EX NO:1A **CAESAR CIPHER**

AIM:

To implement a program for encrypting a plain text and decrypting a cipher text using Caesar Cipher (shift cipher) substitution technique

ALGORITHM DESCRIPTION:

1. It is a type of substitution cipher in which each letter in the plaintext is replaced by a letter some fixed number of positions down the alphabet. For example, with a left shift of 3, D would be replaced by A, E would become B, and so on.

2. The method is named after Julius Caesar, who used it in his private correspondence.

3. The transformation can be represented by aligning two alphabets; the cipher alphabet is the plain alphabet rotated left or right by some number of positions.

4. The encryption can also be represented using modular arithmetic by first transforming the letters into numbers, according to the scheme, A = 0, B = 1, Z = 25.

5. Encryption of a letter x by a shift n can be described mathematically as, $E_n(x) = (x + n) \mod 26$

6. Decryption is performed similarly, $D_n(x) = (x - n) \mod 26$

PROGRAM:

```
import java.util.*;
class caesarCipher
{
public static String encode(String enc, int offset)
{
offset = offset % 26 + 26;
StringBuilder encoded = new
StringBuilder();
 for (char i :enc.toCharArray())
{
if (Character.isLetter(i))
{
if (Character.isUpperCase(i))
{
encoded.append((char) ('A' + (i - 'A' + offset) % 26 ));
}
else
{
encoded.append((char) ('a' + (i - 'a' + offset) % 26 ));
}
}
else
{
encoded.append(i);
}
}
return encoded.toString();
}
public static String decode(String enc, int offset)
{
return encode(enc, 26-offset);
}
public static void main (String[] args) throws java.lang.Exception
{
String msg = "Hello welcome to Security Laboratory";
System.out.println("simulation of Caesar Cipher");
 System.out.println("input message : " + msg);
System.out.printf( "encoded message : ");
System.out.println(caesarCipher.encode(msg, 12));
System.out.printf( "decoded message : ");
System.out.println(caesarCipher.decode(caesarCipher.encode(msg, 12),12));
}
}
```

Output:

```
stdin:
Standard input is empty
stdout:
simulation of Caesar Cipher
input message : Hello welcome to Security Laboratory
encoded message : Tqxxa iqxoayq fa Eqogdufk Xmnadmfadk
decoded message : Hello welcome to SecurityLaboratory
```

RESULT:

Thus the program was executed and verified successfully.

EX NO:1B **PLAY FAIR CIPHER**

AIM:

To implement a program to encrypt a plain text and decrypt a cipher text using play fair Cipher substitution technique.

ALGORITHM DESCRIPTION:

- The Playfair cipher uses a 5 by 5 table containing a key word or phrase.
- To generate the key table, first fill the spaces in the table with the letters of the keyword, then fill the remaining spaces with the rest of the letters of the alphabet in order (usually omitting "Q" to reduce the alphabet to fit; other versions put both "I" and "J" in the same space).
- The key can be written in the top rows of the table, from left to right, or in some other pattern, such as a spiral beginning in the upper-left-hand corner and ending in the centre.
- The keyword together with the conventions for filling in the 5 by 5 table constitutes the cipher key. To encrypt a message, one would break the message into diagrams (groups of2 letters) such that, for example, "HelloWorld" becomes "HE LL OW OR LD", and map them out on the key table. Then apply the following 4 rules, to each pair of letters in the plaintext:
- If both letters are the same (or only one letter is left), add an "X" after the first letter. Encrypt the new pair and continue. Some variants of Playfair use "Q" instead of "X", butany letter, itself uncommon as a repeated pair, will do.
- If the letters appear on the same row of your table, replace them with the letters to their immediate right respectively (wrapping around to the left side of the row if a letter in the original pair was on the right side of the row).
- If the letters appear on the same column of your table, replace them with the letters immediately below respectively (wrapping around to the top side of the column if a letterin the original pair was on the bottom side of the column).
- 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. The order is important – the first letter of the encrypted pair is the one that lies on the same row as the first letter of the plaintext pair.
- To decrypt, use the INVERSE (opposite) of the last 3 rules, and the 1st as-is (dropping any extra "X"s, or "Q"s that do not make sense in the final message when finished).

PROGRAM :

```
import java.awt.Point;
 import java.util.*;
 classplayfairCipher
{
private static char[][] charTable;
private static Point[] positions;
private static String prepareText(String s, boolean chgJtoI)
{
s = s.toUpperCase().replaceAll("[^A-Z]", "");
return chgJtoI ? s.replace("J", "I") : s.replace("Q", "");
}
private static void createTbl(String key, boolean chgJtoI)
{
charTable = new char[5][5];
positions = new Point[26];
String s = prepareText(key + "ABCDEFGHIJKLMNOPQRSTUVWXYZ", chgJtoI);
int len = s.length();
for (int i = 0, k = 0; i < len; i++)
{
char c = s.charAt(i);
if (positions[c - 'A'] == null)
{
charTable[k / 5][k % 5] = c;
positions[c - 'A'] = new Point(k % 5, k / 5);k++;
}
}
}
private static String codec(StringBuilder txt, int dir)
{
int len = txt.length();
for (int i = 0; i < len; i += 2)
{
char a = txt.charAt(i);
char b = txt.charAt(i + 1);
int row1 = positions[a - 'A'].y;
int row2 = positions[b - 'A'].y;
int col1 = positions[a - 'A'].x;
 int col2 = positions[b - 'A'].x;
if (row1 == row2)
{
```

```
col1 = (col1 + dir) % 5;
col2 = (col2 + dir) % 5;
}

else if (col1 == col2)
{
row1 = (row1 + dir) % 5;row2 = (row2 + dir) % 5;
}
else
{
int tmp = col1;
col1 = col2;

col2 = tmp;
}
txt.setCharAt(i, charTable[row1][col1]);
txt.setCharAt(i + 1, charTable[row2][col2]);
}
return txt.toString();
}
private static String encode(String s)
{
String Builder sb = new String Builder(s);
for (int i = 0; i < sb.length(); i += 2)
{
if (i == sb.length() - 1)
{
sb.append(sb.length() % 2 == 1 ? 'X' : "");
}
else if (sb.charAt(i) == sb.charAt(i + 1))
{
sb.insert(i + 1, 'X');
}

}
return codec(sb, 1);
}

private static String decode(String s)
{
return codec(new StringBuilder(s), 4);
}
```

```
public static void main (String[] args) throws java.lang.Exception
{
String key = "mysecretkey";
String txt = "CRYPTOLABS";
  /* make sure string length is even */
/* change J to I */

 boolean chgJtoI = true;
 createTbl(key, chgJtoI);
String enc = encode(prepareText(txt, chgJtoI));
System.out.println("simulation of Playfair Cipher");
 System.out.println("input message : " + txt);
System.out.println("encoded message : " + enc);
 System.out.println("decoded message : " + decode(enc));
}
}
```

OUTPUT:

stdin:

Standard input is empty
stdout:
simulation of Playfair Cipher

input message : CRYPTOLABS
encoded message : MBENKNPRKC
decoded message : CRYPTOLABS

RESULT:

Thus the program was executed and verified successfully.

EX NO: 1C **HILL CIPHER**

AIM:

To implement a program to encrypt and decrypt using the Hill ciphersubstitution technique

ALGORITHM DESCRIPTION:

- The Hill cipher is a substitution cipher invented by Lester S. Hill in 1929.

- Each letter is represented by a number modulo 26. To encrypt a message, each block of nletters is multiplied by an invertible n × n matrix, again 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).

- The cipher can, be adapted to an alphabet with any number of letters.

- All arithmetic just needs to be done modulo the number of letters instead of modulo 26.

PROGRAM :

```
import java.util.*;

classhillCipher

{
/* 3x3 key matrix for 3 characters at once *
/public static int[][] keymat = new int[][]
{ { 1, 2, 1 }, { 2, 3, 2 }, { 2, 2, 1 } };
/* key inverse matrix */
public static int[][] invkeymat = new int[][]
{ { -1, 0, 1 }, { 2, -1, 0 }, { -2, 2, -1 } };
public static String key = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
private static String encode(char a, char b, char c)
{
String ret = "";int x,y, z;
int posa = (int)a - 65;
int posb = (int)b - 65;
int posc = (int)c - 65;
x = posa * keymat[0][0] + posb * keymat[1][0] + posc * keymat[2][0];
y = posa * keymat[0][1] + posb * keymat[1][1] + posc * keymat[2][1];
z = posa * keymat[0][2] + posb * keymat[1][2] + posc * keymat[2][2];
a = key.charAt(x%26);
b = key.charAt(y%26);
c = key.charAt(z%26);
ret = "" + a + b + c;
return ret;
}
private static String decode(char a, char b, char c)
{
String ret = "";int x,y,z;
int posa = (int)a - 65;int posb = (int)b - 65;int posc = (int)c - 65;
x = posa * invkeymat[0][0]+ posb * invkeymat[1][0] + posc * invkeymat[2][0];
y = posa * invkeymat[0][1]+ posb * invkeymat[1][1] + posc * invkeymat[2][1];
z = posa * invkeymat[0][2]+ posb * invkeymat[1][2] + posc * invkeymat[2][2];
a = key.charAt((x%26<0) ? (26+x%26) : (x%26));
b = key.charAt((y%26<0) ? (26+y%26) : (y%26));
c = key.charAt((z%26<0) ? (26+z%26) : (z%26));
ret = "" + a + b + c;
return ret;
}
```

```
public static void main (String[] args) throws java.lang.Exception
{
String msg; String enc = "";String dec = "";

int n;
msg = ("SecurityLaboratory");
 System.out.println("simulation of Hill Cipher");System.out.println("input message : " +
     msg);
msg = msg.toUpperCase();
msg = msg.replaceAll("\\s", ""); /* remove spaces */n = msg.length() % 3;
/* append padding text X */if (n != 0)
{
for(int i = 1; i<= (3-n);i++)
{

msg+= 'X';
}
}
System.out.println("padded message  : " + msg);char[] pdchars = msg.toCharArray();
for (int i=0; i < msg.length(); i+=3)
{
enc += encode(pdchars[i], pdchars[i+1], pdchars[i+2]);
}
System.out.println("encoded message : " + enc);
char[] dechars = enc.toCharArray();
for (int i=0; i< enc.length(); i+=3)
{
dec += decode(dechars[i], dechars[i+1], dechars[i+2]);
}
System.out.println("decoded message : " + dec);
}
```

OUTPUT:

```
stdin:
Standard input is empty
stdout:
simulation of Hill Cipher
input message  : SecurityLaboratory
padded message : SECURITYLABORATORY
 encoded message : EACSDKLCAEFQDUKSXU
decoded message : SECURITYLABORATORY
```

RESULT:

Thus the program was executed and verified successfully.

EX NO:1D **VIGENERE CIPHER**

AIM:

To implement a program for encryption and decryption using vigenere cipher substitution technique

ALGORITHM DESCRIPTION:

- The Vigenere cipher is a method of encrypting alphabetic text by using a series ofdifferent Caesar ciphers based on the letters of a keyword.
- It is a simple form of poly alphabetic substitution.
- To encrypt, a table of alphabets can be used, termed a Vigenere square, or Vigenere table.
- It consists of the alphabet written out 26 times in different rows, each alphabet shifted cyclically to the left compared to the previous alphabet, corresponding to the 26 possible Caesar ciphers.
- At different points in the encryption process, the cipher uses a different alphabet fromone of the rows used.
- The alphabet at each point depends on a repeating keyword.

PROGRAM:

```
import java.util.*;
class vigenere Cipher
{
static String encode(String text, final String key)
{
String res = "";
text = text.toUpperCase();
for (int i = 0, j = 0; i < text.length(); i++)
{
char c = text.charAt(i); if (c <'A' || c > 'Z')
{
continue;
}
res += (char)((c + key.charAt(j) - 2 * 'A') % 26 + 'A');
j = ++j % key.length();
}
return res;
}
static String decode(String text, final String key)
{
String res = "";
text = text.toUpperCase();
for (int i = 0, j = 0; i < text.length(); i++)
 {
char c = text.charAt(i);
if (c < 'A' || c > 'Z')
{
continue;
}
res += (char)((c - key.charAt(j) + 26) % 26 + 'A');
j = ++j % key.length();
}
return res;
}
public static void main (String[] args) throws java.lang.Exception
{
String key = "VIGENERECIPHER";
String msg = "SecurityLaboratory";

System.out.println("simulation of Vigenere Cipher");
```

```
System.out.println("input message   : " + msg);
 String enc = encode(msg, key);
System.out.println("encoded message : " + enc);
System.out.println("decoded message : " + decode(enc, key));


}
}
```

OUTPUT:

stdin: Standard input is empty
stdout: simulation of Vigenere Cipher
input message : Security Laboratory
encoded message : NMIYEMKCNIQVVROWXC
decoded message : SECURITYLABORATORY

RESULT:

Thus the program was executed and verified successfully.

EX NO:1E **RAIL FENCE CIPHER**

AIM:

To implement a program for encryption and decryption using rail fence transposition technique.

ALGORITHM DESCRIPTION:

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

PROGRAM :

```
import java.util.*;
class rail fence Cipher Helper
{
int depth;
String encode(String msg, int depth) throws Exception
{
int r = depth;int l =
msg.length();
 int c= l/depth;int k = 0;
char mat[][] = new char[r][c];
String enc = "";
for (int i=0; i<c; i++)
{
for (int j=0; j<r; j++)
{
if (k != l)
{
mat[j][i] = msg.charAt(k++);
}
else
{
mat[j][i] = 'X';
}
}
}
for (int i=0; i<r; i++)
{
for (int j=0; j<c; j++)
{
enc += mat[i][j];
}
}
return enc;
}
String decode(String encmsg, int depth) throws Exception
{
int r = depth;
int l = encmsg.length();
int c = l/depth;
int k = 0;
```

```
char mat[][] = new char[r][c];

String dec = "";
for (int i=0; i<r; i++)
{
for (int j=0; j< c; j++)

{
mat[i][j] = encmsg.charAt(k++);
}
}
for (int i=0; i<c; i++)
{
for (int j=0; j< r; j++)
{
dec += mat[j][i];
}
}
return dec;
}
}
class railfenceCipher
{
public static void main (String[] args) throws java.lang.Exception
{
railfenceCipherHelper rf = new railfenceCipherHelper();
String msg, enc, dec;
msg = "hellorailfencecipher";
int depth = 2;
enc = rf.encode(msg, depth);
dec = rf.decode(enc, depth);
System.out.println("simulation of Railfence Cipher");
System.out.println("input message : " + msg);
 System.out.println("encoded message : " + enc);
 System.out.printf( "decoded message : " + dec);
}
}
```

OUTPUT:

stdin: Standard input is empty
stdout: simulation of Rail fence Cipher
input message : hellorailfencecipher
encoded message : hloaleccpeelrifneihr
decoded message : hellorailfencecipher

RESULT:

Thus the program was executed and verified successfully.

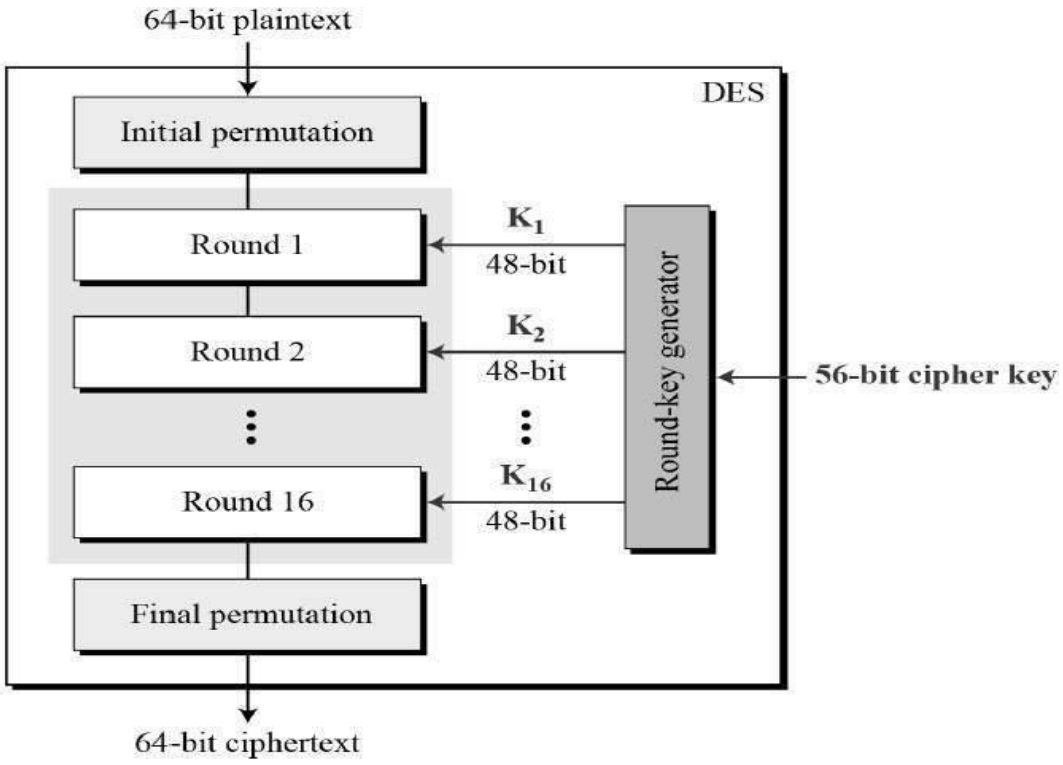
EX NO:2A DATA ENCRYPTION STANDARD (DES)

AIM:

To develop a program to implement Data Encryption Standard for encryption and decryption.

ALGORITHM DESCRIPTION:

- The Data Encryption Standard (DES) is a symmetric-key block cipher published bythe National Institute of Standards and Technology (NIST).
- DES is an implementation of a Feistel Cipher. It uses 16 round Feistel structure. The block size is 64-bit.
- Though, key length is 64-bit, DES has an effective key length of 56 bits, since 8 of the 64 bits of the key are not used by the encryption algorithm (function as check bits only).
- General Structure of DES is depicted in the following illustration



PROGRAM:

DES :-

```
import javax.swing.*;
import java.security.SecureRandom;
import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.SecretKey;
import javax.crypto.spec.SecretKeySpec;
import java.util.Random ;
class DES {
byte[] skey = new byte[1000];
String skeyString;
static byte[] raw;
String inputMessage,encryptedData,decryptedMessage;
public DES() {
try { generateSymmetricKey();
inputMessage=JOptionPane.showInputDialog(null,"Enter message to encrypt");
byte[] ibyte = inputMessage.getBytes();
byte[] ebyte=encrypt(raw, ibyte);
String encryptedData = new String(ebyte);
 System.out.println("Encrypted message "+encryptedData);
JOptionPane.showMessageDialog(null,"Encrypted Data "+"\n"+encryptedData);
byte[] dbyte= decrypt(raw,ebyte);
String decryptedMessage = new String(dbyte);
System.out.println("Decrypted message "+decryptedMessage);
JOptionPane.showMessageDialog(null,"Decrypted Data "+"\n"+decryptedMessage);
}
catch(Exception e)
 {
System.out.println(e);
}
}
void generateSymmetricKey()
 {
Try
 {
Random r = new Random();
intnum = 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
 {
KeyGeneratorkgen = KeyGenerator.getInstance("DES");
SecureRandomsr = SecureRandom.getInstance("SHA1PRNG");
sr.setSeed(seed);
kgen.init(56, sr);
SecretKeyskey = kgen.generateKey();
raw = skey.getEncoded();
return raw;
}

private static byte[] encrypt(byte[] raw, byte[] clear) throws Exception
{
SecretKeySpecskeySpec = 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
 {
SecretKeySpecskeySpec = 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:

RESULT:

Thus the program was executed and verified successfully.

EX NO:2B **RSA ALGORITHM**

AIM:

Develop a program to implement RSA algorithm for encryption and decryption. This cryptosystem is one the initial system. It remains most employed cryptosystem even today. The system was invented by three scholars Ron Rivest, Adi Shamir, and Len Adleman and hence, it is termed as RSA cryptosystem. The two aspects of the RSA cryptosystem, firstly generation of key pair and secondly encryption-decryption algorithms

ALGORITHM DESCRIPTION:

- Generation of RSA Key Pair
- Each person or a party who desires to participate in communication using encryption needs to generate a pair of keys, namely public key and private key.
- The process followed in the generation of keys is described below −
- Generate the RSA modulus (n)
- Select two large primes, p and q.
- Calculate n=p*q. For strong unbreakable encryption, let n be a large number, typically a minimum of 512 bits.
- Find Derived Number (e)
- Number e must be greater than 1 and less than (p − 1)(q − 1).
- There must be no common factor for e and (p − 1)(q − 1) except for 1. In other words two numbers e and (p – 1)(q – 1) are coprime.
- Form the public key
- The pair of numbers (n, e) form the RSA public key and is made public. Interestingly, though n is part of the public key, difficulty in factorizing a large prime number ensures that attacker cannot find in finite time the two primes (p & q) used to obtain n. This is strength of RSA.
- Generate the private key
- Private Key d is calculated from p, q, and e. For given n and e, there is unique number d.
- Number d is the inverse of e modulo (p - 1)(q – 1). This means that d is the number less than (p - 1)(q - 1) such that when multiplied by e, it is equal to 1 modulo (p - 1)(q - 1).
- This relationship is written mathematically as follows ed = 1 mod (p − 1)(q − 1)
- The Extended Euclidean Algorithm takes p, q, and e as input and gives d as output.

PROGRAM :

```
import java.math.BigInteger;
import java.util.Random;
import java.io.*;
class rsaAlg
{
private BigInteger p, q, n, phi, e, d;
 /* public key components */
private int bitLen = 1024;
private int blkSz = 256;
/* block size in bytes */private Random rand;
/* convert bytes to string */
private static String bytesToString(byte[] encrypted)
{
String str = "";
for (byte b : encrypted)
{
str += Byte.to String(b);
}
return str;
}
/* encrypt message */
public byte[] encrypt(byte[] msg)
{
return (new BigInteger(msg)).modPow(e, n).toByteArray();
}
/* decrypt message */
public byte[] decrypt(byte[] msg)
{
return (new BigInteger(msg)).modPow(d, n).toByteArray();
}
/* calculate public key components p, q, n, phi, e, d */public rsaAlg()
{
rand = new Random();
p = BigInteger.probablePrime(bitLen, rand);
q = BigInteger.probablePrime(bitLen, rand);
n = p.multiply(q);
phi = p.subtract(BigInteger.ONE).multiply(q.subtract(BigInteger.ONE));
e = BigInteger.probablePrime(bitLen/2, rand);
while (phi.gcd(e).compareTo(BigInteger.ONE) > 0 &&e.compareTo(phi) < 0)
{
```

```
e.add(BigInteger.ONE);
}
d = e.modInverse(phi);
}
public rsaAlg (BigInteger e, BigInteger d, BigInteger n)
{
this.e = e;this.d = d;this.n = n;


}
public static void main (String[] args) throws java.lang.Exception
{
rsaAlg rsaObj = new rsaAlg();
String msg = "Hello world! Security Laboratory";
 System.out.println("simulation of RSA algorithm");
System.out.println("message(string) : " + msg);
System.out.println("message(bytes): " +bytesToString(msg.getBytes()));
/* encrypt test message */
byte[] ciphertext = rsaObj.encrypt(msg.getBytes());
 System.out.println("ciphertext(bytes) : " + bytesToString(ciphertext));
/* decrypt ciphertext */
byte[] plaintext = rsaObj.decrypt(ciphertext);
System.out.println("plaintext(bytes):”+ bytesToString(plaintext));
System.out.println("plaintext(string) : " + new String(plaintext));
}}
```

OUTPUT:

stdin:
Standard input is empty
stdout:
SIMULATION OF RSA ALGORITHM
message(string) : Hello world! Security Laboratorymessage(bytes) :
721011081081113211911111410810033328310199117114105116121327697981111
149711611
1114121
ciphertext(bytes) : 0-119-42-9610376-981195-8810516-1281042-9-38-23-74856644-
2510705766-94-2310310452-2674-46125-13561120-2616122-111-507-117-5621-962-
57-127-9-
93-537810108-50355612715733-31-10510732-106-6050-8666-
2612570113357436822-46-
7169-402411558-42-11141-50-872210997-61-84-2627-84-36-55-56-1255440196847-
60-
625450-19-35123-901541-126-109-7-5300-117618634-51-67-90-113121-88-
10234124-61198-
24-26741167-568553-43-38873646-1105-21-77-1116358-120-47-98-1-110-97-31-125-
113108-
91-998312234-27-29-11278-6011241-9944-6676-20-9225-12796-48-52-75-117-27-
884-815-
648511153383676-158-116-8573120-87-4467104-9784108111-54830-61-90-38-
11223-119-
4210510-18-20-4090-85-104-8561-120-49-12-120108-23-48-4945-102
plaintext(bytes) :
721011081081113211911111410810033328310199117114105116121327697981
111149711611
1114121
plaintext(string) : Hello world! Security Laboratory

RESULT:

Thus the program was executed and verified successfully.

EX NO:2C DIFFIE HELLMAN KEY EXCHANGE ALGORITHM

AIM:

Develop a program to implement Diffie Hellman Key Exchange Algorithm for encryption and Decryption.

ALGORITHM DESCRIPTION:

- Diffie–Hellman key exchange (D–H) is a specific method of securely exchanging cryptographic keys over a public channel and was one of the first public-key protocols. T

- he Diffie–Hellman key exchange method allows two parties that have no prior knowledge of each other to jointly establish a shared secret key over an insecure channel.
- This key can then be used to encrypt subsequent communications using a symmetric keycipher.

ALGORITHM

- Global Public Elements:
- Let q be a prime number and □ where □ < q and □ is a primitive root of q.
- User A Key Generation: Select private X_A where $X_A < q$
- Calculate public Y_A where $Y_A = □^{X_A} \bmod q$
- User B Key Generation: Select private X_B where $X_B < q$
- Calculate public Y_B where $Y_B = □^{X_B} \bmod q$
- Calculation of Secret Key by User A $K = (Y_B)^{X_A} \bmod q$
- Calculation of Secret Key by User B:

PROGRAM :

```
import java.util.*;class diffieHellmanAlg
{
public static void main (String[] args) throws java.lang.Exception
{
int p = 11; /* publicly known (prime number) */int g = 2;
 /* publicly known (primitive root) */
int x = 9;
 /* only Alice knows this secret */
int y = 4;
 /* only Bob knows this secret */
double aliceSends    = (Math.pow(g,x))%p;
double bobComputes = (Math.pow(aliceSends,y))%p;
double bobSends      = (Math.pow(g,y))%p;
double aliceComputes = (Math.pow(bobSends,x))%p;
double sharedSecret  = (Math.pow(g,(x*y)))%p;
System.out.println(" simulation of Diffie-Hellman key exchange algorithm ");
 System.out.println("aliceSends : " + aliceSends);
 System.out.println("bobComputes : " + bobComputes);
 System.out.println("bobSends : " + bobSends);
System.out.println("aliceComputes:"+aliceComputes);
System.out.println("sharedSecret  : " + sharedSecret);
if((aliceComputes==sharedSecret)&&(aliceComputes==bobComputes))
System.out.println("success: shared secrets matches! " + sharedSecret);
else
System.out.println("error: shared secrets does not match");
}
}
```

OUTPUT:

stdin:
Standard input is empty
stdout:
simulation of Diffie-Hellman key exchange algorithmalice Sends : 6.0
bobComputes : 9.0
bobSends : 5.0
aliceComputes : 9.0
sharedSecret : 9.0
success: shared secrets matches! 9.0

RESULT:

Thus the program was executed and verified successfully.

EX NO: 3 IMPLEMENT MESSAGE DIGEST ALGORITHM (MD5)

AIM:

Develop a program to implement Message Digest Algorithm.

ALGORITHM DESCRIPTION:

- The MD5 message-digest algorithm is a widely used cryptographic hash function producing a 128-bit (16-byte) hash value, typically expressed in text format as a 32-digithexadecimal number.
- MD5 has been utilized in a wide variety of cryptographic applications and is also commonly used to verify data integrity

PROGRAM:

```
import java.util.*; classmd5Alg
{
/* initialize variables A,B,C,D */
private static final int INIT_A = 0x67452301;
 privatestatic  final int INIT_B=(int)0xEFCDAB89L;
 private static final int INIT_C=   (int)0x98BADCFEL;
private static  final int INIT_D = 0x10325476;
/* per-round shift amounts */
private static final int[] SHIFT_AMTS = { 7, 12, 17, 22,5,  9, 14, 20,4, 11, 16, 23,6, 10,
     15, 21 };

/* binary integer part of sin's of integers (Radians) as constants */private static final int[]
     TABLE_T = new int[64];
Static
for (int i = 0; i < 64; i++)
{
TABLE_T[i] = (int)(long)((1L << 32) * Math.abs(Math.sin(i + 1)));
}
}
/* compute message digest (hash value) */ public static byte[] computeMd5(byte[] msg)
{
int  msgLenBytes = msg.length;
  /* msg length (bytes) */
long msgLenBits = (long)msgLenBytes << 3;
   /* msg length (bits) */
 int numBlks = ((msgLenBytes + 8) >>> 6) + 1;
 /* number of blocks */
 int totLen = numBlks << 6;
/* total length */
byte[] padB = new byte[totLen - msgLenBytes];
  /* padding bytes */
/* pre-processing with padding */padB[0] = (byte)0x80;
for (int i = 0; i < 8; i++)
{
padB[padB.length - 8 + i] = (byte)msgLenBits;
msgLenBits >>>= 8;
}
int a = INIT_A;
int b = INIT_B;
int c = INIT_C;
```

```
int d = INIT_D;
int[] buf = new int[16];
for (int i = 0; i < numBlks; i ++)

int idx = i << 6;
for (int j = 0; j < 64; j++, idx++)

{
buf[j >>> 2] = ((int)((idx < msgLenBytes) ? msg[idx] : padB[idx - msgLenBytes]) << 24) |
    (buf[j >>> 2] >>> 8);
}


/* initialize hash value for this chunk */int origA = a;
int origB = b;
int origC = c;
int origD = d;
for (int j = 0;
 j < 64; j++)
{
int div16 = j >>> 4;
int f = 0;
int bufIdx = j;
 /* buf idx */switch (div16)
{
case 0:
{
f = (b & c) | (~b & d);break;
}
case 1:
{
f = (b & d) | (c & ~d);

break;
}
case 2:
{
f = b ^ c ^ d;
bufIdx = (bufIdx * 3 + 5) & 0x0F;
break;
}
```

```
case 3:
{
f = c ^ (b | ~d);
bufIdx = (bufIdx * 7) & 0x0F;
break;
}
}
/* left rotate */
int temp = b + Integer.rotateLeft(a + f + buf[bufIdx] + TABLE_T[j], SHIFT_AMTS[(div16
    << 2) | (j & 3)]);
a = d;d = c;c = b;
b = temp;



}
/* add this chunk's hash to result so far */a += origA;
b += origB;c += origC;

d += origD;

}
byte[] md5 = new byte[16];int cnt = 0;
for (int i = 0; i < 4; i++)
{

int n = (i == 0) ? a : ((i == 1) ? b : ((i == 2) ? c : d));
for (int j = 0; j < 4; j++)
{
md5[cnt++] = (byte)n;n >>>= 8;
}
}
return md5;
}
public static String toHexString(byte[] b)
{
StringBuilder sb = new StringBuilder();
for (int i = 0; i < b.length; i++)
{
sb.append(String.format("%02x", b[i] & 0xFF));
}
return sb.toString();
}
```

```
public static void main (String[] args) throws java.lang.Exception
{
String msg = "hello world";

System.out.println("simulation of MD5 algorithm");
System.out.println("input msg : " + msg);
System.out.println("md5 hash  : " + to HexString(computeMd5(msg.getBytes())));
}
}
```

OUTPUT:

stdin:
Standard input is empty
stdout:
simulation of MD5 algorithminput msg : hello world
md5 hash : 5eb63bbbe01eeed093cb22bb8f5acdc3

RESULT:

Thus the program was executed and verified successfully.

EX NO:4 IMPLEMENT SECURE HASH ALGORITHM

AIM:

Develop a program to implement Secure Hash Algorithm (SHA-1)

ALGORITHM DESCRIPTION:

- Secured Hash Algorithm-1 (SHA-1):
- Step 1: Append Padding Bits….
- Message is “padded” with a 1 and as many 0’s as necessary to bring the message length to 64 bits less than an even multiple of 512.
- Step 2: Append Length....
- 64 bits are appended to the end of the padded message. These bits hold the binary format of 64 bits indicating the length of the original message.
- Step 3: Prepare Processing Functions….
- SHA1 requires 80 processing functions defined as: f(t;B,C,D) = (B AND C) OR ((NOT B) AND D) (0 <= t <= 19)
- f(t;B,C,D) = B XOR C XOR D (20 <= t <= 39) f(t;B,C,D) = (B AND C) OR (B AND D) OR (C AND D) (40 <= t<=59)
- f(t;B,C,D) = B XOR C XOR D (60 <= t <= 79)
- Step 4: Prepare Processing Constants....
- SHA1 requires 80 processing constant words defined as:
- K(t) = 0x5A827999 (0 <= t <= 19) K(t) = 0x6ED9EBA1 (20 <= t <= 39) K(t) = 0x8F1BBCDC (40 <= t <= 59) K(t) = 0xCA62C1D6 (60 <= t <= 79)
- Step 5: Initialize Buffers….
- SHA1 requires 160 bits or 5 buffers of words (32 bits):
- H0 = 0x67452301 H1 = 0xEFCDAB89 H2 = 0x98BADCFE H3 = 0x10325476 H4 = 0xC3D2E1F0
- Step 6: Processing Message in 512-bit blocks (L blocks in total message)….
- This is the main task of SHA1 algorithm which loops through the padded and appended message in 512-bit blocks.

- Input and predefined functions: M[1, 2, ..., L]: Blocks of the padded and appended message f(0;B,C,D), f(1,B,C,D), ..., f(79,B,C,D): 80 Processing Functions K(0), K(1), ...,
- K(79): 80 Processing Constant Words
- H0, H1, H2, H3, H4, H5: 5 Word buffers with initial values

- Step 7: Pseudo Code….
- For loop on k = 1 to L
- (W(0),W(1),...,W(15)) = M[k] */* Divide M[k] into 16 words */*
- For t = 16 to 79 do:
- W(t) = (W(t-3) XOR W(t-8) XOR W(t-14) XOR W(t-16)) <<< 1
- A = H0, B = H1, C = H2, D = H3, E = H4
- For t = 0 to 79 do:
- TEMP = A<<<5 + f(t;B,C,D) + E + W(t) + K(t) E = D, D = C,
- C = B<<<30, B = A, A = TEMP
- End of for loop
- H0 = H0 + A, H1 = H1 + B, H2 = H2 + C, H3 = H3 + D, H4 = H4 + E
- End of for loop
- Output:
- H0, H1, H2, H3, H4, H5: Word buffers with final message digest

PROGRAM :

```
import java.security.*;
public class SHA1 {
public static void main(String[] a)
 {try
{
MessageDigest md = MessageDigest.getInstance("SHA1");
System.out.println("Message digest object info: ");
 System.out.println(" Algorithm = " +md.getAlgorithm());
 System.out.println(" Provider = " +md.getProvider());
System.out.println(" ToString = " +md.toString());
String input = ""; md.update(input.getBytes());
byte[] output = md.digest(); System.out.println();
System.out.println("SHA1(\""+input+"\") = " +bytesToHex(output));
input = "abc";
md.update(input.getBytes());
output = md.digest();
 System.out.println();
```

```
System.out.println("SHA1(\""+input+"\") = " +bytesToHex(output));
input = "abcdefghijklmnopqrstuvwxyz";
 md.update(input.getBytes());
output = md.digest();
System.out.println();
System.out.println("SHA1(\""+input+"\") = " +bytesToHex(output));
System.out.println("");
 }
catch (Exception e)
{
System.out.println("Exception: " +e);
}
}
public static String bytesToHex(byte[] b)
{
char hexDigit[] = {'0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'A', 'B', 'C', 'D', 'E', 'F'};
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:

```
C:\Program Files\Java\jdk1.6.0_20\bin>javac SHA1.java C:\Program
    Files\Java\jdk1.6.0_20\bin>java SHA1 Message digest object info:
Algorithm = SHA1 Provider = SUN version 1.6
ToString = SHA1 Message Digest from SUN,
<initialized> SHA1("") = DA39A3EE5E6B4B0D3255BFEF95601890AFD80709
SHA1("abc") = A9993E364706816ABA3E25717850C26C9CD0D89D
SHA1("abcdefghijklmnopqrstuvwxyz")=32D10C7B8CF96570CA04CE37F2A19D84240
    D3A89
```

RESULT:

Thus the program was executed and verified successfully.

EX NO:5 IMPLEMENT DIGITAL SIGNATURE SCHEME

AIM:

To write a program to implement the digital signature scheme in java

PROGRAM :

```
import java.util.*;
import java.math.BigInteger;
class dsaAlg
{
final static BigInteger one = new BigInteger("1");
 final static BigInteger zero = new BigInteger("0");
/* incrementally tries for next prime */
public static BigInteger getNextPrime(String ans)
{
BigInteger test = new BigInteger(ans);
while (!test.isProbablePrime(99))
{
test = test.add(one);
}
return test;
}
/* finds largest prime factor of n */
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;
}
/* finds a generator mod p */
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();
    /* establish the global public key components */
    BigInteger p = getNextPrime("10600");
     /* approximate prime */BigInteger q = findQ(p.subtract(one));
    BigInteger g = getGen(p,q,randObj);
    /* public key components */
    System.out.println("simulation of Digital Signature Algorithm");
    System.out.println("global public key components are:");
    System.out.println("p is: " + p);
    System.out.println("q is: " + q);
    System.out.println("g is: " + g);
    /* find the private key */
    BigInteger x = new BigInteger(q.bitLength(), randObj);
    x = x.mod(q);
    /* corresponding public key */ BigInteger y = g.modPow(x,p);
    /* random value message */
    BigInteger k = new BigInteger(q.bitLength(), randObj);
    k = k.mod(q);
    /* randomly generated hash value and digital signature */
    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);
    /* secret information */
  System.out.println("secret information are:");
 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("generating digital signature:");
  System.out.println("r is : " + r);
 System.out.println("s is : " + s);
```

```
/* verify the digital signature */
 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("verifying digital signature (checkpoints):");
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("success: digital signature is verified! " + r);
}
else
{
System.out.println("error: incorrect digital signature");
}
}
}
```

OUTPUT:

stdin:
Standard input is empty
stdout:
simulation of Digital Signature Algorithmglobal public key components are:
p is: 10601

q is: 53
g is: 3848
secret information are:
x (private) is: 48k (secret) is: 25
y (public) is: 5885 h (rndhash) is: 8794
generating digital signature:
r is : 4 s is : 16
verifying digital signature (checkpoints):w is : 10
u1 is : 13u2 is : 40v is : 4
success: digital signature is verified! 4

RESULT:

Thus the program was executed and verified successfully.

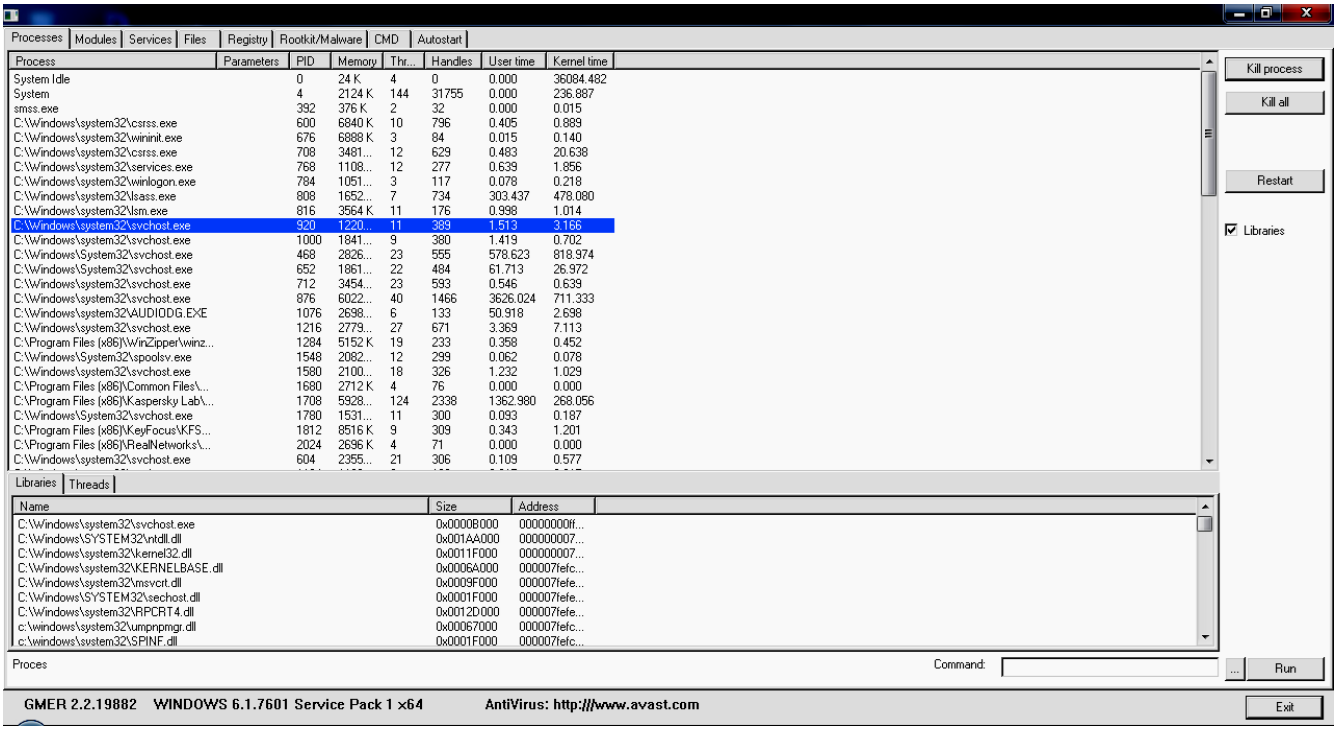
EX.No: 6 INSTALL ROOTKITS AND STUDY VARIETY OF OPTIONS

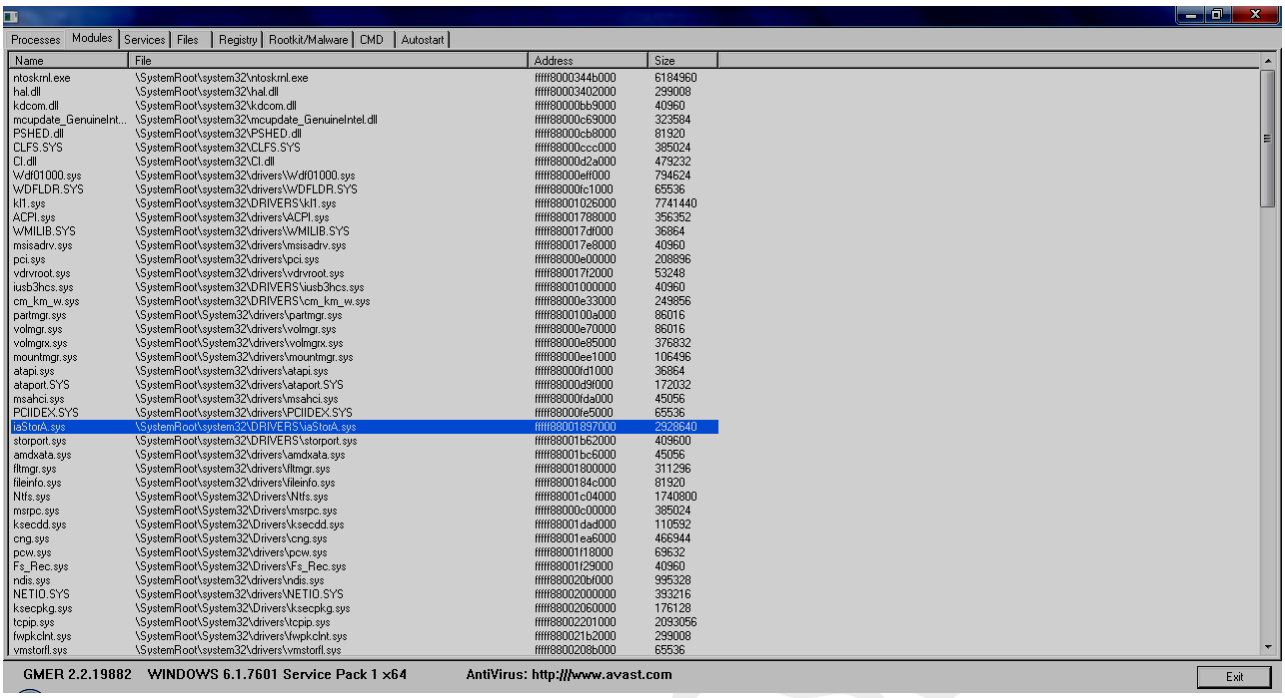
AIM:

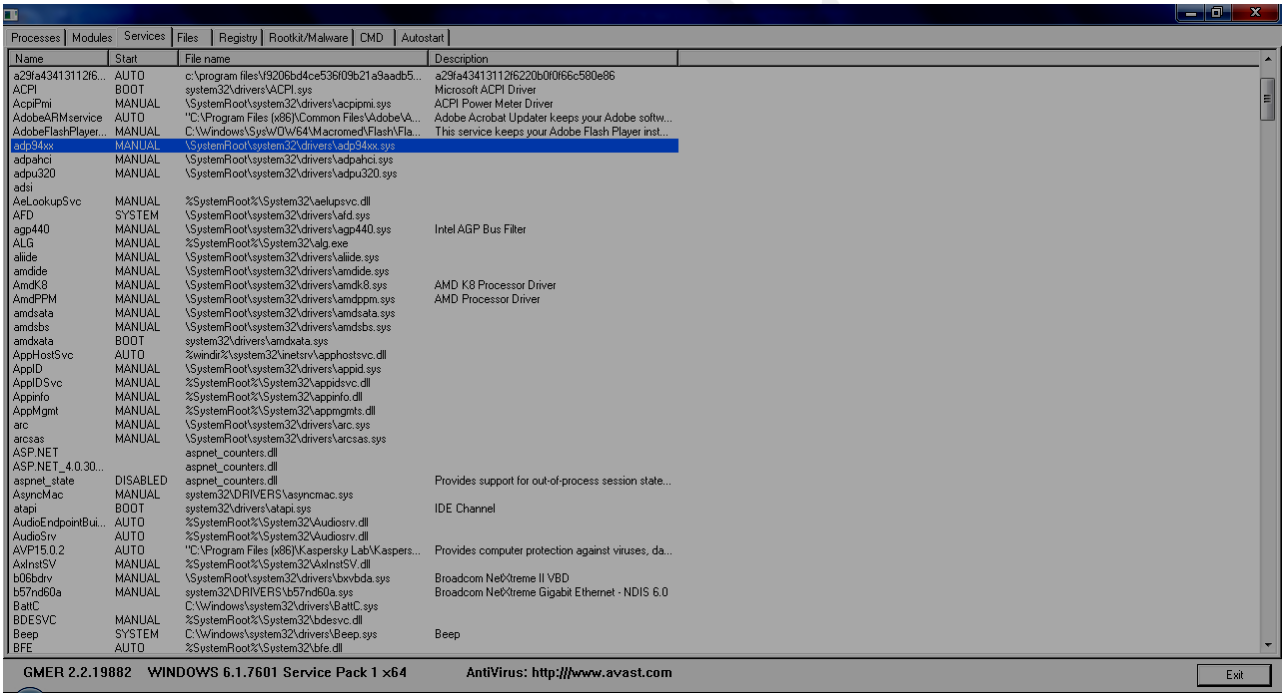
Rootkit is a stealth type of malicious software designed to hide the existence of certain processfrom normal methods of detection and enables continued privileged access to a computer

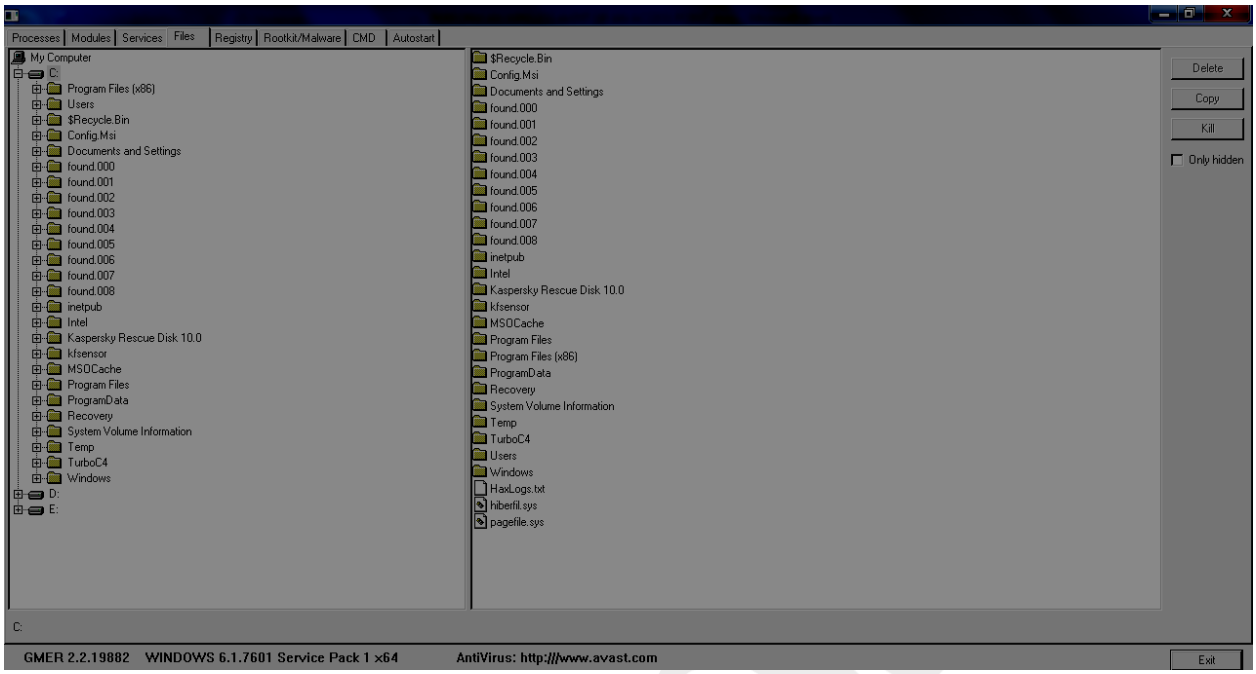
DESCRIPTION :

- Root kit 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.
- Download Rootkit Tool from GMER website. www.gmer.net
- This displays the Processes, Modules, Services, Files, Registry, RootKit/Malwares,Autostart, CMD of local host.
- Select Processes menu and kill any unwanted process if any. Modules menu displays thevarious system files like .sys, .dll
- Services menu displays the complete services running with Autostart, Enable, Disable,System, Boot.
- Files menu displays full files on Hard-Disk volumes.
- Registry displays Hkey_Current_user and Hkey_Local_Machine. Rootkits/Malawaresscans the local drives selected.
- Auto start displays the registry base Auto start applications.
- CMD allows the user to interact with command line utilities or Registry.

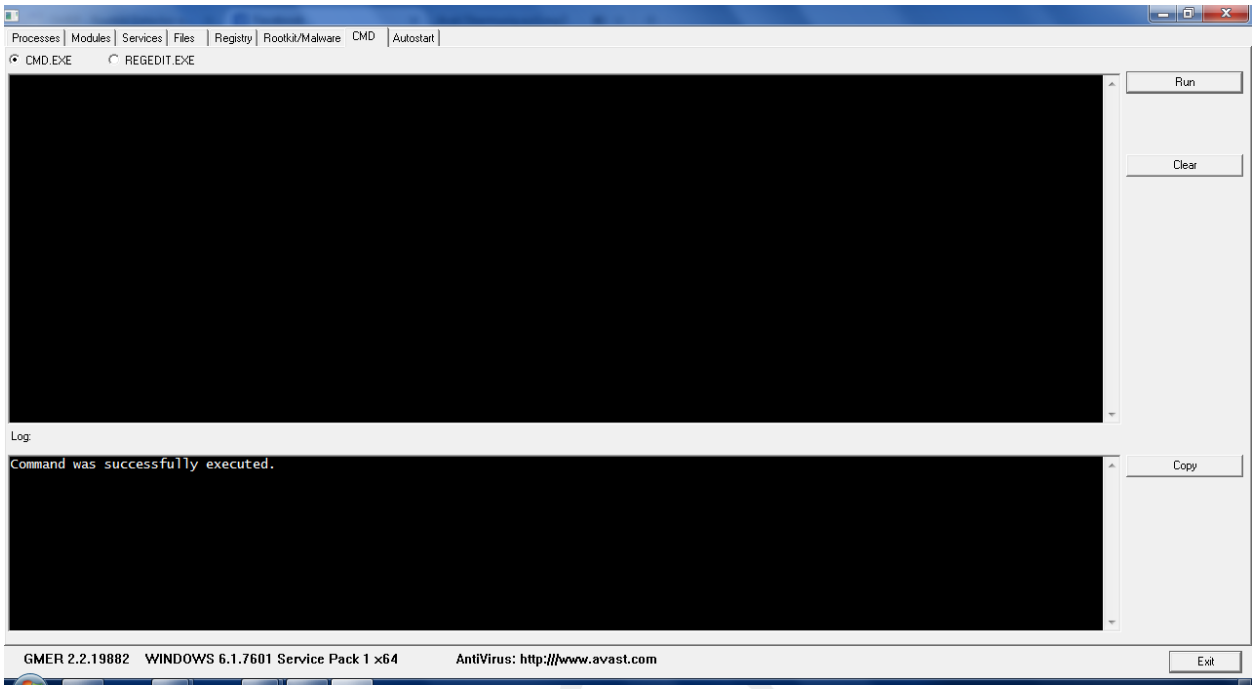


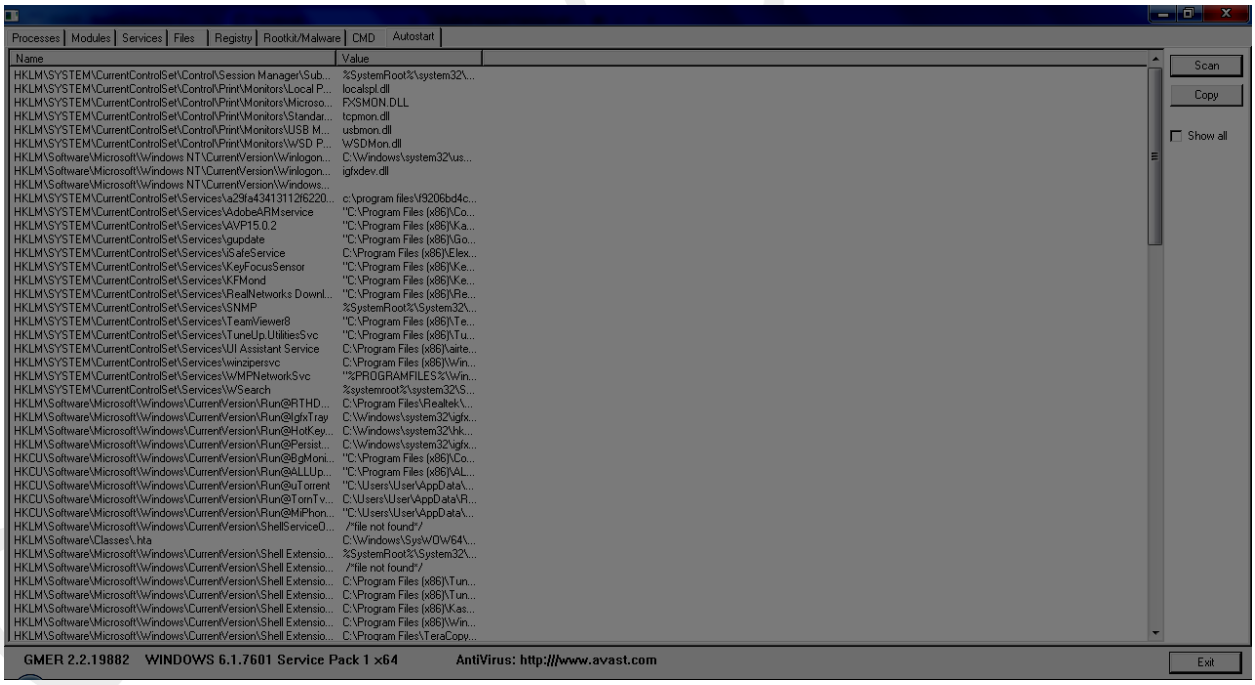

Processes Modules Services Files Registry Rootkit/Malware CMD Autostart
Name File Address Size
ntoskrnl.exe \SystemRoot\system32\ntoskrnl.exe fffff8000344b000 6184960
hal.dll \SystemRoot\system32\hal.dll fffff80003402000 299008
kdcom.dll \SystemRoot\system32\kdcom.dll fffff80000bb9000 40960
mcupdate_GenuineInt... \SystemRoot\system32\mcupdate_GenuineIntel.dll fffff88000c69000 323584
PSHED.dll \SystemRoot\system32\PSHED.dll fffff88000cb8000 81920
CLFS.SYS \SystemRoot\system32\CLFS.SYS fffff88000ccc000 385024
CI.dll \SystemRoot\system32\CI.dll fffff88000d2a000 479232
Wdf01000.sys \SystemRoot\system32\drivers\Wdf01000.sys fffff88000ef1000 794624
WDFLDR.SYS \SystemRoot\system32\drivers\WDFLDR.SYS fffff88000fc1000 65536
kl1.sys \SystemRoot\system32\DRIVERS\kl1.sys fffff88001026000 7741440
ACPI.sys \SystemRoot\system32\drivers\ACPI.sys fffff88001788000 356352
WMILIB.SYS \SystemRoot\system32\drivers\WMILIB.SYS fffff880017df000 36864
msisadrv.sys \SystemRoot\system32\drivers\msisadrv.sys fffff880017e8000 40960
pci.sys \SystemRoot\system32\drivers\pci.sys fffff88000e00000 208896
vdrvroot.sys \SystemRoot\system32\drivers\vdrvroot.sys fffff880017f2000 53248
iusb3hcs.sys \SystemRoot\system32\DRIVERS\iusb3hcs.sys fffff88001000000 40960
cm_km_w.sys \SystemRoot\system32\DRIVERS\cm_km_w.sys fffff88000e33000 249856
partmgr.sys \SystemRoot\System32\drivers\partmgr.sys fffff8800100a000 86016
volmgr.sys \SystemRoot\system32\drivers\volmgr.sys fffff88000e70000 86016
volmgrx.sys \SystemRoot\System32\drivers\volmgrx.sys fffff88000e85000 376832
mountmgr.sys \SystemRoot\System32\drivers\mountmgr.sys fffff88000ee1000 106496
atapi.sys \SystemRoot\system32\drivers\atapi.sys fffff88000fd1000 36864
ataport.SYS \SystemRoot\system32\drivers\ataport.SYS fffff88000d9f000 172032
msahci.sys \SystemRoot\system32\drivers\msahci.sys fffff88000fda000 45056
PCIIDEX.SYS \SystemRoot\system32\drivers\PCIIDEX.SYS fffff88000fe5000 65536
iaStorA.sys \SystemRoot\system32\DRIVERS\iaStorA.sys fffff88001897000 2928640
storport.sys \SystemRoot\system32\DRIVERS\storport.sys fffff88001b62000 409600
amdxata.sys \SystemRoot\system32\drivers\amdxata.sys fffff88001bc6000 45056
fltmgr.sys \SystemRoot\system32\drivers\fltmgr.sys fffff88001800000 311296
fileinfo.sys \SystemRoot\system32\drivers\fileinfo.sys fffff8800184c000 81920
Ntfs.sys \SystemRoot\System32\Drivers\Ntfs.sys fffff88001c04000 1740800
msrpc.sys \SystemRoot\System32\Drivers\msrpc.sys fffff88000c00000 385024
ksecdd.sys \SystemRoot\System32\Drivers\ksecdd.sys fffff88001dad000 110592
cng.sys \SystemRoot\System32\Drivers\cng.sys fffff88001ea6000 466944
pcw.sys \SystemRoot\System32\drivers\pcw.sys fffff88001f18000 69632
Fs_Rec.sys \SystemRoot\System32\Drivers\Fs_Rec.sys fffff88001f29000 40960
ndis.sys \SystemRoot\System32\drivers\ndis.sys fffff880020bf000 995328
NETIO.SYS \SystemRoot\system32\drivers\NETIO.SYS fffff88002000000 393216
ksecpkg.sys \SystemRoot\System32\Drivers\ksecpkg.sys fffff88002060000 176128
tcpip.sys \SystemRoot\System32\drivers\tcpip.sys fffff88002201000 2093056
fwpkclnt.sys \SystemRoot\System32\drivers\fwpkclnt.sys fffff880021b2000 299008
vmstorfl.sys \SystemRoot\system32\drivers\vmstorfl.sys fffff8800208b000 65536
GMER 2.2.19882 WINDOWS 6.1.7601 Service Pack 1 x64 AntiVirus: http://www.avast.com
Exit


Processes Modules Services Files Registry Rootkit/Malware CMD Autostart
Name Start File name Description
a29fa43413112f6... AUTO c:\program files\f9206bd4ce536f09b21a9aadb5... a29fa43413112f6220b0f0f66c580e86
ACPI BOOT system32\drivers\ACPI.sys Microsoft ACPI Driver
AcpiPmi MANUAL \SystemRoot\system32\drivers\acpipmi.sys ACPI Power Meter Driver
AdobeARMservice AUTO "C:\Program Files (x86)\Common Files\Adobe\A... Adobe Acrobat Updater keeps your Adobe softw...
AdobeFlashPlayer... MANUAL C:\Windows\SysWOW64\Macromed\Flash\Fla... This service keeps your Adobe Flash Player inst...
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\aelupsvc.dll
AFD SYSTEM \SystemRoot\system32\drivers\afd.sys
agp440 MANUAL \SystemRoot\system32\drivers\agp440.sys Intel AGP Bus Filter
ALG MANUAL %SystemRoot%\System32\alg.exe
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\inetsrv\apphostsvc.dll
AppID MANUAL \SystemRoot\system32\drivers\appid.sys
AppIDSvc MANUAL %SystemRoot%\System32\appidsvc.dll
Appinfo MANUAL %SystemRoot%\System32\appinfo.dll
AppMgmt MANUAL %SystemRoot%\System32\appmgmts.dll
arc MANUAL \SystemRoot\system32\drivers\arc.sys
arcsas MANUAL \SystemRoot\system32\drivers\arcsas.sys
ASP.NET aspnet_counters.dll
ASP.NET_4.0.30... aspnet_counters.dll
aspnet_state DISABLED aspnet_counters.dll Provides support for out-of-process session state...
AsyncMac MANUAL system32\DRIVERS\asyncmac.sys
atapi BOOT system32\drivers\atapi.sys IDE Channel
AudioEndpointBui... AUTO %SystemRoot%\System32\Audiosrv.dll
AudioSrv AUTO %SystemRoot%\System32\Audiosrv.dll
AVP15.0.2 AUTO "C:\Program Files (x86)\Kaspersky Lab\Kaspers... Provides computer protection against viruses, da...
AxInstSV MANUAL %SystemRoot%\System32\AxInstSV.dll
b06bdrv MANUAL \SystemRoot\system32\drivers\bxvbda.sys Broadcom NetXtreme II VBD
b57nd60a MANUAL system32\DRIVERS\b57nd60a.sys Broadcom NetXtreme Gigabit Ethernet - NDIS 6.0
BattC C:\Windows\system32\drivers\BattC.sys
BDESVC MANUAL %SystemRoot%\System32\bdesvc.dll
Beep SYSTEM C:\Windows\system32\drivers\Beep.sys Beep
BFE AUTO %SystemRoot%\System32\bfe.dll
GMER 2.2.19882 WINDOWS 6.1.7601 Service Pack 1 x64 AntiVirus: http://www.avast.com
Exit


Processes
Modules
Services
Files
Registry
Rootkit/Malware
CMD
Autostart
My Computer
C:
Program Files (x86)
Users
$Recycle.Bin
Config.Msi
Documents and Settings
found.000
found.001
found.002
found.003
found.004
found.005
found.006
found.007
found.008
inetpub
Intel
Kaspersky Rescue Disk 10.0
kfsensor
MSOCache
Program Files
ProgramData
Recovery
System Volume Information
Temp
TurboC4
Windows
D:
E:
HaxLogs.txt
hiberfil.sys
pagefile.sys
Delete
Copy
Kill
Only hidden
C:
GMER 2.2.19882
WINDOWS 6.1.7601 Service Pack 1 x64
AntiVirus: http://www.avast.com
Exit


Processes
Modules
Services
Files
Registry
Rootkit/Malware
CMD
Autostart
My Computer
HKEY_CURRENT_USER
HKEY_LOCAL_MACHINE
Name
Type
Data
Export ...
Save ...
GMER 2.2.19882
WINDOWS 6.1.7601 Service Pack 1 x64
AntiVirus: http://www.avast.com
Exit


Processes
Modules
Services
Files
Registry
Rootkit/Malware
CMD
Autostart
CMD.EXE
REGEDIT.EXE
Run
Clear
Log:
Command was successfully executed.
Copy
GMER 2.2.19882
WINDOWS 6.1.7601 Service Pack 1 x64
AntiVirus: http://www.avast.com
Exit


Processes
Modules
Services
Files
Registry
Rootkit/Malware
CMD
Autostart
Name
Value
Scan
Copy
Show all
GMER 2.2.19882
WINDOWS 6.1.7601 Service Pack 1 x64
AntiVirus: http://www.avast.com
Exit

RESULT:

Thus the program was executed and verified successfully.

EX.No: 7 SETUP A HONEY POT AND MONITOR THE HONEYPOT ON NETWORK

AIM:

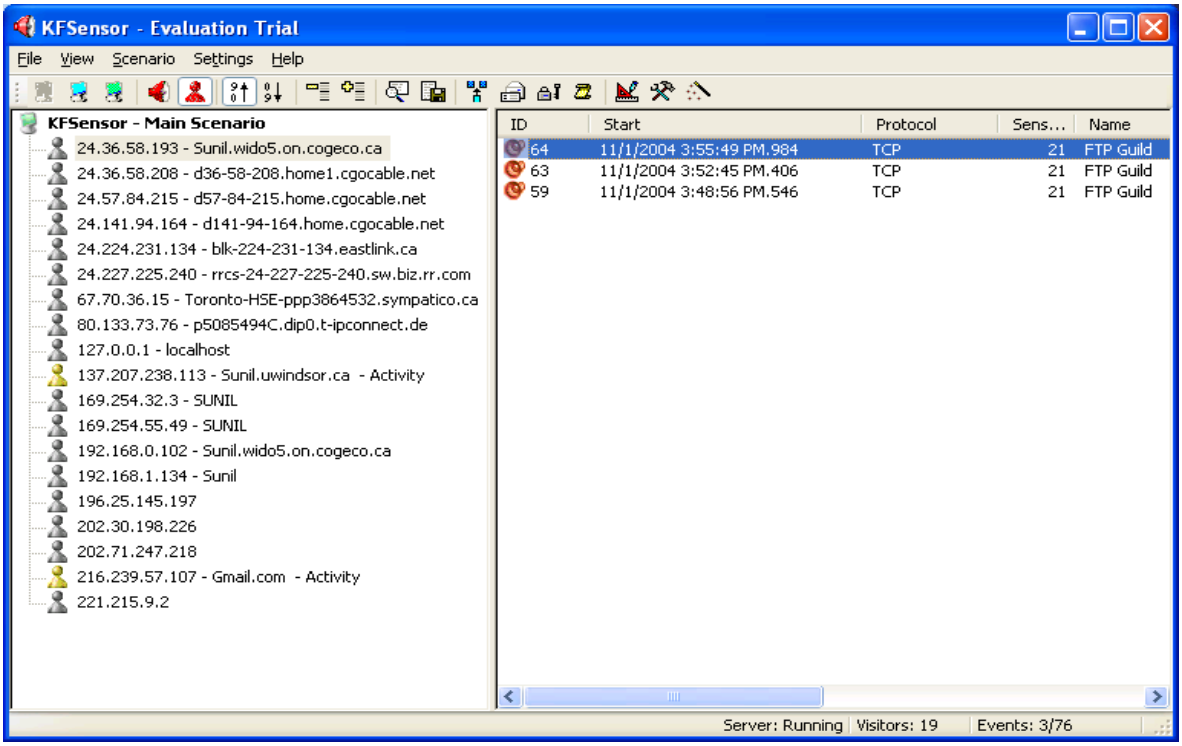
Honey Pot is a device placed on Computer Network specifically designed to capture malicious network traffic. KF Sensor is the tool to setup as honeypot when KF Sensor is running it places a siren icon in the windows system tray in the bottom right of the screen. If there are no alerts then green icon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

STEPS:

- Install winpcap library (mandatory for kfsensor)2.Download kfsensor and install
- Then restart your pc.Configure properly no change needs to do now go to setting option andconfigure according to your attack.
- Now go to your home screen of kf sensor
- You will get some logs about clients .And it will start working
- KFSensor
- Windows based honey pot known as KF Sensor
- It detects an incoming attack or port scanning and reports it to you
- A machine running KFSensor can be treated as just another server on the network,without the need to make complex changes to routers and firewalls.
- How KFSensor Woorks?
- KFSensor is an Intrusion Detection System.
- It performs by opening ports on the machine it is installed on and waiting for connections to be made to those ports.
- By doing this it sets up a target, or a honeypot server, that will record the actions of ahacker.
- Components: KFSensor server
- KFSensor Server- Performs core functionality
- It listens to both TCP and UDP ports on the server machine and interacts with visitorsand generates events.

A daemon that runs at the background (like Unix daemon)

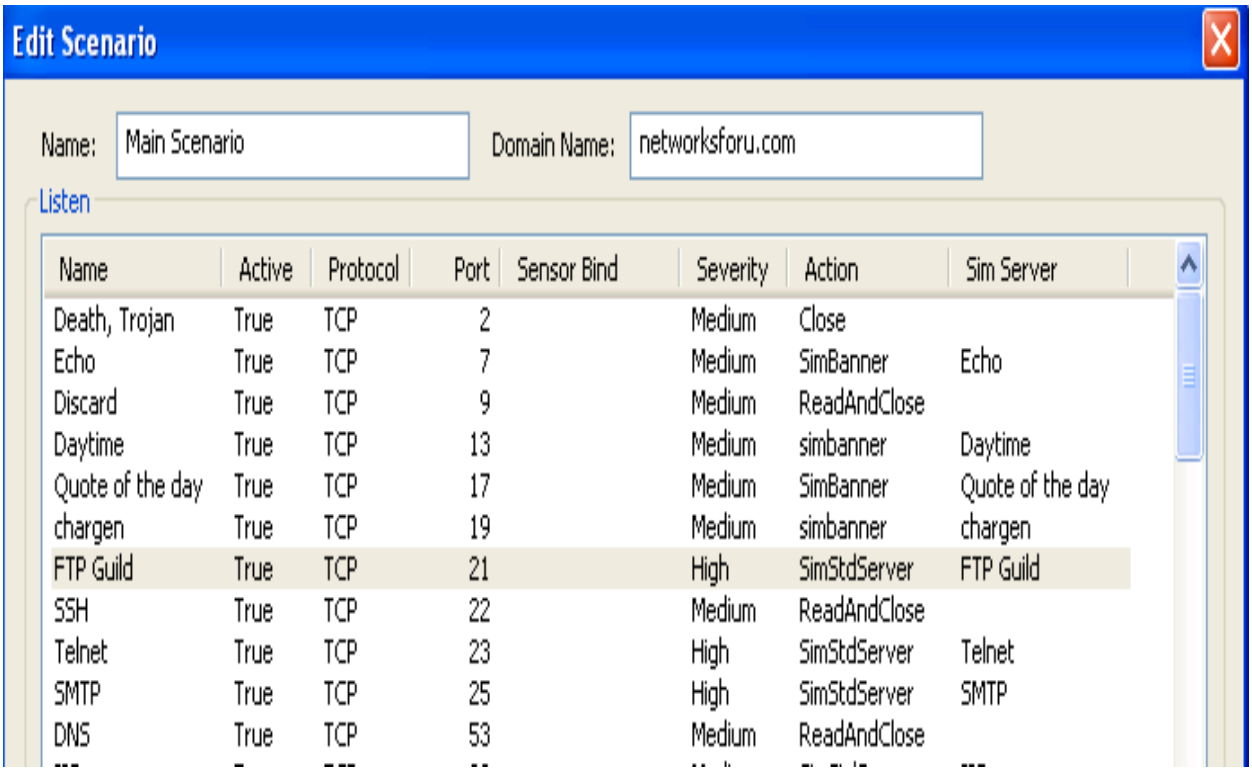
- Components: KFSensor Monitor
- Interprets all the data and alerts captured by server in graphical form.
- Using it you can configure the KFSensor Server and monitor the events generated by theKFSensor Server.
- Sim Server
- Sim server is short for simulated server.
- It is a definition of how KFSensor should emulate real server software.
- There is no limit to the number of Sim Servers that can be defined.
- There are two types of Sim Server available; the Sim Banner and the Sim StandardServer.
- Setting Up a HoneyPot
- You can get educational License from Keyfocus.
- Install WinPCap
- A industry standard network packet capturing library
- Install KFSensor
- KFSensor Monitor



Terminology Visitor

A visitor is an entity that connects to KFSensor.

- Visitors could be hackers, worms, viruses or even legitimate users that have stumbled onto KFSensor by mistake.
- Visitors can also be referred to as the clients of the services provided by KFSensor.Event
- An event is a record of an incident detected by the KFSensor Service.
- For example if a visitor attempts to connect to the simulated web server then an eventdetailing the connection is generated.
- Events are recorded in the log file and displayed in the KFSensor monitor.
- Editing Scenario



Terminology – Rules

- KFSensor is rules based.
- All of the data that was produced was the result of KFSensor detecting certain types of activity and then using a rule to determine what type of action should be taken.
- We can easily modify the existing rules or add your own.
- Edit Active Scenario
- To create or modify rules,
- Scenario menu ->select the Edit Active Scenario command ->you will see a dialog box which contains a summary of all of the existing rules.
- either select a rule and click the Edit button to edit a rule, or you can click the Add button to create a new rule.

Adding a rule

- Click the Add button and you will see the Add Listen dialog box.

- `The first thing that this dialog box asks for is a name. This is just a name for the rule.
- Pick something descriptive though, because the name that you enter is what will show up in the logs whenever the rule is triggered.
- Download Link
- http://www.keyfocus.net/kfsensor/free-trial/

- Write to support@keyfocus.net and request for educational license Installing KFSensor
- Download and install winpcap
- Download and install KFSensor

- Enable Telnet client, server, Internet Information server in Control Panel-> Programs-> Turnwindows features on/off
- Check Telnet client, Telnet server, IIS-> FTP (both options),

- Convert to Native Service
- Convert the stroked off services as native services.

- *Select Scenario ->Edit Active Scenario

- choose the respective service listed in the dialog box opened and press convert to native button and ok.
- Setting up Server
- To start the server

- Settings-> Set Up Wizard

- Go through the wizard, give fictitious mail ids when they are asked and start the server running by pressing the finish button.
- Kfsensor now start showing the captured information in its window.

- FTP Emulation
- Open command prompt

- Type

- Ftp ipaddress

- Enter user name anonymous Enter any password
- Get any file name with path

- Monitor this ftp access in KFSensor monitor

- Right click KFSensor entry, select Event details, see the details captured by the server

- Create visitor rule by right clicking the FTP entry and check either ignore / close under actions in the dialog box that opened.
- Now redo the above said operations at the command prompt and see how the emulation behaves.
- You can see/ modify the created rules in Scenario->edit active visitor rules.

- SMTP Emulation
- open command prompt

- Type telnet ip address 25Helo
- Mail from:<mail-id>Rcpt to:<mail-id> Data
- type contents of mail end that with . in new line

- Check the kfsensor for the captured information.

- IIS emulation
- Create an index.html, store it in c:\keyfocus\kfsensor\files\iis7\wwwroot

- Select scenario->edit sim server

- Choose iis and edit

- Make sure index.html is in first place in the listed htm files in the dialog box
- Check the kfsensor for the captured information.

- DOS attack
- Settings-> DOS attack settings modify (reduce) values in general tab, ICMP and othertabs. Press ok.
- Open command prompt and typePing ip address –t or
- Ping –l 65000 ip address –t

- Check the kfsensor for the DOS attack alerts, open event details in right click menu forfurther details.

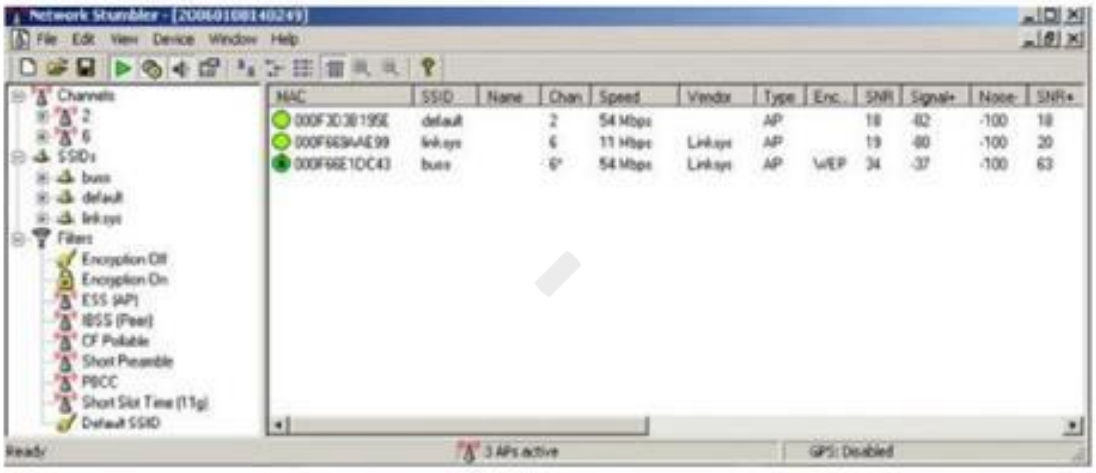
RESULT:

Thus the program was executed and verified successfully.

EX.No: 9 PERFORM WIRELESS AUDIT ON AN ACCESS POINT OR A ROUTER AND DECRYPT WEP AND WPA.

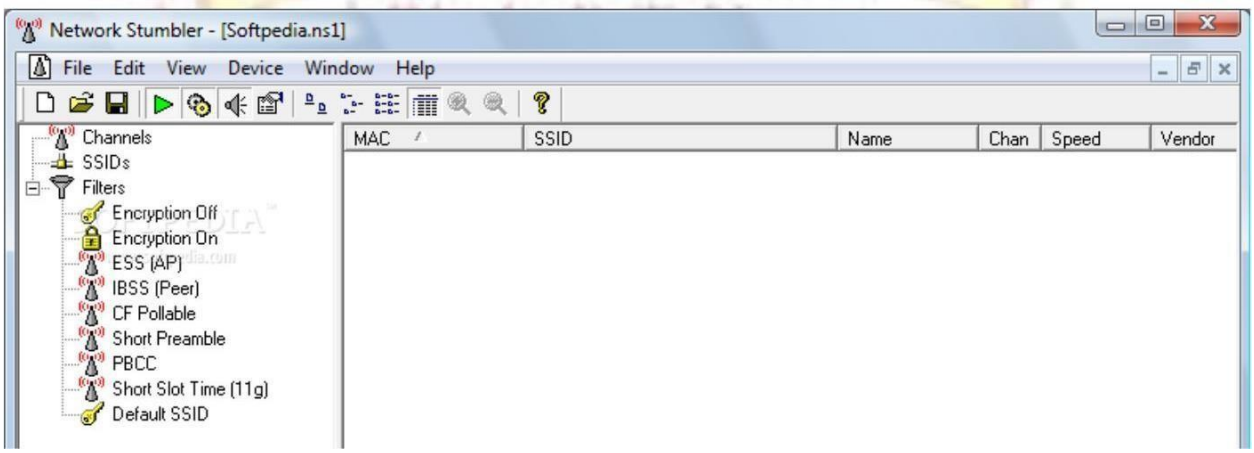
AIM:

NetStumbler (also known as Network Stumbler) aircrack on ubuntu is a tool for windows that facilitates detection of Wireless LANs using the 802.11b, 802.11a and 802.11g WLAN standards. It 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. It's having some compatibility and network adapter issues.

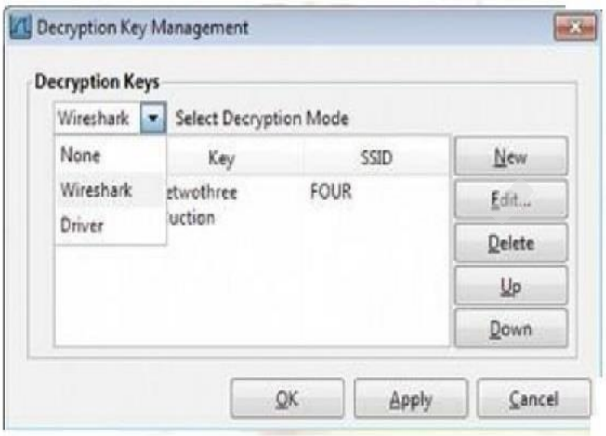


DESCRIPTION :

- NetStumbler (Network Stumbler) is one of the Wi-Fi hacking tool which only compatiblewith windows, this tool also a freeware.
- With this program, we can search for wireless network which open and infiltrate thenetwork.
- Its having some compatibility and network adapter issues.
- Download and install Netstumbler
- It is highly recommended that your PC should have wireless network card in order toaccess wireless router.
- Now Run Netstumbler in record mode and configure wireless card.
- 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 avery weak and GREY indicates a signal loss.
- Lock symbol with GREEN bubble indicates the Access point has encryption enabled.
- MAC assigned to Wireless Access Point is displayed on right hand pane.
- The next column displays the Access points Service Set Identifier[SSID] which is usefulto crack the password.
- To decrypt use WireShark tool by selecting Edit preferences IEEE 802.11
- Enter the WEP keys as a string of hexadecimal numbers as A1B2C3D4E5 Adding Keys: Wireless Toolbar
- If you are using the Windows version of Wireshark and you have an AirPcap adapteryou 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



- 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 program was executed and verified successfully.

EX.No:10 DEMONSTRATE INTRUSION DETECTION SYSTEM (IDs) USING ANY TOOLANY TOOL (SNORT OR ANY OTHER S/W)

AIM:

Snort is an open source network intrusion detection system (NIDS) has the ability to perform real-time traffic analysis and packet logging on internet protocol (IP) networks. Snort performs protocol analysis, content searching and matching. Snort can be configured in three main modes: sniffer, packet logger, and network intrusion detection.

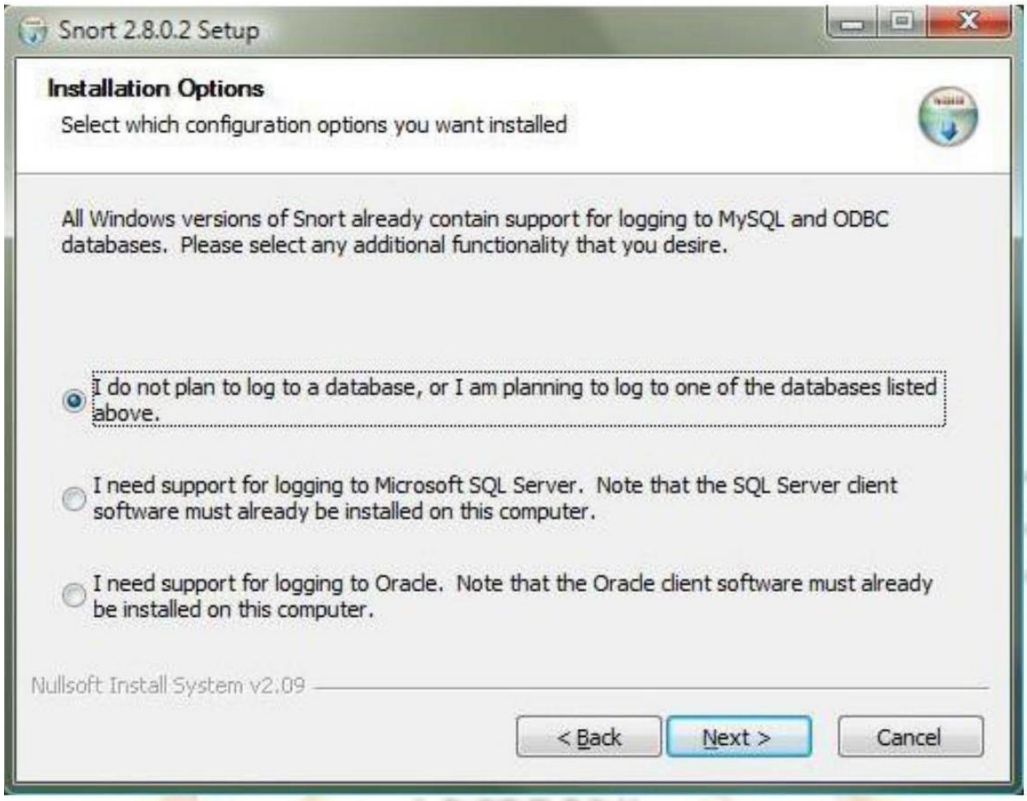
Description:

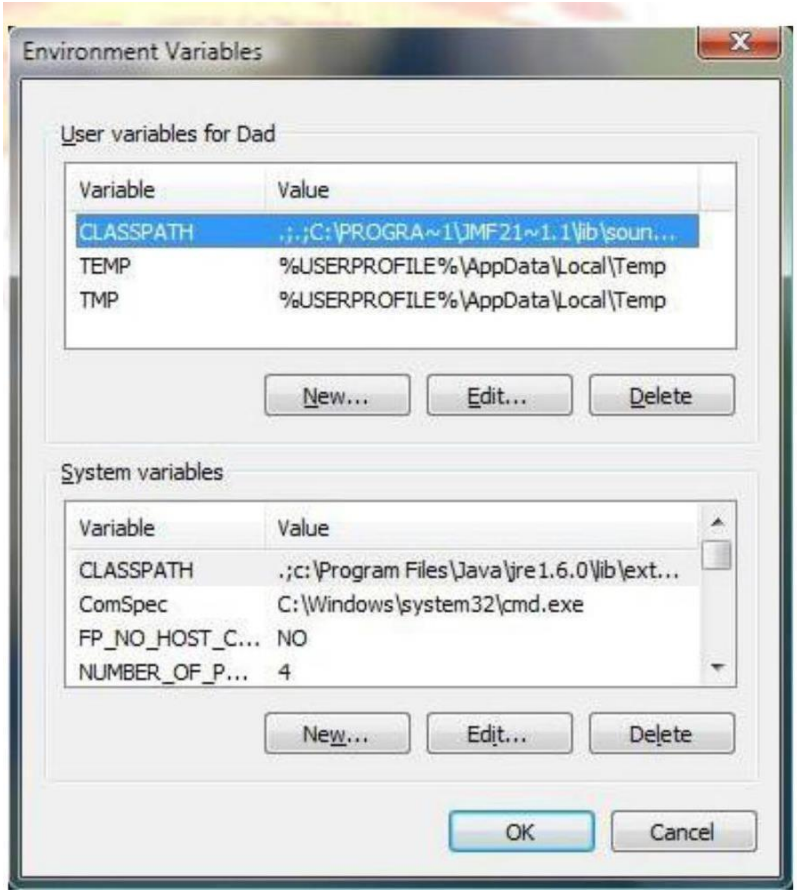
- Snort Installation Steps
- Getting and Installing Necessary ToolsInstallaing Packages
- Snort: <Snort_2_9_8_2_installer.exe>WinPcap: <WinPcap_4_1_3.exe>
- Snort rules: <snortrules-snapshot-2982.tar.gz>Once Completed
- Change the Snort program directory:
- <c:\cd \Snort\bin
- Check the installed version for Snort:
- <c:\Snort\bin> snort -V
- Check to see what network adapters are on your system
- <c:\Snort\bin> snort -W> Configure Snort with snort.conf
- <snort.conf> is located in <c:\Snort\ect>Contains nine steps:
- Set the network variables
- Change <HOME_NET> to your home network IP address range <10.6.2.1/24>
- Change <EXTERNAL_NET> to <!$HOME_NET>
- This expression means the external network will be defined as - any IP not part of homenetwork
- Check for <HTTP_PORTS>
- Change var <RULE_PATH> - actual path of rule files. i.e <c:\Snort\rules>
- Change var <PREPROC_RULE_PATH> - actual path of preprocessor rule files
- i.e <c:\Snort\preproc_rules>
- Comment <#> <SO_RULE_PATH> - as windows Snort doesn't use shared objectrules
- Configure trusted <white.list> and untrusted <black.list> IP address - reputation preprocessor
- Configure the decoder
- No changes in this part
- Set the default directory for Snort logs i.e <c:\Snort\logs> 3.Configure the base

detection engine

- No changes in this part 4.Configure dynamic loaded libraries
- Change the dynamic loaded library path references
- i.e. <dynamic preprocess direc c:\Snort\lib\snort_dynamicpreprocessor>
- i.e. <dynamic engine direc c:\Snort\lib\snort_dynamicengine\sf_engine.dll>
- Comment out <dynamic detection directory> declaration 5.Configure preprocessors
- Many Preprocessors are used by Snort - Check Snort manual before setting them.
- Comment on <inline packet normalization preprocessor> This preprocessor is used when Snort is in-line IPS mode>
- For general purpose Snort usage - check these preprocessors are activefrag3
- stream5 http_inpectftp_telnet smtp
- dnsssl
- sensitive_data
- Confgiure output plugins
- Be default Snort uses only one output plugings - <default:unified2>
- Want to use Syslog output pluging - activate it by uncommenting.
- Uncomment and edit the syslog output line
- <output alert_syslog: host=127.0.0.1:514, LOG_AUTH LOG_ALERT> Note: If you are going to use syslog - install <Syslog Server>
- Uncomment metadata reference lines
- <include classification.config and include reference.config> 7.Customise your rule set
- Initial test, reduce the number of rules loaded at start-up, uncomment <local.rules>
- First time users, comment most of include statements.8.Customise preprocessor and decoder rule set
- Uncomment the first two lines in Step 8
- <include $PREPROC_RULE_PATH\preprocessor.rules>
- <include $PREPROC_RULE_PATH\decoder.rules>
- If you enables the sensitive_data preprocessor <step 5> uncomment
- <include $PREPROC_RULE_PATH\sensitive-data.rules>
- Make sure rules you declare - available in <c:\Snort\preproc_rules>9.Customise shart object rule set
- Comment on lines
- Uncomment <include threshold.conf>Generating Alerts
- This is for validation of Snort
- Open <local.rules> in a text editor
- Start typing this:

- <alert icmp any any -> any any (msg:"ICMP Testing Rule"; sid:1000001; rev:1;)
- <alert tcp any any -> any 80 (msg:"TCP Testing Rule"; sid:1000002; rev:1;)
- <alert udp any any -> any any (msg:"UDP Testing Rule"; sid:1000003; rev:1;)
- Save as <local.rules>
- Open <CMD> and run it as <ADMINISTRATOR>
- Start Snort <c:\Snort\bin> snort -i 2 -c c:\Snort\etc\snort.conf -A console
- Open <CMD> no need to be an ADMINISTRATOR
- Send a <PING> command to your local gateway: <c:\> ping 10.6.0.1>
- Open a web browser and browse to any web page
- You can see the alerts Snort produces and shows it in First terminal.




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
New...
Edit...
Delete
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
New...
Edit...
Delete
OK
Cancel

RESULT:

Thus the program was executed and verified successfully.