/1/2012 9:52:41 AM.734	0.000	UDP	138	NBT Datagram ...	CIVILDEPT		NBT DGRAM Packet:
12	3/1/2012 9:52:38 AM.750	0.000	UDP	138	NBT Datagram ...	BTPS-PC		NBT DGRAM Packet:
11	3/1/2012 9:52:31 AM.078	0.000	UDP	67	DHCP	BTPS-PC		DHCP: Boot Request:
10	3/1/2012 9:52:28 AM.953	0.000	TCP	80	IIS	COMP2	IIS view script s...	PROPFIND /CanonLB

Server: Running | Visitors: 15 | Events: 44|44



```
0.0.0.0 - com15 - R...
24.51.76.132 - Rece...
31.131.81.158 - Rec...
46.43.209.12 - Rece...
46.63.6.244 - MORD...
46.102.77.223 - Rec...
46.180.169.132 - R...
46.241.118.5 - Rece...
49.248.13.190 - smt...
58.137.151.26 - Rec...
59.23.151.204 - Rec...
59.126.122.92 - Rec...
59.144.55.226 - AES...
59.148.20.211 - 069...
61.34.107.100 - Rec...
71.43.42.154 - rrCF-...
77.238.93.227 - Rec...
78.60.251.98 - Rece...
78.97.105.133 - Rec...
78.97.160.74 - Rece...
78.111.104.187 - R...
79.114.172.206 - 79...
79.125.89.51 - IP-0...
79.170.188.185 - DC...
82.80.130.160 - CA...
84.240.34.146 - ACE...
85.122.41.194 - Rec...
85.204.143.139 - R...
87.69.221.227 - USE...
88.80.107.148 - HO...
88.208.208.192 - se...
89.43.159.230 - Rec...
92.114.113.151 - R...
93.118.250.195 - R...
```

rt	Duration	Pr...	Sens...	Name	Visitor	Sig. Message	Received
2012 9:58:30 AM.217	0.000	UDP	1523	UDP Packet	MICROSOF-6566EA		8[D7][00]EM[C6 80]K#[A8 E6 A6 8...
2012 9:58:29 AM.903	0.000	UDP	1522	UDP Packet	MICROSOF-6566EA		H[15 1A 00 C9]%[FE][AC EF 16 06 ...
2012 10:10:29 AM....	0.016	TCP	80	IIS	COMP2	IIS view script s...	PROPFIND /Canon%BP HTTP/1.1[0D ...
2012 10:10:29 AM....	0.000	TCP	80	IIS	COMP2	IIS view script s...	OPTIONS / HTTP/1.1[0D 0A]translat...
2012 9:58:29 AM.501	0.000	UDP	1520	UDP Packet	78.97.105.133		[F4 1F 13 00 BE][A5 C0][91]'([1D]...
2012 9:58:29 AM.287	0.000	UDP	1518	UDP Packet	78.97.105.133		[illegible]
2012 9:58:29 AM.020	0.000	UDP	1516	UDP Packet	112.205.4.149.pldt.net		[illegible]
2012 9:58:28 AM.525	0.000	UDP	1515	UDP Packet	112.205.4.149.pldt.net		[1E 83 18 00 80 0A][8D][83[A3]...
2012 9:58:28 AM.196	0.000	UDP	1514	UDP Packet	78.111.104.187		[9F 82],[D0][E7 07 C7 80 AF][EF]...
2012 9:58:27 AM.857	0.000	UDP	1513	UDP Packet	78.111.104.187		Y[9E 0F 00][B1 C9 F6][E8][8A]...
2012 9:58:27 AM.310	0.000	UDP	1510	UDP Packet	188-173-82-171.next...		[8B F8 11 00 E9 CF][18][15 C4 DA...
2012 9:58:26 AM.667	0.000	UDP	1507	UDP Packet	CAMERAS		[A7][5][00][E8 19 0E 84 96 D1 1C 98...
2012 9:58:26 AM.655	0.000	UDP	1509	UDP Packet	smtp.tmil.co.in		[0B 84]%[00][8A[97 DF C5][84 DA E...
2012 9:58:26 AM.561	0.000	UDP	1508	UDP Packet	smtp.tmil.co.in		s[13 00]%[F7 F0 94 AD][81 CF][5[...
2012 9:58:26 AM.329	0.000	UDP	1506	UDP Packet	CAMERAS		[8A 9F 17 00 8A 00]-[18 C3 E1 EF]-[...
2012 9:58:25 AM.971	0.000	UDP	1504	UDP Packet	bacd02d6.virtua.com.br		[illegible]
2012 9:58:25 AM.380	0.000	UDP	1503	UDP Packet	bacd02d6.virtua.com.br		H[15 1A 00 C9][FE][AC EF 16 D6 ...
2012 9:58:24 AM.818	0.000	UDP	1502	UDP Packet	customer144131.meg...		[illegible]
2012 9:58:24 AM.349	0.000	UDP	1501	UDP Packet	customer144131.meg...		[1E 83 18 00 80 0A][8D][85[A3][C...
2012 9:58:13 AM.601	0.000	UDP	1494	UDP Packet	CC_KOMP4		F.[17 00 F1][98][E2]... [01 89 ...
2012 9:58:13 AM.212	0.000	UDP	1493	UDP Packet	CC_KOMP4		[A5 15 15 00 CE][0A][F7][D8 ...
2012 10:09:29 AM....	0.000	TCP	80	IIS	COMP2	IIS view script s...	PROPFIND /Canon%BP HTTP/1.1[0D ...
2012 10:09:29 AM....	0.000	TCP	80	IIS	COMP2	IIS view script s...	OPTIONS / HTTP/1.1[0D 0A]translat...
2012 9:58:12 AM.932	0.000	UDP	1492	UDP Packet	112.206.183.74.pldt....		[illegible]
2012 9:58:12 AM.473	0.000	UDP	1491	UDP Packet	112.206.183.74.pldt....		[E5 14 00]_[D5 D6 0A][10]_[C3]...
2012 9:58:11 AM.897	0.000	UDP	1490	UDP Packet	178-168-84-215.star...		[illegible]
2012 9:58:11 AM.386	0.000	UDP	1488	UDP Packet	178-168-84-215.star...		[illegible]
2012 9:58:10 AM.164	0.000	UDP	1484	UDP Packet	189-74-127-249.mega...		[illegible]
2012 9:58:09 AM.676	0.000	UDP	1483	UDP Packet	189-74-127-249.mega...		[illegible]
2012 9:58:08 AM.813	0.000	UDP	1480	UDP Packet	ACER		[1F 84][00 06 C8 F2][9F A4][F8 ...
2012 9:58:08 AM.486	0.000	UDP	1479	UDP Packet	ACER		[illegible]
2012 9:58:07 AM.272	0.000	UDP	1472	UDP Packet	CHANGEME1		[illegible]
2012 9:58:06 AM.959	0.000	UDP	1471	UDP Packet	CHANGEME1		[82]K[00 F6 F7 0E 04 C7 16][D4 A...
2012 9:58:06 AM.557	0.000	UDP	1470	UDP Packet	78.97.160.74		[FD 82 18 00][8E]![ED D1 A9 E5 ...
2012 9:58:06 AM.331	0.000	UDP	1469	UDP Packet	78.97.160.74		[A5 EC][00 FE][01 9E FD][AD B...

```
Server: Running   Visitors: 107   Events: 254/254
```

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.

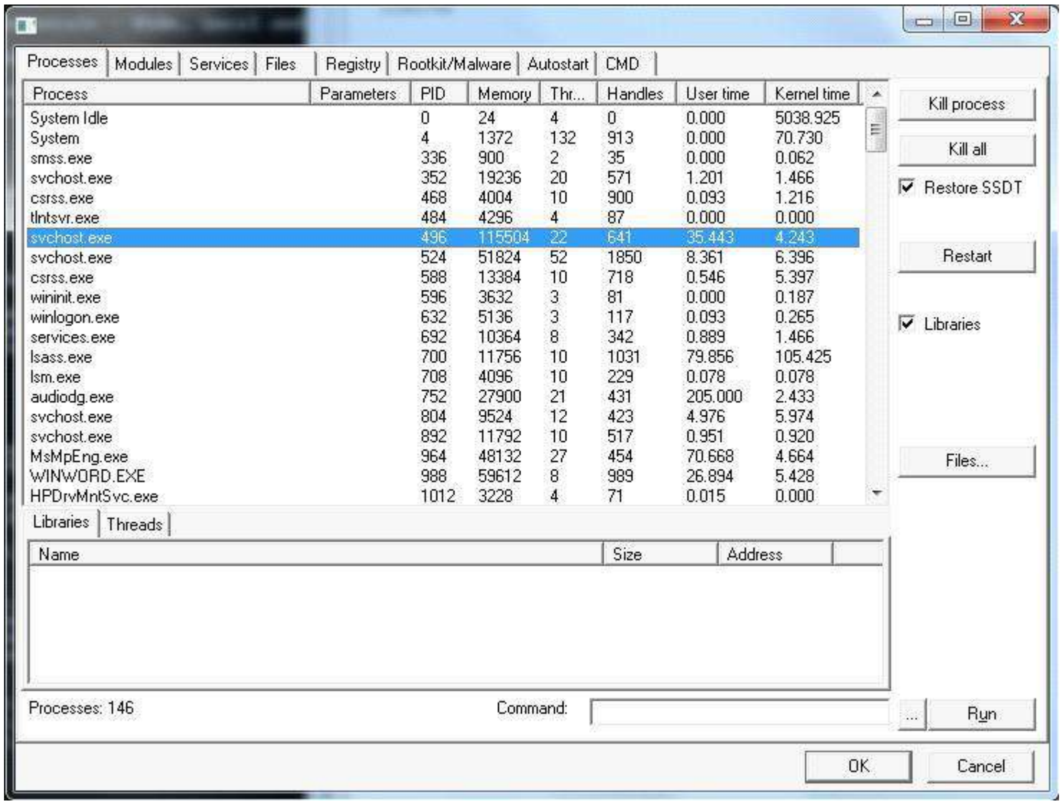
STEP-7: Registry displays Hkey_Current_user and Hkey_Local_Machine.

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:

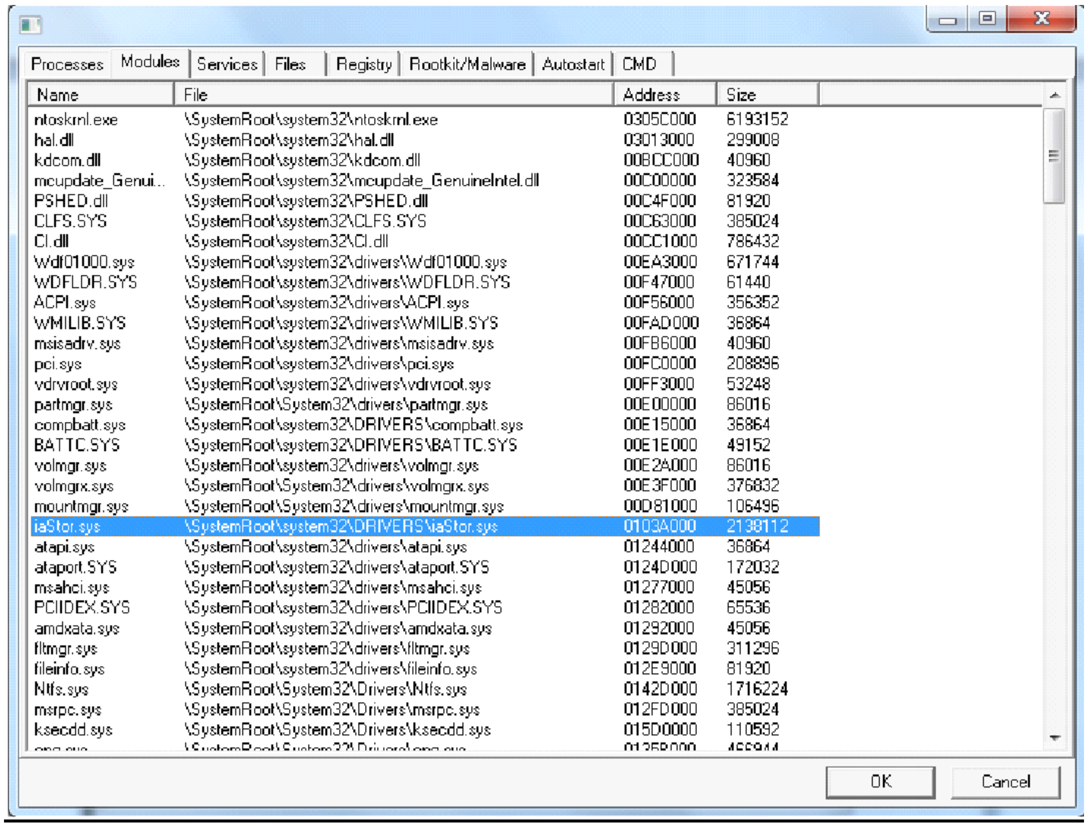


Process	Parameters	PID	Memory	Thr...	Handles	User time	Kernel time
System Idle		0	24	4	0	0.000	5038.925
System		4	1372	132	913	0.000	70.730
smss.exe		336	900	2	35	0.000	0.062
svchost.exe		352	19236	20	571	1.201	1.466
csrss.exe		468	4004	10	900	0.093	1.216
tlntsvr.exe		484	4296	4	87	0.000	0.000
svchost.exe		496	115504	22	641	35.443	4.243
svchost.exe		524	51824	52	1850	8.361	6.396
csrss.exe		588	13384	10	718	0.546	5.397
wininit.exe		596	3632	3	81	0.000	0.187
winlogon.exe		632	5136	3	117	0.093	0.265
services.exe		692	10364	8	342	0.889	1.466
lsass.exe		700	11756	10	1031	79.856	105.425
lsm.exe		708	4096	10	229	0.078	0.078
audiodg.exe		752	27900	21	431	205.000	2.433
svchost.exe		804	9524	12	423	4.976	5.974
svchost.exe		892	11792	10	517	0.951	0.920
MsMpEng.exe		964	48132	27	454	70.668	4.664
WINWORD.EXE		988	59612	8	989	26.894	5.428
HPDrvMntSvc.exe		1012	3228	4	71	0.015	0.000

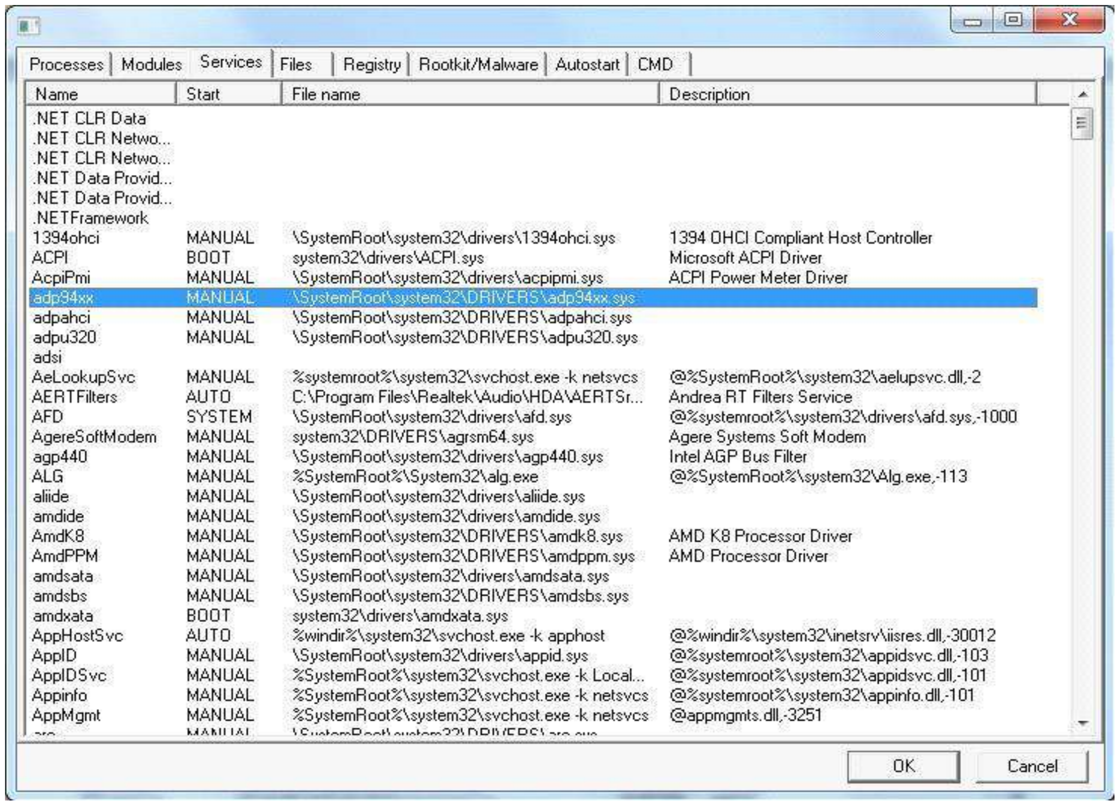
Name	Size	Address

Processes: 146

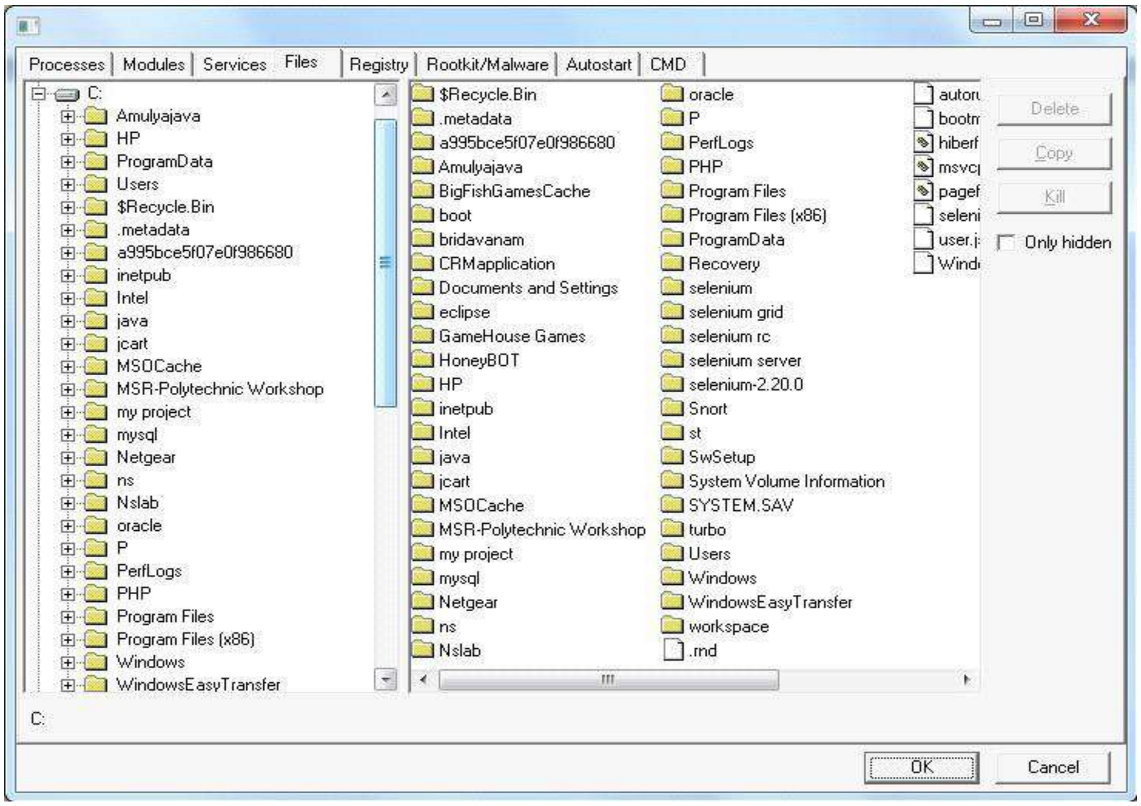
Command:



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
mcupdate_Genui...	\SystemRoot\system32\mcupdate_GenuineIntel.dll	00C00000	323584
PSHED.dll	\SystemRoot\system32\PSHED.dll	00C4F000	81920
CLFS.SYS	\SystemRoot\system32\CLFS.SYS	00C63000	385024
CI.dll	\SystemRoot\system32\CI.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	00FAD000	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



Name	Start	File name	Description
.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



C:
Amulyajava
HP
ProgramData
Users
$Recycle.Bin
.metadata
a995bce5f07e0f986680
inetpub
Intel
java
jcart
MSOCache
MSR-Polytechnic Workshop
my project
mysql
Netgear
ns
Nslab
oracle
P
PerfLogs
PHP
Program Files
Program Files (x86)
Windows
WindowsEasyTransfer

$Recycle.Bin
.metadata
a995bce5f07e0f986680
Amulyajava
BigFishGamesCache
boot
bridavanam
CRMapplication
Documents and Settings
eclipse
GameHouse Games
HoneyBOT
HP
inetpub
Intel
java
jcart
MSOCache
MSR-Polytechnic Workshop
my project
mysql
Netgear
ns
Nslab
oracle
P
PerfLogs
PHP
Program Files
Program Files (x86)
ProgramData
Recovery
selenium
selenium grid
selenium rc
selenium server
selenium-2.20.0
Snort
st
SwSetup
System Volume Information
SYSTEM.SAV
turbo
Users
Windows
WindowsEasyTransfer
workspace
.rnd
autoru
bootm
hiberf
msvcp
pagef
seleni
user.j
Wind

C:

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.
- ✓ 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.

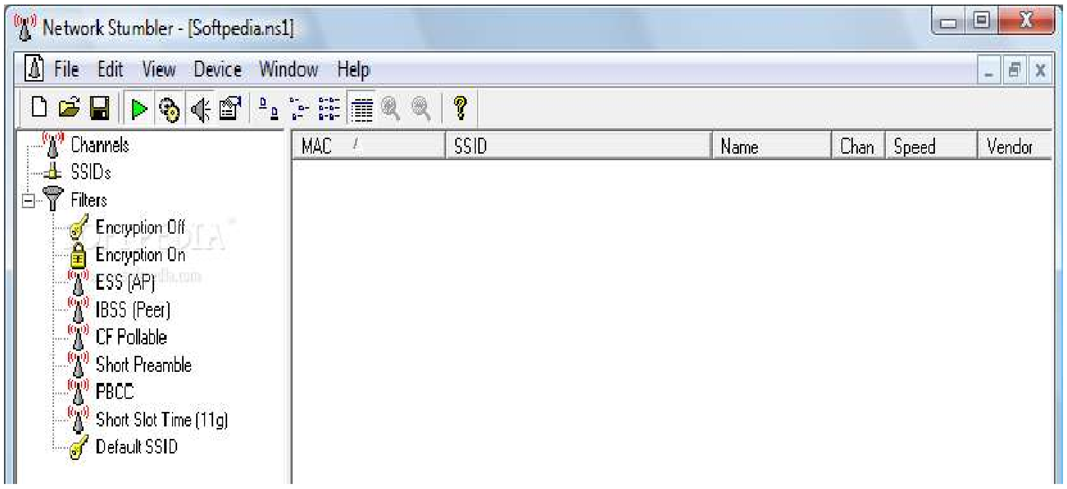
STEP-6: MAC assigned to Wireless Access Point is displayed on right hand pane.

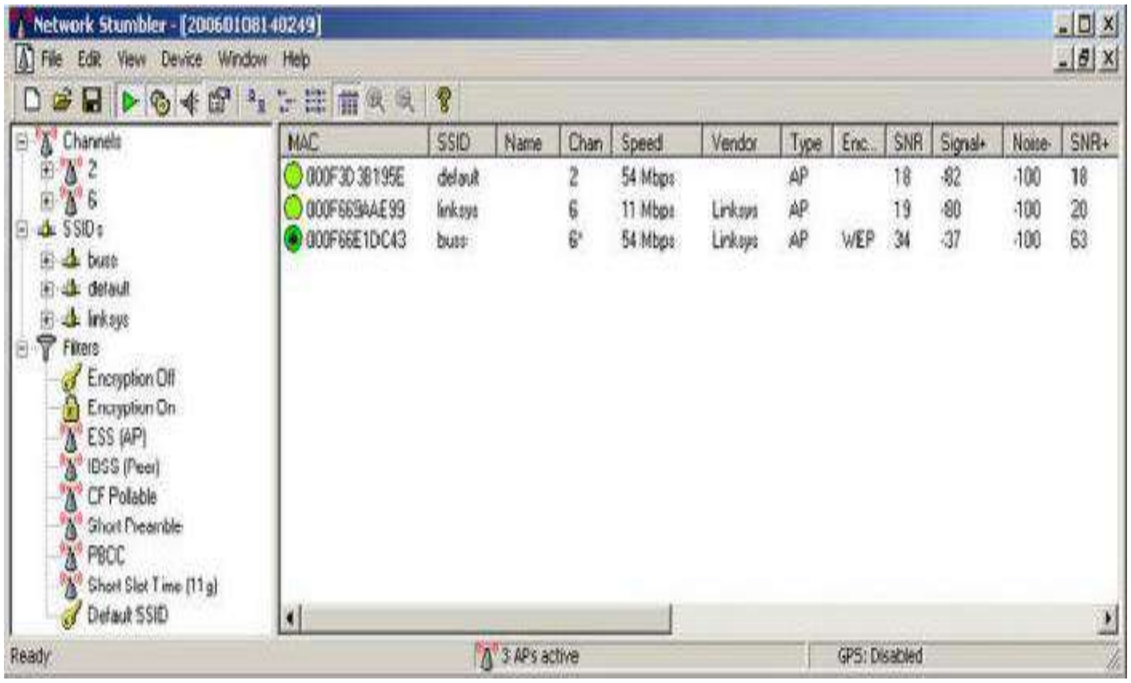
STEP-7: The next column displays the Access points Service Set Identifier[SSID] which is useful to crack the password.

STEP-8: To decrypt use WireShark tool by selecting Edit ☐ preferences ☐ IEEE 802.11.

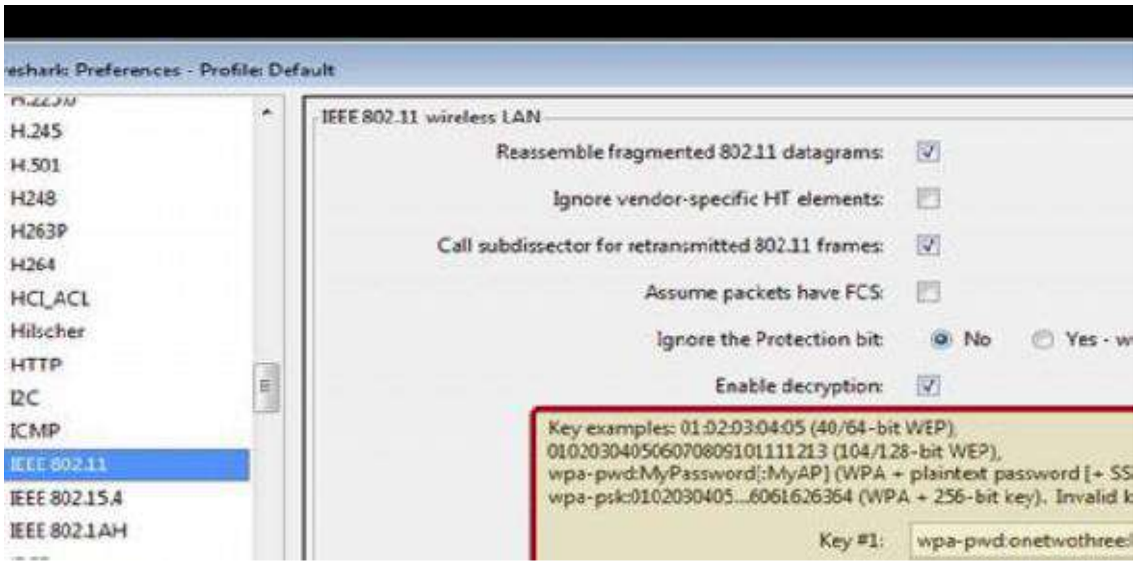
STEP-9: Enter the WEP keys as a string of hexadecimal numbers as A1B2C3D4E5.

SCREENSHOTS:





MAC	SSID	Name	Chan	Speed	Vendor	Type	Enc...	SNR	Signal+	Noise-	SNR+
000F3D38195E	default		2	54 Mbps		AP		18	-82	-100	18
000F669AAE99	linksys		6	11 Mbps	Linksys	AP		19	-80	-100	20
000F66E1DC43	buss		6*	54 Mbps	Linksys	AP	WEP	34	-37	-100	63



eshark: Preferences - Profile: Default

IEEE 802.11 wireless LAN

Reassemble fragmented 802.11 datagrams: ✓

Ignore vendor-specific HT elements: ☐

Call subdissector for retransmitted 802.11 frames: ✓

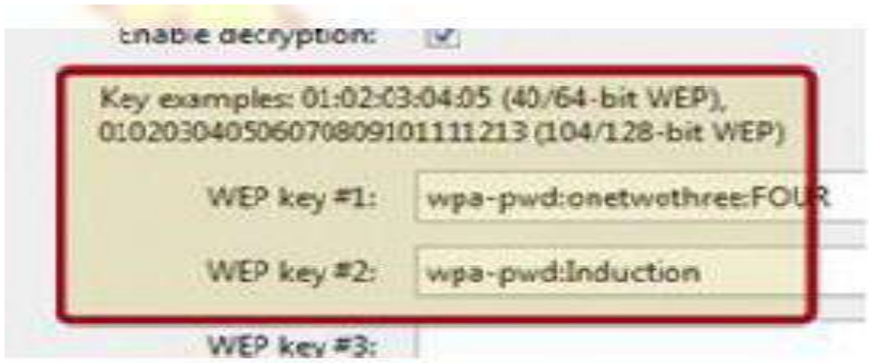
Assume packets have FCS: ☐

Ignore the Protection bit: [✓ No] Yes - w

Enable decryption: ✓

Key examples: 01:02:03:04:05 (40/64-bit WEP),
010203040506070809101111213 (104/128-bit WEP),
wpa-pwd:MyPassword[:MyAP] (WPA + plaintext password [+ SS
wpa-psk:0102030405...6061626364 (WPA + 256-bit key). Invalid k

Key #1: wpa-pwd:onetwothree:



Enable decryption: ✓

Key examples: 01:02:03:04:05 (40/64-bit WEP),
010203040506070809101111213 (104/128-bit WEP)

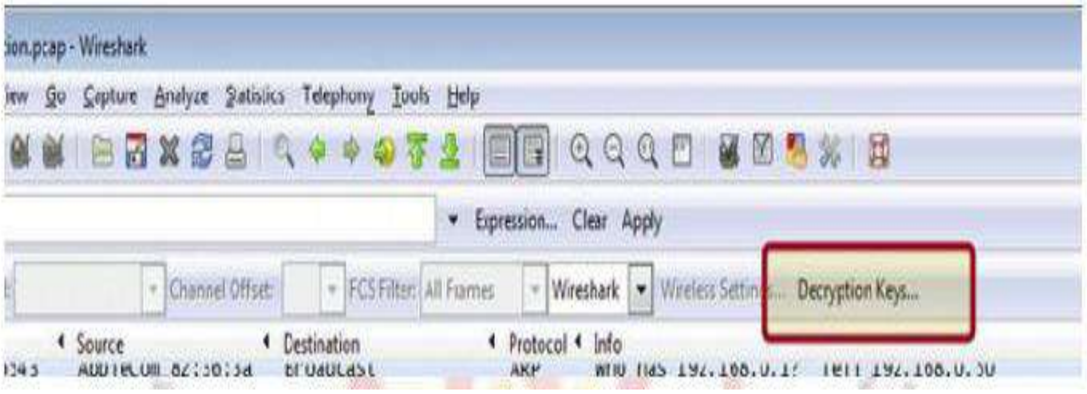
WEP key #1: wpa-pwd:onetwothree:FOUR

WEP key #2: wpa-pwd:Induction

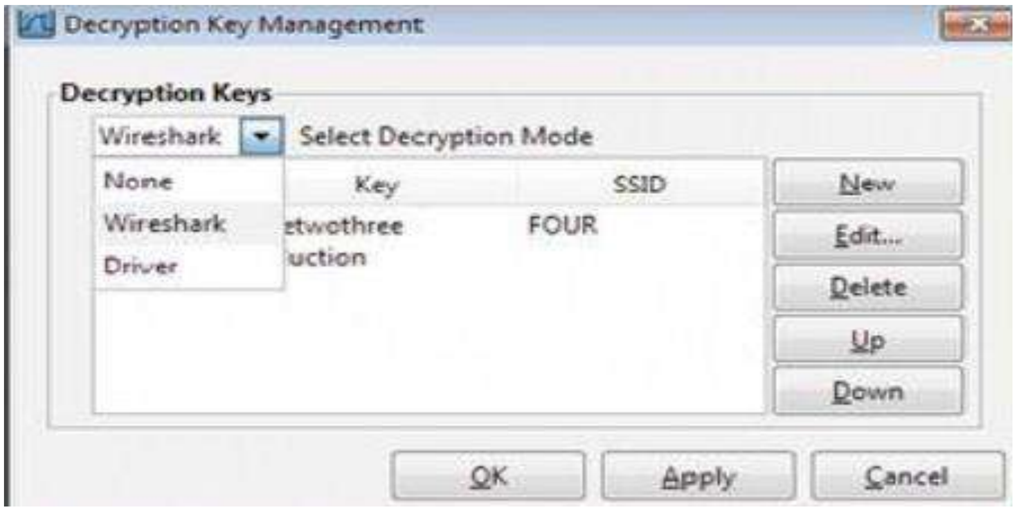
WEP key #3:

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:


ion.pcap - Wireshark
iew Go Capture Analyze Statistics Telephony Tools Help
Expression... Clear Apply
Channel Offset: FCS Filter: All Frames Wireshark Wireless Settings... Decryption Keys...
Source Destination Protocol Info

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



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

WORKING WITH SNORT TOOL TO DEMONSTRATE INTRUSION DETECTION SYSTEM

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
- ✓ 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 cnfigure snort to run in its most basic NIDS form, logging packets that trigger rules specifies in the snort.conf.

PROCEDURE:

STEP-1: Sniffer mode☐ snort –v ☐ Print out the TCP/IP packets header on the screen.

STEP-2: Snort –vd ☐ Show the TCP/IP ICMP header with application data in transit.

STEP-3: 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.

STEP-4: 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.

STEP-5: snort –l c:\log –b ☐ 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.

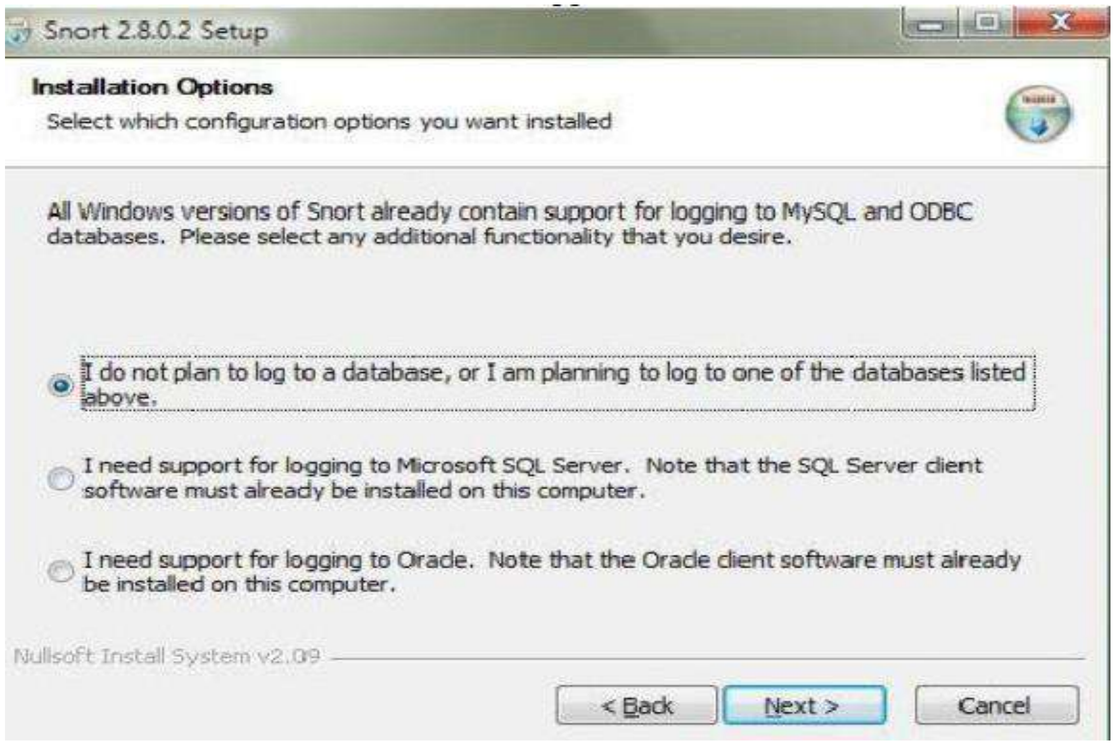
STEP-10: Skip the WinPcap driver installation.

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 ☐ path and variable value ☐ c:\snort\bin.

STEP-13: Click OK button and then close all dialog boxes. Open command prompt and type the following commands:

INSTALLATION PROCESS :


Snort 2.8.0.2 Setup

Installation Options

Select which configuration options you want installed

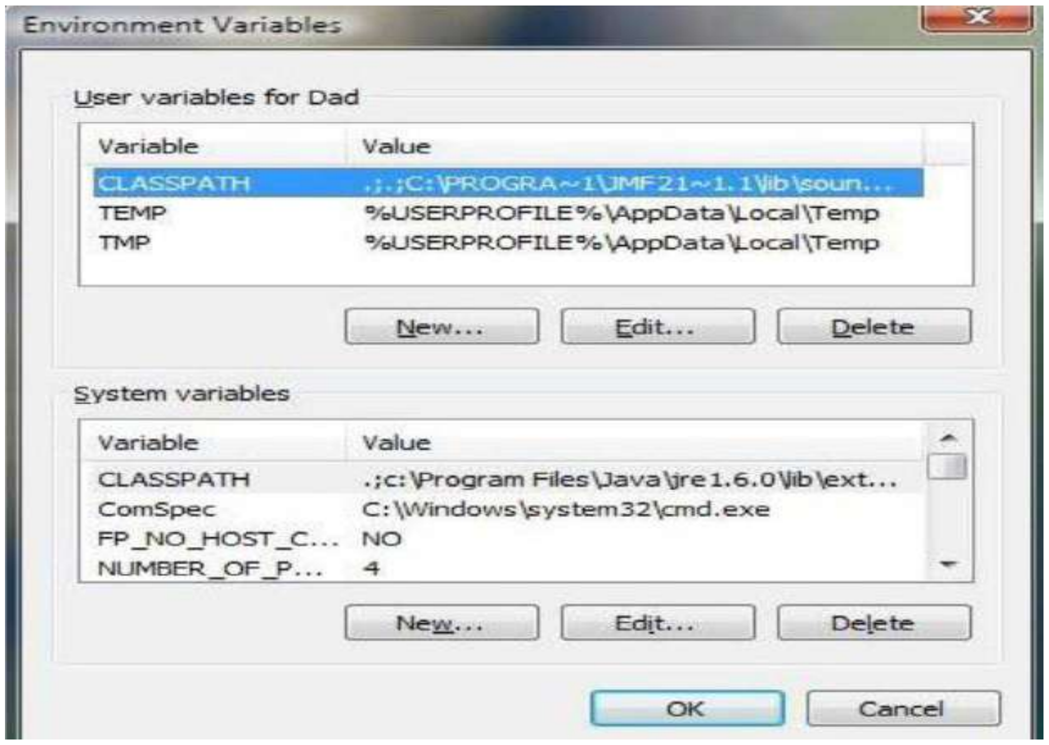
All Windows versions of Snort already contain support for logging to MySQL and ODBC databases. Please select any additional functionality that you desire.

[✓ I do not plan to log to a database, or I am planning to log to one of the databases listed above.]

I need support for logging to Microsoft SQL Server. Note that the SQL Server client software must already be installed on this computer.

I need support for logging to Oracle. Note that the Oracle client software must already be installed on this computer.

Nullsoft Install System v2.09

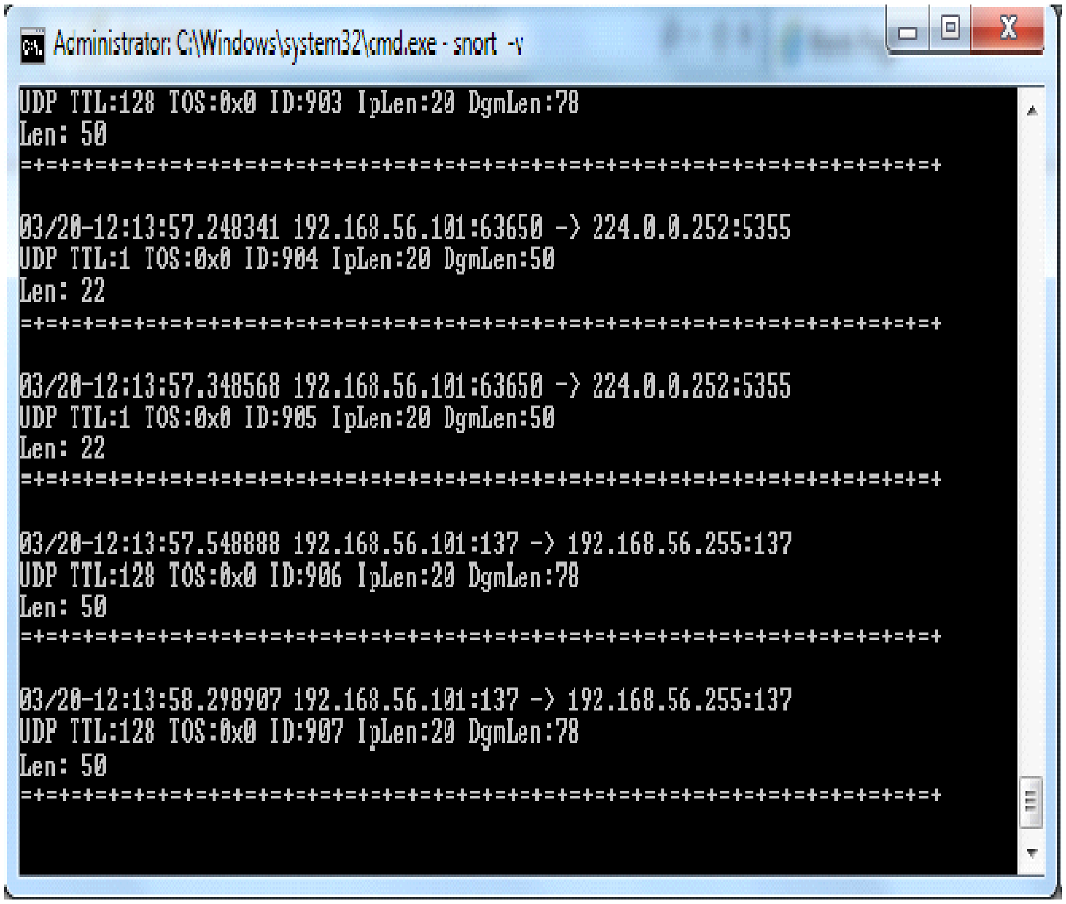

Environment Variables

User variables for Dad

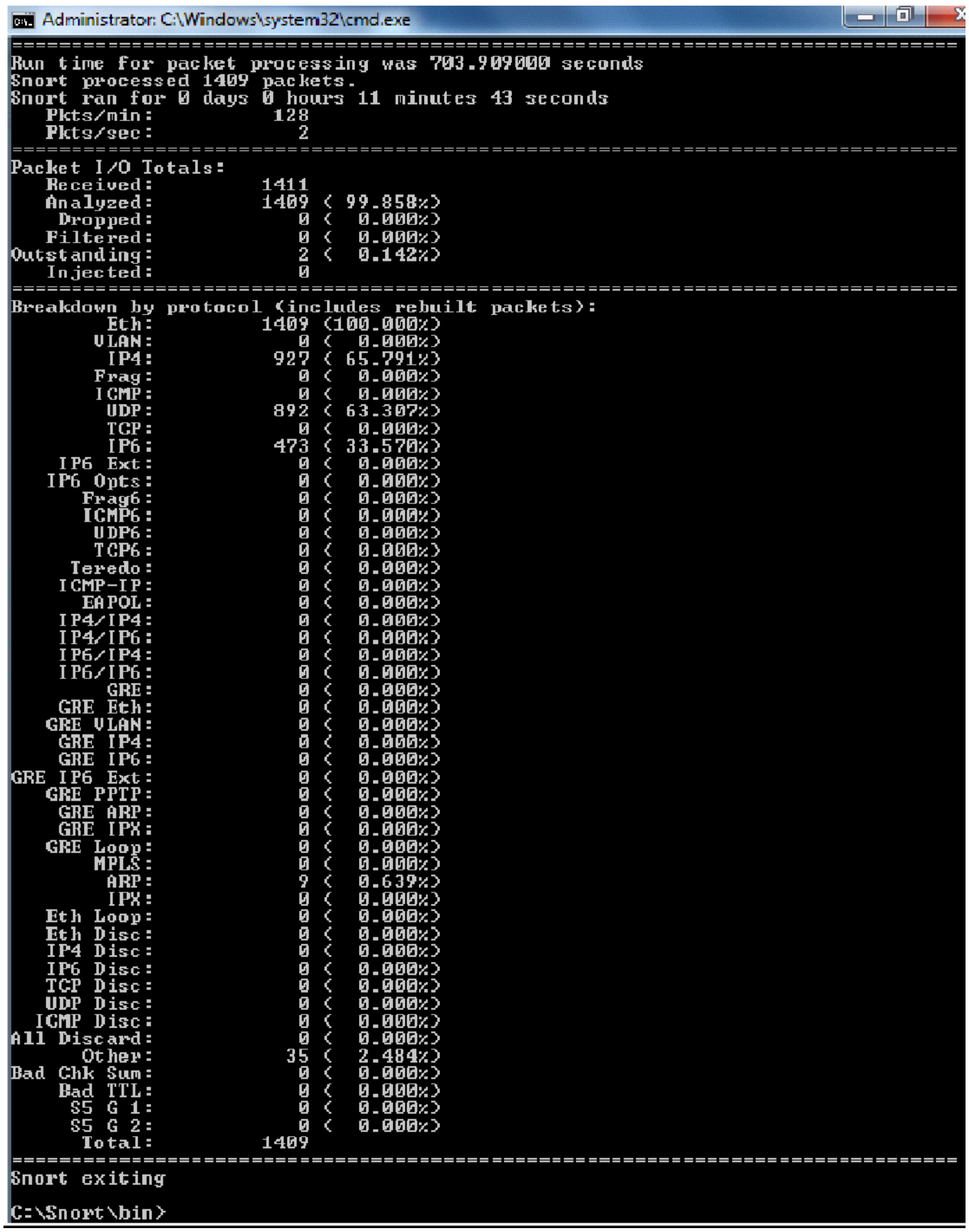
Variable	Value
CLASSPATH	.;.;C:\PROGRA~1\JMF21~1.1\lib\soun...
TEMP	%USERPROFILE%\AppData\Local\Temp
TMP	%USERPROFILE%\AppData\Local\Temp

System variables

Variable	Value
CLASSPATH	.;c:\Program Files\Java\jre1.6.0\lib\ext...
ComSpec	C:\Windows\system32\cmd.exe
FP_NO_HOST_C...	NO
NUMBER_OF_P...	4



```
UDP TTL:128 TOS:0x0 ID:903 IpLen:20 DgmLen:78
Len: 50
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

03/20-12:13:57.248341 192.168.56.101:63650 -> 224.0.0.252:5355
UDP TTL:1 TOS:0x0 ID:904 IpLen:20 DgmLen:50
Len: 22
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

03/20-12:13:57.348568 192.168.56.101:63650 -> 224.0.0.252:5355
UDP TTL:1 TOS:0x0 ID:905 IpLen:20 DgmLen:50
Len: 22
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

03/20-12:13:57.548888 192.168.56.101:137 -> 192.168.56.255:137
UDP TTL:128 TOS:0x0 ID:906 IpLen:20 DgmLen:78
Len: 50
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

03/20-12:13:58.298907 192.168.56.101:137 -> 192.168.56.255:137
UDP TTL:128 TOS:0x0 ID:907 IpLen:20 DgmLen:78
Len: 50
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```



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/sec:            2
===============================================================================
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%)
       VLAN:            0 (  0.000%)
        IP4:          927 ( 65.791%)
       Frag:            0 (  0.000%)
       ICMP:            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 (  0.000%)
     Teredo:            0 (  0.000%)
    ICMP-IP:            0 (  0.000%)
      EAPOL:            0 (  0.000%)
    IP4/IP4:            0 (  0.000%)
    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 VLAN:            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%)
       MPLS:            0 (  0.000%)
        ARP:            9 (  0.639%)
        IPX:            0 (  0.000%)
   Eth Loop:            0 (  0.000%)
   Eth Disc:            0 (  0.000%)
   IP4 Disc:            0 (  0.000%)
   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%)
     S5 G 1:            0 (  0.000%)
     S5 G 2:            0 (  0.000%)
      Total:         1409
===============================================================================
Snort exiting

C:\Snort\bin>
```

RESULT:

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