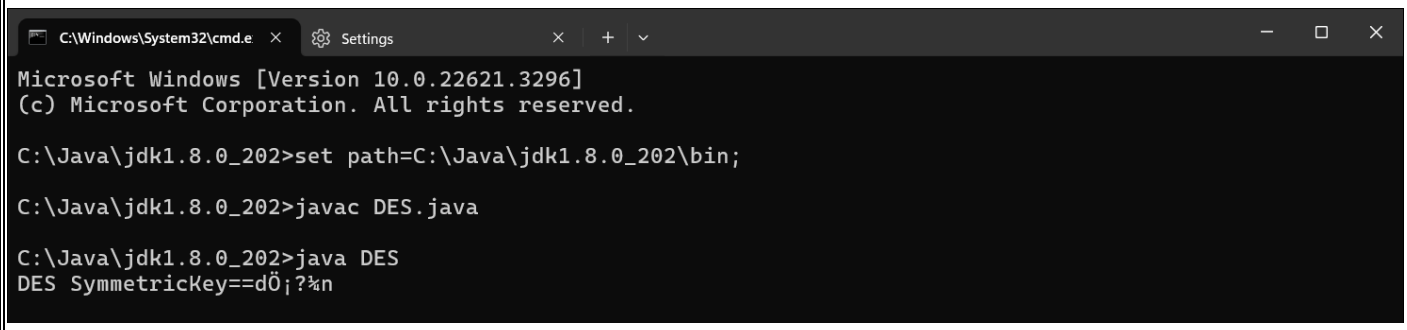
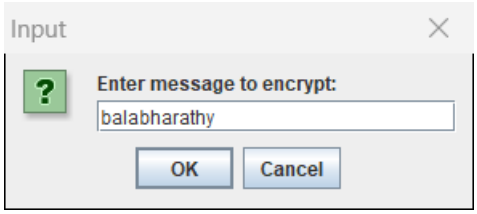
PROGRAM:

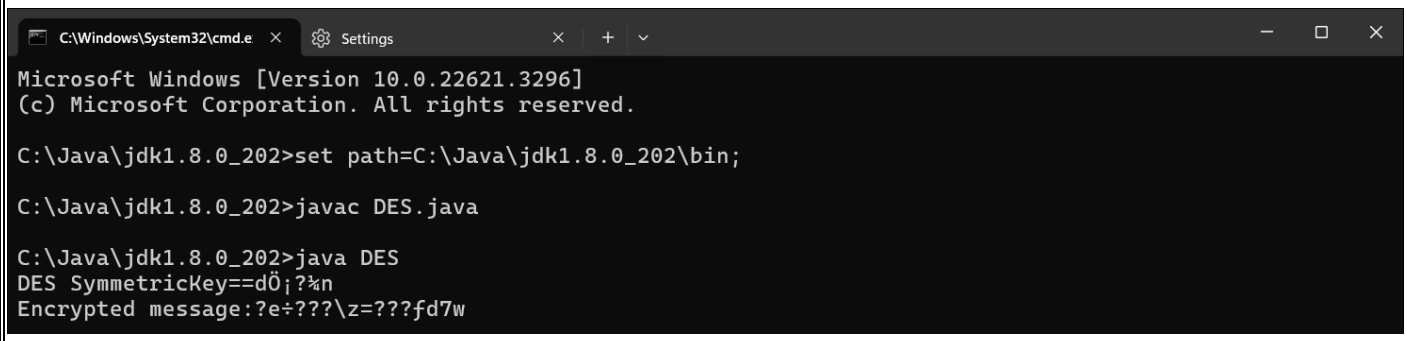
```
import javax.swing.*;
import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.SecretKey;
import javax.crypto.spec.SecretKeySpec;
import java.security.SecureRandom;
import java.util.Random;

class DES {
   byte[] skey = new byte[1000];
   String skeystring;
   static byte[] raw;
   String inputmessage, encryptedata, decryptedmessage;

   public DES() {
      try {
         generatesymmetrickey();
         inputmessage = JOptionPane.showInputDialog(null, "Enter message to encrypt:");
         byte[] ibyte = inputmessage.getBytes();
         byte[] ebyte = encrypt(raw, ibyte);
         String encrypteddata = new String(ebyte);
         System.out.println("Encrypted message:" + encrypteddata);
         JOptionPane.showMessageDialog(null, "Encrypted Data" + "\n" + encrypteddata);
         byte[] dbyte = decrypt(raw, ebyte);
         String decryptedmessage = new String(dbyte);
         System.out.println("Decrypted message:" + decryptedmessage);
         JOptionPane.showMessageDialog(null, "Decrypted Data " + "\n" +
decryptedmessage);
      } catch (Exception e) {
         System.out.println(e);
      }
   }

   void generatesymmetrickey() {
      try {
         Random r = new Random();
         int num = r.nextInt(10000);
         String knum = String.valueOf(num);
         byte[] knumb = knum.getBytes();
         skey = getRawKey(knumb);
         skeystring = new String(skey);
         System.out.println("DES SymmetricKey=" + skeystring);
      } catch (Exception e) {
```

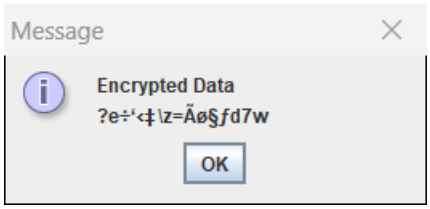
```
            System.out.println(e);
        }
    }

    private static byte[] getRawKey(byte[] seed) throws Exception {
        KeyGenerator kgen = KeyGenerator.getInstance("DES");
        SecureRandom sr = SecureRandom.getInstance("SHA1PRNG");
        sr.setSeed(seed);
        kgen.init(56, sr);
        SecretKey skey = kgen.generateKey();
        raw = skey.getEncoded();
        return raw;
    }

    private static byte[] encrypt(byte[] raw, byte[] clear) throws Exception {
        SecretKey seckey = new SecretKeySpec(raw, "DES");
        Cipher cipher = Cipher.getInstance("DES");
        cipher.init(Cipher.ENCRYPT_MODE, seckey);
        byte[] encrypted = cipher.doFinal(clear);
        return encrypted;
    }

    private static byte[] decrypt(byte[] raw, byte[] encrypted) throws Exception {
        SecretKey seckey = new SecretKeySpec(raw, "DES");
        Cipher cipher = Cipher.getInstance("DES");
        cipher.init(Cipher.DECRYPT_MODE, seckey);
        byte[] decrypted = cipher.doFinal(encrypted);
        return decrypted;
    }

    public static void main(String args[]) {
        DES des = new DES();
    }
}
```

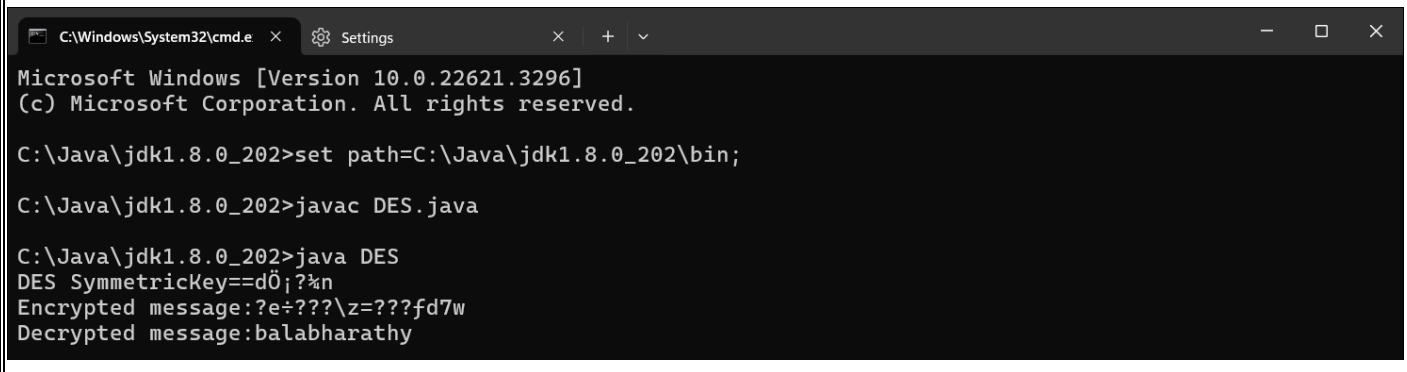
OUTPUT:

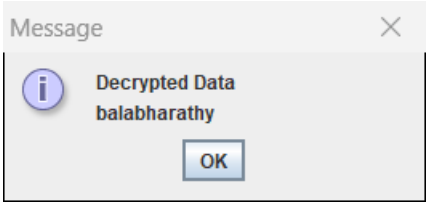

C:\Windows\System32\cmd.e
Settings
Microsoft Windows [Version 10.0.22621.3296]
(c) Microsoft Corporation. All rights reserved.
C:\Java\jdk1.8.0_202>set path=C:\Java\jdk1.8.0_202\bin;
C:\Java\jdk1.8.0_202>javac DES.java
C:\Java\jdk1.8.0_202>java DES
DES SymmetricKey==dÖ¡?%n


Input
Enter message to encrypt:
balabharathy
OK
Cancel


C:\Windows\System32\cmd.e
Settings
Microsoft Windows [Version 10.0.22621.3296]
(c) Microsoft Corporation. All rights reserved.
C:\Java\jdk1.8.0_202>set path=C:\Java\jdk1.8.0_202\bin;
C:\Java\jdk1.8.0_202>javac DES.java
C:\Java\jdk1.8.0_202>java DES
DES SymmetricKey==dÖ¡?%n
Encrypted message:?e÷???\z=???fd7w


Message
Encrypted Data
?e÷'‹‡\z=Ãø§fd7w
OK


C:\Windows\System32\cmd.e
Settings
Microsoft Windows [Version 10.0.22621.3296]
(c) Microsoft Corporation. All rights reserved.
C:\Java\jdk1.8.0_202>set path=C:\Java\jdk1.8.0_202\bin;
C:\Java\jdk1.8.0_202>javac DES.java
C:\Java\jdk1.8.0_202>java DES
DES SymmetricKey==dÖ¡?%n
Encrypted message:?e÷???\z=???fd7w
Decrypted message:balabharathy


Message
Decrypted Data
balabharathy
OK

PROGRAM:

```
import java.io.UnsupportedEncodingException;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.util.Arrays;
import java.util.Base64;
import javax.crypto.Cipher;
import javax.crypto.spec.SecretKeySpec;
public class AES
{
  private static SecretKeySpec secretKey;
  private static byte[] key;
  public static void setKey(String myKey) {
    MessageDigest sha = null;
    try {
      key = myKey.getBytes("UTF-8");
      sha = MessageDigest.getInstance("SHA-1");
      key = sha.digest(key);
      key= Arrays.copyOf(key, 16);
      secretKey= new SecretKeySpec(key, "AES");
    } catch (NoSuchAlgorithmException e) {
      e.printStackTrace();
    } catch (UnsupportedEncodingException e) {
      e.printStackTrace();
    }
  }
  public static String encrypt(String strToEncrypt, String secret) {
    try {
      setKey(secret);
      Cipher cipher = Cipher.getInstance("AES/ECB/PKCS5Padding");
      cipher.init(Cipher.ENCRYPT_MODE, secretKey);
      return Base64.getEncoder().encodeToString(cipher.doFinal(strToEncrypt.getBytes
("UTF-8")));
    } catch (Exception e) {
      System.out.println("Error while encrypting: " + e.toString());
    }
    return null;
  }
  public static String decrypt(String strToDecrypt, String secret) {
    try {
      setKey(secret);
      Cipher cipher = Cipher.getInstance("AES/ECB/PKCS5PADDING");
      cipher.init(Cipher.DECRYPT_MODE, secretKey);
      return new String(cipher.doFinal(Base64.getDecoder().decode(strToDecrypt)));
```

```
    } catch (Exception e) {
      System.out.println("Error while decrypting: " + e.toString());
    }
    return null;
  }
  public static void main(String[] args) {
    System.out.println("Enter the secret key: ");
    String secretKey= System.console().readLine();
    System.out.println("Enter the original URL: ");
    String originalString= System.console().readLine();
    String encryptedString = AES.encrypt(originalString, secretKey);
    String decryptedString = AES.decrypt(encryptedString, secretKey);
    System.out.println("URL Encryption Using AES Algorithm\n ---------- ");
    System.out.println("Original URL : " + originalString);
    System.out.println("Encrypted URL : " + encryptedString);
    System.out.println("Decrypted URL : " + decryptedString);
  }
}
```

OUTPUT:

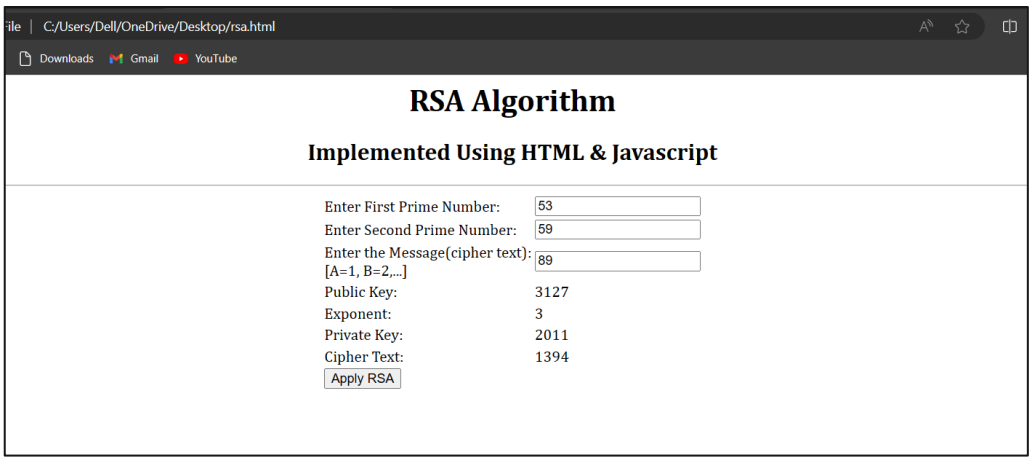
```
C:\Java\jdk1.8.0_202>javac AES.java

C:\Java\jdk1.8.0_202>java AES
Enter the secret key:
annaUniversity
Enter the original URL:
www.annauniv.edu
URL Encryption Using AES Algorithm
 -----------
Original URL : www.annauniv.edu
Encrypted URL : vibpFJW6Cvs5Y+L7t4N6YWWe07+JzS1d3CU2h3mEvEg=
Decrypted URL : www.annauniv.edu
```

PROGRAM:

```
<html>
 <head>
  <title>RSA Encryption</title>
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
 </head>
 <body>
  <center>
  <h1>RSA Algorithm</h1>
  <h2>Implemented Using HTML & Javascript</h2>
  <hr>
  <table>
  <tr>
  <td>Enter First Prime Number:</td>
  <td><input type="number" value="53" id="p"></td>
  </tr>
  <tr>
  <td>Enter Second Prime Number:</td>
  <td><input type="number" value="59" id="q"></p> </td>
  </tr>
  <tr>
  <td>Enter the Message(cipher text):<br>[A=1, B=2,...]</td>
  <td><input type="number" value="89" id="msg"></p> </td>
   </tr>
  <tr>
  <td>Public Key:</td>
  <td><p id="publickey"></p> </td>
  </tr>
  <tr>
  <td>Exponent:</td>
  <td><p id="exponent"></p> </td>
  </tr>
  <tr>
  <td>Private Key:</td>
  <td><p id="privatekey"></p></td>
  </tr>
  <tr>
  <td>Cipher Text:</td>
  <td><p id="ciphertext"></p> </td>
  </tr>
  <tr>
  <td><button onclick="RSA();">Apply RSA</button></td>
  </tr>
  </table> </center>
```

```
</body>
<script type="text/javascript">
  function RSA()
  {
    var gcd, p, q, no, n, t, e, i, x;
    gcd = function (a, b) { return (!b) ? a : gcd(b, a % b); };
    p = document.getElementById('p').value;
    q = document.getElementById('q').value;
    no = document.getElementById('msg').value;
    n = p * q;
    t = (p - 1) * (q - 1);
    for (e = 2; e < t; e++)
    {
      if (gcd(e, t) == 1)
      {
        break;
      }
    }
    for (i = 0; i < 10; i++)
    {
      x = 1 + i * t
      if (x % e == 0)
      {
        d = x / e;
        break;
      }
    }
    ctt = Math.pow(no, e).toFixed(0);
    ct = ctt % n;
    dtt = Math.pow(ct, d).toFixed(0);
    dt = dtt % n;
    document.getElementById('publickey').innerHTML = n;
    document.getElementById('exponent').innerHTML = e;
    document.getElementById('privatekey').innerHTML = d;
    document.getElementById('ciphertext').innerHTML = ct;
  }
</script>
</html>
```

OUTPUT:


ile | C:/Users/Dell/OneDrive/Desktop/rsa.html
Downloads
Gmail
YouTube
RSA Algorithm
Implemented Using HTML & Javascript
Enter First Prime Number:
53
Enter Second Prime Number:
59
Enter the Message(cipher text):
[A=1, B=2,...]
89
Public Key:
3127
Exponent:
3
Private Key:
2011
Cipher Text:
1394
Apply RSA

PROGRAM:

```
import java.io.*;
import java.math.BigInteger;
class dh
{
 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 lessthan "+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("Diffie-Hellman secret key was calculated.");
 }
}
```

OUTPUT:

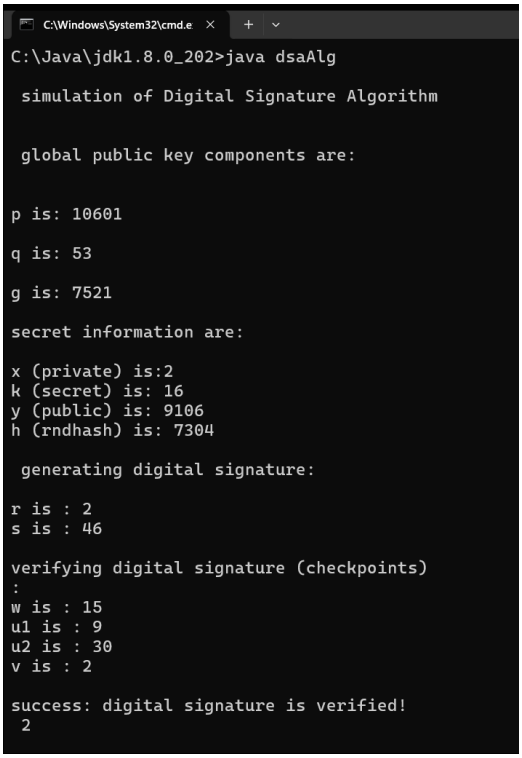
```
C:\Java\jdk1.8.0_202>set path=C:\Java\jdk1.8.0_202\jre\bin;

C:\Java\jdk1.8.0_202>java dh
Enter prime number:
11
Enter primitive root of 11:7
Enter value for x less than 11:
3
R1=2
Enter value for y lessthan 11:6
R2=4
Key calculated at Sender's side:9
Key calculated at Receiver's side:9
Diffie-Hellman secret key was calculated.
```

PROGRAM:

```
import java.util.*;
import java.math.BigInteger;
class dsaAlg {
 final static BigInteger one = new BigInteger("1");
 final static BigInteger zero = new BigInteger("0");
 public static BigInteger getNextPrime(String ans)
 {
  BigInteger test = new BigInteger(ans);
  while (!test.isProbablePrime(99))
  e:
  {
   test = test.add(one);
  }
  return test;
 }
 public static BigInteger findQ(BigInteger n)
 {
  BigInteger start = new BigInteger("2");
  while (!n.isProbablePrime(99))
  {
   while (!((n.mod(start)).equals(zero)))
   {
    start = start.add(one);
   }
   n = n.divide(start);
  }
  return n;
 }
 public static BigInteger getGen(BigInteger p, BigInteger q,Random r)
 {
  BigInteger h = new BigInteger(p.bitLength(), r);
  h = h.mod(p);
  return h.modPow((p.subtract(one)).divide(q), p);
 }
 public static void main (String[] args) throws java.lang.Exception
{
  Random randObj = new Random();
  BigInteger p = getNextPrime("10600"); /* approximate prime */
  BigInteger q = findQ(p.subtract(one));
  BigInteger g = getGen(p,q,randObj);
  System.out.println(" \n simulation of Digital Signature Algorithm \n");
  System.out.println(" \n global public key components are:\n");
  System.out.println("\np is: " + p);
```

```
  System.out.println("\nq is: " + q);
  System.out.println("\ng is: " + g);
  BigInteger x = new BigInteger(q.bitLength(), randObj);
  x = x.mod(q);
  BigInteger y= g.modPow(x,p);
  BigInteger k = new BigInteger(q.bitLength(), randObj);
  k = k.mod(q);
  BigInteger r = (g.modPow(k,p)).mod(q);
  BigInteger hashVal = new BigInteger(p.bitLength(),randObj);
  BigInteger kInv = k.modInverse(q);
  BigInteger s = kInv.multiply(hashVal.add(x.multiply(r)));
  s = s.mod(q);
  System.out.println("\nsecret information are:\n");
  System.out.println("x (private) is:" + x);
  System.out.println("k (secret) is: " + k);
  System.out.println("y (public) is: " + y);
  System.out.println("h (rndhash) is: " + hashVal);
  System.out.println("\n generating digital signature:\n");
  System.out.println("r is : " + r);
  System.out.println("s is : " + s);
  BigInteger w = s.modInverse(q);
  BigInteger u1 = (hashVal.multiply(w)).mod(q);
  BigInteger u2 = (r.multiply(w)).mod(q);
  BigInteger v = (g.modPow(u1,p)).multiply(y.modPow(u2,p));
  v = (v.mod(p)).mod(q);
  System.out.println("\nverifying digital signature (checkpoints)\n:");
  System.out.println("w is : " + w);
  System.out.println("u1 is : " + u1);
  System.out.println("u2 is : " + u2);
  System.out.println("v is : " + v);
  if (v.equals(r))
  {
    System.out.println("\nsuccess: digital signature is verified!\n " + r);
  }
  else
  {
    System.out.println("\n error: incorrect digitalsignature\n ");
  }
 }
}
```

OUTPUT:


C:\Windows\System32\cmd.e
C:\Java\jdk1.8.0_202>java dsaAlg
simulation of Digital Signature Algorithm
global public key components are:
p is: 10601
q is: 53
g is: 7521
secret information are:
x (private) is:2
k (secret) is: 16
y (public) is: 9106
h (rndhash) is: 7304
generating digital signature:
r is : 2
s is : 46
verifying digital signature (checkpoints)
:
w is : 15
u1 is : 9
u2 is : 30
v is : 2
success: digital signature is verified!
2

Ex.No: 4	Installation of Wireshark, TCPdump and observe the data transferred in client-server communication using UDP/TCP and Identify the UDP/TCP datagram.

Aim:

To install wireshark, TCPdump and observe the data transferred in client-server communicationusing UDP/TCP and Identify the UDP/TCP datagram.

Wireshark:

Wireshark is an open-source tool for profiling network traffic and analyzing packets. Such tool isoften referred as a network analyzer, network protocol analyzer or sniffer.

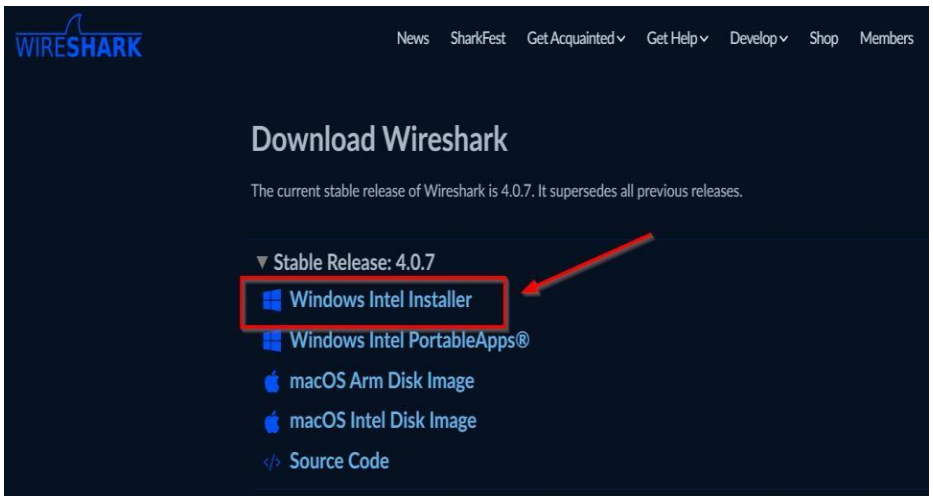
It is used to understand how communication takes place across a network and to analyze what wentwrong when an issue in communication arises.

It captures network traffic from ethernet, Bluetooth, wireless (IEEE.802.11), token ring, and framerelay connections, among others, and stores that data for offline analysis.

Wireshark allows you to filter the log before the capture starts or during analysis, For example, you can set a filter to see TCP traffic between two IP addresses, or you can set it only to show you the packets sent from one computer. The filters in Wireshark are one of the primary reasons it has become the standardtool for packet analysis.

Installation of Wireshark:

Step 1: Your first step is to head to the Wireshark download page **https://www.wireshark.org/download.html** and locate the Windows installer.



Step 2: You will be presented with the Wireshark wizard to guide you through the installation. Click "Next."

Step 3: Next, you can review, agree to the license agreement, and click "Noted" to continue.

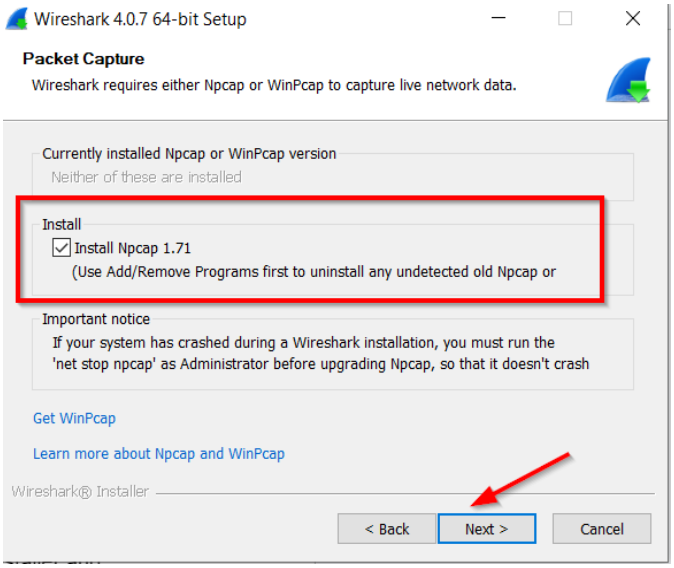
Step 4: You will be asked what components you want to install. You can make your

choice and thenclick “Next.”

Step 5: Choose a directory to install Wireshark in, showing you the space required to install it.

Step 6: Install **Ncap.**

Ncap is an open-source library for packet capture and network analysis which allows Wireshark to capture and analyze network traffic effectively. It enhances Wireshark's capabilities byproviding optimized packet capture.



Step 7: The next screen will ask if you want to install USBPcap, an open-source USB packet capture utility that lets you capture raw USB traffic, helping analyze and troubleshoot USB devices,this is not mandatory.

Click “Install” to begin the installation.

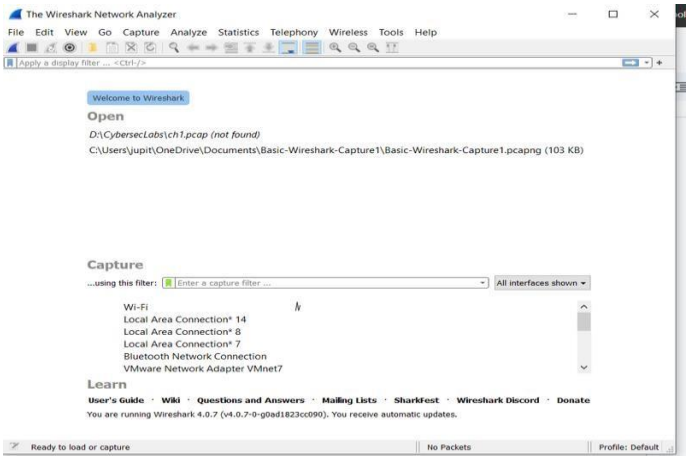
Step 8: Wireshark will now begin the installation process. A window will pop up duringinstallation to install cap.

Step 9: Ncap will begin the installation; click “Next” once complete.

Step 10: Wireshark will now complete its installation. Once complete, you can click “Next.”

Step 11: On the last window, click “Finish” to complete the setup**.**

Step 12: Wireshark will now be installed, and you can begin packet capturing.



When you install the wireshark program, the wireshark GUI with no data will be displayed.

Select one of the wireshark interface, eth0, eth1 will be displayed. Click “Start”for interface eth0to begin the Packet capture.

All packets being sent/received from/by the computer are now being captured by wireshark. Click”Start”.

Wireshark User Interface:

The wireshark interface has 5 major components;

- The **Command menus** are the standard pulldown menus located at top.
- The **Packet listing window** displays a one-line summary for each packet captured, it includes Packet number, Packet captured time, Packet’s source & destination address, Protocol type, Protocol specific information.
- The **Packet header details** window provides about packet selected in the packet listing window. It includes details about Ethernet frame and IP datagram of the packet. If the packet has been carried over by TCP/UDP, that details will also be displayed.
- **Packet contents window** display entire contents of the captured frame in both ASCII and hexadecimal format.
- In the **Packet display filter field,** the protocol name or other information can be entered to filter the information displayed in packet listing window.

Capturing Packets:

After installing and downloading wireshark, Launch it and click the name of an interface underInyerface List to start capturing packets.

Test Run:

Start any browser→ Start the wireshark software → Select an interface → Stop wireshark packetcapture once the browser has been displayed.

Colour coding: Packets will be highlighted in blue, green, black which helps to identify thetypes of traffic.

Green→ TCP traffic, Dark Blue → DNS traffic, Light Blue → UDP traffic, Black → TCPpackets with problems.

Inspecting Packets:

Click on any packet and go to the bottom pane.

Inspecting Packet flow:

We have a live packet data that contains all protocol message exchanged between yourcomputer and other network entities.

To filter the connection and to get a clear data type "http" in the filtering field. Note that directly typing the destination will not work as wireshark doesn't have ability to discern the protocolsfield.

To get more precise data set

http.host==www.netwoksecurity.eduRight click on any

packet → Select "Follow UDP Stream".

Close the window, change filter back to

"http.host==www.networksecurity.edu" follow apacket from the list that matches the filter.Use "Contains with other protocols."

TCPdump:

TCP (Transmission Control Protocol) facilitates the transmission of packets from source todestination.

Tcpdump is a command line utility that allows you to capture and analyze network traffic goingthrough your system. It is often used to help troubleshoot network issues, as well as a security tool.

It is a network monitoring and management utility that captures and records TCP/IP data on the run time. Tcpdump is designed to provide statistics about the number of packets received and captured at the operating node for network performance analysis, debugging and diagnosing networkbottlenecks and other network-oriented tasks.

Identifying UDP/TCP datagram:

IP packets have 8-bit header (Protocol for v4 and Next Header in v6) which determines which transport-layer protocol is used in the payload. For example, if it's 6, the payload is a TCP segment,and if it's 17 then that is an UDP.

TCP is connection-oriented while UDP is connectionless.

TCP sends data in a particular sequence, whereas there is no fixed order for UDP protocol.

Result:

Thus, the installation of Wireshark, TCPdump and observing the data transferred in client-servercommunication using UDP/TCP and Identifying the UDP/TCP datagram has been executed successfully.

PROGRAM:

KeyStore

Command to create KeyStore file:

keytool -genkeypair -keyalg RSA -keysize 2048 -validity 365 -alias myserverkey -keystore samlKeystore.jks -storepass password -keypass password -dname "CN=localhost,OU=Unknown,O=Unknown,L=Unknown,ST=Unknown,C=Unknown"

Server.java

```
import javax.net.ssl.*;
import java.io.*;
import java.security.*;

public class Server {
    public static void main(String[] args) {
        try {
            // Load the keystore
            char[] keystorePassword = "password".toCharArray();
            char[] keyPassword = "password".toCharArray();
            KeyStore keyStore = KeyStore.getInstance("JKS");
            try (FileInputStream fis = new FileInputStream("samlKeystore.jks")) {
                keyStore.load(fis, keystorePassword);
            }

            // Set up the key manager factory
            KeyManagerFactory kmf = KeyManagerFactory.getInstance("SunX509");
            kmf.init(keyStore, keyPassword);

            // Set up the SSL context
            SSLContext sslContext = SSLContext.getInstance("TLS");
            sslContext.init(kmf.getKeyManagers(), null, null);

            // Create the server socket factory
            SSLServerSocketFactory ssf = sslContext.getServerSocketFactory();
            SSLServerSocket serverSocket = (SSLServerSocket) ssf.createServerSocket(9999);

            System.out.println("Server started. Waiting for client connection...");

            // Accept client connections
            SSLSocket socket = (SSLSocket) serverSocket.accept();

            // Set up input and output streams
```

```
        BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
        PrintWriter out = new PrintWriter(socket.getOutputStream(), true);

        // Read message from client
        String message = in.readLine();
        System.out.println("Received message from client: " + message);

        // Send response back to client
        out.println("Message received by server");

        // Close streams and socket
        out.close();
        in.close();
        socket.close();
        serverSocket.close();
      } catch (Exception e) {
        e.printStackTrace();
      }
   }
}
```

Client.java

```
import javax.net.ssl.*;
import java.io.*;
import java.security.*;

public class Client {
   public static void main(String[] args) throws Exception {
      // Load the truststore
      char[] truststorePassword = "password".toCharArray();
      KeyStore trustStore = KeyStore.getInstance("JKS");
      FileInputStream fis = new FileInputStream("samlKeystore.jks");
      trustStore.load(fis, truststorePassword);

      // Set up the trust manager factory
      TrustManagerFactory tmf = TrustManagerFactory.getInstance("SunX509");
      tmf.init(trustStore);

      // Set up the SSL context
      SSLContext sslContext = SSLContext.getInstance("TLS");
      sslContext.init(null, tmf.getTrustManagers(), null);
```

```
        // Create the socket factory
        SSLSocketFactory sf = sslContext.getSocketFactory();
        SSLSocket socket = (SSLSocket) sf.createSocket("localhost", 9999);

        // Set up input and output streams
        PrintWriter out = new PrintWriter(socket.getOutputStream(), true);
        BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));

        // Send message to server
        out.println("Hello from client");

        // Read response from server
        String response = in.readLine();
        System.out.println("Response from server: " + response);

        // Close streams and socket
        out.close();
        in.close();
        socket.close();
    }
}
```

Output:

```
D:\NS LAB>keytool -genkeypair -keyalg RSA -keysize 2048 -validity 365 -alias myserverkey -keystore samlKeystore.jks -storepass password -keypass password -dna
me "CN=localhost,OU=Unknown,O=Unknown,L=Unknown,ST=Unknown,C=Unknown"
Generating 2,048 bit RSA key pair and self-signed certificate (SHA384withRSA) with a validity of 365 days
        for: CN=localhost, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown, C=Unknown

D:\NS LAB>javac Server.java

D:\NS LAB>java Server.java
Server started. Waiting for client connection...
Received message from client: Hello from client

D:\NS LAB>
```

```
D:\NS LAB>javac Client.java

D:\NS LAB>java Client.java
Response from server: Message received by server

D:\NS LAB>
```

Ex.No: 6	Experiment Eavesdropping, Dictionary Attack, MITM Attacks.

Aim:

To experiment Eavesdropping, Dictionary Attack, MITM Attacks.

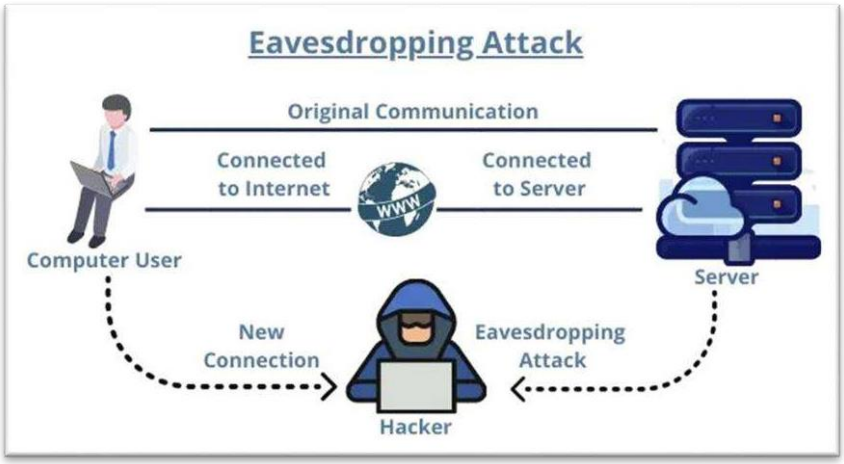
EAVESDROPPING:

Eavesdropping refers to the unauthorised and unseen intervention of a private, live conversation.

Sniffing or Eavesdropping pertains to the act of acquiring or intercepting data by capturing thecommunication flow within a network using a packet sniffer tool.

This technique involves monitoring the packets of information passing through the network,allowing unauthorized access to sensitive data, akin to theft or unauthorized interception of information.

During the transmission of data across networks, if the data packets lack encryption, they become vulnerable to interception, enabling unauthorized parties to read the contents of these network packetswith the use of a sniffer.



Categories of Network Sniffing:

Active and Passive Sniffing attacks are two distinct categories of network sniffing techniquesused by attackers to intercept and analyze data traffic.

1. **Active Sniffing:**

Active Sniffing is performed through a Switch and it is easy to detect.

It involves more direct interaction with the network traffic. Instead of just observing and capturing data, the attacker actively injects or modifies packets within the communication flow.

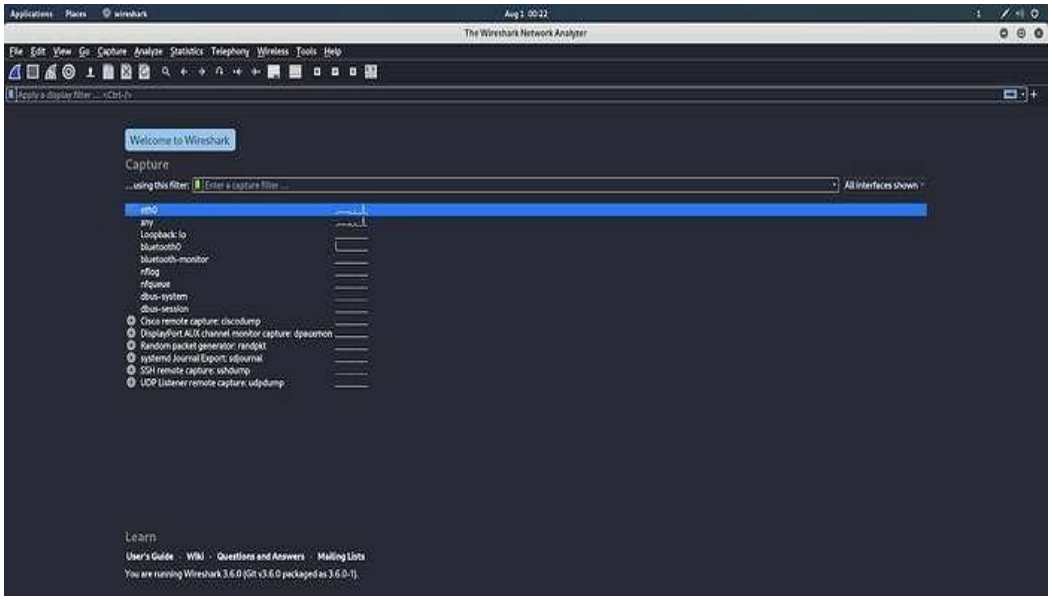
2.Passive Sniffing:

Passive Sniffing is performed through a Hub which is difficult to detect.

It involves silently capturing and monitoring network traffic without altering or modifying thedata being transmitted. The attacker's presence is relatively discreet, as they do not actively participate in the communication process. They just observe the data that flows through the network, looking for sensitive/crucial information that is not encrypted.

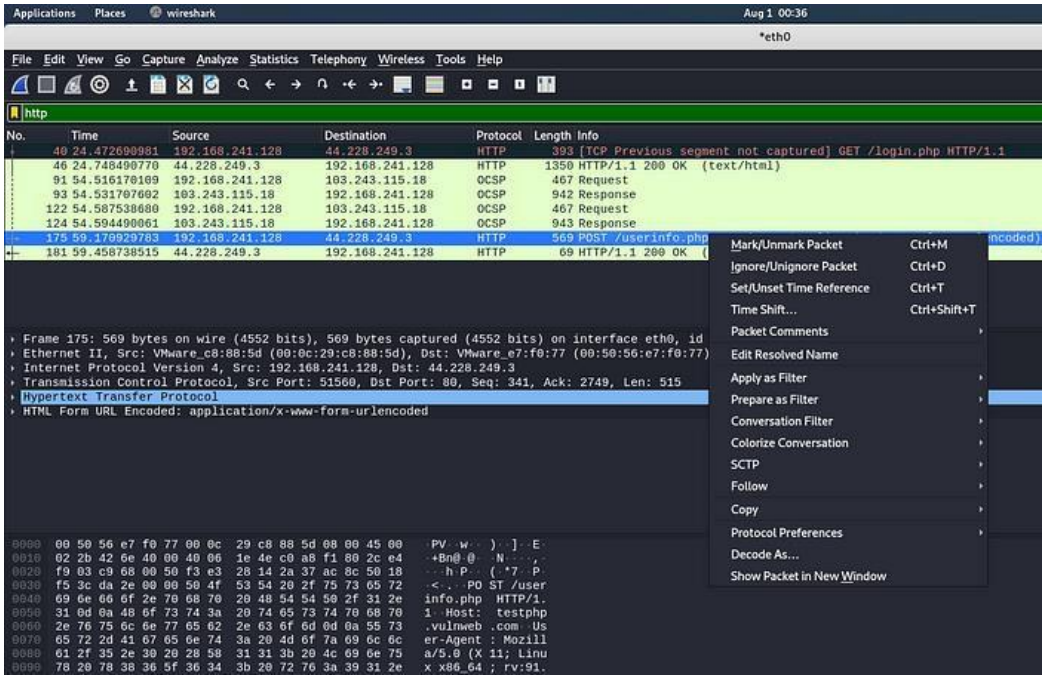
Experimenting Eavesdropping:

Step 1: Launch the Wireshark software on your computer and choose the 'eth0' option, In your webbrowser, input the URL we want to capture login credentials from.



Step 2: Input the login credentials, which are 'test', and then click on the login button.

Step 3: Then by entering 'http' in the filter section, the captured packets using the HTTP protocol willbe shown. Choose 'Follow' to access additional options, then select 'http stream' from the available choices.



Step 4: Explore the provided information, and you will uncover the login credentials.

Output:

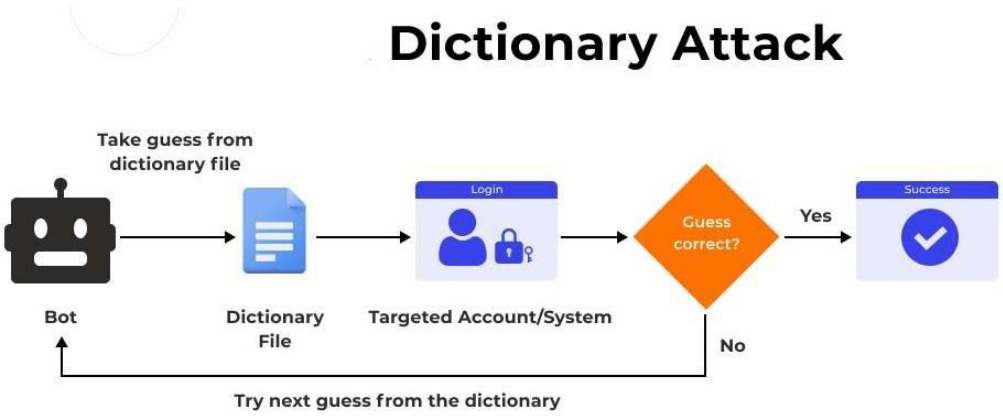
DICTIONARY ATTACK:

A Dictionary Attack is an attack vector used by the attacker to break in a system, which is password protected, by putting technically every word in a dictionary as a form of password for thatsystem. This attack vector is a form of Brute Force Attack.

Like the brute force attack, the dictionary attack aims to break in by logging in using usernameand password combinations. It is only inefficient as far as its overall success rate: automated scriptscan do this in a matter of seconds.

A hacker will look for applications and websites that don't lock a user out quickly for incorrectusername and password combinations and don't require other forms of authentication when signing in. Sites that allow simple passwords are especially vulnerable.

Suppose the target website or application does not adequately monitor suspicious behavior likethis or has lax password rules. In that case, the website runs a high risk of data disclosure resulting from a dictionary attack. Leaked password databases have become a common feature of modern dictionary attacks. Attempting to log in with username and password combinations used multiple times elsewhere makes these dictionary attacks much more successful and potentially harder to detect on the application or website's end.



Result:

Thus, Eavesdropping and Dictionary Attack have been implemented successfully.

Ex.No: 7	Experiment with Sniff Traffic using ARP Poisoning.

Aim:

To Experiment Sniff Traffic using ARP Poisoning.

ARP Poisoning:

Address Resolution Protocol (ARP) poisoning is an attack that involves sending spoofed ARP messages over a local area network. It's also known as ARP spoofing, ARP poison routing and ARPcache poisoning.

ARP poisoning is a type of man-in-the-middle attack that can be used to stop network traffic, change it, or intercept it. The technique is often used to initiate further offensives, such as sessionhijacking or denial-of-service.

The relationship between a given MAC address and its IP address is kept in a table known as theARP cache. When a packet heading towards a host on a LAN gets to the gateway, the gateway uses ARP to associate the MAC or physical host address with its correlating IP address.

The host then searches through its ARP cache. If it locates the corresponding address, it is used toconvert the format and packet length. Otherwise, ARP will send out a request packet that asks other machines on the local network if they know the correct address. When a machine replies with the address, the ARP cache is updated.

ARP Poisoning Countermeasures:

We can use several methods to prevent ARP poisoning, each with its own positives and negatives. These include static ARP entries, encryption, VPNs, packet sniffing, Poisoning detectionsoftware, OS security,etc.

Static ARP entries:

This solution involves a lot of administrative overhead and is only recommended forsmaller networks. It involves adding an ARP entry for every machine on a network into each individual computer.

Mapping the machines with sets of static IP and MAC addresses helps to preventspoofing attacks, because the machines can ignore ARP replies**.**

Encryption:

Protocols such as HTTPS and SSH can also help to reduce the chances of a successful ARP poisoning attack. When traffic is encrypted, the attacker would have to

go to the additional stepof tricking the target's browser into accepting an illegitimate certificate.

VPN: If it is just a single person making a potentially dangerous connection, such as using public wifi at an airport, then a VPN will encrypt all of the data that travels between the client and theexit server.

Operating System Security:

This measure is dependent on the OS been used. The following are the basic techniquesused by various operating systems.

- Linux: These work by ignoring unsolicited ARP reply packets.
- Microsoft Windows: The ARP cache behavior can be configured via the registry. The following list includes some of the software that can be used to protect networks against sniffing;

  AntiARP- provides protection against both passive and active sniffing

  Agnitum Outpost Firewall-provides protection against passive sniffing

  XArp- provides protection against both passive and active sniffing
- Mac OS: ArpGuard can be used to provide protection. It protects against both active and passive sniffing.

Sniff Traffic:

Network sniffing is the process of intercepting data packets sent over a network. This can bedone by the specialized software program or hardware equipment. Sniffing can be used to;

- Capture sensitive data such as login credentials
- Eavesdrop on chat messages
- Capture files have been transmitted over a network.

Types of Sniffing:

Passive sniffing is intercepting packages transmitted over a network that uses a hub. It is calledpassive sniffing because it is difficult to detect. It is also easy to perform as the hub sends broadcastmessages to all the computers on the network.

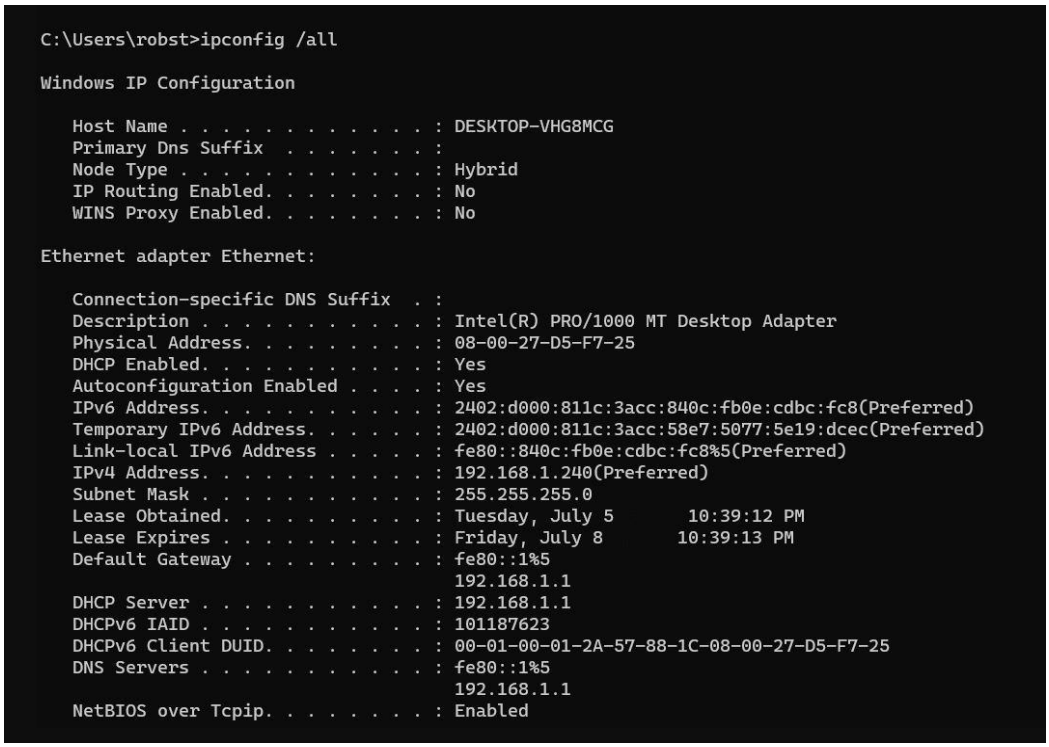
Active sniffing is intercepting packages transmitted over a network that uses a switch. There are two main methods used to sniff switch linked networks, ARP Poisoning, and MAC flooding**.**

Sniff Traffic using ARP Poisoning:

Step 1: Open the command prompt and Enter the command.

```
ipconfig /all
```

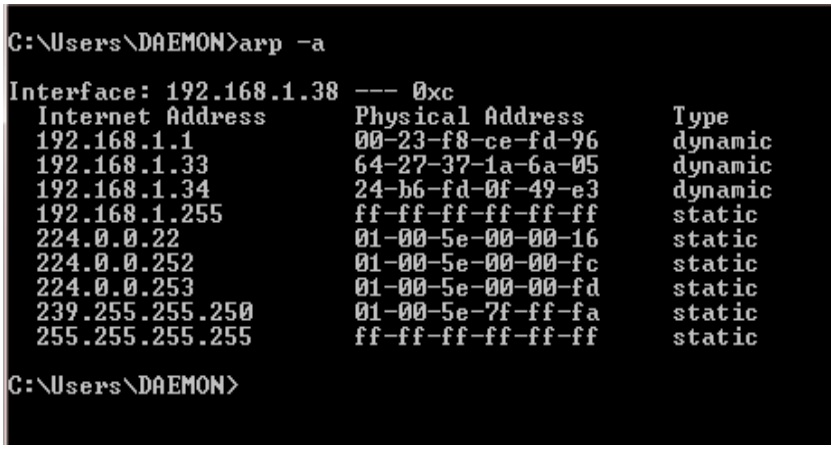
Detailed information about all the network connections available on your computer will be displayed. The results shown below are for a broadband modem to show the MAC address and IPv4format and wireless network to show IPv6 format.



Step 2: apr command calls the ARP configure program located in Windows/System32 directory

-a is the parameter to display to contents of the ARP cache.

```
arp –a
```



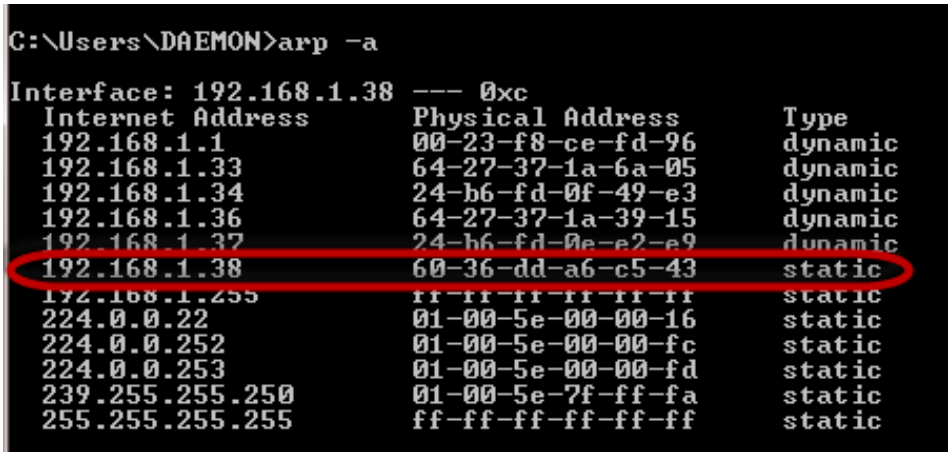
Step 3: Static entries are added manually and are deleted when the computer is restarted.

Step 4: After getting the IP/MAC address, enter the following command.

```
arp -s 192.168.1.38 60-36-DD-A6-C5-43
```

Step 5: To view the ARP cache

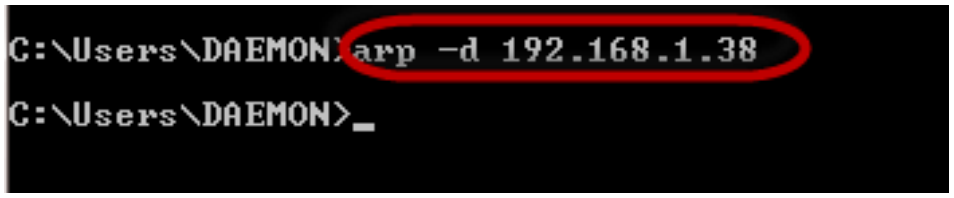
```
arp -a
```



The IP address has been resolved to the MAC address we provided and it is of a static type.

Step 6: Command to remove an entry.

```
arp -d 192.168.1.38
```



ARP poisoning works by sending fake MAC addresses to the switch.

Result:

Thus, the Sniff Traffic using ARP Poisoning have been executed successfully.

Ex. No. : 8 Date:	INTRUSION DETECTION SYSTEM (IDS)

AIM:

To demonstrate the Intrusion Detection System (IDS) using Snort software tool.

STEPS ON CONFIGURING AND INTRUSION DETECTION:

1. Download Snort from the Snort.org website. (http://www.snort.org/snort-downloads)
2. Download Rules(https://www.snort.org/snort-rules). You must register to get the rules. (You should download these often)
3. Double-click on the .exe to install snort. This will install snort in the "C:\Snort" folder.It isimportant to have WinPcap (https://www.winpcap.org/install/) installed
4. Extract the Rules file. You will need WinRAR for the .gz file.
5. Copy all files from the "rules" folder of the extracted folder. Now paste the rules into *"C:\Snort\rules"* folder.
6. Copy the "snort.conf" file from the "etc" folder of the extracted folder. You must paste it into"C:\Snort\etc" folder. Overwrite any existing file. Remember if you modify your snort.conf file and download a new file, you must modify it for Snort to work.
7. Open a command prompt (cmd.exe) and navigate to the folder "C:\Snort\bin" folder. (at thePrompt, type cd\snort\bin)
8. To start (execute) snort in sniffer mode use following command:
snort -dev -i 3
-i indicates the interface number. You must pick the correct interface number. In my case, it is 3.
-dev is used to run snort to capture packets on your network.

To check the interface list, use following command:
snort -W

Downloaded by K L NEELA (klneelaaut@gmail.com)

Finding an interface

You can tell which interface to use by looking at the Index number and finding Microsoft. As you can see in the above example, the other interfaces are for VMWare. My interface is 3.

9. To run snort in IDS mode, you will need to configure the file "snort.conf" according to your network environment.

10. To specify the network address that you want to protect in snort.conf file, look for the following line.

var HOME_NET 192.168.1.0/24 (You will normally see any here)

11. You may also want to set the addresses of DNS_SERVERS, if you have some on your network.

Example:

example snort

12. Change the RULE_PATH variable to the path of rules folder.

var RULE_PATH c:\snort\rules

pathto rules

13. Change the pathof all library files with the name and path on your system. and you must change the path of snort_dynamicpreprocessorvariable.

C:\Snort\lib\snort_dynamiccpreprocessor

You need to do this to all library files in the "C:\Snort\lib" folder. The old path might be: "/usr/local/lib/…". you will need to replace that path with your system path. Using C:\Snort\lib

14. Change the path ofthe "dynamicengine" variable value in the "snort.conf" file..

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

15 Add the paths 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 commented with a #.
include $RULE_PATH/icmp.rules
17. You can also remove the comment of ICMP-info rules comment, if it is commented.
include $RULE_PATH/icmp-info.rules
18. To add log files to store alerts generated by snort, search for the "output log" test in snort.conf and add the following line:
output alert_fast: snort-alerts.ids
19. Comment (add a #) the whitelist $WHITE_LIST_PATH/white_list.rules and the blacklist

Change the nested_ip inner , \ to nested_ip inner #, \
20. Comment out (#) 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.
22. To start snort in IDS mode, run the following command:

snort -c c:\snort\etc\snort.conf -l c:\snort\log -i 3
(Note: 3 is used for my interface card)

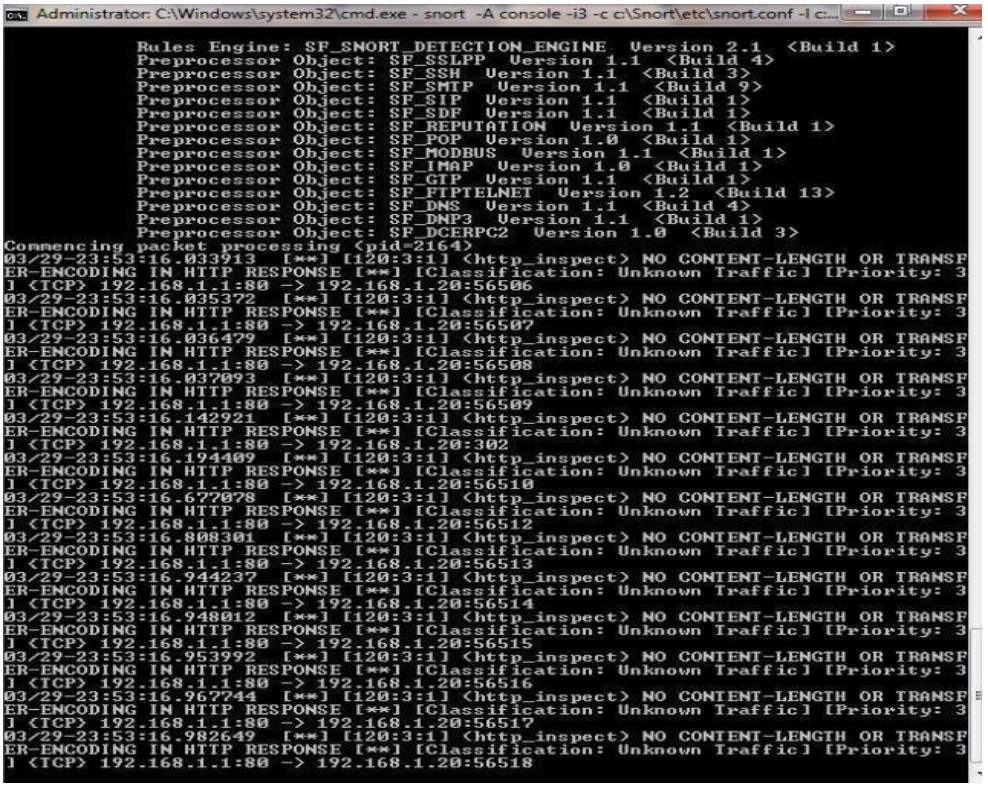
Ifa log is created, select the appropriate program to open it. You can use WordPard or NotePad++ to read the file.

To generate Log files in ASCII mode, you can use following command while running snort in IDS mode:
snort -A console -i3 -c c:\Snort\etc\snort.conf -l c:\Snort\log -K ascii

23. Scan the computer that is running snort from another computer by using PING or NMap (ZenMap).

After scanning or during the scan you can check the snort-alerts.ids file in the log folder to insure it is logging properly. You will see IP address folders appear.

Snort monitoring traffic –



RESULT:

Thus the Intrusion Detection System(IDS) has been demonstrated using the Open Source Intrusion Detection Tool Snort.

Ex.No. :09 **Monitoring Malware Using Rootkit Hunter**

AIM:

To install a rootkit hunter and find the malwares in a computer.

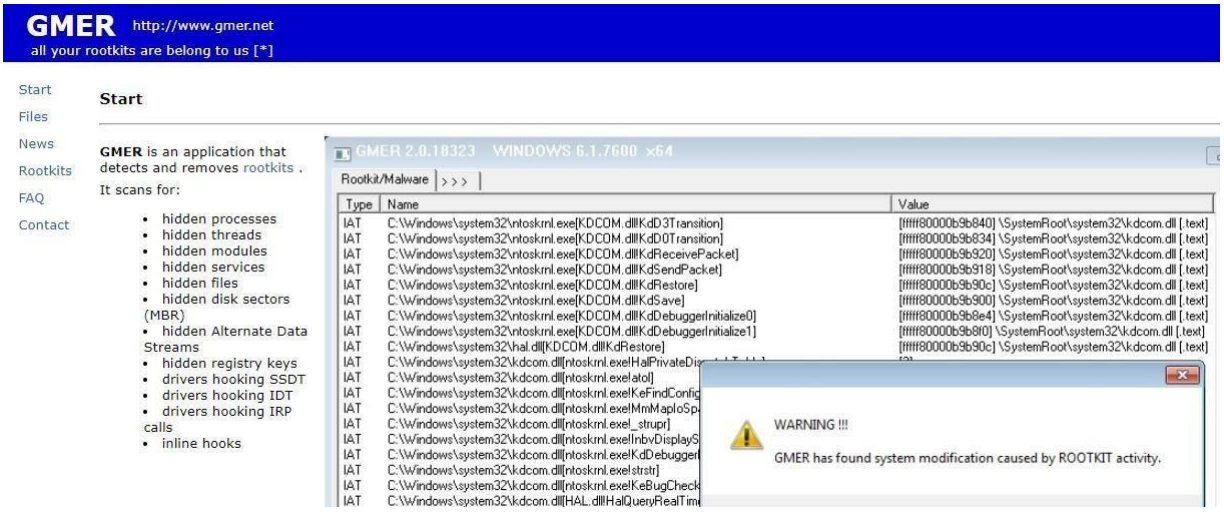
ROOTKIT HUNTER:

- rkhunter (Rootkit Hunter) is a Unix-based tool that scans for rootkits, backdoors andpossible local exploits.
- It does this by comparing SHA-1 hashes of important files with known good ones in online databases, searching for default directories (of rootkits), wrong permissions, hidden files, suspicious strings in kernel modules, and special tests for Linux and FreeBSD.
- rkhunter is notable due to its inclusion in popular operating systems (Fedora, Debian,etc.)
- The tool has been written in Bourne shell, to allow for portability. It can run on almostall UNIX-derived systems.

GMER ROOTKIT TOOL:

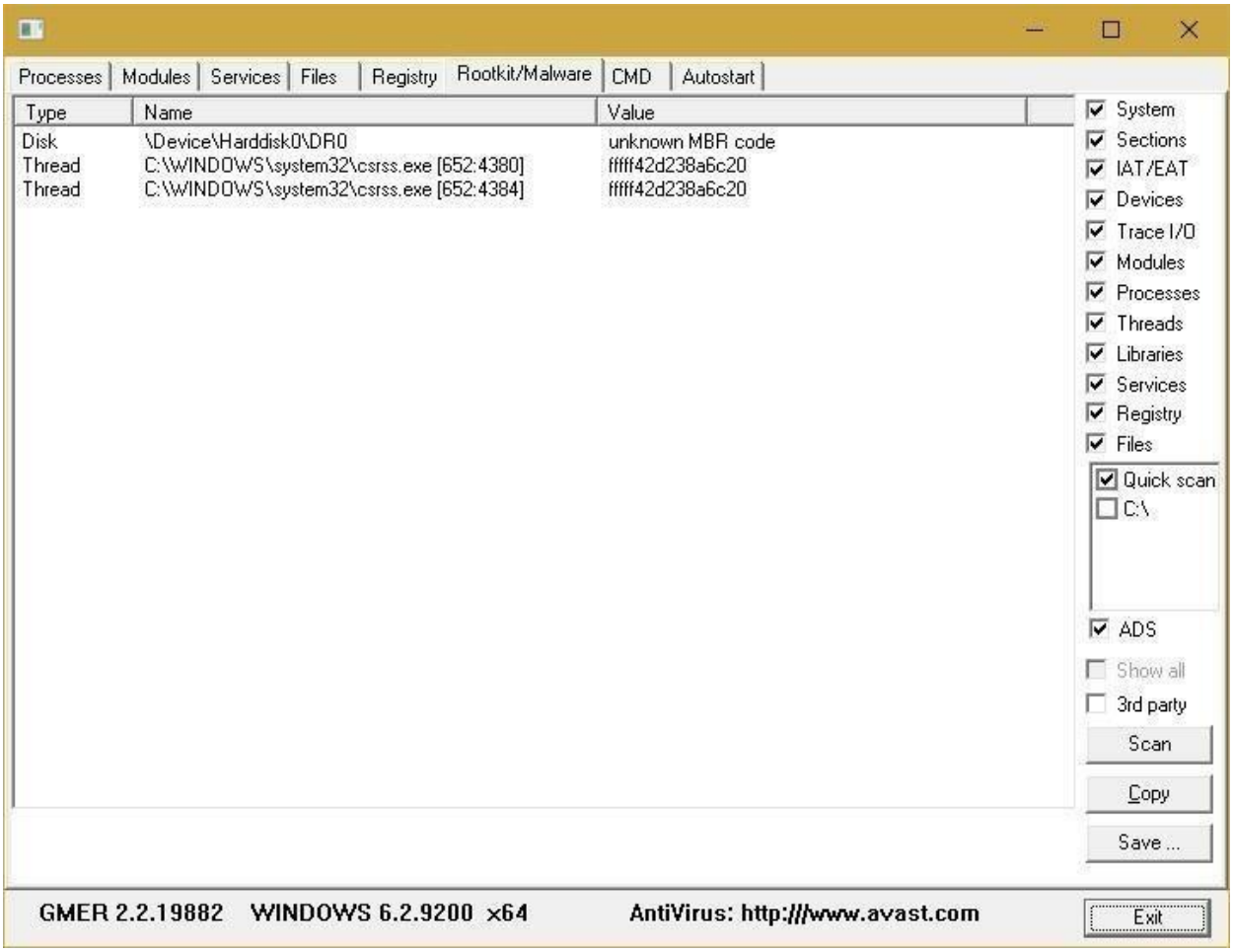
- GMER is a software tool written by a Polish researcher Przemysław Gmerek, fordetecting and removing rootkits.
- It runs on Microsoft Windows and has support for Windows NT, 2000, XP, Vista, 7, 8
  and 10. With version 2.0.18327 full support for Windows x64 is added.

Step 1



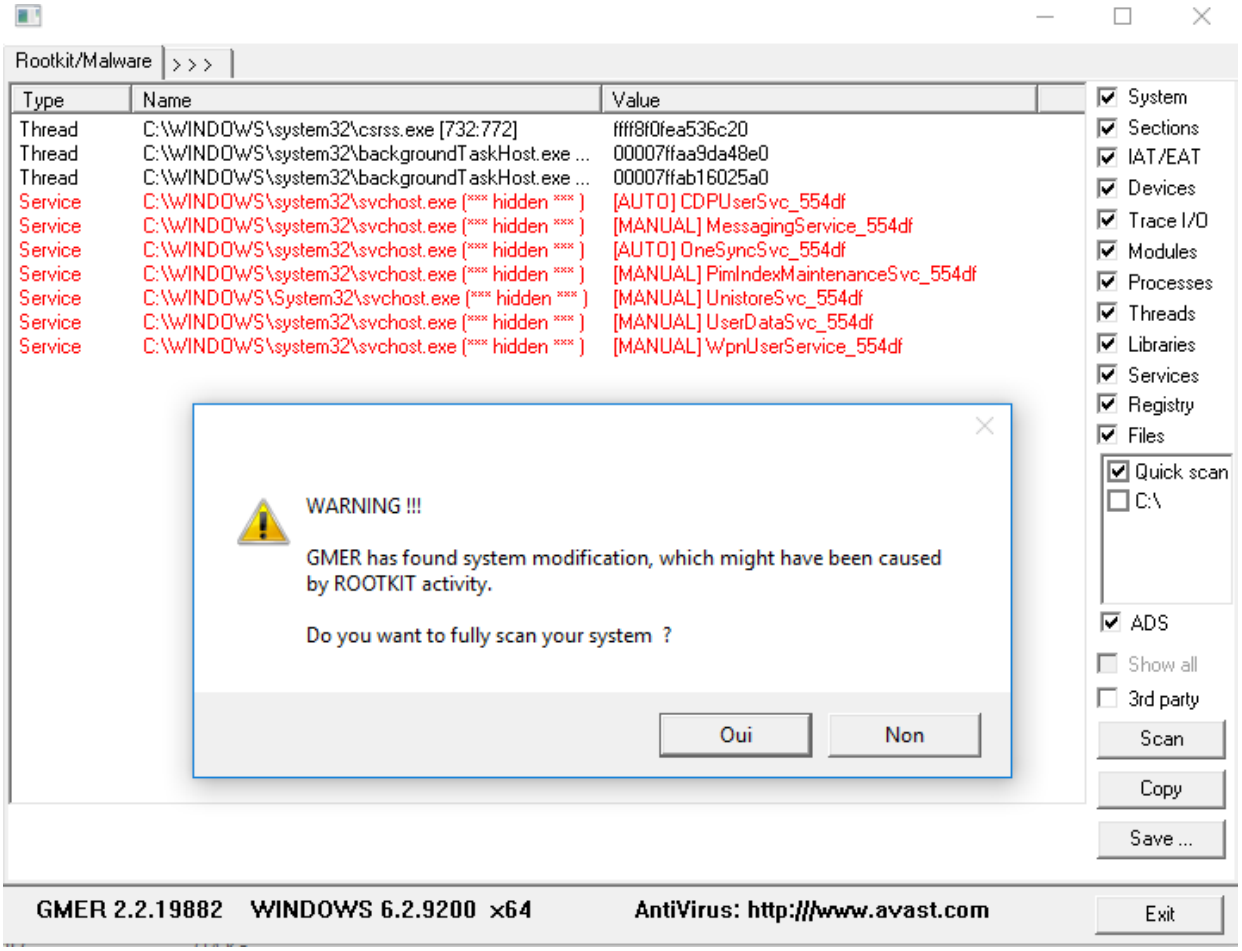
Visit GMER's website (see Resources) and download the GMER executable. Click the "Download EXE" button to download the program with a random file name, assome rootkits will close “gmer.exe” before you can open it.

Step 2



Double-click the icon for the program.
Click the "Scan" button in the lower-right corner of the dialog box. Allow the program toscan your entire hard drive.

Step 3



When the program completes its scan, select any program or file listed in red. Right-click it and select "Delete." If the red item is a service, it may be protected. Right-click the service and select "Disable." Reboot your computer and run the scan again, this time selecting "Delete" when that service is detected. When your computer is free of Rootkits, close the program and restart your PC.

RESULT:

A rootkit hunter software tool *gmer* has been installed and the rootkits have been detected.

Ex. No : 10	Study to configure Firewall, VPN

AIM:

To study and configure firewall and VPN.

PROCEDURE FOR CONFIGURING FIREWALL AND VPN:

Working with Windows Firewall in Windows 7

Firewall in Windows 7

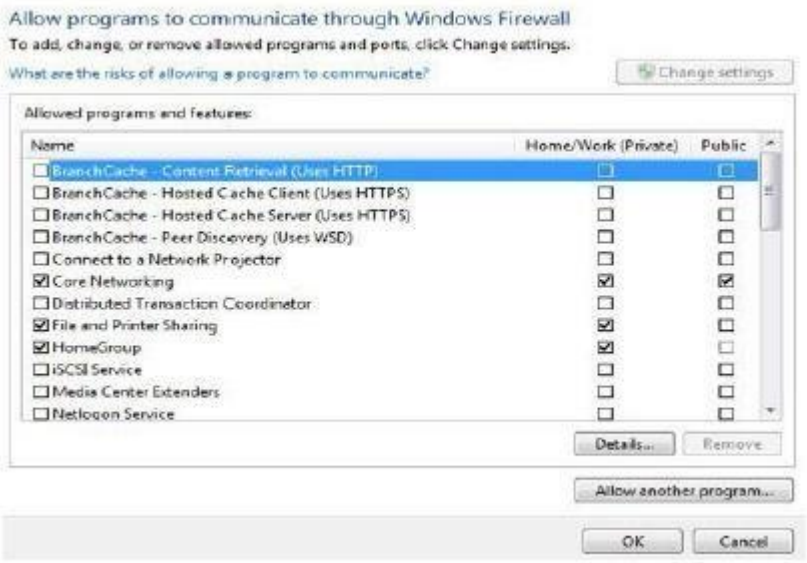
Windows 7 comes with two firewalls that work together. One is the Windows Firewall, and the other is Windows Firewall with Advanced Security (WFAS). The main difference between them is the complexity of the rules configuration. Windows Firewall uses simple rules that directly relate to a program or a service. The rules in WFAS can be configured based on protocols, ports, addresses and authentication. By default, both firewalls come with predefined set of rules that allow us to utilize network resources. This includes things like browsing the web, receiving e-mails, etc.Other standard firewall exceptions are File and Printer Sharing, Network Discovery, Performance Logs and Alerts, Remote Administration, Windows Remote Management, Remote Assistance, Remote Desktop, Windows Media Player, Windows Media Player Network Sharing Service.

With firewall in Windows 7 we can configure inbound and outbound rules. By default, all outbound traffic is allowed, and inbound responses to that traffic are also allowed. Inbound traffic initiated from external sources is automatically blocked.

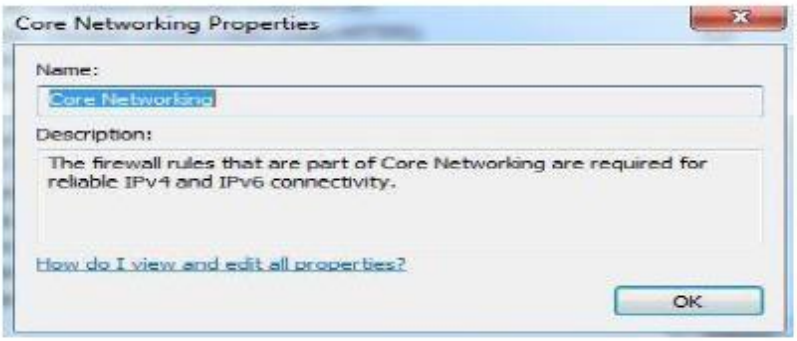
Configuring Windows Firewall

To open Windows Firewall we can go to Start > Control Panel > Windows Firewall.

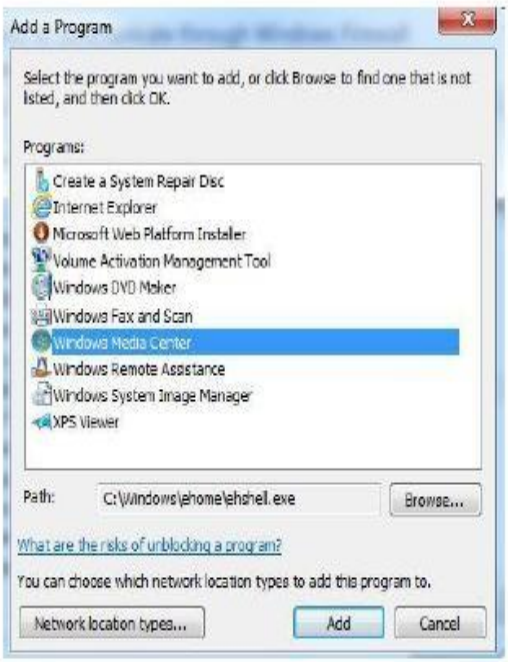
it is also configured to block all connections to programs that are not on the list of allowed programs.To configure exceptions we can go to the menu on the left and select "Allow a program or feature trough Windows Firewall" option.



To change settings in this window we have to click the "Change settings" button.the Core Networking feature is allowed on both private and public networks, while the File and Printer Sharing is only allowed on private networks.



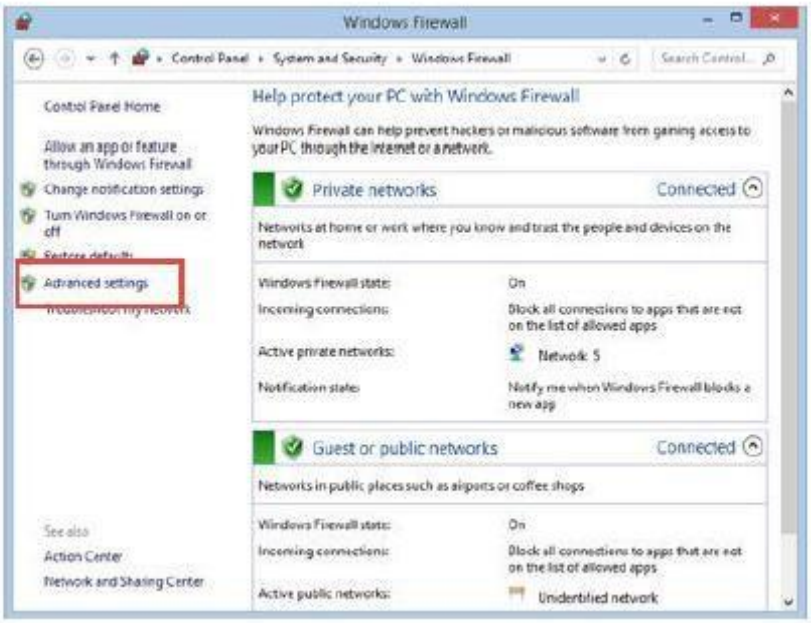
If we have a program on our computer that is not in this list, we can manually add it by clicking on the "Allow another program" button.



a Program

Program will be allowed to communicate by clicking on the "Network location types" button.

Windows Firewall can be turned off completely. To do that selects the "Turn Windows Firewall on or off" option from the menu on the left.



Windows Firewall is actually a Windows service. As you know, services can be stopped and started. If the Windows Firewall service is stopped, the Windows Firewallwill not work.

RESULT

Thus the firewall configuration and VPN installations are studied successfully.