NETWORK AND WEB APPLICATION SECURITY LABORATORY EXPERIMENTS WITH SOLUTIONS

LIST OF EXPERIMENTS:

1. Apply DES algorithm for practical applications.
2. Apply AES algorithm for practical applications
3. Implement RSA Algorithm using HTML and JavaScript
4. Implement the Diffie-Hellman Key Exchange algorithm for a given problem.
5. Calculate the message digest of a text using the SHA-1 algorithm
6. Implement the SIGNATURE SCHEME - Digital Signature Standard.
7. Demonstrate intrusion detection system (ids) using any tool eg. Snort or any other s/w.
8. Automated Attack and Penetration Tools Exploring N-Stalker, a

   Vulnerability Assessment Tool
9. Defeating Malware - Building Trojans, Rootkit Hunter

DESCRIPTION OF MAJOR SOFTWARE USED

JAVA

Java is a high-level programming language originally developed by Sun Microsystems. Java runs on a variety of platforms, such as Windows, Mac OS, and the various versions of UNIX. Java programming were "Simple, Robust, Portable, Platform-independent, Secured, High Performance, Multithreaded, Architecture Neutral, Object-Oriented, Interpreted, and Dynamic". It was originally designed for developing programs for set-top boxes and handheld devices, but later became a popular choice for creating web applications.

Installation requires you to download an executable file available at the manual Java download page, which includes all the files needed for the complete installation at the user's discretion. There is no need to remain connected to the Internet during the installation. The file can also be copied to and installed on another computer that is not connected to the Internet. Administrative permission is required in order to install Java on Microsoft Windows

The Java syntax is similar to C++, but is strictly an object-oriented programming language. For example, most Java programs contain classes, which are used to define objects, and methods, which are assigned to individual classes. Java is also known for being stricter than C++, meaning variables and functions must be explicitly defined. This means Java source code may produce errors or "exceptions" more easily than other languages, but it also limits other types of errors that may be caused by undefined variables or unassigned types.

Unlike Windows executables (.EXE files) or Macintosh applications (.APP files), Java programs are not run directly by the operating system. Instead, Java programs are interpreted by the Java Virtual Machine, or JVM, which runs on multiple platforms. This means all Java programs are multiplatform and can run on different platforms, including Macintosh, Windows, and Unix computers. However, the JVM must be installed for Java applications or applets to run at all. Fortunately, the JVM is included as part of the Java Runtime Environment (JRE), which is available as a free download.

SNORT

Snort really isn’t very hard to use, but there are a lot of command line options to play with, and it’s not always obvious which ones go together well. This file aims to make using Snort easier for new users. Before we proceed, there are a few basic concepts you should understand about Snort. Snort can be configured to run in three modes:

- Sniffer mode, which simply reads the packets off of the network and displays them for you in a continuous stream on the console (screen).

- Packet Logger mode, which logs the packets to disk.

- Network Intrusion Detection System (NIDS) mode, which performs detection and analysis on network traffic. This is the most complex and configurable mode.

Snort is based on libpcap (for library packet capture), a **tool** that is widely **used** in TCP/IP traffic sniffers and analyzers. Through protocol analysis and content searching and matching, **Snort** detects attack methods, including denial of service, buffer overflow, CGI attacks, stealth port scans, and SMB probes.

N-STALKER

N-STALKER is a world leader in Web Application Security solutions since 2000. It has started providing the first commercial and most complete HTTP Security Scanner, holding the largest signatures database available in the market – more than 39,000 attack signatures. Our products are delivered to hundreds of customers distributed in more than 30 different countries around the world. Back in 2000, N-Stalker's challenge was to provide complete solutions for your Web server infrastructure, which ended up with the release of N-Stealth HTTP Security Scanner. Nowadays, N- Stalker is seeking to provide the most complete solution for your enterprise web applications, the N-Stalker Web Application Security Scanner Suite.

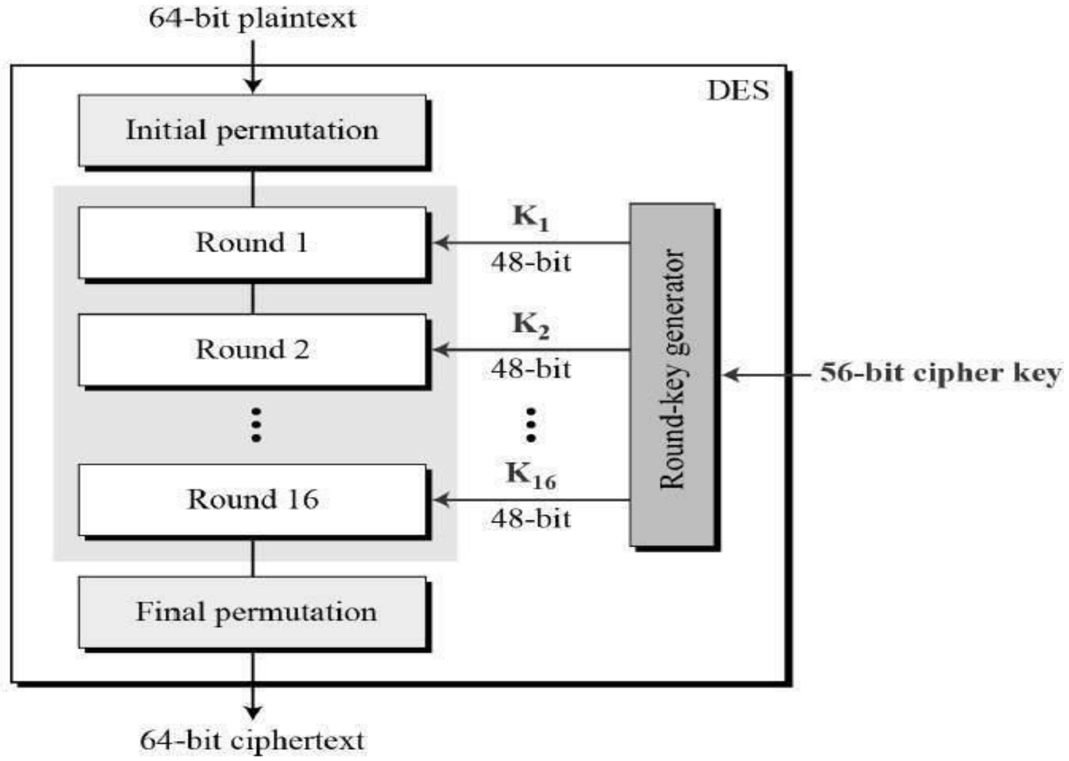
EX.No.: 1	DATA ENCRYPTION STANDARD (DES)

AIM:

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

PRELAB DISCUSSION:

- The Data Encryption Standard (DES) is a symmetric-key block cipher published by the 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



ALGORITHM:

1. Process the key.
   i. Get a 64-bit key from the user.
   ii. Calculate the key schedule.
      1. Perform the following permutation on the 64-bit key. The parity bits are discarded, reducing the key to 56 bits. Bit 1 of the permuted block is bit 57 of the original key, bit 2 is bit 49, and so on with bit 56 being bit 4 of the original key.
      2. Split the permuted key into two halves. The first 28 bits are called C[0] and the last 28 bits are called D[0].

3. Calculate the 16 subkeys. Start with i = 1.

    1. Perform one or two circular left shifts on both C[i-1] and D[i-1] to get C[i] and D[i], respectively. The number of shifts per iteration are given in the table below.

       Iteration # 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

       Left Shifts 1 1 2 2 2 2 2 2 1 2 2 2 2 2 2 1

    2. Permute the concatenation C[i]D[i] as indicated below. This will yield K[i], which is 48 bits long.
    3. Loop back to 1.ii.c.1 until K[16] has been calculated.

2 Process a 64-bit data block.

- i. Get a 64-bit data block. If the block is shorter than 64 bits, it should be padded as appropriate for the application.
- ii. Perform the initial permutation on the data block.
- iii. Split the block into two halves. The first 32 bits are called L[0], and the last 32 bits are called R[0].
- iv. Apply the 16 subkeys to the data block. Start with i = 1.

  a. Expand the 32-bit R[i-1] into 48 bits according to the bit-selection function Expansion (E)

  b. Exclusive-or E(R[i-1]) with K[i].

  c. Break E(R[i-1]) xor K[i] into eight 6-bit blocks. Bits 1-6 are B[1], bits 7-12 are B[2], and so on with bits 43-48 being B[8].

  d. Substitute the values found in the S-boxes for all B[j]. Start with j = 1. All values in the S-boxes should be considered 4 bits wide.

    - i. Take the 1st and 6th bits of B[j] together as a 2-bit value (call it m) indicating the row in S[j] to look in for the substitution.
    - ii. Take the 2nd through 5th bits of B[j] together as a 4-bit value(call it n) indicating the column in S[j] to find the substitution.
    - iii. Replace B[j] with S[j][m][n].
    - iv. Loop back to 2.iv.d.i until all 8 blocks have been replaced.

  e. Permute the concatenation of B[1] through B[8]

  f. Exclusive-or the resulting value with L[i-1]. Thus, all together, your R[i] = L[i-1] xor P(S[1](B[1])...S[8](B[8])), where B[j] is a 6-bit block of E(R[i-1]) xor K[i]. (The function for R[i] is written as, R[i] = L[i-1] xor f(R[i-1], K[i]).)

  g. L[i] = R[i-1].

  h. Loop back to 2.iv.a until K[16] has been applied.

- v. Perform the final permutation on the block R[16]L[16].

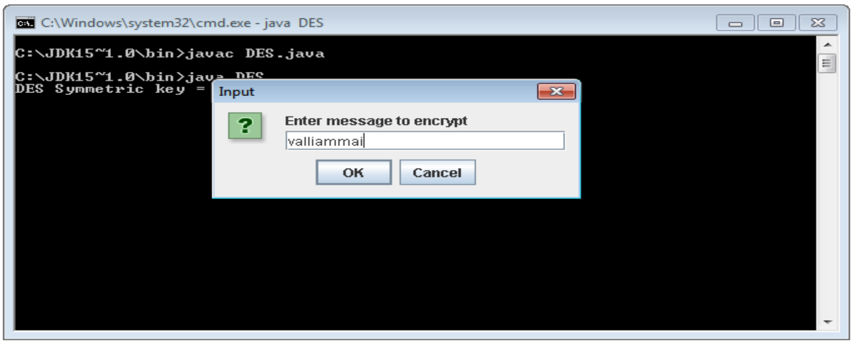
3. Decryption : Use the keys K[i] in reverse order. That is, instead of applying K[1] for the first iteration, apply K[16], and then K[15] for the second, on down to K[1]

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();
}
}
```

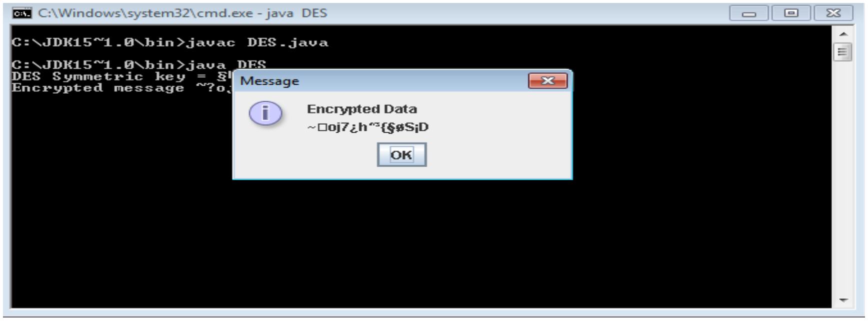
OUTPUT:



```
C:\JDK15~1.0\bin>javac DES.java

C:\JDK15~1.0\bin>java DES
DES Symmetric key =
```

Enter message to encrypt

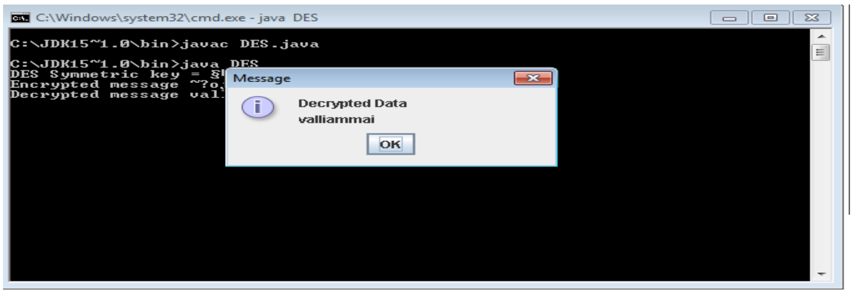
valliammai



```
C:\JDK15~1.0\bin>javac DES.java

C:\JDK15~1.0\bin>java DES
DES Symmetric key = §
Encrypted message ~?oj
```

Encrypted Data

~□oj7¿h^³{§øS¡D



```
C:\JDK15~1.0\bin>javac DES.java

C:\JDK15~1.0\bin>java DES
DES Symmetric key = §
Encrypted message ~?o
Decrypted message val
```

Decrypted Data
valliammai

VIVA QUESTIONS (PRELAB and POSTLAB):

1. DES follows which basic stream cipher?
2. The DES Algorithm Cipher System consists of how many rounds (iterations) each with a round key?
3. What is the key length of the DES algorithm?
4. In the DES algorithm, although the key size is 64 bits only 48bits are used for the encryption procedure, the rest are parity bits. Is it true or false?
5. In the DES algorithm, what is the size of the round key and the Round Input?
6. In the DES algorithm how the Round Input is expanded to 48 ?
7. What is size of the Initial Permutation table/matrix
8. How many unique substitution boxes are in DES after the 48 bit XOR operation?
9. In the DES algorithm the 64 bit key input is shortened to 56 bits by ignoring every 4th bit. Is it true or false?

<u>RESULT:</u>

Thus the program to implement DES encryption technique was developed and executed successfully

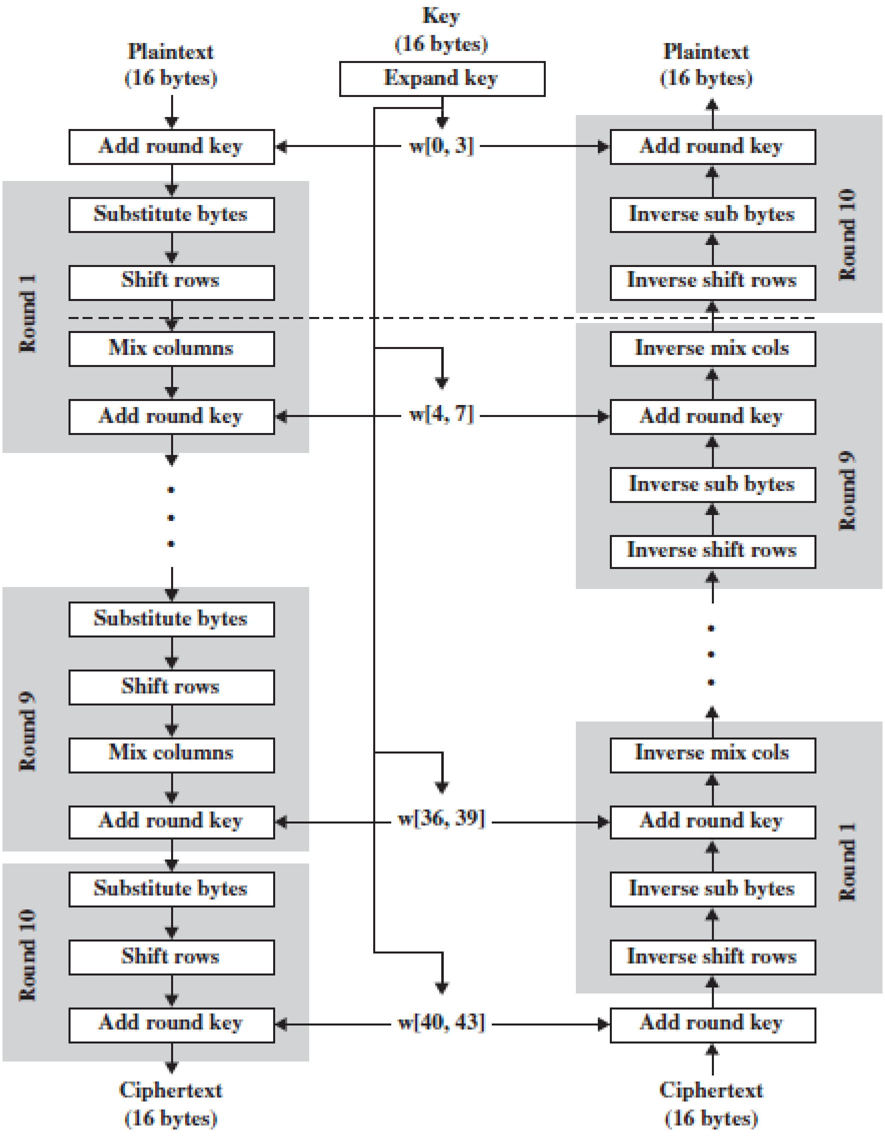
EX.No.: 2 AES ALGORITHM

AIM:

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

PRELAB DISCUSSION:

The cipher takes a plaintext block size of 128 bits, or 16 bytes. The key length can be 16, 24, or 32 bytes (128, 192, or 256 bits). The algorithm is referred to as AES-128, AES-192, or AES-256, depending on the key length.



(a) Encryption (b) Decryption

The input to the encryption and decryption algorithms is a single 128-bit block. In FIPS PUB 197, this block is depicted as a 4 * 4 square matrix of bytes. This block is copied into the **State**array, which is modified at each stage of encryption or decryption. After the final stage, **State** is copied to an output matrix. Similarly, the key is depicted as a square matrix of bytes. This key is then expanded into an array of key schedule words. Each word is four bytes, and the total key schedule is 44 words for the 128-bit key. Note that the ordering of bytes within a matrix is by column. So, for example, the first four bytes of a 128-bit plaintext input to the encryption cipher occupy the first column of the **in** matrix, the second four bytes occupy the second column, and so on. Similarly, the first four bytes of the expanded key, which form a word, occupy the first column of the **w** matrix.

The cipher consists of *N* rounds, where the number of rounds depends on the key length: 10 rounds for a 16-byte key, 12 rounds for a 24-byte key, and 14 rounds for a 32-byte key. The first *N* - 1 rounds consist of four distinct transformation functions: SubBytes, ShiftRows, MixColumns, and AddRoundKey. The final round contains only three transformations, and there is a initial single transformation (AddRoundKey) before the first round, which can be considered Round 0. Each transformation takes one or more 4 * 4 matrices

PROGRAM:

```
package com.includehelp.stringsample;

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

/**
 * Program to Encrypt/Decrypt String Using AES 128 bit Encryption Algorithm
 */
public class EncryptDecryptString
{
        private static final String encryptionKey          = "ABCDEFGHIJKLMNOP";
        private static final String characterEncoding       = "UTF-8";
        private static final String cipherTransformation    = "AES/CBC/PKCS5PADDING";
        private static final String aesEncryptionAlgorithem = "AES";
   /**
   * Method for Encrypt Plain String Data
   * @param plainText
   * @return encryptedText
   */
        public static String encrypt(String plainText)
        {
                String encryptedText = "";
                try
                {
                     Cipher cipher = Cipher.getInstance(cipherTransformation);
                      byte[] key      = encryptionKey.getBytes(characterEncoding);
                      SecretKeySpec secretKey = new SecretKeySpec(key, aesEncryptionAlgorithem);
                      IvParameterSpec ivparameterspec = new IvParameterSpec(key);
                      cipher.init(Cipher.ENCRYPT_MODE, secretKey, ivparameterspec);
                      byte[] cipherText = cipher.doFinal(plainText.getBytes("UTF8"));
```

```
                    Base64.Encoder encoder = Base64.getEncoder();
                    encryptedText = encoder.encodeToString(cipherText);
                } catch (Exception E)
                {
                        System.err.println("Encrypt Exception : "+E.getMessage());
                 }
                return encryptedText;
            }
    /**

     * Method For Get encryptedText and Decrypted provided String
     * @param encryptedText
     * @return decryptedText
     */
        public static String decrypt(String encryptedText)
        {
                String decryptedText = "";
                try
                {
                        Cipher cipher = Cipher.getInstance(cipherTransformation);
                        byte[] key = encryptionKey.getBytes(characterEncoding);
                    SecretKeySpec secretKey = new SecretKeySpec(key, aesEncryptionAlgorithem);
                        IvParameterSpec ivparameterspec = new IvParameterSpec(key);
                        cipher.init(Cipher.DECRYPT_MODE, secretKey, ivparameterspec);
                        Base64.Decoder decoder = Base64.getDecoder();
                        byte[] cipherText = decoder.decode(encryptedText.getBytes("UTF8"));
                        decryptedText = new String(cipher.doFinal(cipherText), "UTF-8");
                } catch (Exception E)
                {
                        System.err.println("decrypt Exception : "+E.getMessage());
                }
                return decryptedText;
        }

        public static void main(String[] args)
        {
                Scanner sc = new Scanner(System.in);
                System.out.println("Enter String : ");
                String plainString = sc.nextLine();

                String encyptStr = encrypt(plainString);
                String decryptStr = decrypt(encyptStr);

                System.out.println("Plain   String  :  "+plainString);
                System.out.println("Encrypt  String  :  "+encyptStr);
                System.out.println("Decrypt String : "+decryptStr);
        }  }
```

OUTPUT:

```
Enter String : Hello World
Plain String : Hello World
Encrypt String : IMfL/ifkuvkZwG/v2bn6Bw==
Decrypt String : Hello World
```

VIVA QUESTIONS (PRELAB and POSTLAB):

This set of Cryptography Multiple Choice Questions & Answers (MCQs) focuses on “The Data Encryption Standard (DES) and It’s Strength”.

1. DES follows

a)Hash Algorithm   b)Caesars Cipher   c)Feistel Cipher Structure   d)SP Networks

2. The DES Algorithm Cipher System consists of ________ _ rounds (iterations) each with a round key

a) 12   b) 18   c) 9   d) 16

3. The DES algorithm has a key length of

a) 128 Bits   b) 32 Bits   c) 64 Bits   d) 16 Bits

4. In the DES algorithm, although the key size is 64 bits only 48bits are used for the encryption procedure, the rest are parity bits.

a) True   b) False

5. In the DES algorithm the round key is______ ___bit and the Round Input is ____ ________ bits.

a) 48, 32   b) 64,32   c) 56, 24   d) 32, 32

6. In the DES algorithm the Round Input is 32 bits, which is expanded to 48 bits via _____ _ __

a) Scaling of the existing bits   b) Duplication of the existing bits

c) Addition of zeros   d) Addition of ones

7. The Initial Permutation table/matrix is of size

a) 16×8   b) 12×8   c) 8×8   d) 4×8

8. The number of unique substitution boxes in DES after the 48 bit XOR operation are

a) 8   b) 4   c) 6   d) 12

9. In the DES algorithm the 64 bit key input is shortened to 56 bits by ignoring every 4th bit.

a) True   b) False

RESULT:

Thus the program to implement AES encryption technique was developed and executed successfully.

EX.No.: 3	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

PRELAB DISCUSSION:

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.

ALGORITHM:

1. Key Generation
   i. Choose two distinct prime numbers p and q.
   ii. Find n such that n = pq, n will be used as the modulus for both the public and private keys.
   iii. Find the totient of n, $\phi(n)$ $\phi(n)=(p-1)(q-1)$
   iv. Choose an e such that $1 < e < \phi(n)$, and such that e and $\phi(n)$ share no divisors other than 1 (e and $\phi(n)$ are relatively prime). e is kept as the public

key exponent

v. Determine d (using modular arithmetic) which satisfies the congruence relation

$de \equiv 1 \pmod{\phi(n)}$.

The public key has modulus n and the public (or encryption) exponent e. The private key has modulus n and the private (or decryption) exponent d, which is kept secret.

2. Encryption

   $c \equiv m^e \pmod{n}$.

3. Decryption:

   $m \equiv c^d \pmod{n}$.

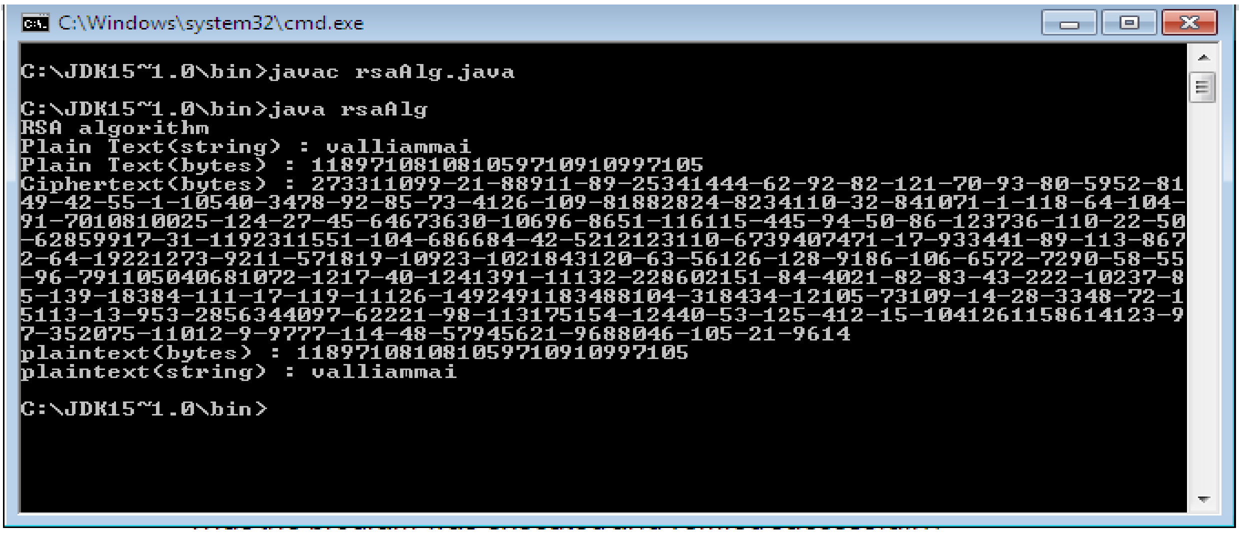
4. Stop.

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.toString(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.a dd(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:



```
C:\JDK15~1.0\bin>javac rsaAlg.java

C:\JDK15~1.0\bin>java rsaAlg
RSA algorithm
Plain Text(string) : valliammai
Plain Text(bytes) : 118971081081059710910997105
Ciphertext(bytes) : 273311099-21-88911-89-25341444-62-92-82-121-70-93-80-5952-81
49-42-55-1-10540-3478-92-85-73-4126-109-81882824-8234110-32-841071-1-118-64-104-
91-7010810025-124-27-45-64673630-10696-8651-116115-445-94-50-86-123736-110-22-50
-62859917-31-1192311551-104-686684-42-5212123110-6739407471-17-933441-89-113-867
2-64-19221273-9211-571819-10923-1021843120-63-56126-128-9186-106-6572-7290-58-55
-96-791105040681072-1217-40-1241391-11132-228602151-84-4021-82-83-43-222-10237-8
5-139-18384-111-17-119-11126-1492491183488104-318434-12105-73109-14-28-3348-72-1
5113-13-953-2856344097-62221-98-113175154-12440-53-125-412-15-1041261158614123-9
7-352075-11012-9-9777-114-48-57945621-9688046-105-21-9614
plaintext(bytes) : 118971081081059710910997105
plaintext(string) : valliammai

C:\JDK15~1.0\bin>
```

VIVA QUESTIONS (PRELAB and POSTLAB):

1. What Is RSA?
2. Are Strong Primes Necessary In RSA?
3. Can Users Of RSA Run Out Of Distinct Primes?
4. What Are The Alternatives To RSA?
5. How fast is RSA?
6. What would it take to break RSA?
7. How large a modulus key should be used in RSA?
8. How large should the primes be?
9. How do you know if a number is prime?
10. How is RSA used for authentication in practice?
11. What do you mean by Secret Key Cryptography and Public Key Cryptography? How they are different from one another?
12. What exactly do you know about RSA?

RESULT:

Thus the program for implementation of RSA algorithm was executed and verified successfully.

EX.No.: 4	DIFFIEE HELLMAN KEY EXCHANGE ALGORITHM

AIM:

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

PRELAB DISCUSSION:

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. The 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 key cipher. This algorithm uses arithmetic modulus as the basis of its calculation. Suppose Alice and Bob follow this key exchange procedure with Eve acting as a man in middle interceptor (or the bad guy).

Here are the calculation steps followed in this algorithm that make sure that eve never gets to know the final keys through which actual encryption of data takes place. First, both Alice and Bob agree upon a prime number and another number that has no factor in common. Lets call the prime number as **p** and the other number as **g**. Note that **g** is also known as the generator and **p** is known as prime modulus. Now, since eve is sitting in between and listening to this communication so eve also gets to know **p** and **g**. Now, the modulus arithmetic says that **r** = (**g** to the power **x**) mod **p.** So **r** will always produce an integer between 0 and **p**. The first trick here is that given **x** (with **g** and **p** known) , its very easy to find **r**. But given **r** (with **g** and **p** known) its difficult to deduce **x**. One may argue that this is not that difficult to crack but what if the value of **p**is a very huge prime number? Well, if this is the case then deducing **x** (if **r** is given) becomes almost next to impossible as it would take thousands of years to crack this even with supercomputers. This is also called the discrete logarithmic problem. Coming back to the communication, all the three Bob, Alice and eve now know **g** and **p**. Now, Alice selects a random private number **xa** and calculates (**g** to the power **xa**) mod **p** = **ra**. This resultant **ra** is sent on the communication channel to Bob. Intercepting in between, eve also comes to know **ra**. Similarly Bob selects his own random private number **xb**, calculates (**g** to the power **xb**) mod **p** = **rb** and sends this **rb** to Alice through the same communication channel. Obviously eve also comes to know about **rb**. So eve now has information about **g**, **p**, **ra** and **rb**. Now comes the heart of this algorithm. Alice calculates (**rb** to the power **xa**) mod **p** = **Final key** which is equivalent to **(g to the power (xa*xb)) mod p**. Similarly Bob calculates **(ra to the power xb) mod p** = **Final key** which is again equivalent to **(g to the power(xb * xa)) mod p**. So both Alice and Bob were able to calculate a common **Final key** without sharing each others private random number and eve sitting

in between will not be able to determine the **Final key** as the private numbers were never transferred.

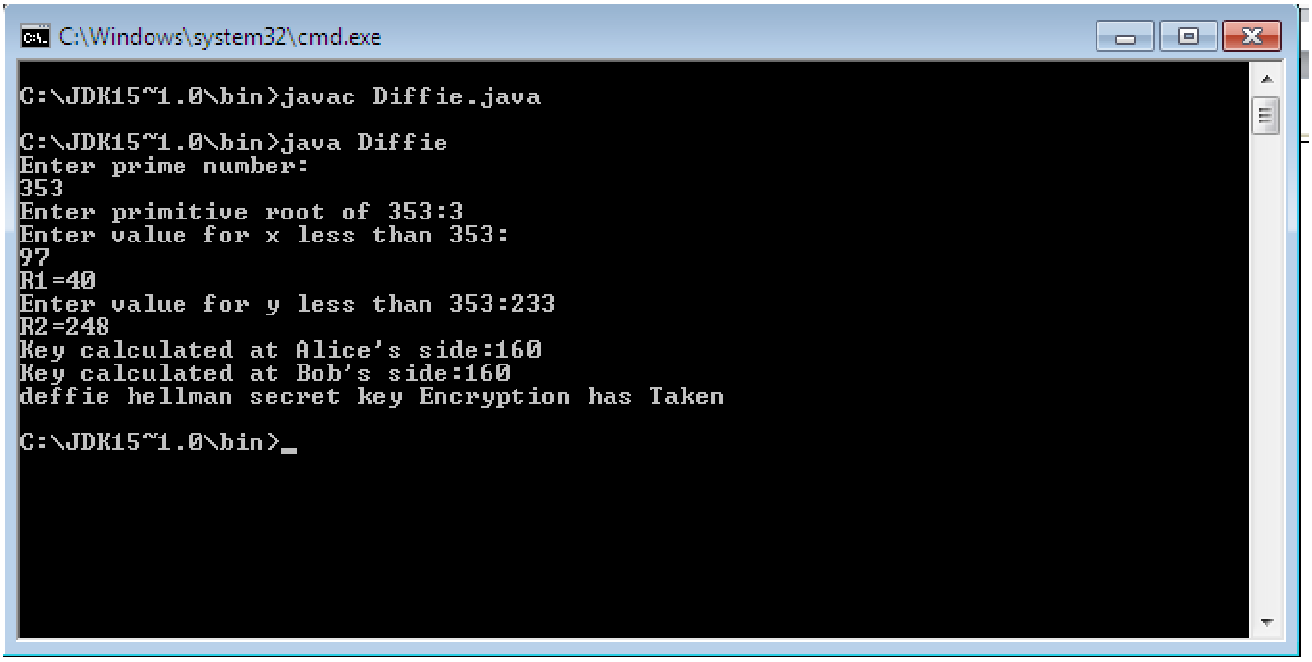
ALGORITHM

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

PROGRAM:

```
import java.io.*;
import java.math.BigInteger;
class Diffie
{
   public static void main(String[]args)throws IOException
   {
      BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
      System.out.println("Enter prime number:");
      BigInteger p=new BigInteger(br.readLine());
      System.out.print("Enter primitive root of "+p+":");
      BigInteger g=new BigInteger(br.readLine());
      System.out.println("Enter value for x less than "+p+":");
      BigInteger x=new BigInteger(br.readLine());
      BigInteger R1=g.modPow(x,p);
      System.out.println("R1="+R1);
      System.out.print("Enter value for y less than "+p+":");
      BigInteger y=new BigInteger(br.readLine());
      BigInteger R2=g.modPow(y,p);
      System.out.println("R2="+R2);
      BigInteger k1=R2.modPow(x,p);
      System.out.println("Key calculated at Sender's side:"+k1);
      BigInteger k2=R1.modPow(y,p);
      System.out.println("Key calculated at Receiver's side:"+k2);
      System.out.println("deffie hellman secret key Encryption has Taken");
   }
}
```

OUTPUT:



```
C:\JDK15~1.0\bin>javac Diffie.java

C:\JDK15~1.0\bin>java Diffie
Enter prime number:
353
Enter primitive root of 353:3
Enter value for x less than 353:
97
R1=40
Enter value for y less than 353:233
R2=248
Key calculated at Alice's side:160
Key calculated at Bob's side:160
deffie hellman secret key Encryption has Taken

C:\JDK15~1.0\bin>_
```

VIVA QUESTIONS (PRELAB and POSTLAB):

1. What's the difference between Diffie-Hellman and RSA?
2. Does Diffie Hellman guarantee secrecy?
3. Why is RSA preferred over Diffie-Hellman if they are both used to establish shared key?
4. Are there any one way operations that could be used for Diffie-Hellman post quantum?
5. Why is Diffie-Hellman required whenRSA is already used for key exchange in TLS?
6. What is Authenticated Diffie-Hellman Key Agreement?
7. How secure is ECDH if the public keys are never shared?
8. Which is better when the secret is leaked, RSA or Diffie-Hellman?
9. What role does RSA play in DH-RSA cipher suite?
10. Why is Diffie-Hellman used alongside public keys?
11. Is Diffie-Hellman key exchange based on one-way function or trapdoor function?

RESULT:

Thus the program to implement Diffie-Hellman Key Exchange algorithm was developed and executed successfully

EX.No.: 5	**IMPLEMENT SECURE HASH FUNCTION (SHA)**

AIM:

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

PRELAB DISCUSSION:

In cryptography, SHA-1 (Secure Hash Algorithm 1) is a cryptographic hash function which takes an input and produces a 160-bit (20-byte) hash value known as a message digest - typically rendered as a hexadecimal number, 40 digits long.

Secure Hashing Algorithms, also known as SHA, are a family of cryptographic functions designed to keep data secured. It works by transforming the data using a hash function: an algorithm that consists of bitwise operations, modular additions, and compression functions. The hash function then produces a fixed size string that looks nothing like the original. These algorithms are designed to be one-way functions, meaning that once they’re transformed into their respective hash values, it’s virtually impossible to transform them back into the original data. A few algorithms of interest are SHA-1, SHA-2, and SHA-5, each of which was successively designed with increasingly stronger encryption in response to hacker attacks. SHA-0, for instance, is now obsolete due to the widely exposed vulnerabilities.

A common application of SHA is to encrypting passwords, as the server side only needs to keep track of specific user’s hash value, rather than the actual password. This is helpful in case an attacker hacks the database, as they will only find the hashed functions and not the actual passwords, so if they were to input the hashed value as a password, the hash function will convert it into another string and subsequently deny access. Additionally, SHA exhibit the avalanche effect, where the modification of very few letters being encrypted cause a big change in output; or conversely, drastically different strings produce similar hash values. This effect causes hash values to not give any information regarding the input string, such as its original length. In addition, SHAs are also used to detect the tampering of data by attackers, where if a text file is slightly changed and barely noticeable, the modified file’s hash value will be different than the original file’s hash value, and the tampering will be rather noticeable.

ALGORITHM:

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.
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.
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)

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)

5. Initialize Buffers: SHA1 requires 160 bits or 5 buffers of words (32 bits):

   H0 = 0x67452301 H1 = 0xEFCDAB89
   H2 = 0x98BADCFE H3 = 0x10325476
   H4 = 0xC3D2E1F0

6. Processing Message in 512-bit blocks (L blocks in total message)

   i. This is the main task of SHA1 algorithm which loops through the padded and appended message in 512-bit blocks.

   ii. 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

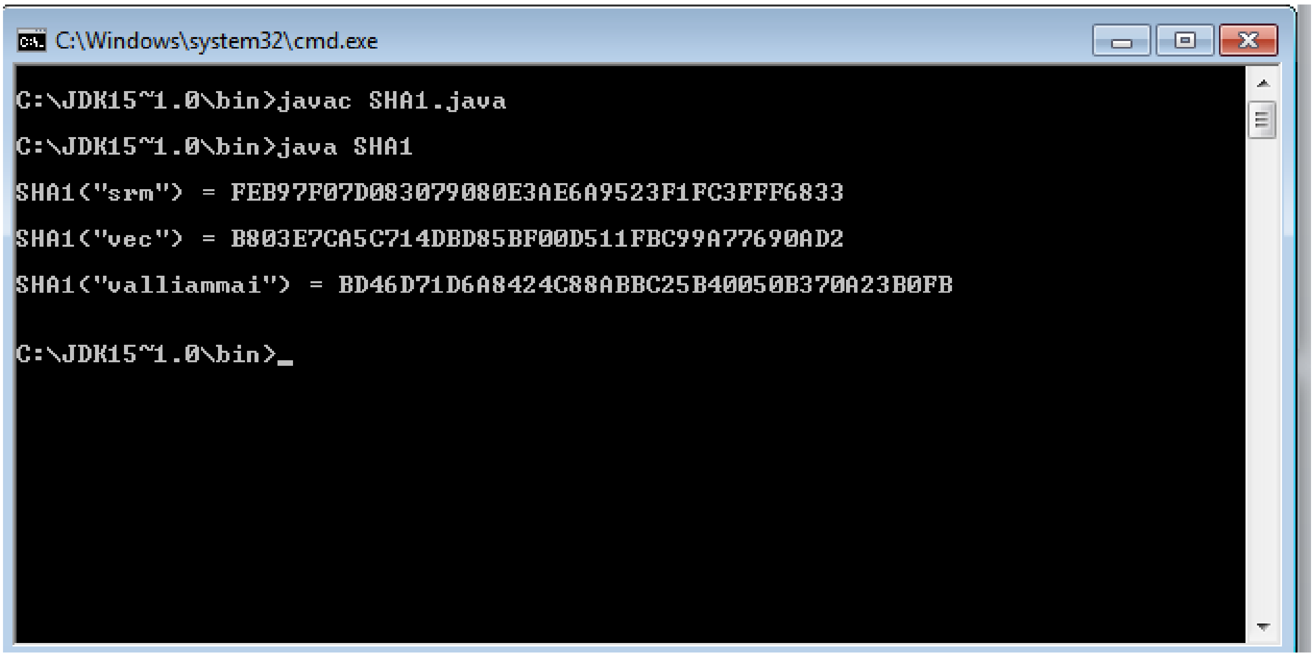
   K(0), K(1), ..., K(79): 80 Processing Constant Words
   H0, H1, H2, H3, H4, H5: 5 Word buffers with initial values

```
7.  For loop on k = 1 to L
            1. (W(0),W(1),...,W(15)) = M[k] /* Divide M[k] into 16 words */
8.  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
```

PROGRAM

```
import java.security.*;
public class SHA1 {
public static void main(String[] a) {
try {
MessageDigest md = MessageDigest.getInstance("SHA1");
 String input = "srm";
md.update(input.getBytes());
byte[] output = md.digest();
System.out.println();
System.out.println("SHA1(\""+input+"\") = " +bytesToHex(output));
input = "vec";
md.update(input.getBytes());
output = md.digest();
System.out.println();
System.out.println("SHA1(\""+input+"\") = " +bytesToHex(output));
input = "valliammai";
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:\JDK15~1.0\bin>javac SHA1.java

C:\JDK15~1.0\bin>java SHA1

SHA1("srm") = FEB97F07D083079080E3AE6A9523F1FC3FFF6833

SHA1("vec") = B803E7CA5C714DBD85BF00D511FBC99A77690AD2

SHA1("valliammai") = BD46D71D6A8424C88ABBC25B40050B370A23B0FB

C:\JDK15~1.0\bin>_
```

VIVA QUESTIONS (PRELAB and POSTLAB):

1. SHA-1 produces a hash value of how many bits?
2. What is the number of round computation steps in the SHA-256 algorithm?
3. In SHA-512, the message is divided into blocks size of how many bits for the hash computation.
4. What is the maximum length of the message (in bits) that can be taken by SHA-512?
5. What is the length of the message in SHA-512 after it is padded?
6. Describe the big-endian format?
7. In SHA-512, the registers 'a' to 'h' are obtained by taking the first 64 bits of the fractional parts of the cube roots of the first 8 prime numbers. Is it true or false?
8. What is the size of W (in bits) in the SHA-512 processing of a single 1024- bit block?
9. In the SHA-512 processing of a single 1024- bit block, how the round constants are obtained?
10. What is the maximum length of the message (in bits) that can be taken by SHA-512?
11. What does the figure represent?
12. Among the registers 'a' to 'h' how many involve permutation in each round?

RESULT:

Thus the program to implement Secure Hash Algorithm was developed and executed successfully.

EX.No.: 6	IMPLEMENT DIGITAL SIGNATURE SCHEME

AIM:

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

PRELAB DISCUSSION:

Digital signatures are based on public key cryptography, also known as asymmetric cryptography. Using a public key algorithm such as RSA, one can generate two keys that are mathematically linked: one private and one public. To create a digital signature, signing software (such as an email program) creates a one-way hash of the electronic data to be signed. The private key is then used to encrypt the hash. The encrypted hash -- along with other information, such as the hashing algorithm -- is the digital signature. The reason for encrypting the hash instead of the entire message or document is that a hash function can convert an arbitrary input into a fixed length value, which is usually much shorter. This saves time since hashing is much faster than signing. The value of the hash is unique to the hashed data. Any change in the data, even changing or deleting a single character, results in a different value. This attribute enables others to validate the integrity of the data by using the signer's public key to decrypt the hash. If the decrypted hash matches a second computed hash of the same data, it proves that the data hasn't changed since it was signed. If the two hashes don't match, the data has either been tampered with in some way (integrity) or the signature was created with a private key that doesn't correspond to the public key presented by the signer (authentication).

A digital signature can be used with any kind of message -- whether it is encrypted or not -- simply so the receiver can be sure of the sender's identity and that the message arrived intact. Digital signatures make it difficult for the signer to deny having signed something (non-repudiation)
-- assuming their private key has not been compromised -- as the digital signature is unique to both the document and the signer, and it binds them together. A digital certificate, an electronic document that contains the digital signature of the certificate-issuing authority, binds together a public key with an identity and can be used to verify a public key belongs to a particular person or entity. Most modern email programs support the use of digital signatures and digital certificates, making it easy to sign any outgoing emails and validate digitally signed incoming messages. Digital signatures are also used extensively to provide proof of authenticity, data integrity and non- repudiation of communications and transactions conducted over the Internet.

ALGORITHM:

1. Choose a prime number q, which is called the prime divisor.
2. Choose another primer number p, such that p-1 mod q = 0. p is called the prime modulus.
3. Choose an integer g, such that 1 < g < p, g**q mod p = 1 and g = h**((p–1)/q) mod p. q is also called g's multiplicative order modulo p.
4. Choose an integer, such that 0 < x < q.
5. Compute y as g**x mod p.
6. Package the public key as {p,q,g,y}, {p,q,g,x}.

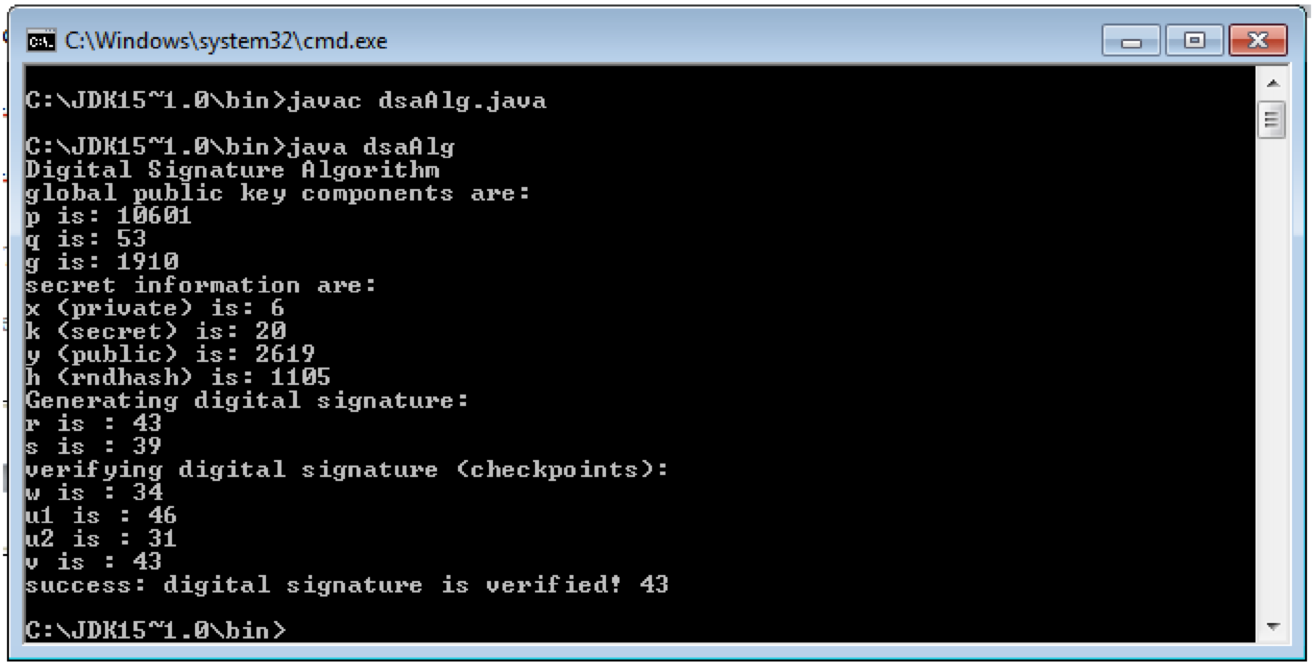
7. Generate the message digest h, using a hash algorithm like SHA1.
8. Generate a random number k, such that 0 < k < q.
9. Compute r as (g**k mod p) mod q. If r = 0, select a different k.
10. Compute i, such that k*i mod q = 1. i is called the modular multiplicative inverse of k modulo q.
11. Compute s = i*(h+r*x) mod q. If s = 0, select a different k.
12. Package the digital signature as {r,s}.
13. Generate the message digest h, using the same hash algorithm.
14. Compute w, such that s*w mod q = 1. w is called the modular multiplicative inverse of s modulo q.
15. Compute u1 = h*w mod q. Compute u2 = r*w mod q.
16. Compute v = (((g**u1)*(y**u2)) mod p) mod q.
17. If v == r, the digital signature is valid.

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("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



```
C:\JDK15~1.0\bin>javac dsaAlg.java

C:\JDK15~1.0\bin>java dsaAlg
Digital Signature Algorithm
global public key components are:
p is: 10601
q is: 53
g is: 1910
secret information are:
x (private) is: 6
k (secret) is: 20
y (public) is: 2619
h (rndhash) is: 1105
Generating digital signature:
r is : 43
s is : 39
verifying digital signature (checkpoints):
w is : 34
u1 is : 46
u2 is : 31
v is : 43
success: digital signature is verified! 43

C:\JDK15~1.0\bin>
```

VIVA QUESTIONS (PRELAB and POSTLAB):

1. What is a digital signature?
2. What does a digital signature look like?
3. What is an electronic document?
4. Does that mean that the authenticity of any electronic document can be verified by a digital signature?
5. What is it like to actually sign an electronic document?
6. Can you actually see the signer's handwritten signature?
7. How do I get a digital signature certificate?
8. What is a certificate? What does it mean to "publish" a certificate?
9. How am I identified as the signer?
10. If my private key is stored on my computer, can't someone sign the documents without my permission by getting access to the computer?
11. Can a digital signature be forged?
12. What are the responsibilities and the liability of a digital signature certificate subscriber?
13. What are the practical uses of a digital signature?

RESULT:

Thus the program to implement Digital Signature was developed and executed successfully

EX.No.: 7	**DEMONSTRATE INTRUSION DETECTION SYSTEM (IDs) USING ANY TOOL (SNORT OR ANY OTHER S/W)**

AIM:

To demonstrate intrusion detection system (ids) using the tool snort

PRELAB DISCUSSION and PROCEDURE:

1. Configure and Use Snort IDS on Windows

Steps to configure Snort on Widnows machine and how to use it for detection of attacks.

2. Steps:

1. Download Snort from "http://www.snort.org/" website.

2. Also download Rules from the same website. You need to sign up to get rules for registered users.

3. Click on the Snort_(version-number)_Installer.exe file to install it. By-default it will install snort in the "C:\Snort" directory.

4. Extract downloaded Rules file: snortrules-snapshot-(number).tar.gz

5. Copy all files from the "rules" directory of the extracted folder and paste them into "C:\Snort\rules" directory.

6. Copy "snort.conf" file from the "etc" directory of the extracted folder and paste it into "C:\Snort\etc" directory. Overwrite existing file if there is any.

7. Open command prompt (cmd.exe) and navigate to directory "C:\Snort\bin" directory.

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

```
snort -dev -i 2
```

-i indicate interface number.

-dev is used to run snort to capture packets.

To check interface list use following command: snort -W

9. To execute snort in IDS mode, we need to configure a file "snort.conf" according to our network environment.

10. Set up network address we want to protect in snort.conf file. To dothat look for "HOME_NET" and add your IP address.

```
var HOME_NET 10.1.1.17/8
```

11. You can also set addresses or DNS_SERVERS, if you have any. otherwise go to the next step.

12. Change RULE_PATH variable with the path of rules directory.

```
var RULE_PATH c:\snort\rules
```

13. Change the path of all libraries with the name and path on your system. or change path of snort_dynamicpreprocessorvariable.

```
sor file C:\Snort\lib\snort_dynamiccpreprocessor\sf_dcerpc.dll
```

You need to do this to all library files in the "C:\Snort\lib" directory. The old path might be something like: "/usr/local/lib/...". you need to replace that path with you system path.

14. Change path of the "dynamicengine" variable value in the "snort.conf" file with the path of your system. Such as:

```
dynamicengine C:\Snort\lib\snort_dynamicengine\sf_engine.dll
```

15 Add complete path for "include classification.config" and "include reference.config" files.

```
include c:\snort\etc\classification.config
include c:\snort\etc\reference.config
```

16. Remove the comment on the line to allow **ICMP** rules, if it is alredy commented.

```
include $RULE_PATH/icmp.rules
```

17. Similary, remove the comment of ICMP-info rules comment, if it is already commented.

```
include $RULE_PATH/icmp-info.rules
```

18 To add log file to store alerts generated by snort, search for "output log" test and add following line:

```
output alert_fast: snort-alerts.ids
```

19. Comment whitelist $WHITE_LIST_PATH/white_list.rules and blacklist $BLACK_LIST_PATH/black_list.rules lines. Also ensure that you add change the line above $WHITE_LIST_PATH

Change nested_ip inner , \ to nested_ip inner #, \

20. Comment following lines:

```
#preprocessor  normalize_ip4
#preprocessor normalize_tcp: ips ecn stream
#preprocessor normalize_icmp4
#preprocessor normalize_ip6
#preprocessor normalize_icmp6
```

21. Save the "snort.conf" file and close it.

22. Go to the "C:\Snort\log" directory and create a file: snort-alerts.ids

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

```
snort -c c:\snort\etc\snort.conf -l c:\snort\log -i 2
```

Above command will generate log file that will not be readable without using a tool. To read it use following command:

```
C:\Snort\Bin\> snort -r ..\log\log-filename
```

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

```
snort -A console -i2 -c c:\Snort\etc\snort.conf -l c:\Snort\log -K ascii
```

24. Scan the computer running snort from another computer using PING or launch attack. Then check snort-alerts.ids file the log folder.

VIVA QUESTIONS (PRE LAB and POSTLAB)

1. When discussing IDS/IPS, what is a signature?
2. "Semantics-aware" signatures automatically generated by Nemean are based on traffic at which two layers?
3. Which of the following is used to provide a baseline measure for comparison of IDSes?
4. What Is Ips And Ids?
5. What Are The Functions Of Intrusion Detection?
6. What Is Ids In Networking?
7. Explain Host Based (hids)?
8. What Is An Intrusion Detection System?
9. What is the advantage of anomaly detection?
10. Define a false positive.
11. One of the most obvious places to put an IDS sensor is near the firewall. Where exactly in relation to the firewall is the most productive placement?
12. What is the purpose of a shadow honeypot?
13. At which two traffic layers do most commercial IDSes generate signatures?
14. An IDS follows a two-step process consisting of a passive component and an active component.
15. Which of the following is part of the active component?

RESULT:

Thus the intrusion detection system (ids) using the tool snort program was demonstrated and verified successfully.

EX.No.: 8	AUTOMATED ATTACK AND PENETRATION TOOLS EXPLORING N-STALKER, A VULNERABILITY ASSESSMENT TOOL

AIM:

To explore automated and penetration tools on network (KF Sensor)

PRELAB DISCUSSION:

HONEYPOTS

When it comes to computer security, honeypots are all the rage. Honeypots can detect unauthorized activities that might never be picked up by a traditional intrusion detection system. Furthermore, since almost all access to a honeypot is unauthorized, nearly everything in a honeypot's logs is worth paying attention to. Honeypots can act as a decoy to keep hackers away from your production servers. At the same time though, a honeypot can be a little tricky to deploy. In this article, I will walk you through the process of deploying a honeypot.

INTRODUCTION

There are many different types of honeypot systems. Honeypots can be hardware appliances or they can be software based. Software based firewalls can reside on top of a variety of operating systems. For the most part though, honeypots fall into two basic categories; real and virtual.

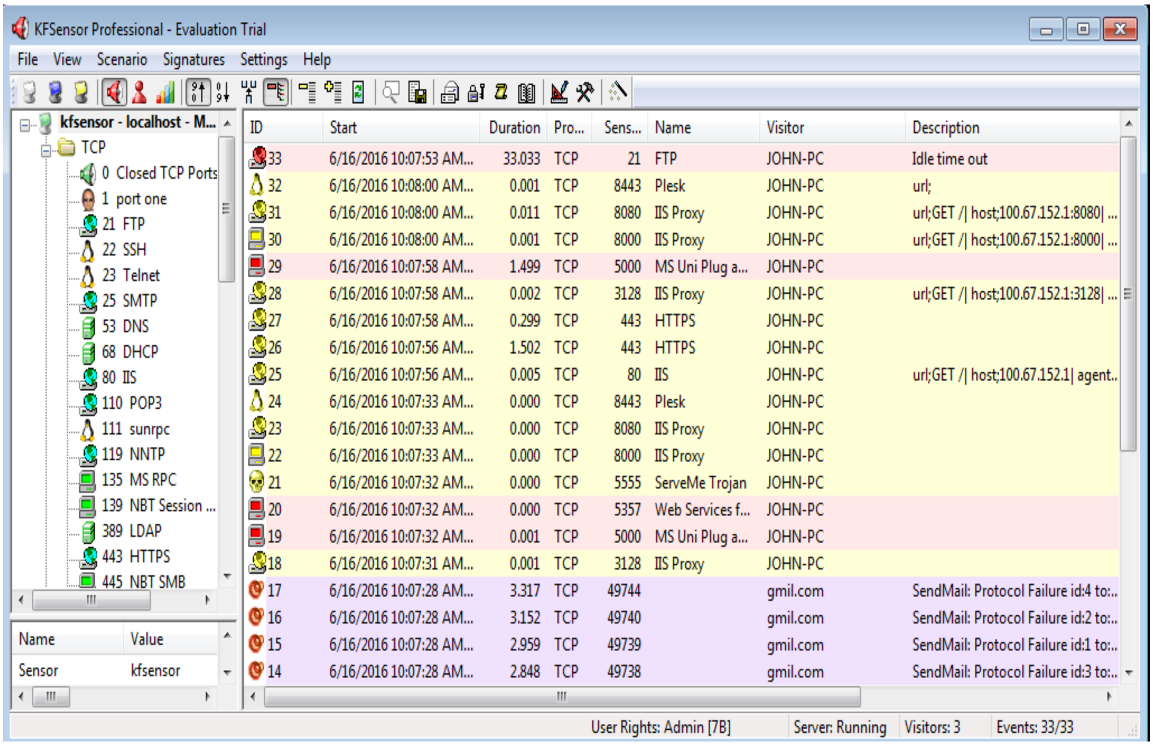
A virtual honeypot is essentially an emulated server. There are both hardware and software implementations of virtual honeypots. For example, if a network administrator was concerned that someone might try to exploit an FTP server, the administrator might deploy a honeypot appliance that emulates an FTP server.

Downloading and installing KF Sensor

- The KF Sensor download consists of a 1.7 MB self-extracting executable file.
- Download the file and copy it into an empty folder on your computer.
- When you double click on the file, it will launch a very basic Setup program.
- The only thing special that you need to know about the Setup process is that it will require a reboot

Using KFSensor

Step1: You will see the main KFSensor screen shown



kfsensor - localhost - M...
TCP
0 Closed TCP Ports
1 port one
21 FTP
22 SSH
23 Telnet
25 SMTP
53 DNS
68 DHCP
80 IIS
110 POP3
111 sunrpc
119 NNTP
135 MS RPC
139 NBT Session ...
389 LDAP
443 HTTPS
445 NBT SMB

Name	Value
Sensor	kfsensor

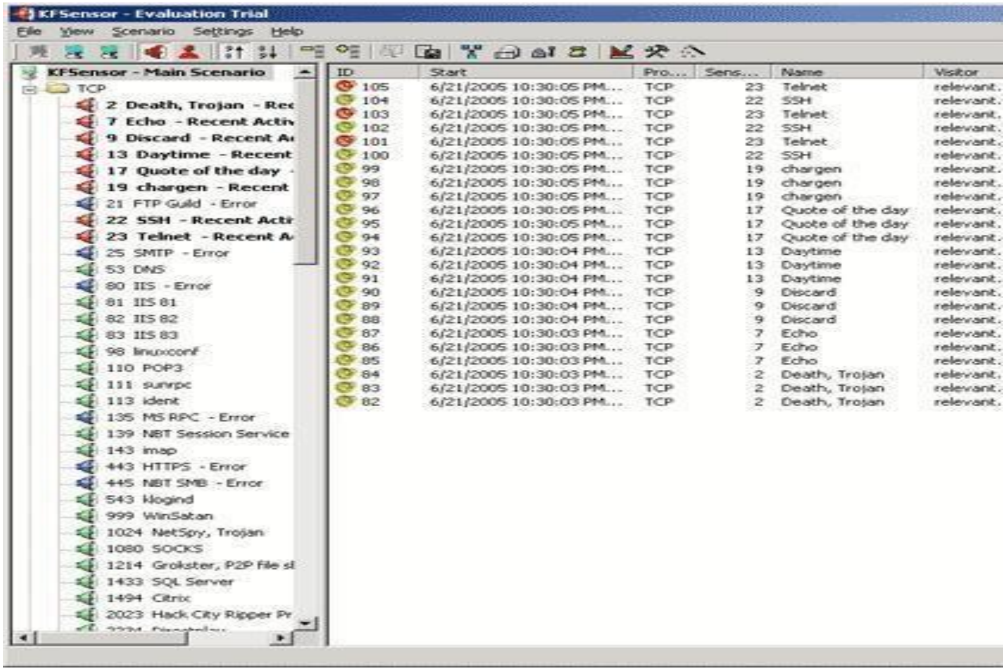
ID	Start	Duration	Pro...	Sens...	Name	Visitor	Description
33	6/16/2016 10:07:53 AM...	33.033	TCP	21	FTP	JOHN-PC	Idle time out
32	6/16/2016 10:08:00 AM...	0.001	TCP	8443	Plesk	JOHN-PC	url;
31	6/16/2016 10:08:00 AM...	0.011	TCP	8080	IIS Proxy	JOHN-PC	url;GET /| host;100.67.152.1:8080| ...
30	6/16/2016 10:08:00 AM...	0.001	TCP	8000	IIS Proxy	JOHN-PC	url;GET /| host;100.67.152.1:8000| ...
29	6/16/2016 10:07:58 AM...	1.499	TCP	5000	MS Uni Plug a...	JOHN-PC	
28	6/16/2016 10:07:58 AM...	0.002	TCP	3128	IIS Proxy	JOHN-PC	url;GET /| host;100.67.152.1:3128| ...
27	6/16/2016 10:07:58 AM...	0.299	TCP	443	HTTPS	JOHN-PC	
26	6/16/2016 10:07:56 AM...	1.502	TCP	443	HTTPS	JOHN-PC	
25	6/16/2016 10:07:56 AM...	0.005	TCP	80	IIS	JOHN-PC	url;GET /| host;100.67.152.1| agent..
24	6/16/2016 10:07:33 AM...	0.000	TCP	8443	Plesk	JOHN-PC	
23	6/16/2016 10:07:33 AM...	0.000	TCP	8080	IIS Proxy	JOHN-PC	
22	6/16/2016 10:07:33 AM...	0.000	TCP	8000	IIS Proxy	JOHN-PC	
21	6/16/2016 10:07:32 AM...	0.000	TCP	5555	ServeMe Trojan	JOHN-PC	
20	6/16/2016 10:07:32 AM...	0.000	TCP	5357	Web Services f...	JOHN-PC	
19	6/16/2016 10:07:32 AM...	0.001	TCP	5000	MS Uni Plug a...	JOHN-PC	
18	6/16/2016 10:07:31 AM...	0.001	TCP	3128	IIS Proxy	JOHN-PC	
17	6/16/2016 10:07:28 AM...	3.317	TCP	49744		gmil.com	SendMail: Protocol Failure id:4 to:..
16	6/16/2016 10:07:28 AM...	3.152	TCP	49740		gmil.com	SendMail: Protocol Failure id:2 to:..
15	6/16/2016 10:07:28 AM...	2.959	TCP	49739		gmil.com	SendMail: Protocol Failure id:1 to:..
14	6/16/2016 10:07:28 AM...	2.848	TCP	49738		gmil.com	SendMail: Protocol Failure id:3 to:..

User Rights: Admin [7B] Server: Running Visitors: 3 Events: 33/33

- ➢ As you can see, the column on the left contains a list of port numbers and what the port is typically used for.
- ➢ If the icon to the left of a port listing is green, it means that KFSensor is actively monitoring that port for attacks.
- ➢ If the icon is blue, it means that there has been an error and KFSensor is not watching for exploits aimed at that particular port.

Testing the software

- ➢ Once you've got the software up and running, one of the best things that you can do is to test the software by launching a port scan against the machine that's running KFSensor.
  - ➢ For the port scan, we using the HostScan.
- ➢ It simply scans a block of IP addresses, looking for open ports. Figure B shows how the KFSensor reacts to a partial port scan.
- ➢ If you look at Figure B, you will notice that the icons next to ports that were scanned turn red to indicate recent activity.

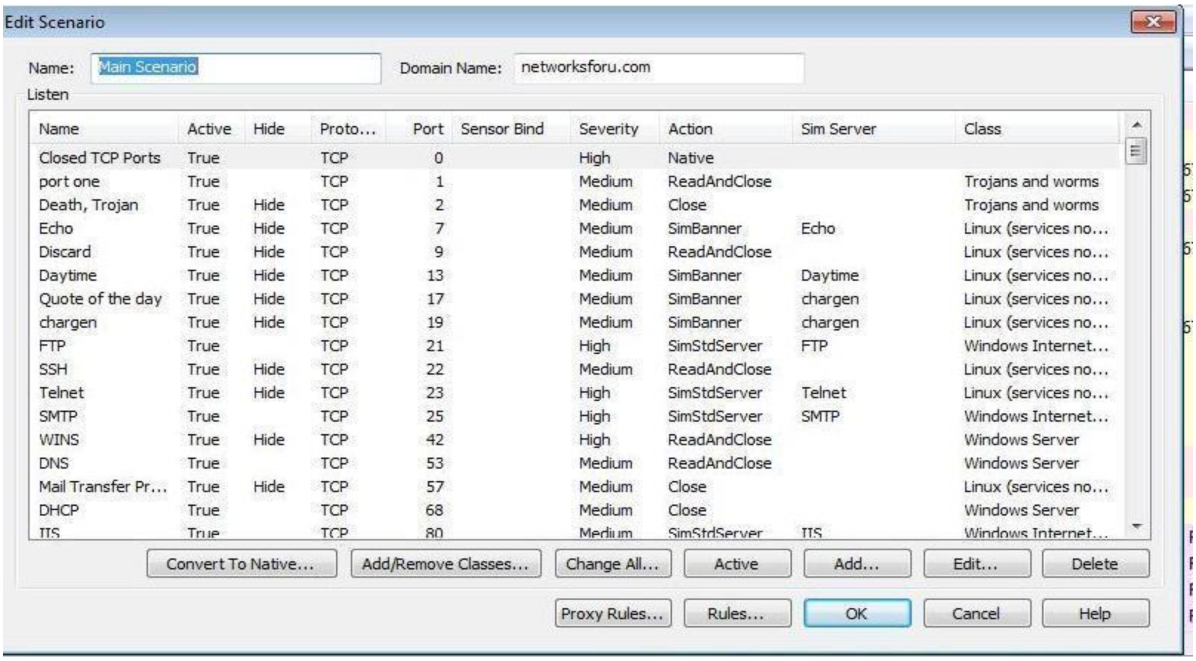


KFSensor - Main Scenario
- TCP
  - 2 Death, Trojan - Rec
  - 7 Echo - Recent Activ
  - 9 Discard - Recent Ac
  - 13 Daytime - Recent
  - 17 Quote of the day -
  - 19 chargen - Recent
  - 21 FTP Guild - Error
  - 22 SSH - Recent Acti
  - 23 Telnet - Recent A
  - 25 SMTP - Error
  - 53 DNS
  - 80 IIS - Error
  - 81 IIS 81
  - 82 IIS 82
  - 83 IIS 83
  - 98 linuxconf
  - 110 POP3
  - 111 sunrpc
  - 113 ident
  - 135 MS RPC - Error
  - 139 NBT Session Service
  - 143 imap
  - 443 HTTPS - Error
  - 445 NBT SMB - Error
  - 543 klogind
  - 999 WinSatan
  - 1024 NetSpy, Trojan
  - 1080 SOCKS
  - 1214 Grokster, P2P file sl
  - 1433 SQL Server
  - 1494 Citrix
  - 2023 Hack City Ripper Pr

ID	Start	Pro...	Sens...	Name	Visitor
105	6/21/2005 10:30:05 PM...	TCP	23	Telnet	relevant.
104	6/21/2005 10:30:05 PM...	TCP	22	SSH	relevant.
103	6/21/2005 10:30:05 PM...	TCP	23	Telnet	relevant.
102	6/21/2005 10:30:05 PM...	TCP	22	SSH	relevant.
101	6/21/2005 10:30:05 PM...	TCP	23	Telnet	relevant.
100	6/21/2005 10:30:05 PM...	TCP	22	SSH	relevant.
99	6/21/2005 10:30:05 PM...	TCP	19	chargen	relevant.
98	6/21/2005 10:30:05 PM...	TCP	19	chargen	relevant.
97	6/21/2005 10:30:05 PM...	TCP	19	chargen	relevant.
96	6/21/2005 10:30:05 PM...	TCP	17	Quote of the day	relevant.
95	6/21/2005 10:30:05 PM...	TCP	17	Quote of the day	relevant.
94	6/21/2005 10:30:05 PM...	TCP	17	Quote of the day	relevant.
93	6/21/2005 10:30:04 PM...	TCP	13	Daytime	relevant.
92	6/21/2005 10:30:04 PM...	TCP	13	Daytime	relevant.
91	6/21/2005 10:30:04 PM...	TCP	13	Daytime	relevant.
90	6/21/2005 10:30:04 PM...	TCP	9	Discard	relevant.
89	6/21/2005 10:30:04 PM...	TCP	9	Discard	relevant.
88	6/21/2005 10:30:04 PM...	TCP	9	Discard	relevant.
87	6/21/2005 10:30:03 PM...	TCP	7	Echo	relevant.
86	6/21/2005 10:30:03 PM...	TCP	7	Echo	relevant.
85	6/21/2005 10:30:03 PM...	TCP	7	Echo	relevant.
84	6/21/2005 10:30:03 PM...	TCP	2	Death, Trojan	relevant.
83	6/21/2005 10:30:03 PM...	TCP	2	Death, Trojan	relevant.
82	6/21/2005 10:30:03 PM...	TCP	2	Death, Trojan	relevant.

Modifying the Honeypot's behavior

- To create or modify rules, select the Edit Active Scenario command from the scenario menu.
- When you do, you will see a dialog box which contains a summary of all of the existing rules.
- You can 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.
- Both procedures work similarly.



Name: Main Scenario

Domain Name: networksforu.com

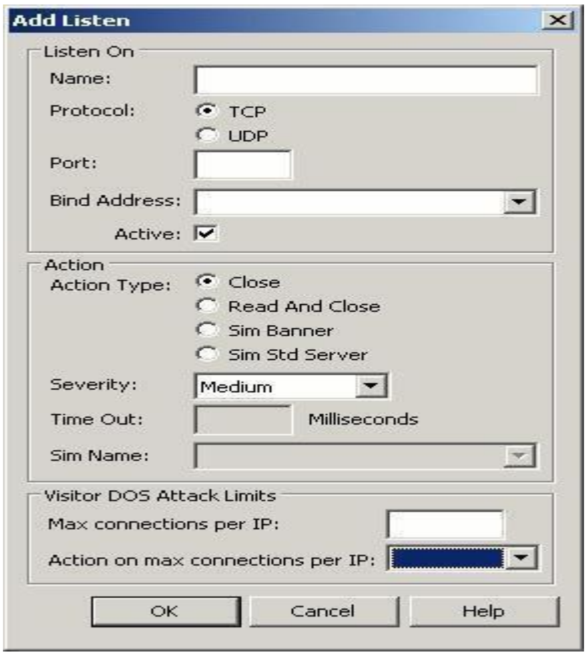
Listen

Name	Active	Hide	Proto...	Port	Sensor Bind	Severity	Action	Sim Server	Class
Closed TCP Ports	True		TCP	0		High	Native		
port one	True		TCP	1		Medium	ReadAndClose		Trojans and worms
Death, Trojan	True	Hide	TCP	2		Medium	Close		Trojans and worms
Echo	True	Hide	TCP	7		Medium	SimBanner	Echo	Linux (services no...
Discard	True	Hide	TCP	9		Medium	ReadAndClose		Linux (services no...
Daytime	True	Hide	TCP	13		Medium	SimBanner	Daytime	Linux (services no...
Quote of the day	True	Hide	TCP	17		Medium	SimBanner	chargen	Linux (services no...
chargen	True	Hide	TCP	19		Medium	SimBanner	chargen	Linux (services no...
FTP	True		TCP	21		High	SimStdServer	FTP	Windows Internet...
SSH	True	Hide	TCP	22		Medium	ReadAndClose		Linux (services no...
Telnet	True	Hide	TCP	23		High	SimStdServer	Telnet	Linux (services no...
SMTP	True		TCP	25		High	SimStdServer	SMTP	Windows Internet...
WINS	True	Hide	TCP	42		High	ReadAndClose		Windows Server
DNS	True		TCP	53		Medium	ReadAndClose		Windows Server
Mail Transfer Pr...	True	Hide	TCP	57		Medium	Close		Linux (services no...
DHCP	True		TCP	68		Medium	Close		Windows Server
IIS	True		TCP	80		Medium	SimStdServer	IIS	Windows Internet...

Click the Add button and you will see the Add Listen dialog box, shown in Figure D.

- 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.

Click on Add Button



Listen On

Name:

Protocol: ● TCP ○ UDP

Port:

Bind Address:

Active: ☑

Action

Action Type: ● Close ○ Read And Close ○ Sim Banner ○ Sim Std Server

Severity: Medium

Time Out: Milliseconds

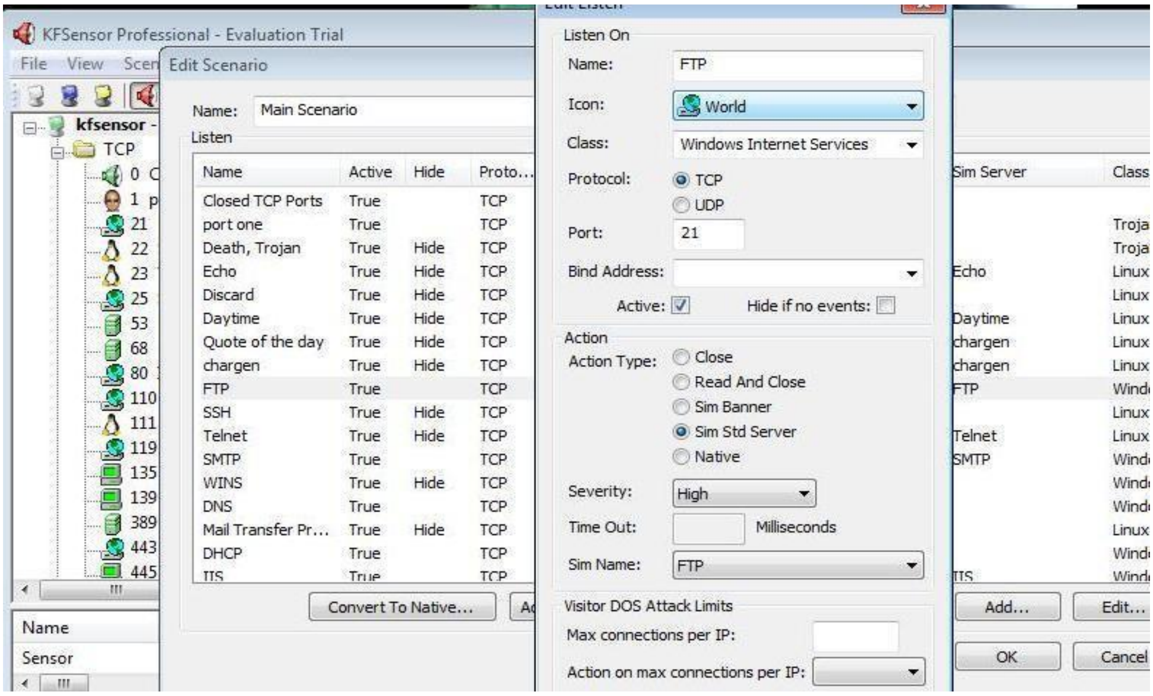
Sim Name:

Visitor DOS Attack Limits

Max connections per IP:

Action on max connections per IP:

Click on Edit Button



Name: Main Scenario

Listen

Name	Active	Hide	Proto...	Sim Server	Class
Closed TCP Ports	True		TCP		
port one	True		TCP		Troja
Death, Trojan	True	Hide	TCP		Troja
Echo	True	Hide	TCP	Echo	Linux
Discard	True	Hide	TCP		Linux
Daytime	True	Hide	TCP	Daytime	Linux
Quote of the day	True	Hide	TCP	chargen	Linux
chargen	True	Hide	TCP	chargen	Linux
FTP	True		TCP	FTP	Wind
SSH	True	Hide	TCP		Linux
Telnet	True	Hide	TCP	Telnet	Linux
SMTP	True		TCP	SMTP	Wind
WINS	True	Hide	TCP		Wind
DNS	True		TCP		Wind
Mail Transfer Pr...	True	Hide	TCP		Linux
DHCP	True		TCP		Wind
IIS	True		TCP	IIS	Wind

Edit Listen

Listen On

Name: FTP

Icon: World

Class: Windows Internet Services

Protocol: ● TCP ○ UDP

Port: 21

Bind Address:

Active: ☑ Hide if no events: □

Action

Action Type: ○ Close ○ Read And Close ○ Sim Banner ● Sim Std Server ○ Native

Severity: High

Time Out: Milliseconds

Sim Name: FTP

Visitor DOS Attack Limits

Max connections per IP:

Action on max connections per IP:

- The next few fields are protocol, port, and Bind Address. These fields allow you to choose what the rule is listening for. For example, you could configure the rule to listen to TCP port 1023 on IP address 192.168.1.100. The bind address portion of the rule is optional though. If you leave the bind address blank, the rule will listen across all of the machine's NICs.

- Now that you have defined the listener, it's time to configure the action that the rule takes when traffic is detected on the specified port. Your options are close, read and close, Sim Banner, and SimStd Server.

- The close option tells the rule to just terminate the connection. Read and close logs the information and then terminates the connection. The SimStd Server and Sim Banner options pertain to server emulation. The Sim Banner option allows you to perform a very simple server emulation, such as what you might use to emulate an FTP server.

- The Sim STD Server option allows you to emulate a more complex server, such as an IIS server.
- If you choose to use one of the sim options, you will have to fill in the simulator's name just below the Time Out field.
- The other part of the Action section that's worth mentioning is the severity section. KFSensor treated some events as severe and other events as a more moderate threat. The dialog box's Severity drop down list allows you to determine what level of severity should be associated with the event that you are logging.
- The final portion of the Add Listen dialog box is the Visitor DOS Attack Limits section. This section allows you to prevent denial of service attacks against KFSensor. You can determine the maximum number of connections to the machine per IP address (remember that this applies on a per rule basis).
- If your threshold is exceeded, you can choose to either ignore the excessive connections or you can lock out the offending IP address.
- Now that you have configured the new rule, select the Active Button to Enable/Disable. The new rule should now be in effect.

VIVA QUESTIONS (PRE LAB and POSTLAB):

1. How to build and use a Honeypot
2. List the types of interactions
3. Define honey tokens
4. Give the advantages of honey pot
5. Mention the protocols used by IPSec to provide security.
6. How arethe passwords stored in password file in UNIX operating system?
7. Which system is used to protect credit card transactions on the internet?
8. What is meant by Block Cipher?
9. Define Plain Text.

RESULT:

Thus the program to explore automated attack and penetration tools has been completed successfully

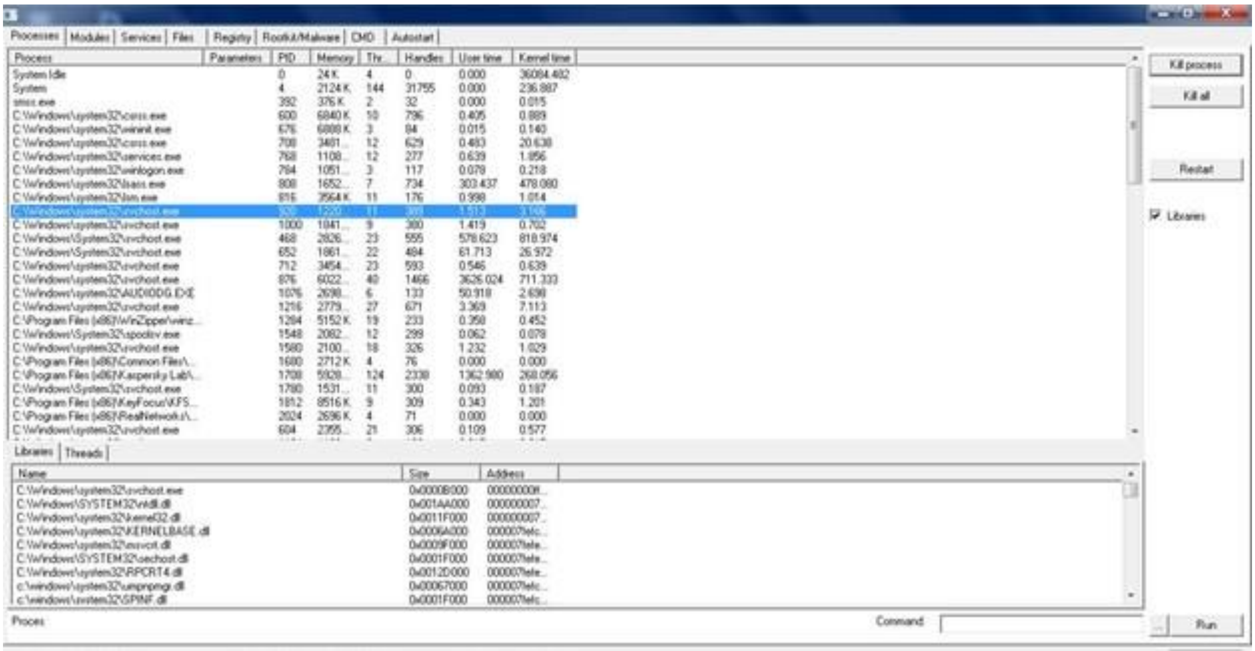
EX.No.: 9	**DEFEATING MALWARE – ROOTKIT HUNTER**

AIM

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.

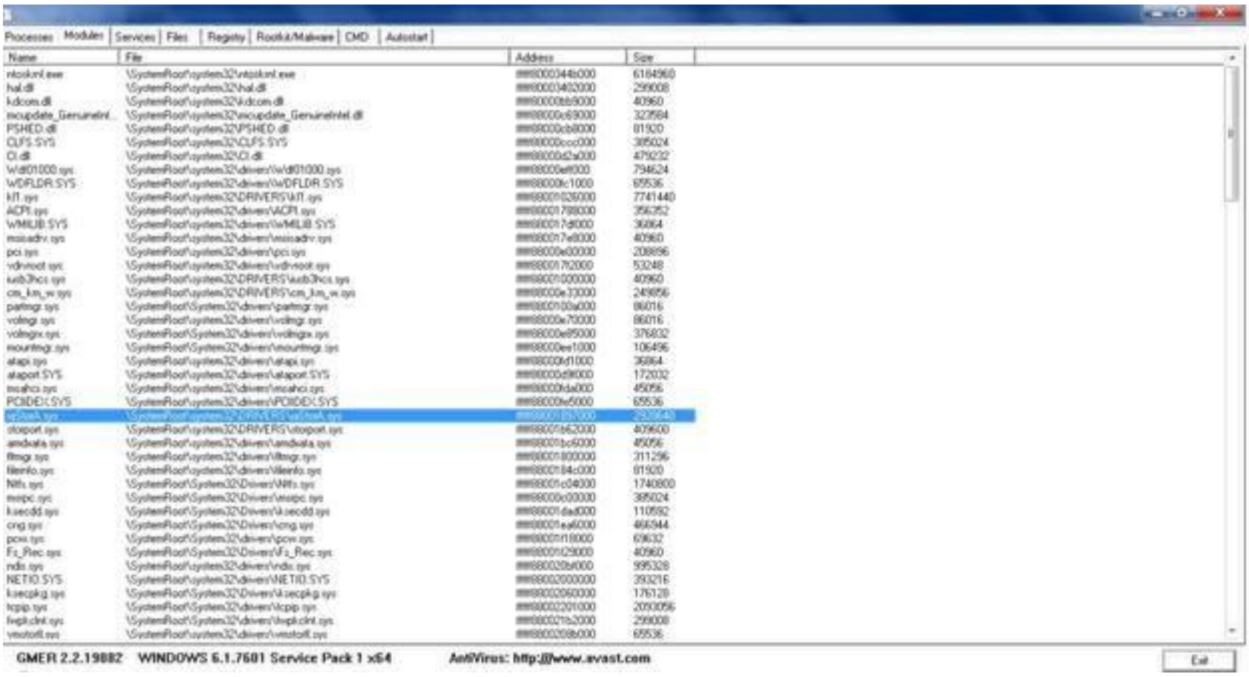
PRELAB DISCUSSION and PROCEDURE :

- 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 the various 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/Malawares scans the local drives selected.
- **Autostart** displays the registry base Autostart applications.
- CMD allows the user to interact with command line utilities or Registry.

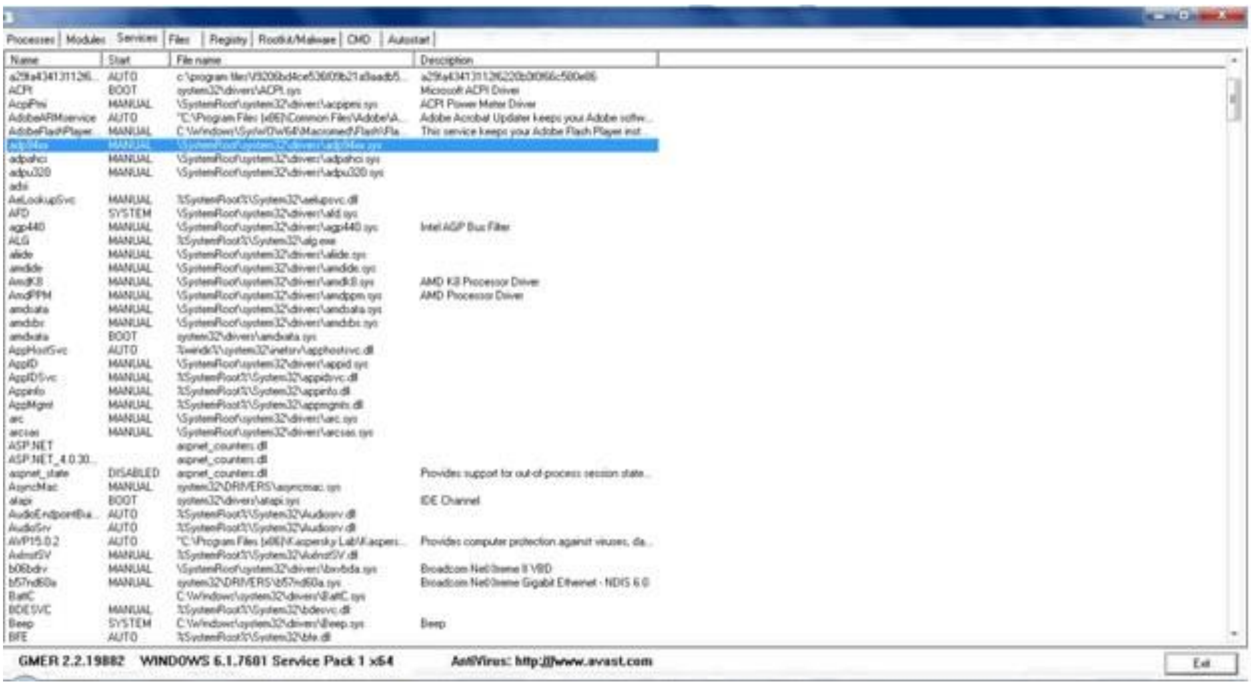


Process	Parameters	PID	Memory	Thr...	Handles	User time	Kernel time
System Idle		0	24 K	4	0	0.000	36084.402
System		4	2124 K	144	31755	0.000	236.887
smss.exe		392	376 K	2	32	0.000	0.015
C:\Windows\system32\csrss.exe		600	6840 K	10	796	0.405	0.889
C:\Windows\system32\wininit.exe		676	6888 K	3	84	0.015	0.140
C:\Windows\system32\csrss.exe		708	3481...	12	629	0.483	20.638
C:\Windows\system32\services.exe		768	1108...	12	277	0.639	1.856
C:\Windows\system32\winlogon.exe		784	1051...	3	117	0.078	0.218
C:\Windows\system32\lsass.exe		808	1652...	7	734	303.437	478.080
C:\Windows\system32\lsm.exe		816	3564 K	11	176	0.998	1.014
C:\Windows\system32\svchost.exe		920	1220...	11	389	1.513	3.166
C:\Windows\system32\svchost.exe		1000	1041...	9	380	1.419	0.702
C:\Windows\System32\svchost.exe		468	2826...	23	555	578.623	818.974
C:\Windows\System32\svchost.exe		652	1861...	22	484	61.713	26.972
C:\Windows\system32\svchost.exe		712	3454...	23	593	0.546	0.639
C:\Windows\system32\svchost.exe		876	6022...	40	1466	3626.024	711.333
C:\Windows\system32\AUDIODG.EXE		1076	2698...	6	133	50.918	2.698
C:\Windows\system32\svchost.exe		1216	2779...	27	671	3.369	7.113
C:\Program Files (x86)\WinZipper\winz...		1284	5152 K	19	233	0.358	0.452
C:\Windows\System32\spoolsv.exe		1548	2082...	12	299	0.062	0.078
C:\Windows\system32\svchost.exe		1580	2100...	18	326	1.232	1.029
C:\Program Files (x86)\Common Files\...		1680	2712 K	4	76	0.000	0.000
C:\Program Files (x86)\Kaspersky Lab\...		1708	5928...	124	2338	1362.980	268.056
C:\Windows\System32\svchost.exe		1780	1531...	11	300	0.093	0.187
C:\Program Files (x86)\KeyFocus\KFS...		1812	8516 K	9	309	0.343	1.201
C:\Program Files (x86)\RealNetworks\...		2024	2696 K	4	71	0.000	0.000
C:\Windows\system32\svchost.exe		604	2355...	21	306	0.109	0.577

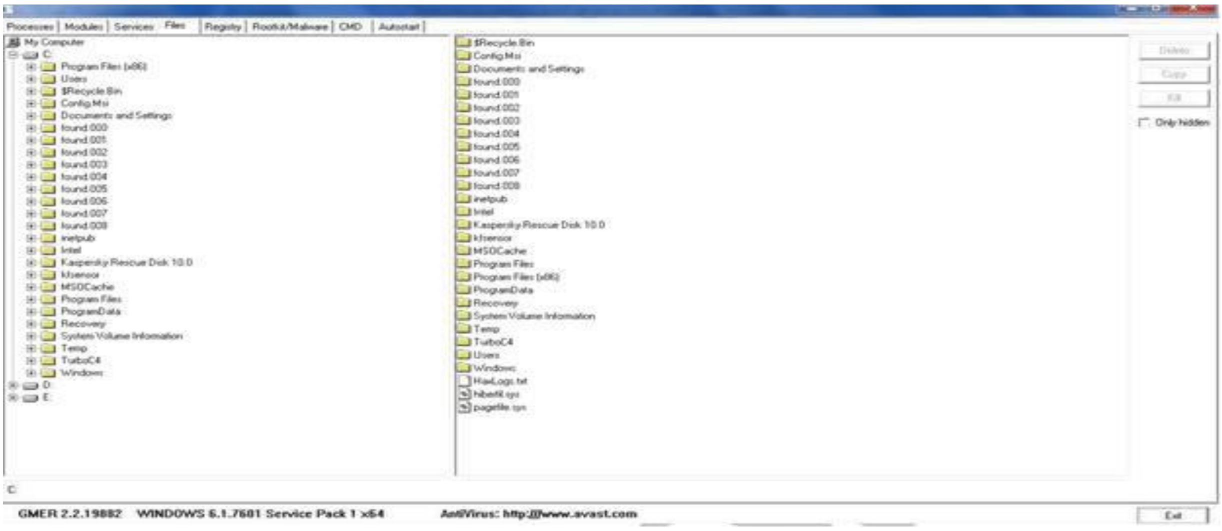
Name	Size	Address
C:\Windows\system32\svchost.exe	0x0000B000	00000000ff...
C:\Windows\SYSTEM32\ntdll.dll	0x001AA000	000000007...
C:\Windows\system32\kernel32.dll	0x0011F000	000000007...
C:\Windows\system32\KERNELBASE.dll	0x0006A000	000007fefc...
C:\Windows\system32\msvcrt.dll	0x0009F000	000007fefe...
C:\Windows\SYSTEM32\sechost.dll	0x0001F000	000007fefe...
C:\Windows\system32\RPCRT4.dll	0x0012D000	000007fefe...
c:\windows\system32\umpnpmgr.dll	0x00067000	000007fefc...
c:\windows\system32\SPINF.dll	0x0001F000	000007fefc...



```
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         fffff88000eff000   794624
WDFLDR.SYS           \SystemRoot\system32\drivers\WDFLDR.SYS           fffff88000fc1000   65536
klif.sys             \SystemRoot\system32\DRIVERS\klif.sys             fffff88001026000   7741440
ACPI.sys             \SystemRoot\system32\drivers\ACPI.sys             fffff88001798000   356352
WMILIB.SYS           \SystemRoot\system32\drivers\WMILIB.SYS           fffff880017ef000   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         fffff88001712000   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          fffff88001187000   2520640
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
```



```
Name                  Start     File name                                         Description
a29a4341311126...     AUTO      c:\program files\9206bd4ce536f09b21a9aadb5...     a29a4341311262202b0f066c500e86
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
AudioEndpointBu...    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
```

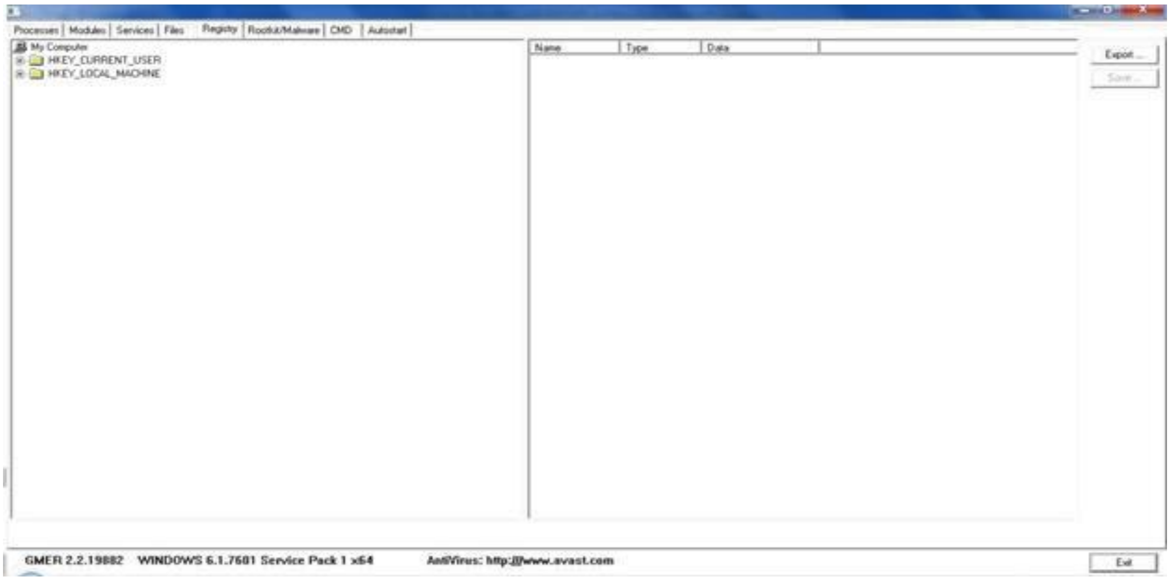


```
My Computer
C:
  Program Files (x86)
  Users
  $Recycle.Bin
  Config.Msi
  Documents and Settings
  found.000
  found.001
  found.002
  found.003
  found.004
  found.005
  found.006
  found.007
  found.008
  inetpub
  Intel
  Kaspersky Rescue Disk 10.0
  kfsensor
  MSOCache
  Program Files
  ProgramData
  Recovery
  System Volume Information
  Temp
  TurboC4
  Windows
D:
E:
```

```
$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
Program Files (x86)
ProgramData
Recovery
System Volume Information
Temp
TurboC4
Users
Windows
HawLogs.txt
hiberfil.sys
pagefile.sys
```

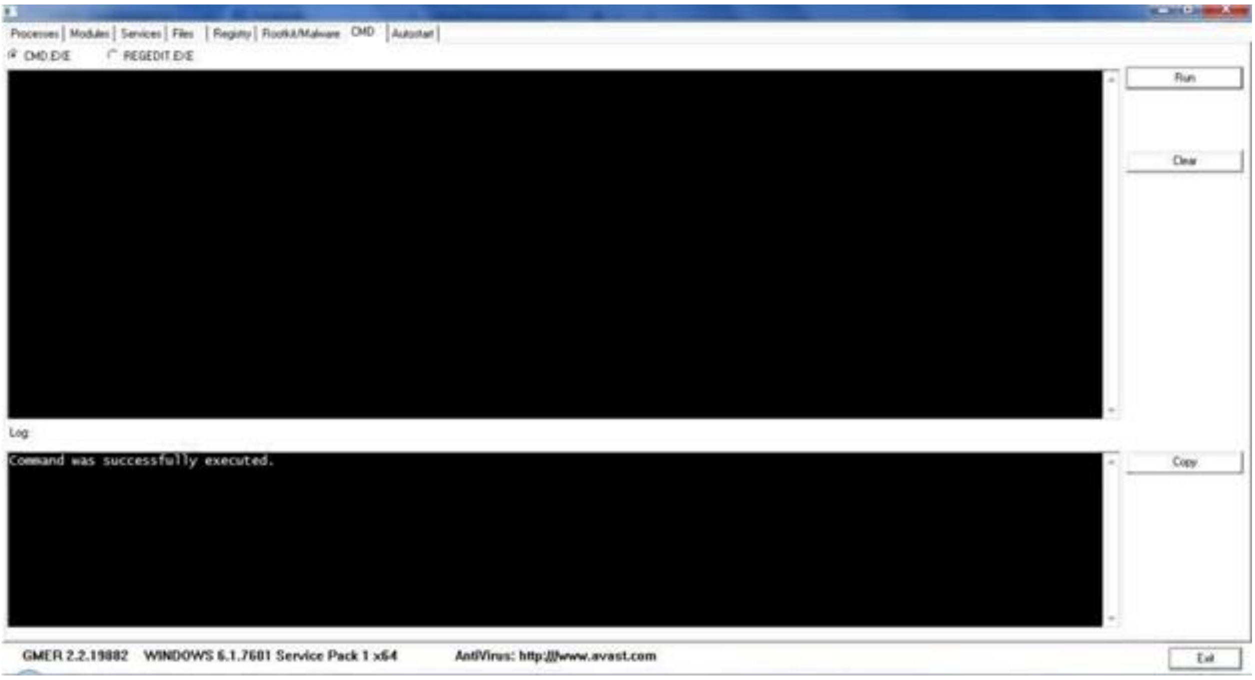
C:

GMER 2.2.19882 WINDOWS 6.1.7601 Service Pack 1 x64 AntiVirus: http://www.avast.com



```
My Computer
  HKEY_CURRENT_USER
  HKEY_LOCAL_MACHINE
```

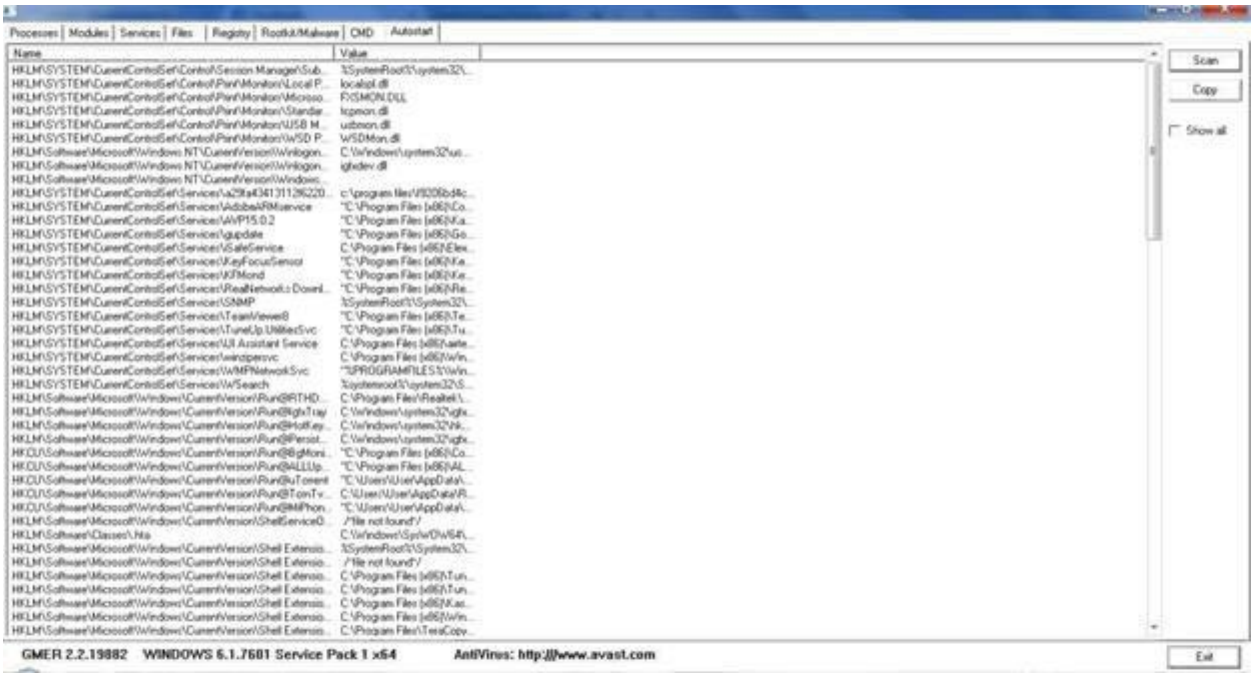
GMER 2.2.19882 WINDOWS 6.1.7601 Service Pack 1 x64 AntiVirus: http://www.avast.com



● CMD.EXE ○ REGEDIT.EXE

```
Command was successfully executed.
```

GMER 2.2.19882 WINDOWS 6.1.7601 Service Pack 1 x64 AntiVirus: http://www.avast.com



Name	Value
HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\Sub...	%SystemRoot%\system32\...
HKLM\SYSTEM\CurrentControlSet\Control\Print\Monitors\Local P...	localspl.dll
HKLM\SYSTEM\CurrentControlSet\Control\Print\Monitors\Microso...	FXSMON.DLL
HKLM\SYSTEM\CurrentControlSet\Control\Print\Monitors\Standar...	tcpmon.dll
HKLM\SYSTEM\CurrentControlSet\Control\Print\Monitors\USB M...	usbmon.dll
HKLM\SYSTEM\CurrentControlSet\Control\Print\Monitors\WSD P...	WSDMon.dll
HKLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon...	C:\Windows\system32\us...
HKLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon...	igfxdev.dll
HKLM\Software\Microsoft\Windows NT\CurrentVersion\Windows...	
HKLM\SYSTEM\CurrentControlSet\Services\a29fa4341311286220...	c:\program files\f9206bd4c...
HKLM\SYSTEM\CurrentControlSet\Services\AdobeARMservice	"C:\Program Files (x86)\Co...
HKLM\SYSTEM\CurrentControlSet\Services\AVP15.0.2	"C:\Program Files (x86)\Ka...
HKLM\SYSTEM\CurrentControlSet\Services\gupdate	"C:\Program Files (x86)\Go...
HKLM\SYSTEM\CurrentControlSet\Services\iSafeService	C:\Program Files (x86)\Elex...
HKLM\SYSTEM\CurrentControlSet\Services\KeyFocusSensor	"C:\Program Files (x86)\Ke...
HKLM\SYSTEM\CurrentControlSet\Services\KFMond	"C:\Program Files (x86)\Ke...
HKLM\SYSTEM\CurrentControlSet\Services\RealNetworks Downl...	"C:\Program Files (x86)\Re...
HKLM\SYSTEM\CurrentControlSet\Services\SNMP	%SystemRoot%\System32\...
HKLM\SYSTEM\CurrentControlSet\Services\TeamViewer8	"C:\Program Files (x86)\Te...
HKLM\SYSTEM\CurrentControlSet\Services\TuneUp.UtilitiesSvc	"C:\Program Files (x86)\Tu...
HKLM\SYSTEM\CurrentControlSet\Services\UI Assistant Service	C:\Program Files (x86)\aite...
HKLM\SYSTEM\CurrentControlSet\Services\windipersvc	C:\Program Files (x86)\Win...
HKLM\SYSTEM\CurrentControlSet\Services\WMPNetworkSvc	"%PROGRAMFILES%\Win...
HKLM\SYSTEM\CurrentControlSet\Services\WSearch	%systemroot%\system32\S...
HKLM\Software\Microsoft\Windows\CurrentVersion\Run@RTHD...	C:\Program Files\Realtek\...
HKLM\Software\Microsoft\Windows\CurrentVersion\Run@IgfxTray	C:\Windows\system32\igfx...
HKLM\Software\Microsoft\Windows\CurrentVersion\Run@HotKey...	C:\Windows\system32\hk...
HKLM\Software\Microsoft\Windows\CurrentVersion\Run@Persist...	C:\Windows\system32\igfx...
HKCU\Software\Microsoft\Windows\CurrentVersion\Run@BgMoni...	"C:\Program Files (x86)\Co...
HKCU\Software\Microsoft\Windows\CurrentVersion\Run@ALLUp...	"C:\Program Files (x86)\AL...
HKCU\Software\Microsoft\Windows\CurrentVersion\Run@uTorrent	"C:\Users\User\AppData\...
HKCU\Software\Microsoft\Windows\CurrentVersion\Run@TomTv...	C:\Users\User\AppData\R...
HKCU\Software\Microsoft\Windows\CurrentVersion\Run@MiPhon...	"C:\Users\User\AppData\...
HKLM\Software\Microsoft\Windows\CurrentVersion\ShellServiceO...	/*file not found*/
HKLM\Software\Classes\.hta	C:\Windows\SysWOW64\...
HKLM\Software\Microsoft\Windows\CurrentVersion\Shell Extensio...	%SystemRoot%\System32\...
HKLM\Software\Microsoft\Windows\CurrentVersion\Shell Extensio...	/*file not found*/
HKLM\Software\Microsoft\Windows\CurrentVersion\Shell Extensio...	C:\Program Files (x86)\Tun...
HKLM\Software\Microsoft\Windows\CurrentVersion\Shell Extensio...	C:\Program Files (x86)\Tun...
HKLM\Software\Microsoft\Windows\CurrentVersion\Shell Extensio...	C:\Program Files (x86)\Kas...
HKLM\Software\Microsoft\Windows\CurrentVersion\Shell Extensio...	C:\Program Files (x86)\Win...
HKLM\Software\Microsoft\Windows\CurrentVersion\Shell Extensio...	C:\Program Files\TeraCopy...

GMER 2.2.19882 WINDOWS 6.1.7601 Service Pack 1 x64 AntiVirus: http://www.avast.com

VIVA QUESTIONS (PRE LAB and POSTLAB):

1. List any two web security threats.
2. List any two design goals for a firewall.
3. Define Security Mechanism.
4. What is S/MIME?
5. Define IPSec.
6. What are the key features of SET?
7. What is the use of public key encryption scheme?
8. Identify the possible threats for RSA algorithm.
9. List out the general schemes for the distribution of public keys.
10. What are the areas where Kerberos Version 5 addresses the limitation of Version 4?

RESULT:

Thus the Rootkits tool was installed and its various options were verified successfully

Best wishes,
Dr. Mustafa Basthikodi