



ccs354 network ecurity

Network Security (Anna University)



Scan to open on Studocu

LIST OF EXPERIMENTS

<i>Ex. No.</i>	<i>Date</i>	<i>Name of the Experiment</i>	<i>Staffe Sign</i>
1(a).		Implementing symmetric key algorithms -DES	
1(b)		Implementing symmetric key algorithms - AES	
2(a)		Implementing asymmetric key algorithms	
2(b)		Implementing Key exchange algorithms	
3.		Calculate the message digest of a text using the SHA-1 algorithm.	
4.		Implement the SIGNATURE SCHEME - Digital Signature Standard.	
5.		Installation of Wire shark, tcpdump and observe data transferred in client-servercommunication using UDP/TCP and identify the UDP/TCP datagram.	
6.		Check message integrity and confidentiality using SSL	
7		Experiment Eavesdropping, Dictionary attacks, MITM attacks	
8		Experiment with Sniff Traffic using ARP Poisoning	
9		Demonstrate intrusion detection system using any tool.	
10		Explore network monitoring tools	
11		Study to configure Firewall, VPN	

Ex. No : 1(a)
Date :

Data Encryption Standard (DES) Algorithm
(User Message Encryption)

AIM:

To use Data Encryption Standard (DES) Algorithm for a practical application like User Message Encryption.

ALGORITHM:

1. Create a DES Key.
2. Create a Cipher instance from Cipher class, specify the following information and separated by a slash (/).
 - a. Algorithm name
 - b. Mode (optional)
 - c. Padding scheme (optional)
3. Convert String into **Byte[]** array format.
4. Make Cipher in encrypt mode, and encrypt it with **Cipher.doFinal()** method.
5. Make Cipher in decrypt mode, and decrypt it with **Cipher.doFinal()** method.

PROGRAM:

DES.java

```
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;

import javax.crypto.BadPaddingException;
import javax.crypto.Cipher;
import javax.crypto.IllegalBlockSizeException;
import javax.crypto.KeyGenerator;
import javax.crypto.NoSuchPaddingException;
import javax.crypto.SecretKey;

public class DES
{
    public static void main(String[] argv) {

        try{
            System.out.println("Message Encryption Using DES Algorithm\n-----");
            KeyGenerator keygenerator = KeyGenerator.getInstance("DES");
            SecretKey myDesKey = keygenerator.generateKey();
            Cipher desCipher;
            desCipher = Cipher.getInstance("DES/ECB/PKCS5Padding");
            desCipher.init(Cipher.ENCRYPT_MODE, myDesKey);
            byte[] text = "Secret Information ".getBytes();
            System.out.println("Message [Byte Format] : " + text);
            System.out.println("Message : " + new String(text));
            byte[] textEncrypted = desCipher.doFinal(text);
            System.out.println("Encrypted Message: " + textEncrypted);
            desCipher.init(Cipher.DECRYPT_MODE, myDesKey);
            byte[] textDecrypted = desCipher.doFinal(textEncrypted);

            System.out.println("Decrypted Message: " + new String(textDecrypted));
```

```

        }catch(NoSuchAlgorithmException e){
            e.printStackTrace();
        }catch(NoSuchPaddingException e){
            e.printStackTrace();
        }catch(InvalidKeyException e){
            e.printStackTrace();
        }catch(IllegalBlockSizeException e){
            e.printStackTrace();
        }catch(BadPaddingException e){
            e.printStackTrace();
        }
    }
}

```

OUTPUT:

Message Encryption Using DES Algorithm

 Message [Byte Format] : [B@4dcbadb4

Message : Secret Information

Encrypted Message: [B@504bae78

Decrypted Message: Secret Information

RESULT:

Thus the java program for DES Algorithm has been implemented and the output verified successfully.

Ex. No : 1(b) Date: Advanced Encryption Standard (AES) Algorithm (URL Encryption)

AIM:

To use Advanced Encryption Standard (AES) Algorithm for a practical application like URL Encryption.

ALGORITHM:

1. AES is based on a design principle known as a substitution–permutation.
2. AES does not use a Feistel network like DES, it uses variant of Rijndael.
3. It has a fixed block size of 128 bits, and a key size of 128, 192, or 256 bits.
4. AES operates on a 4×4 column-major order array of bytes, termed the state

PROGRAM:

AES.java

```
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) {
    final String secretKey = "annaUniversity";

    String originalString = "www.annauniv.edu";
    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:

URL Encryption Using AES Algorithm

```

-----
Original URL : www.annauniv.edu
Encrypted URL : vibpFJW6Cvs5Y+L7t4N6YWWe07+JzS1d3CU2h3mEvEg=
Decrypted URL : www.annauniv.edu

```

RESULT:

Thus the java program for AES Algorithm has been implemented for URL Encryption and the output verified successfully.

Ex. No : 2(a) Date:
RSA Algorithm

AIM:

To implement RSA (Rivest–Shamir–Adleman) algorithm by using HTML and Javascript.

ALGORITHM:

1. Choose two prime number p and q
2. Compute the value of n and ϕ
3. Find the value of e (public key)
4. Compute the value of d (private key) using $\text{gcd}()$
5. Do the encryption and decryption
 - a. Encryption is given as,

$$c = t^e \bmod n$$

- b. Decryption is given as,

$$t = c^d \bmod n$$

PROGRAM:

rsa.html

```
<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>
    </table>
  </center>
</body>
</html>
```

```

</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:

RSA Algorithm

Implemented Using HTML & Javascript

Enter First Prime Number:	<input type="text" value="53"/>
Enter Second Prime Number:	<input type="text" value="59"/>
Enter the Message(cipher text): [A=1, B=2,...]	<input type="text" value="89"/>
Public Key:	3127
Exponent:	3
Private Key:	2011
Cipher Text:	1394
<input type="button" value="Apply RSA"/>	

RESULT:

Thus the RSA algorithm has been implemented using HTML & CSS and the output has been verified successfully.

Ex. No : 2(b) Date:

Diffie-Hellman key exchange algorithm

AIM:

To implement the Diffie-Hellman Key Exchange algorithm for a given problem .

ALGORITHM:

1. Alice and Bob publicly agree to use a modulus $p = 23$ and base $g = 5$ (which is a primitive root modulo 23).
2. Alice chooses a secret integer $a = 4$, then sends Bob $A = g^a \bmod p$
 - o $A = 5^4 \bmod 23 = 4$
3. Bob chooses a secret integer $b = 3$, then sends Alice $B = g^b \bmod p$
 - o $B = 5^3 \bmod 23 = 10$
4. Alice computes $s = B^a \bmod p$
 - o $s = 10^4 \bmod 23 = 18$
5. Bob computes $s = A^b \bmod p$
 - o $s = 4^3 \bmod 23 = 18$
6. Alice and Bob now share a secret (the number 18).

PROGRAM:

DiffieHellman.java

```
class DiffieHellman {  
    public static void main(String args[]) {  
        int p = 23; /* publicly known (prime number) */  
        int g = 5; /* publicly known (primitive root) */  
        int x = 4; /* only Alice knows this secret */  
        int y = 3; /* only Bob knows this secret */  
        double aliceSends = (Math.pow(g, x)) % p;  
        double bobComputes = (Math.pow(aliceSends, y)) % p;  
        double bobSends = (Math.pow(g, y)) % p;  
        double aliceComputes = (Math.pow(bobSends, x)) % p;  
        double sharedSecret = (Math.pow(g, (x * y))) % p;  
        System.out.println("simulation of Diffie-Hellman key exchange algorithm\n-----  
.....");  
        System.out.println("Alice Sends : " + aliceSends);  
        System.out.println("Bob Computes : " + bobComputes);  
        System.out.println("Bob Sends : " + bobSends);  
        System.out.println("Alice Computes : " + aliceComputes);  
        System.out.println("Shared Secret : " + sharedSecret);  
    }  
}
```

```

/* shared secrets should match and equality is transitive */
if ((aliceComputes == sharedSecret) && (aliceComputes == bobComputes))
    System.out.println("Success: Shared Secrets Matches! " + sharedSecret);
else
    System.out.println("Error: Shared Secrets does not Match");
}
}

```

OUTPUT:

simulation of Diffie-Hellman key exchange algorithm

```

-----
Alice Sends : 4.0
Bob Computes : 18.0
Bob Sends : 10.0
Alice Computes : 18.0
Shared Secret : 18.0
Success: Shared Secrets Matches! 18.0

```

RESULT:

Thus the *Diffie-Hellman key exchange algorithm* has been implemented using Java Program and the output has been verified successfully.

Ex. No : 3

Date: **SHA-1 Algorithm**

AIM:

To Calculate the message digest of a text using the SHA-1 algorithm.

ALGORITHM:

1. Append Padding Bits
2. Append Length - 64 bits are appended to the end
3. Prepare Processing Functions
4. Prepare Processing Constants
5. Initialize Buffers
6. Processing Message in 512-bit blocks (L blocks in total message)

PROGRAM:

sha1.java

```
import java.security.*;
```

```
public class sha1 {  
    public static void main(String[] a) {  
        try {  
            MessageDigest md = MessageDigest.getInstance("SHA1");  
            System.out.println("Message digest object info:\n-----");  
            System.out.println("Algorithm=" + md.getAlgorithm());  
            System.out.println("Provider=" + md.getProvider());  
            System.out.println("ToString=" + md.toString());  
            String input = "";  
            md.update(input.getBytes());  
            byte[] output = md.digest();  
            System.out.println();  
            System.out.println("SHA1(\"" + input + "\")=" + bytesToHex(output));  
            input = "abc";  
            md.update(input.getBytes());  
            output = md.digest();  
            System.out.println();  
            System.out.println("SHA1(\"" + input + "\")=" + bytesToHex(output));  
            input = "abcdefghijklmnopqrstuvwxyz";
```

```

        md.update(input.getBytes());

        output = md.digest();
        System.out.println();
        System.out.println("SHA1(\"" + input + "\")=" + bytesToHex(output));
        System.out.println();
    } catch (Exception e) {
        System.out.println("Exception:" + e);
    }
}

private 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 (byte aB : b) {
        buf.append(hexDigit[(aB >> 4) & 0x0f]);
        buf.append(hexDigit[aB & 0x0f]);
    }

    return buf.toString();
}
}

```

OUTPUT:

Message digest object info:

```

-----
Algorithm=SHA1
Provider=SUN version 12
ToString=SHA1 Message Digest from SUN, <initialized>

```

SHA1("")=DA39A3EE5E6B4B0D3255BFEF95601890AFD80709

SHA1("abc")=A9993E364706816ABA3E25717850C26C9CD0D89D

SHA1("abcdefghijklmnopqrstuvwxyz")=32D10C7B8CF96570CA04CE37F2A19D84240D3A89

RESULT:

Thus the *Secure Hash Algorithm (SHA-1)* has been implemented and the output has been verified successfully.

Ex. No : 4

Digital Signature Standard

Date:

AIM:

To implement the SIGNATURE SCHEME - Digital Signature Standard.

ALGORITHM:

1. Create a KeyPairGenerator object.
2. Initialize the KeyPairGenerator object.
3. Generate the KeyPairGenerator. ...
4. Get the private key from the pair.
5. Create a signature object.
6. Initialize the Signature object.
7. Add data to the Signature object
8. Calculate the Signature

PROGRAM:

```
import java.security.KeyPair;
import java.security.KeyPairGenerator;
import java.security.PrivateKey;
import java.security.Signature;
import java.util.Scanner;

public class CreatingDigitalSignature {
    public static void main(String args[]) throws Exception {

        Scanner sc = new Scanner(System.in);
        System.out.println("Enter some text");
        String msg = sc.nextLine();

        KeyPairGenerator keyPairGen = KeyPairGenerator.getInstance("DSA");

        keyPairGen.initialize(2048);

        KeyPair pair = keyPairGen.generateKeyPair();

        PrivateKey privKey = pair.getPrivate();

        Signature sign = Signature.getInstance("SHA256withDSA");
        sign.initSign(privKey);
        byte[] bytes = "msg".getBytes();

        sign.update(bytes);

        byte[] signature = sign.sign();

        System.out.println("Digital signature for given text: "+new String(signature, "UTF8"));
    }
}
```

OUTPUT:

Enter some text

Hi how are you

Digital signature for given text: 0=@gRD???-?.???? /yGL?i??a!?

RESULT:

Thus the Digital Signature Standard Signature Scheme has been implemented and the output has been verified successfully.

Installation of Wire shark, tcpdump and observe data transferred in client- server communication

Ex. No : 5

Date:

Aim:

To installation of Wire shark, tcpdump and observe data transferred in client-server communication using UDP/TCP and identify the UDP/TCP datagram.

Introduction:

The first part of the lab introduces packet sniffer, Wireshark. Wireshark is a free open-source network protocol analyzer. It is used for network troubleshooting and communication protocol analysis. Wireshark captures network packets in real time and display them in human-readable format. It provides many advanced features including live capture and offline analysis, three-pane packet browser, coloring rules for analysis. This document uses Wireshark for the experiments, and it covers Wireshark installation, packet capturing, and protocol analysis.



Figure 1: Wireshark in Kali Linux

Background

TCP/IP Network Stack

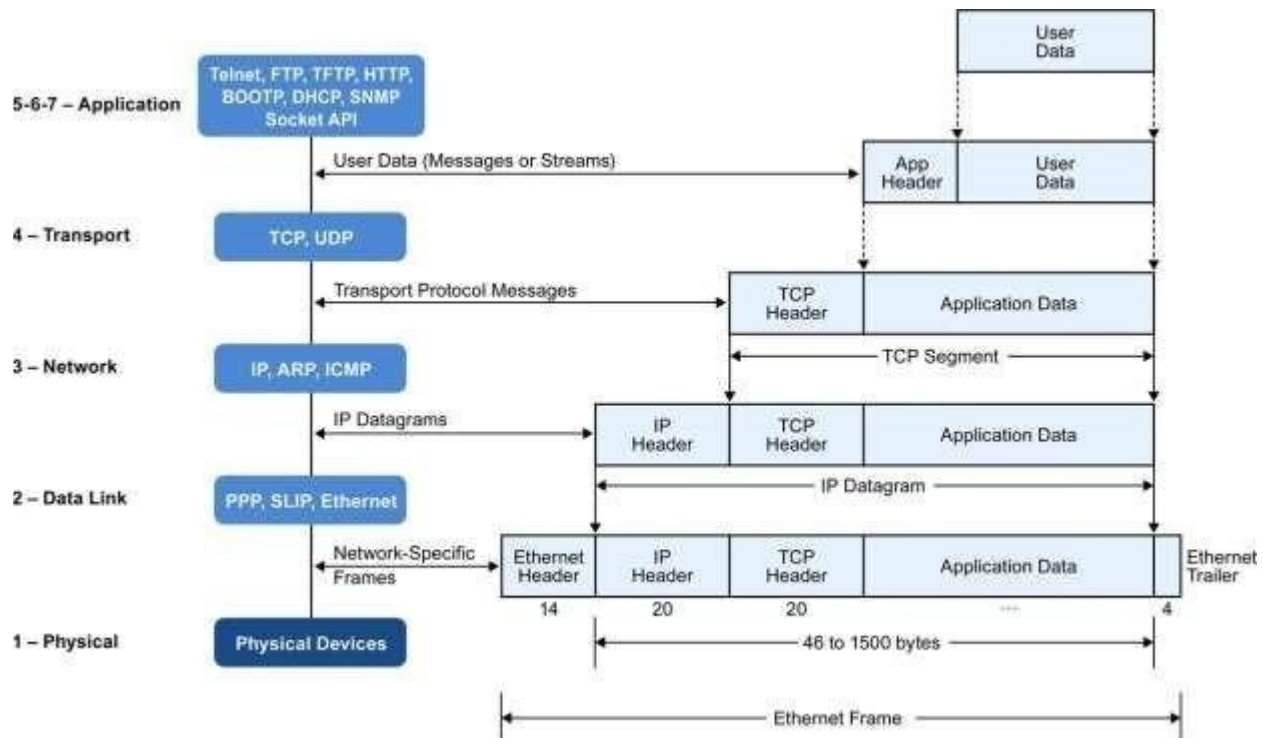


Figure 2: Encapsulation of Data in the TCP/IP Network Stack

In the CSC 4190 Introduction to Computer Networking (one of the prerequisite courses), TCP/IP network stack is introduced and studied. This background section briefly explains the concept of TCP/IP network stack to help you better understand the experiments. TCP/IP is the most commonly used network model for Internet services. Because its most important protocols, the Transmission Control Protocol (TCP) and the Internet Protocol (IP) were the first networking protocols defined in this standard, it is named as TCP/IP. However, it contains multiple layers including application layer, transport layer, network layer, and data link layer.

- *Application Layer:* The application layer includes the protocols used by most applications for providing user services. Examples of application layer protocols are Hypertext

Transfer Protocol (HTTP), Secure Shell (SSH), File Transfer Protocol (FTP), and Simple Mail Transfer Protocol (SMTP).

- *Transport Layer*: The transport layer establishes process-to-process connectivity, and it provides end-to-end services that are independent of underlying user data. To implement the process-to-process communication, the protocol introduces a concept of port. The examples of transport layer protocols are Transport Control Protocol (TCP) and User Datagram Protocol (UDP). The TCP provides flow- control, connection establishment, and reliable transmission of data, while the UDP is a connectionless transmission model.
- *Internet Layer*: The Internet layer is responsible for sending packets to across networks. It has two functions: 1) Host identification by using IP addressing system (IPv4 and IPv6); and 2) packets routing from source to destination. The examples of Internet layer protocols are Internet Protocol (IP), Internet Control Message Protocol (ICMP), and Address Resolution Protocol (ARP).
- *Link Layer*: The link layer defines the networking methods within the scope of the local network link. It is used to move the packets between two hosts on the same link. An common example of link layer protocols is Ethernet.

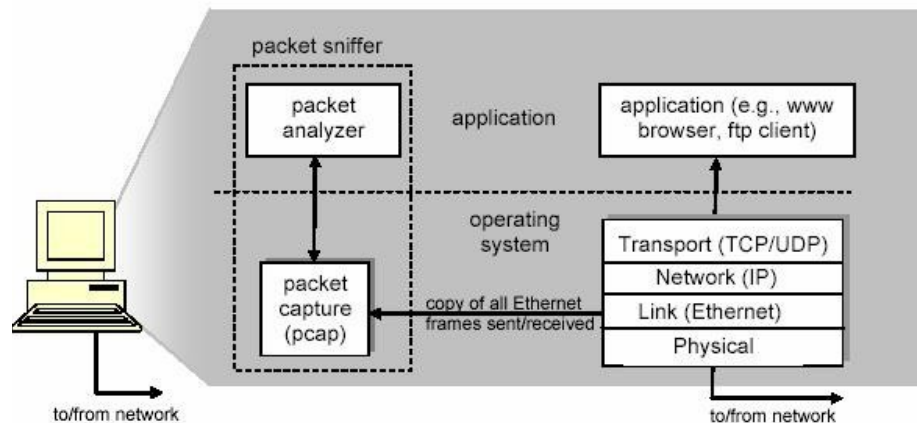
Packet Sniffer

Packet sniffer is a basic tool for observing network packet exchanges in a computer. As the name suggests, a packet sniffer captures (“sniffs”) packets being sent/received from/by your computer; it will also typically store and/or display the contents of the various protocol fields in these captured packets. A packet sniffer itself is passive. It observes messages being sent and received by applications and protocols running on your computer, but never sends packets itself.

Figure 3 shows the structure of a packet sniffer. At the right of **Figure 3** are the protocols (in this case, Internet protocols) and applications (such as a web browser or ftp client) that normally run on your computer. The packet sniffer, shown within the dashed rectangle in **Figure 3** is an addition to the usual software in your computer, and consists of two parts. The packet capture library receives a copy of every link-layer frame that is sent from or received by your computer. Messages exchanged by higher layer protocols such as HTTP, FTP, TCP, UDP, DNS, or IP all are eventually encapsulated in link-layer frames that are transmitted over physical media such as an Ethernet cable. In Figure 1, the assumed physical media is an Ethernet, and so all upper-layer protocols are eventually encapsulated within an Ethernet frame. Capturing all link-layer frames thus gives you access to all messages sent/received from/by all protocols and applications

executing in your computer.

The second component of a packet sniffer is the packet analyzer, which displays the contents of all fields within a protocol message. In order to do so, the packet analyzer



Packet Sniffer Structure

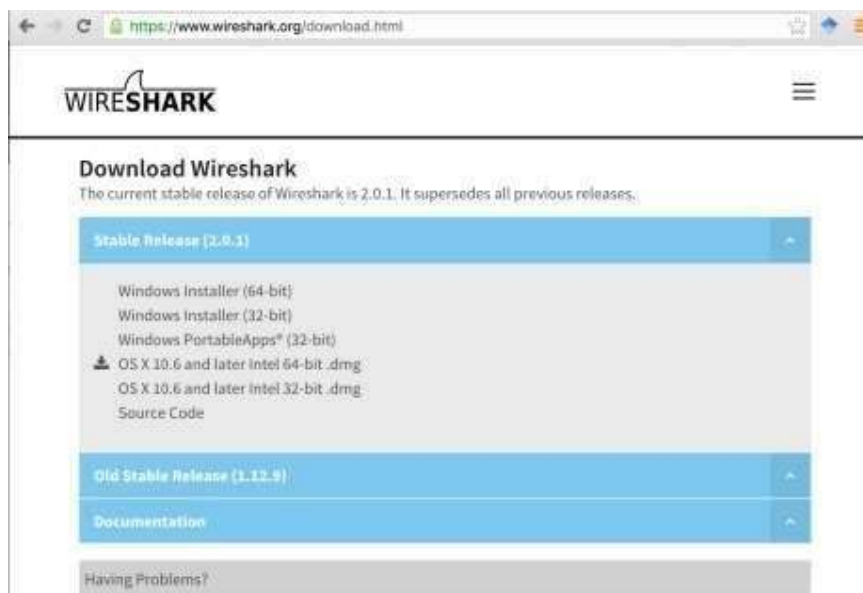
must “understand” the structure of all messages exchanged by protocols. For example, suppose we are interested in displaying the various fields in messages exchanged by the HTTP protocol in **Figure 3**. The packet analyzer understands the format of Ethernet frames, and so can identify the IP datagram within an Ethernet frame. It also understands the IP datagram format, so that it can extract the TCP segment within the IP datagram. Finally, it understands the TCP segment structure, so it can extract the HTTP message contained in the TCP segment. Finally, it understands the HTTP protocol and so, for example, knows that the first bytes of an HTTP message will contain the string “GET,” “POST,” or “HEAD”.

We will be using the Wireshark packet sniffer [<http://www.wireshark.org/>] for these labs, allowing us to display the contents of messages being sent/received from/by protocols at different levels of the protocol stack. (Technically speaking, Wireshark is a packet analyzer that uses a packet capture library in your computer). Wireshark is a free network protocol analyzer that runs on Windows, Linux/Unix, and Mac computers.

Getting Wireshark

The Kai Linux has Wireshark installed. You can just launch the Kali Linux VM and open Wireshark there. Wireshark can also be downloaded from here:

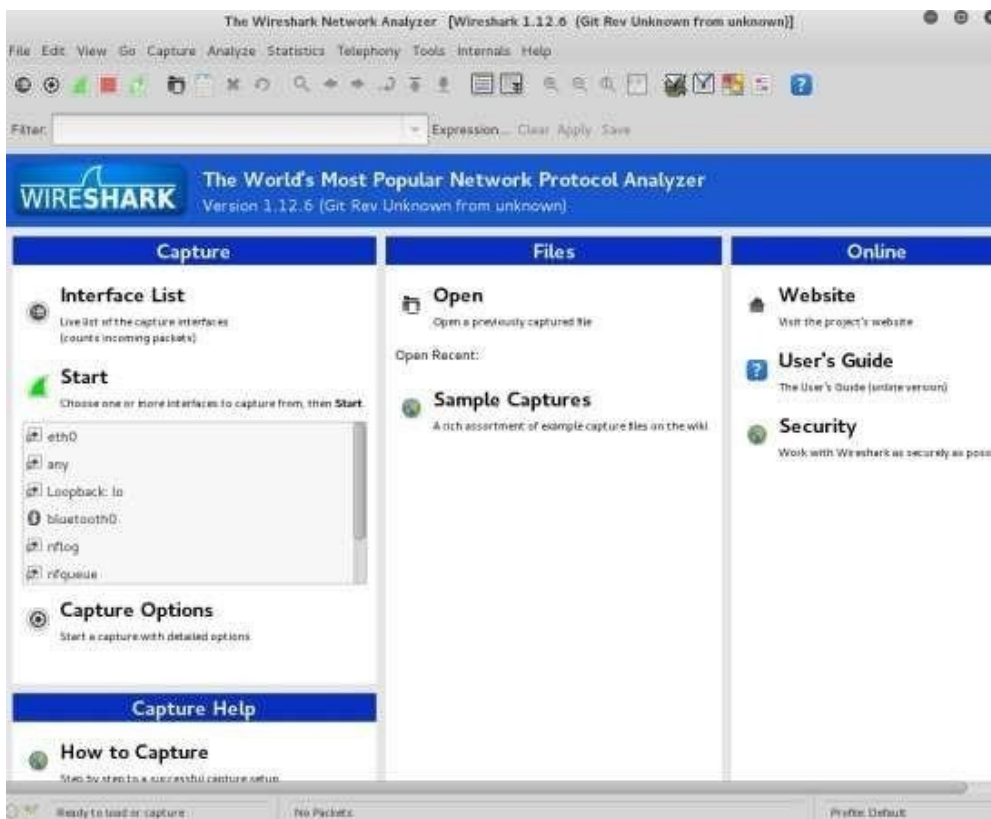
<https://www.wireshark.org/download.html>



(Download Page of Wireshark)

Starting Wireshark:

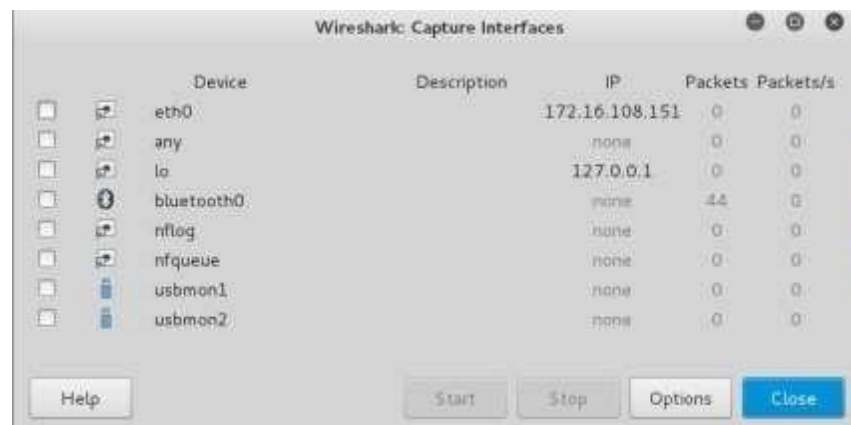
When you run the Wireshark program, the Wireshark graphic user interface will be shown as **Figure 5**. Currently, the program is not capturing the packets.



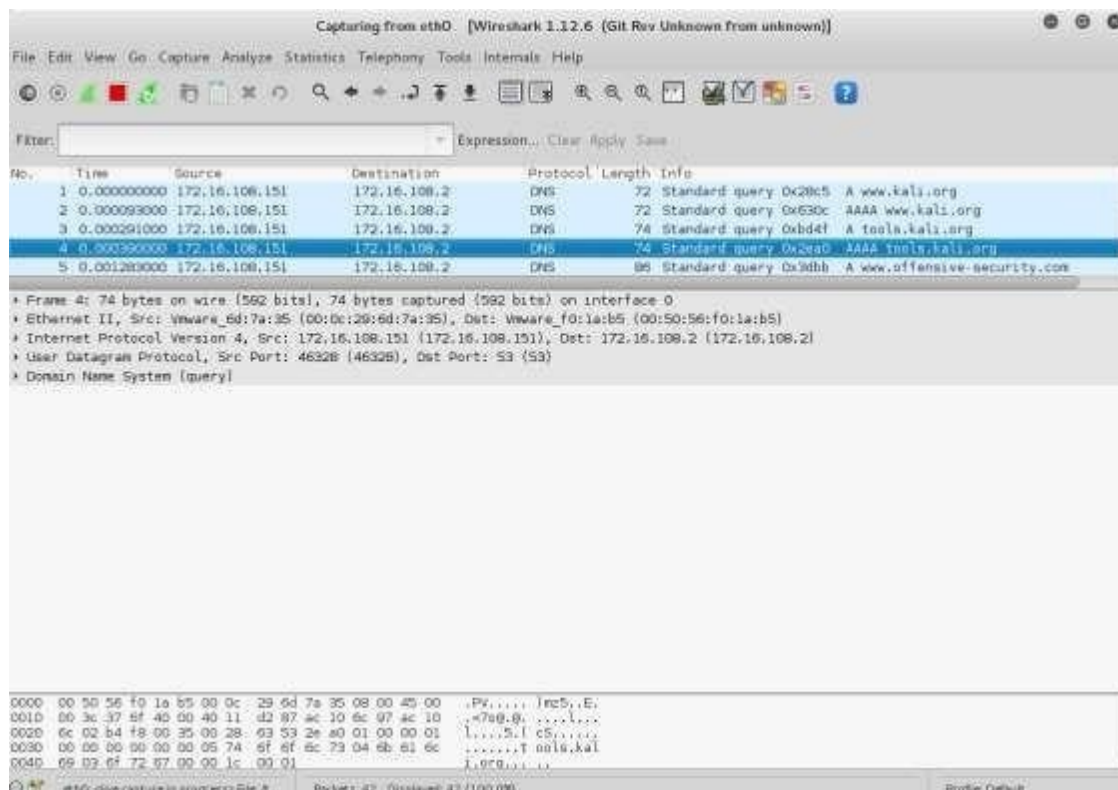
Initial Graphic User Interface of Wireshark

Then, you need to choose an interface. If you are running the Wireshark on your laptop, you need to select WiFi interface. If you are at a desktop, you need to select the Ethernet interface being used. Note that there could be multiple interfaces. In general, you can select any interface but that does not mean that traffic will flow through that interface. The network interfaces (i.e., the physical connections) that your computer has to the network are shown. The attached **Figure 6** was taken from my computer.

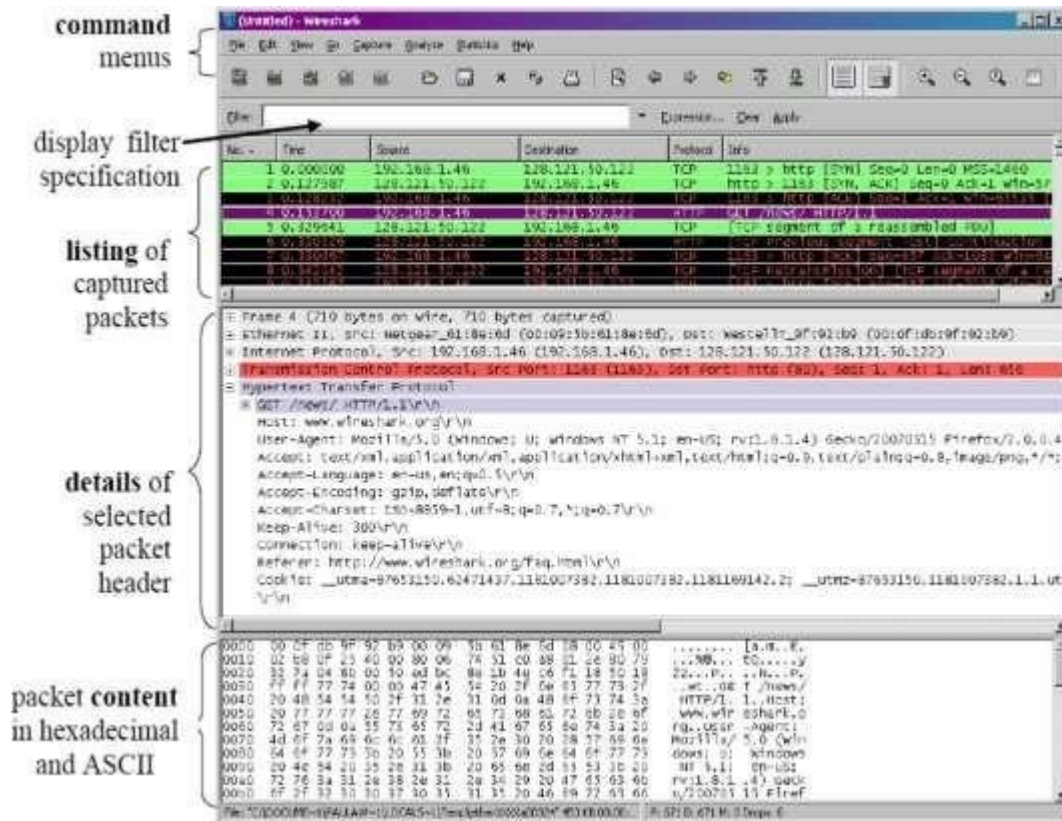
After you select the interface, you can click start to capture the packets as shown in **Figure 7**.



Capture Interfaces in Wireshark



Capturing Packets in Wireshark



(Wireshark Graphical User Interface on Microsoft Windows)

The Wireshark interface has five major components:

The **command menus** are standard pulldown menus located at the top of the window. Of interest to us now is the File and Capture menus. The File menu allows you to save captured packet data or open a file containing previously captured packet data, and exit the Wireshark application. The Capture menu allows you to begin packet capture.

The **packet-listing window** displays a one-line summary for each packet captured, including the packet number (assigned by Wireshark; this is not a packet number contained in any protocol's header), the time at which the packet was captured, the packet's source and destination addresses, the protocol type, and protocol-specific information contained in the packet. The packet listing can be sorted according to any of these categories by clicking on a column name. The protocol type field lists the highest- level protocol that sent or received this packet, i.e., the protocol that is the source or ultimate sink for this packet.

The **packet-header details window** provides details about the packet selected (highlighted) in the packet-listing window. (To select a packet in the packet-listing window, place the cursor over the packet's one- line summary in the packet-listing window and click with the left mouse button.). These details include information about the Ethernet frame and IP datagram that

contains this packet. The amount of Ethernet and IP-layer detail displayed can be expanded or minimized by clicking on the right- pointing or down- pointing arrowhead to the left of the Ethernet frame or IP datagram line in the packet details window. If the packet has been carried over TCP or UDP, TCP or UDP details will also be displayed, which can similarly be expanded or minimized. Finally, details about the highest-level protocol that sent or received this packet are also provided.

The **packet-contents window** displays the entire contents of the captured frame, in both ASCII and hexadecimal format.

Towards the top of the Wireshark graphical user interface, is the **packet display filter field**, into which a protocol name or other information can be entered in order to filter the information displayed in the packet-listing window (and hence the packet-header and packet-contents windows). In the example below, we'll use the packet-display filter field to have Wireshark hide (not display) packets except those that correspond to HTTP messages.

Capturing Packets

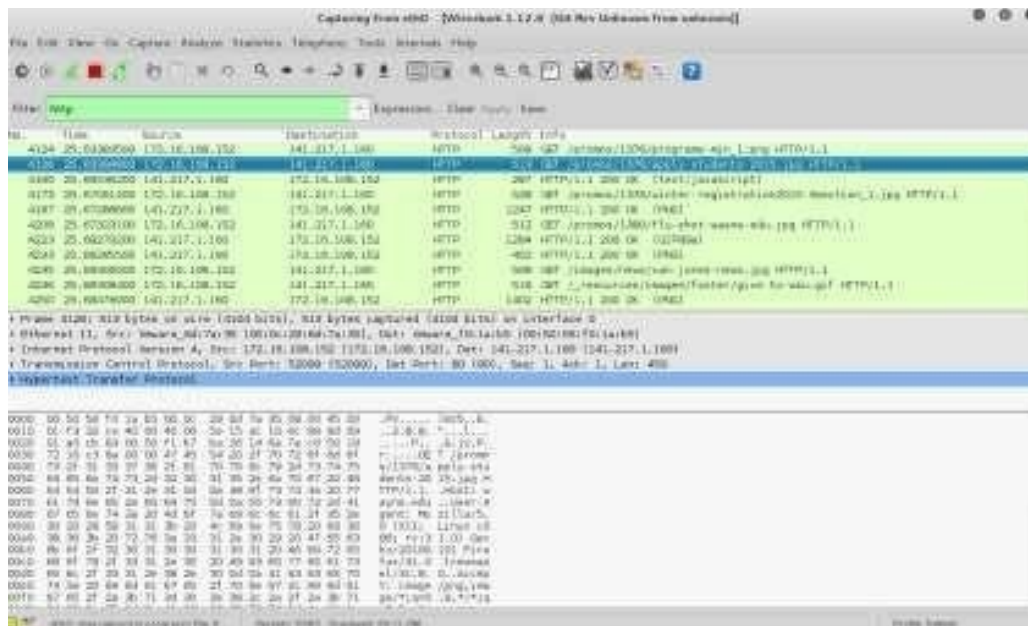
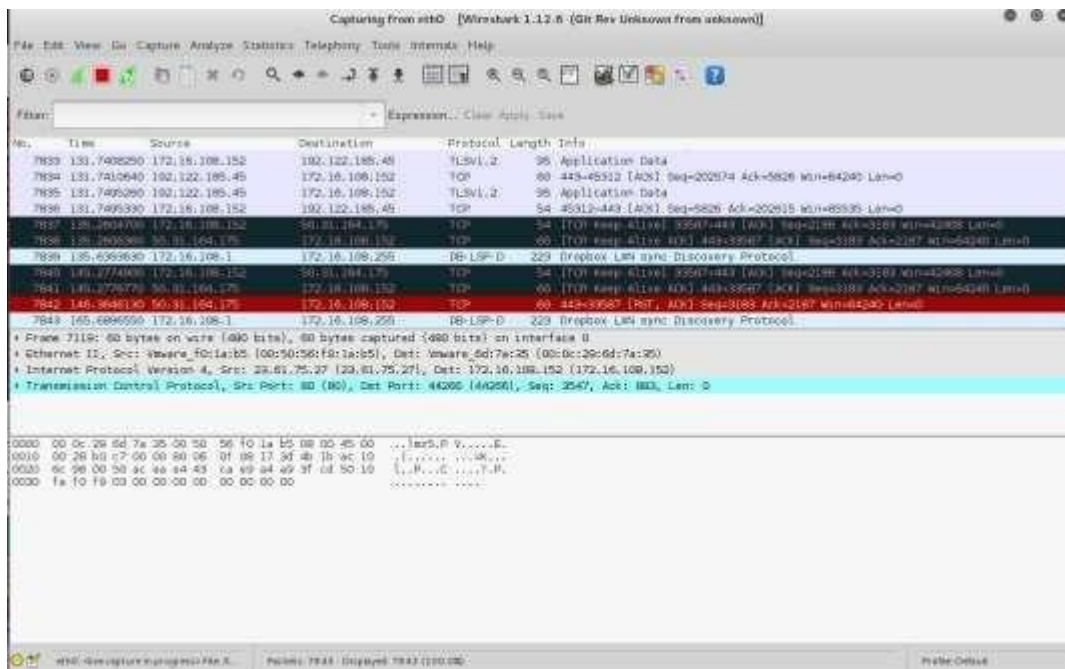
After downloading and installing Wireshark, you can launch it and click the name of an interface under Interface List to start capturing packets on that interface. For example, if you want to capture traffic on the wireless network, click your wireless interface.

Test Run

Do the following steps:

1. Start up the Wireshark program (select an interface and press start to capture packets).
2. Start up your favorite browser (ceweasel in Kali Linux).
3. In your browser, go to Wayne State homepage by typing www.wayne.edu.
4. After your browser has displayed the <http://www.wayne.edu> page, stop Wireshark packet capture by selecting stop in the Wireshark capture window. This will cause the Wireshark capture window to disappear and the main Wireshark window to display all

packets captured since you began packet capture see image below:

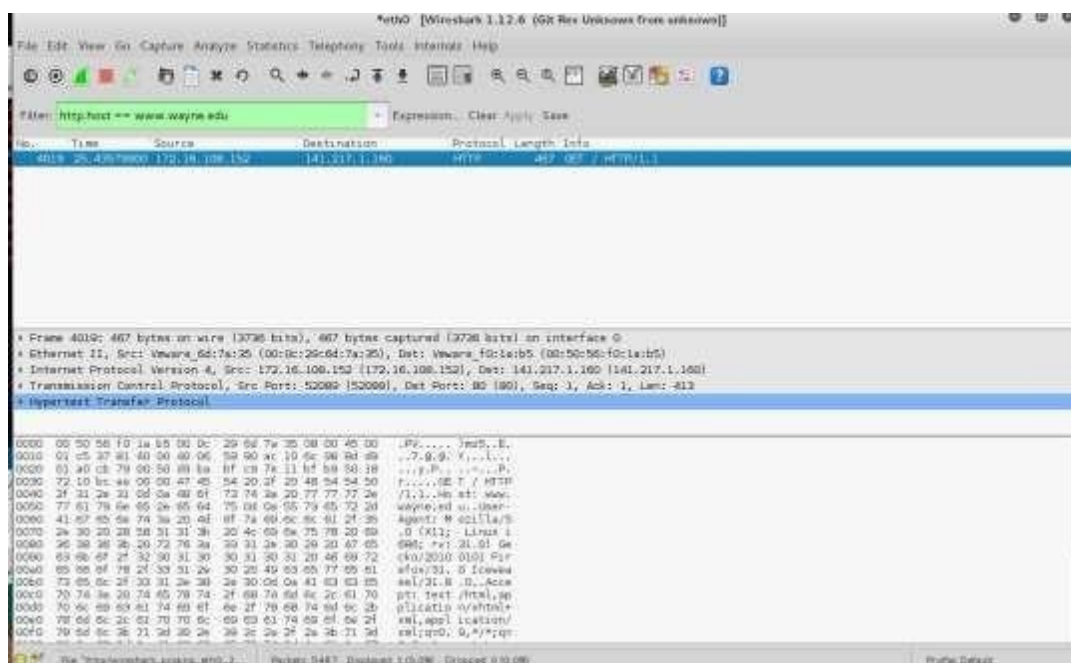


5. Color Coding: You'll probably see packets highlighted in green, blue, and black. Wireshark uses colors to help you identify the types of traffic at a glance. By default, green is TCP traffic, dark blue is DNS traffic, light blue is UDP traffic, and black identifies TCP packets with problems — for example, they could have been delivered out-of-order.
6. You now have live packet data that contains all protocol messages exchanged between your computer and other network entities! However, as you will notice the HTTP messages are not clearly shown because there are many other packets included in the packet capture. Even though the only action you took was to open your browser, there are many other programs in your computer that communicate via the network in the

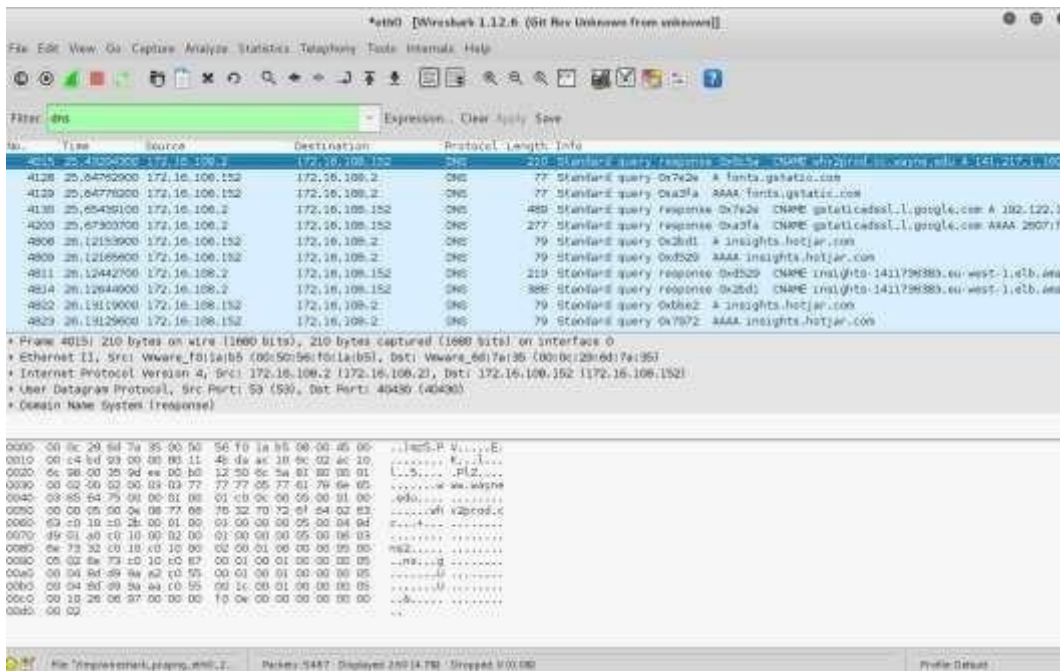
background. To filter the connections to the ones we want to focus on, we have to use the filtering functionality of Wireshark by typing “http” in the filtering field as shown below:

Notice that we now view only the packets that are of protocol HTTP. However, we also still do not have the exact communication we want to focus on because using HTTP as a filter is not descriptive enough to allow us to find our connection to <http://www.wayne.edu>. We need to be more precise if we want to capture the correct set of packets.

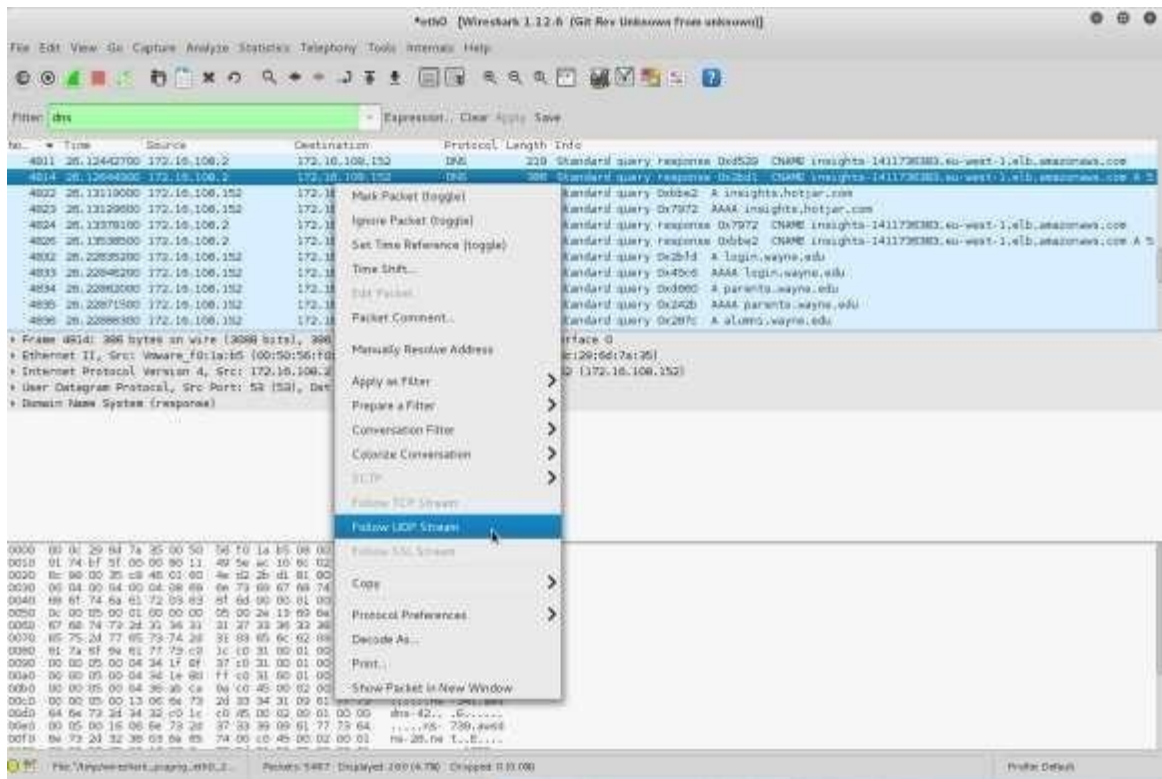
7. To further filter packets in Wireshark, we need to use a more precise filter. By setting the `http.host www.wayne.edu`, we are restricting the view to packets that have as an http host the www.wayne.edu website. Notice that we need two equal signs to perform the match not just one. See the screenshot below:

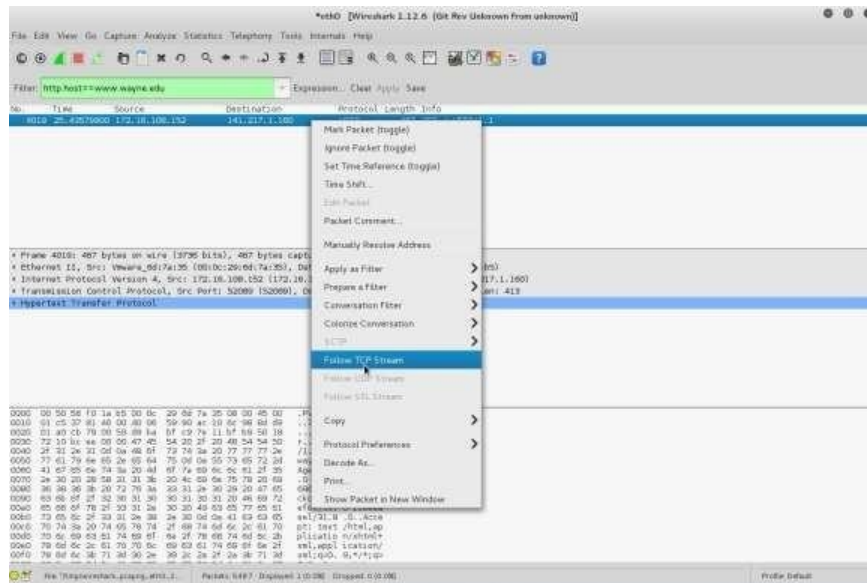


8. Now, we can try another protocol. Let's use Domain Name System (DNS) protocol as an example here.



conversations (also called network flows), select one of the packets and press the right mouse button (if you are on a Mac use the command button and click), you should see something similar to the screen below:





10. If we close this window and change the filter back to “http.hos ww.wayne.edu” and then follow a packet from the list of packets that match that filter, we should get the something similar to the following screens. Note that we click on **Follow TCP Stream** this time.



Result:

Installation of Wire shark, tcpdump and observe data transferred in client-server communication using UDP/TCP and identify the UDP/TCP datagram.

Ex. No : 6

Check message integrity and confidentiality using SSL

Date:

Aim

SSL Session in Details

Handshaking - Ciphersuit Negotiation

Client sends a plaintext Client_Hello message and suggests some cryptographic parameters (collectively called ciphersuit) to be used for their communication session. The Client_Hello message also contains a 32-byte random number denoted as client_random. For example,

Client_Hello:

Protocol Version: TLSv1 if you can, else SSLv3.

Key Exchange: RSA if you can, else Diffie-Hellman.

Secret Key Cipher Method: 3DES if you can, else DES.

Message Digest: SHA-1 if you can, else MD5.

Data Compression Method: PKZip if you can, else gzip.

Client Random Number: 32 bytes.

The stronger method (in terms of security) shall precede the weaker one, e.g. RSA (1024-bit) precedes DH, 3DES precedes DES, SHA-1 (160-bit) precedes MD5 (128-bit).

Server responds with a plaintext Server_Hello to state the ciphersuit of choice (server decides on the ciphersuit). The message also contains a 32-byte random number denoted as server_random. For example,

Server_Hello:

Protocol Version: TLSv1.

Key Exchange: RSA.

Secret Key Cipher Method: DES.

Message Digest: SHA-1.

Data Compression Method: PKZip.

Server Random Number: 32 bytes.

Handshaking - Key Exchange

The server sends its digital certificate to the client, which is supposedly signed by a root CA. The client uses the root CA's public key to verify the server's certificate (trusted root-CAs' public key are pre-installed inside the browser). It then retrieves the server's public key from the server's certificate. (If the server's certificate is signed by a sub-CA, the client has to build a digital certificate chain, leading to a trusted root CA, to verify the server's certificate.)

The server can optionally request for the client's certificate to authenticate the client. In practice, server usually does not authenticate the client. This is because:

- Server authenticates client by checking the credit card in an e-commerce transaction.
- Most clients do not have a digital certificate.
- Authentication via digital certificate takes time and the server may lose an impatient client.

The next step is to establish the Session Key:

1. The client generates a 48-byte (384-bit) random number called `pre_master_secret`, encrypts it using the verified server's public key and sends it to the server.
2. Server decrypts the `pre_master_secret` using its own private key. Eavesdroppers cannot decrypt the `pre_master_secret`, as they do not possess the server's private key.
3. Client and server then independently and simultaneously create the session key, based on the `pre_master_secret`, `client_random` and `server_random`. Notice that both the server and client contribute to the session key, through the inclusion of the random number exchange in the hello messages. Eavesdroppers can intercept `client_random` and `server_random` as they are sent in plaintext, but cannot decrypt the `pre_master_secret`.
4. In a SSL/TLS session, the session key consists of 6 secret keys (to thwart crypto-analysis). 3 secret keys are used for client-to-server messages, and the other 3 secret keys are used for server-to-client messages. Among the 3 secret keys, one is used for encryption (e.g., DES secret key), one is used for message integrity (e.g., HMAC) and one is used for cipher initialization. (Cipher initialization uses a random plaintext called Initial Vector (IV) to prime the cipher pump.)
5. Client and server use the `pre_master_secret` (48-byte random number created by the client and exchange securely), `client_random`, `server_random`, and a pseudo-random function (PRF) to generate a `master_secret`. They can use the `master_secret`, `client_random`, `server_random`, and the pseudo-random function (PRF) to generate all the 6 shared secret keys. Once the secret keys are generated, the `pre_master_secret` is no longer needed and should be deleted.
6. From this point onwards, all the exchanges are encrypted using the session key.
7. The client sends Finished handshake message using their newly created session key. Server responds with a Finished handshake message.

Message Exchange

Client and server can use the agreed-upon session key (consists of 6 secret keys) for secure exchange of messages.

Sending messages:

1. The sender compresses the message using the agreed-upon compression method (e.g., PKZip, gzip).
2. The sender hashes the compressed data and the secret HMAC key to make an HMAC, to assure message integrity.
3. The sender encrypts the compressed data and HMAC using encryption/decryption secret key, to assure message confidentiality.

Retrieve messages:

1. The receiver decrypts the ciphertext using the encryption/decryption secret key to retrieve the compressed data and HMAC.

2. The receiver hashes the compressed data to independently produce the HMAC. It then verifies the generated HMAC with the HMAC contained in the message to assure message integrity.
3. The receiver un-compresses the data using the agreed-upon compression method to recover the plaintext.

The following diagram shows the sequence of the SSL messages for a typical client/server session.

A SSL Session Trace

We could use OpenSSL's `s_client` (with debug option) to produce a SSL session trace.

> openssl s_client ?

(Display the available options)

The following command turns on the debug option and forces the protocol to be TLSv1:

> openssl s_client -connect localhost:443 -CAfile ca.crt -debug -tls1

Loading 'screen' into random state - done

CONNECTED(00000760)

write to 00988EB0 [009952C8] (102 bytes => 102 (0x66))

```
0000 - 16 03 01 00 61 01 00 00-5d 03 01 40 44 35 27 5c  ....a...].@D5\
0010 - 5a e8 74 26 e9 49 37 e2-06 3b 1c 6d 77 37 d1 ae  Z.t&.I7...;mw7..
0020 - 44 07 86 47 98 fa 84 1a-8d f4 72 00 00 36 00 39  D..G....r..6.9
0030 - 00 38 00 35 00 16 00 13-00 0a 00 33 00 32 00 2f  .8.5.....3.2./
0040 - 00 07 00 66 00 05 00 04-00 63 00 62 00 61 00 15  ...f....c.b.a..
0050 - 00 12 00 09 00 65 00 64-00 60 00 14 00 11 00 08  ....e.d.`.....
0060 - 00 06 00 03 01                .....
0066 - <SPACES/NULS>
```

read from 00988EB0 [00990AB8] (5 bytes => 5 (0x5))

```
0000 - 16 03 01 00 2a.....*
```

read from 00988EB0 [00990ABD] (42 bytes => 42 (0x2A))

```
0000 - 02 00 00 26 03 01 40 44-35 27 cc ef 2b 51 e1 b0  ...&..@D5'..+Q..
0010 - 44 1f ef c4 83 72 df 37-4f 9b 2b dd 11 50 13 87  D....r.7O.+..P..
0020 - 91 0a a2 d2 28 b9 00 00-16                ....(....
002a - <SPACES/NULS>
```

read from 00988EB0 [00990AB8] (5 bytes => 5 (0x5))

```
0000 - 16 03 01 02 05                .....
```

read from 00988EB0 [00990ABD] (517 bytes => 517 (0x205))

```
0000 - 0b 00 02 01 00 01 fe 00-01 fb 30 82 01 f7 30 82  ....0. .0.
0010 - 01 60 02 01 01 30 0d 06-09 2a 86 48 86 f7 0d 01  .`...0...*.H....
```



```

0020 - 01 04 05 00 30 4d 31 0b-30 09 06 03 55 04 06 13 ....0M1.0...U...
0030 - 02 55 53 31 10 30 0e 06-03 55 04 0b 13 07 74 65 .US1.0...U...te
0040 - 73 74 31 30 31 31 0c 30-0a 06 03 55 04 03 13 03 st1011.0...U....
0050 - 63 68 63 31 1e 30 1c 06-09 2a 86 48 86 f7 0d 01 chc1.0...*.H....
0060 - 09 01 16 0f 63 68 63 40-74 65 73 74 31 30 31 2e.....chc@test101.
0070 - 63 6f 6d 30 1e 17 0d 30-34 30 32 32 36 30 36 35 com0..040226065
0080 - 36 35 34 5a 17 0d 30 35-30 32 32 35 30 36 35 36 654Z 0502250656
0090 - 35 34 5a 30 3b 31 0b 30-09 06 03 55 04 06 13 02 54Z0;1.0...U....
00a0 - 55 53 31 0c 30 0a 06 03-55 04 03 13 03 63 68 63 US1.0...U...chc
00b0 - 31 1e 30 1c 06 09 2a 86-48 86 f7 0d 01 09 01 16 1.0...*.H.....
00c0 - 0f 63 68 63 40 74 65 73-74 31 30 31 2e 63 6f 6d .chc@test101.com
00d0 - 30 81 9f 30 0d 06 09 2a-86 48 86 f7 0d 01 01 01 0..0...*.H.....
00e0 - 05 00 03 81 8d 00 30 81-89 02 81 81 00 cd e4 9e .....0.....
00f0 - 7c b6 d2 34 4e d3 53 46-25 c7 53 88 25 60 e6 46 |.4N.SF%.S.%`.F
0100 - db 64 3a 73 61 92 ac 23-92 cd 2c 94 a9 8f c6 7f .d:sa.#.,.....
0110 - 47 73 c0 d9 8d 34 b7 2c-dd c9 86 bd 82 6f ce ac Gs...4.,.....o..
0120 - d8 e2 ba 0f e5 f5 3a 67-2c 89 1a 1b 03 eb 21 85 .....:g,....!.
0130 - 28 e3 29 98 84 ed 46 75-82 fa 0f 30 a3 a9 a5 71 (.)...Fu...0..q
0140 - 46 4c d6 0d 17 c4 19 fd-44 fb e2 18 46 a6 9d ab FL.....D...F...
0150 - 91 de 6b a1 7f fe 30 06-28 5d d8 d3 29 00 c3 1d .k...0.(.)...
0160 - 4c 13 00 61 8f f3 85 51-f5 68 d8 69 25 02 03 01 L..a...Q.h.i%...
0170 - 00 01 30 0d 06 09 2a 86-48 86 f7 0d 01 01 04 05 ..0...*.H.....
0180 - 00 03 81 81 00 29 fd bf-5a ed 70 8f 53 a4 e9 14 .....).Z.p.S...
0190 - 4c 5e ba 84 c6 54 1b f2-c0 3c c4 30 0f 7f 12 80 L^...T...<.0....
01a0 - 4e 01 b7 fd 39 50 f1 41-0d d8 aa 77 d9 87 25 1a N...9P.A...w..%.
01b0 - 1e e2 97 88 4f 53 75 c8-70 22 6a 01 61 0f 51 3e.....OSu.p"j.a.Q>
01c0 - 13 19 9c 64 f2 76 14 e8-85 25 23 a2 11 c4 8c f8 ...d.v...%#.....
01d0 - 23 2c d1 c3 d3 71 3a e6-71 54 10 07 dc 72 ff ee #,...q.:qT...r..
01e0 - e8 3e cf 8e 77 73 e9 9f-f5 9a 90 60 4d a0 aa 03 >..ws.....`M...
01f0 - 32 1f 11 6f 2e 9a 5f 3c-77 05 22 0c 81 bf 29 96 2..o.._5 (0x5))
0000 - 16 03 01 01 8d .....

```

read from 00988EB0 [00990ABD] (397 bytes => 397 (0x18D))

```

0000 - 0c 00 01 89 00 80 e6 96-9d 3d 49 5b e3 2c 7c f1.....=I[.,|.
0010 - 80 c3 bd d4 79 8e 91 b7-81 82 51 bb 05 5e 2a 20 ....y.....Q..^*
0020 - 64 90 4a 79 a7 70 fa 15-a2 59 cb d5 23 a6 a6 ef d.Jy.p...Y.#...
0030 - 09 c4 30 48 d5 a2 2f 97-1f 3c 20 12 9b 48 00 0e ..0H../.<..H..
0040 - 6e dd 06 1c bc 05 3e 37-1d 79 4e 53 27 df 61 1e n...>7.yNS'.a.
0050 - bb be 1b ac 9b 5c 60 44-cf 02 3d 76 e0 5e ea 9b .....`D..=v.^..
0060 - ad 99 1b 13 a6 3c 97 4e-9e f1 83 9e b5 db 12 51 .....<.N.....Q
0070 - 36 f7 26 2e 56 a8 87 15-38 df d8 23 c6 50 50 85 6.&.V...8.#.PP.
0080 - e2 1f 0d d5 c8 6b 00 01-02 00 80 11 3f 5f fa e4 .....k.....?_..
0090 - 79 9a 0b d9 e0 67 37 c4-2a 88 22 b0 95 b7 a7 be y....g7.*.".....
00a0 - 93 79 9d 51 ae 31 47 99-df 47 dd 80 5e 3d 2a 4a .y.Q.1G..G..^=*J
00b0 - 29 8b fd c1 63 5e 48 e8-e3 fd ac 95 1b 3a 5f 75 )...c^H.....:u
00c0 - 98 2d 3c 9c ba 68 18 7b-be 38 2c 69 3d 41 b7 c3 .-<..h.{.8,i=A..
00d0 - 08 a1 da b0 a8 a4 fe 9a-d6 1e 56 ff 4c 8c 6e 6b.....V.L.nk
00e0 - 18 f1 ec 9d 22 a9 90 27-c1 c6 2c 0e bd 0e 13 d4 ....".'.,.....
00f0 - fd b2 c9 8f 6f bb 8e 06-e0 b5 1f f7 87 03 5f a8 ....o....._.
0100 - 12 4f bb ce ba f1 76 fb-80 08 37 00 80 30 99 ad .O....v...7..0..

```

```

0110 - 9b fc 3a 14 6b a8 2c c5-fe 7b bd 1c 92 ec 19 a6 ...k,...{.....
0120 - 75 2d 69 4e f4 9f 74 60-5d d4 3e 06 97 38 bc b5 u-iN..t']>..8..
0130 - 0e 3c 1f f2 99 e6 55 4a-36 42 a8 f2 b7 32 2a 1e .<...UJ6B. .2*.
0140 - a3 87 b3 f3 79 43 28 d1-7a 0d db 7c 11 26 f3 68 ....yC(.z..|.&.h
0150 - b1 73 b6 78 4b f3 22 20-e4 f7 27 08 ab 74 92 92 .s.xK." ..'!t..
0160 - 79 26 61 40 1e e9 90 11-e8 b1 cf 99 d9 9f c7 68 y&a@.....h
0170 - 48 e8 f2 a5 d5 d7 0e e1-88 9a bd 0f 40 85 af 2d H.....@.-
0180 - da 76 3a 10 6e b9 38 4d-37 9c 41 c8 9f .v:..n.8M7.A..

```

```

read from 00988EB0 [00990AB8] (5 bytes => 5 (0x5))
0000 - 16 03 01 00 04 .....

```

```

read from 00988EB0 [00990ABD] (4 bytes => 4 (0x4))
0000 - 0e .
0004 - <SPACES/NULS>

```

```

write to 00988EB0 [00999BE0] (139 bytes => 139 (0x8B))
0000 - 16 03 01 00 86 10 00 00-82 00 80 63 c2 3c 69 26 .....c...dU.....]n..
0030 - 05 f1 db 44 f3 13 a8 24-3a 76 0e 3e 1a 6e 55 0c ...D. $.v.>.nU.
0040 - 31 9b 04 99 30 ff 8f d2-8d 8e 0d b1 67 ac 43 ee 1...0.....g.C.
0050 - b2 3f d3 c7 c5 33 81 e1-3f d2 47 6f 5d 8a fb 4c ?...3..?.Go]..L
0060 - 62 c7 23 b3 f7 ad 3c a9-0c 87 4a 08 07 55 ba 06 b.#...<...J..U..
0070 - 34 18 0c 5f d9 35 f0 2b-90 9a 9d 6b 87 62 41 0f 4.._5.+..k.bA.
0080 - b3 47 74 5f 5b b8 59 5a-b2 21 dd .Gt_[.YZ.!.

```

```

write to 00988EB0 [00999BE0] (6 bytes => 6 (0x6))
0000 - 14 03 01 00 01 01 .....

```

```

write to 00988EB0 [00999BE0] (45 bytes => 45 (0x2D))
0000 - 16 03 01 00 28 0f 31 83-e0 f8 91 fa 33 98 68 46 ....(1....3.hF
0010 - c0 60 83 66 28 fe d3 a5-00 f0 98 d5 df 22 72 2d .`.f(....."r-
0020 - e4 40 9b 96 3b 4c f9 02-13 a7 e7 77 74 .@..;L....wt

```

```

read from 00988EB0 [00990AB8] (5 bytes => 5 (0x5))
0000 - 14 03 01 00 01 .....

```

```

read from 00988EB0 [00990ABD] (1 bytes => 1 (0x1))
0000 - 01 .

```

```

read from 00988EB0 [00990AB8] (5 bytes => 5 (0x5))
0000 - 16 03 01 00 28.....(

```

```

read from 00988EB0 [00990ABD] (40 bytes => 40 (0x28))
0000 - d4 0b a6 b7 e8 91 09 1e-e4 1e fc 44 5f 80 cc a1 .....D_...
0010 - 5d 51 55 3e 62 e8 0f 78-07 f6 2f cd f9 bc 49 8d ]QU>b..x../. .I.
0020 - 56 5b e8 b2 09 2c 18 52- V[...R
---

```

Certificate chain

0 s:/C=US/CN=chc/emailAddress=chc@test101.com

i:/C=US/OU=test101/CN=chc/emailAddress=chc@test101.com

Server certificate

-----BEGIN CERTIFICATE-----

MIIB9zCCAWACAQEwDQYJKoZIhvcNAQEEBQAwwTTELMakGA1UEBhMCVVMxEDAObgNV

BA5TB3Rlc3QxMDExDDAKBgNVBAMTA2NoYzEeMBwGCSqGSIb3DQEJARYPY2hjQHRlc3QxMDEuY29tMB4XDTA0MDIyNjA2NTY1NFoXDTA1MDIyNTA2NTY1NFowOzELMAkGA1UEBhMCVVMxDDAKBgNVBAMTA2NoYzEeMBwGCSqGSIb3DQEJARYPY2hjQHRlc3Qx

MDEuY29tMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDN5J58ttI0TtNTRiXH
U4glYOZG22Q6c2GSrCOSzSyUqY/Gf0dzwNmNNLcs3cmGvYJvzqzY4roP5fU6ZyyJ
GhsD6yGFKOMpmITtRnWC+g8wo6mlcUZM1g0XxBn9RPviGEamnauR3muhf/4wBihd
2NMpAMMdTBMAYY/zhVH1aNhpJQIDAQABMA0GCSqGSIb3DQEBAQUAA4GBACn9v1rt
cI9TpOkUTF66hMZUG/LAPMQwD38SgE4Bt/05UPFBDdiqd9mHJRoe4peIT1N1yHAI
agFhD1E+ExmcZPJ2FOiFJSOIecSM+CMs0cPTcTrmcVQQB9xy/+7oPs+Od3Ppn/Wa
kGBNoKoDMh8Rby6aXzx3BSIMgb8plq3LOxiu

-----END CERTIFICATE-----

subject=/C=US/CN=chc/emailAddress=chc@test101.com

issuer=/C=US/OU=test101/CN=chc/emailAddress=chc@test101.com

No client certificate CA names sent

SSL handshake has read 1031 bytes and written 292 bytes

New, TLSv1/SSLv3, Cipher is EDH-RSA-DES-CBC3-SHA

Server public key is 1024 bit

SSL-Session:

Protocol : TLSv1

Cipher : EDH-RSA-DES-CBC3-SHA

Session-ID:

Session-ID-ctx:

Master-Key: 57FDDAF85C7D287F9F9A070E8784A29C75E788DA2757699B

20F3CA50E7EE01A66182A71753B78DA218916136D50861AE

Key-Arg : None

Start Time: 1078211879

Timeout : 7200 (sec)

Verify return code: 0 (ok)

GET /test.html HTTP/1.0

write to 00988EB0 [009952C8] (82 bytes => 82 (0x52))

0000 - 17 03 01 00 18 74 fa 45-35 2d b1 24 59 cf ad 96t.E5-.\$Y...

0010 - 34 30 01 7d be 8e 70 f9-41 62 11 f1 36 17 03 01 40.}.p.Ab..6...

0020 - 00 30 56 61 ba 2d d3 58-5d e6 6a 83 78 07 87 7a .0Va.-X].j.x..z
0030 - db b2 a7 40 c7 6d c1 4a-20 3b 82 7d aa 15 e8 65 ...@.m.J ;.}...e
0040 - 3b 92 bd c8 20 e9 9d 41-f1 77 51 d9 ae 31 c4 2c ;...A.wQ..1.,
0050 - 32 5a 2Z

write to 00988EB0 [009952C8] (58 bytes => 58 (0x3A))

0000 - 17 03 01 00 18 39 2f df-43 75 91 13 34 1b 12 049/.Cu..4...
0010 - 7d ef 8d e1 86 54 4f 67-c8 1d cd 07 a4 17 03 01 }....TOg.....
0020 - 00 18 53 d9 22 9d eb 6e-8b 79 f8 e4 82 2f ba ea ..S."..n.y.../.
0030 - 03 a5 3f 12 85 2e 9f 64-ff dc ..?....d..

read from 00988EB0 [00990AB8] (5 bytes => 5 (0x5))

0000 - 17 03 01 01 48.....H

read from 00988EB0 [00990ABD] (328 bytes => 328 (0x148))

0000 - bd eb 8b 9c 01 ac 73 30-8f ca a4 8b 2a 6f bd 02s0....*o..
0010 - d7 fc 71 18 61 47 f2 1d-70 8b 10 7d 98 28 a4 50 ..q.aG..p..}..(P
0020 - f3 0f 42 e8 c5 e1 3e 53-34 bd c7 62 34 1b 5e 8c ..B...>S4..b4.^.
0030 - 99 2d 89 c6 b3 f0 19 96-22 97 43 b8 8f 9d 76 42 ..-.....".C..vB
0040 - 95 a5 7c db 3b 22 dd 57-29 8d e8 d4 28 3e 89 d8 ..|;"..W)...(>..
0050 - 46 e5 dc 35 51 56 f8 44-d1 82 44 a0 65 b0 93 22 F..5QV.D..D.e.."
0060 - 4b 0a eb 07 26 c9 2a e2-45 4c de 07 0c bb 3e c6 K...&.*.EL...>..
0070 - bc 37 94 cd ec 94 2f 35-76 37 13 4d 0f 88 9c b1 ..7..../5v7.M....
0080 - d7 1c 58 8a 35 5b 32 bc-12 2b 9c e6 5b d4 86 bd ..X.5[2..+..[...
0090 - 39 fc 99 18 79 ec f7 53-db 59 74 49 da 07 69 54 9...y..S.YtL..iT
00a0 - f4 66 aa 36 34 39 f9 0b-87 50 9e 76 db 9f d0 44 ..f.649...P.v..D
00b0 - 0c 0d e7 65 80 9b b8 51-56 3d d0 db aa 55 ff ca ...e...QV=...U..
00c0 - 74 38 24 c1 8c d7 32 cf-ab 03 b3 59 29 0f 80 18 t8\$...2....Y)...
00d0 - 6a d4 e0 7e fd 41 8c f7-1d 81 12 a7 00 b3 71 39 j..~.A.....q9
00e0 - 78 1e 3c 17 42 d4 99 22-69 7b 2d 09 ef d8 6e f4 x.<.B.."i{....n.
00f0 - 64 f6 61 34 72 8c 89 f5-a8 ea 1c b1 0d 08 ff 17 d.a4r.....
0100 - 51 3e 46 2b 38 75 61 6a-1e 34 f4 14 14 38 0d 5e Q>F+8uaj.4..8.^
0110 - 6e ba db ef 83 88 ee a5-2c 18 5a 0c 27 e3 d9 19 n.....,Z.'...
0120 - 6c a3 12 c0 a1 3d e1 14-96 d3 1a f9 c9 f2 aa d6 l....=.....
0130 - 12 d5 36 ae 36 f2 18 f5-df c6 ef 34 d7 7d 2b 70 ..6.6....4.}+p
0140 - 99 88 47 93 91 09 56 b1- ..G...V..

HTTP/1.1 **200 OK**

Date: Tue, 02 Mar 2004 07:18:08 GMT

Server: Apache/1.3.29 (Win32) mod_ssl/2.8.16 OpenSSL/0.9.7c

Last-Modified: Sat, 07 Feb 2004 10:53:25 GMT

ETag: "0-23-4024c3a5"

Accept-Ranges: bytes

Content-Length: 35

Connection: close

Content-Type: text/html

<h1>Home page on main server</h1>

read from 00988EB0 [00990AB8] (5 bytes => 5 (0x5))

0000 - 15 03 01 00 18

.....

read from 00988EB0 [00990ABD] (24 bytes => 24 (0x18))

0000 - a5 47 51 bd aa 0f 9b e4-ac d4 28 f2 d0 a0 c8 fa .GQ.....(.....

0010 - 2c d4 e5 e4 be c5 01 85-

,.....

closed

write to 00988EB0 [009952C8] (29 bytes => 29 (0x1D))

0000 - 15 03 01 00 18 d4 19 b9-59 88 88 c0 c9 38 ab 5cY...8.\

0010 - 98 8c 43 fd b8 9e 14 3d-77 5e 4c 68 03 ..C....=w^Lh.

Trace Analysis

The data to be transmitted is broken up into series of fragments. Each fragment is protected for integrity using HMAC. (more)

Each SSL record begins with a 5-byte header:

- Byte 0: Record Content Type. Four Content Types are defined, as follows:

Content Type	Hex Code	Description
Handshake	0x16	The record carries a handshaking message
Application_Data	0x17	Encrypted Application Data
Change_Cipher_Spec	0x14	To indicate a change in encryption methods.
Alert	0x15	To signal various types of errors

- Byte 1 & 2: SSL version (0x0301 for TLSv1, 0x0300 for SSLv3).
- Byte 3 & 4: The record length, excluding the 5-byte header.

Let us begin looking into the handshake message contained within a SSL record (of Content Type 0x16). The handshake message has a 4-byte header:

- Byte 0: Handshake Type, as follows:

Handshake Type	Hex Code
hello_request	0x00
client_hello	0x01
server_hello	0x02
certificate	0x0b
server_key_exchange	0x0c
certificate_request	0x0d
server_hello_done	0x0e
certificate_verify	0x0f
client_key_exchange	0x10
finished	0x14

- Byte 1 - 3: The message length, excluding the 3-byte header.

Hence, a `client_hello` record will begin with a 5-byte record header, followed by a 4-byte handshake message header. For example,

Client_Hello

The first handshake message is always sent by the client, called `client_hello` message. In this message, the client tells the server its preferences in terms of protocol version, ciphersuit, and compression method. The client also includes a 32-byte random number (`client_random`) in the message, which is made up of a 4-byte GMT Unix time (seconds since 1970), plus another 28 random bytes.

You must refer to RFC2246 for the structure of the `Client_Hello` message.

Bytes	Len	Value	Description
00	1	16	Record Content Type - Handshake Message
01-02	2	03 01	SSL version - TLSv1
03-04	2	00 61	Record Length
05	1	01	Handshake Type - Client_Hello
06-08	3	00 00 5d	Message Length (0x61-4 = 0x5d)
09-0A	2	03 01	Client preferred version (client_version) - TLSv1
0B-0E	4	40 44 35 27	GMT Time
0C-2A	28	5c ... 72	28 random bytes Client_Random
2B	1	00	Session ID Length 0 (for resuming the session)
2C-2D	2	00 36	Ciphersuit Length - 27 choices (2-byte each)
2E-63	54	The 27 Ciphersuits (See Table)
64	1	01	Compression Method Length - 1
65	1	00	Compression Method: NULL.

Ciphersuit Code used in `Client_Hello` and `Server_Hello` messages is tabulated as follows:

Cipher Suite	Auth	Key Exchange	Encryption	Hash	Code
RSA_WITH_NULL_MD5	RSA	RSA	NULL	MD5	0001
RSA_WITH_NULL_SHA	RSA	RSA	NULL	SHA	0002
RSA_EXPORT_WITH_RC4_40_MD5	RSA	RSA_EXPORT	RC4_40	MD5	0003
RSA_WITH_RC4_128_MD5	RSA	RSA	RC4_128	MD5	0004
RSA_WITH_RC4_128_SHA	RSA	RSA	RC4_128	SHA	0005
RSA_EXPORT_WITH_RC2_CBC_40_MD5	RSA	RSA_EXPORT	RC2_40_CBC	MD5	0006

RSA_WITH_IDEA_CBC_SHA	RSA RSA	IDEA_CBC	SHA 0007
RSA_EXPORT_WITH_DES40_CBC_SHA	RSA RSA_EXPORT	DES40_CBC	SHA 0008
RSA_WITH_DES_CBC_SHA	RSA RSA	DES_CBC	SHA 0009
RSA_WITH_3DES_EDE_CBC_SHA	RSA RSA	3DES_EDE_CB C	SHA 000A
DH_DSS_EXPORT_WITH_DES40_CBC_SHA	RSA DH_DSS_EXP	DES_40_CBC	SHA 000B
DH_DSS_WITH_DES_CBC_SHA	DSS DH	DES_CBC	SHA 000C
DH_DSS_WITH_3DES_EDE_CBC_SHA	DSS DH	3DES_EDE_CB C	SHA 000D
DH_RSA_EXPORT_WITH_DES40_CBC_SHA	RSA DH_EXPORT	DES_40_CBC	SHA 000E
DH_RSA_WITH_DES_CBC_SHA	RSA DH	DES_CBC	SHA 000F
DH_RSA_WITH_3DES_EDE_CBC_SHA	DSS DH	3DES_EDE_CB C	SHA 0010
DHE_DSS_EXPORT_WITH_DES40_CBC_SHA	DSS DH_EXPORT	RC4_40	SHA 0011
DHE_DSS_WITH_DES_CBC_SHA	DSS DHE	RC4_128	SHA 0012
DHE_DSS_WITH_3DES_EDE_CBC_SHA	DSS DHE	DES_40_CBC	SHA 0013
DHE_RSA_EXPORT_WITH_DES40_CBC_SHA	RSA DHE_EXPORT	DES_CBC	SHA 0014
DHE_RSA_WITH_DES_CBC_SHA	RSA DH	DES_CBC	SHA 0015
DHE_RSA_WITH_3DES_EDE_CBC_SHA	RSA DHE	3DES_EDE_CB C	SHA 0016
DH_anon_EXPORT_WITH_RC4_40_MD5	- DH_EXPORT	RC4_40	MD 5 0017
DH_anon_WITH_RC4_128_MD5	- DH	RC4_128	MD 5 0018
DH_anon_EXPORT_WITH_DES40_CBC_SHA	- DH_EXPORT	DES_40_CBC	SHA 0019
DH_anon_WITH_DES_CBC_SHA	- DH	DES_CBC	SHA 001A
DH_anon_WITH_3DES_EDE_CBC_SHA	- DH	3DES_EDE_CB C	SHA 001B

Server_Hello

In response to the `client_hello` message, the server returns a `server_hello` message to tell the client its choice of protocol version, ciphersuit and compression method. The server also includes a 32-byte random number (`server_random`) in the message.

Bytes	Len	Value	Description
00	1	16	Record Content Type - Handshake Message
01-02	2	03 01	SSL version - TLSv1
03-04	2	00 2a	Record Length
05	1	02	Handshake Type - Server_Hello
06-08	3	00 00 26	Message Length
09-0A	2	03 01	Protocol Version Chosen - TLSv1
0B-0E	4	40 44 35 27	GMT Time (sec since 1970)
0C-2A	28	cc ... b9	28 random bytes Server_Random
2B	1	00	Session ID Length 0 (for resuming the session)
2C-2D	2	00 16	Ciphersuit Chosen: DHE_RSA_WITH_3DES_EDE_CBC_SHA
2E	1	00	Compression Method Chosen: NULL.

Certificate

The certificate message consists of a chain of X.509 certificates in the correct order. The first certificate belongs to the server, and the next certificate contains the key that certifies the first certificate (i.e., the server's certificate), and so on. The client uses the server's public key (contained inside the server's certificate) to either encrypt the `pre_master_secret` or verify the `server_key_exchange`, depending on which ciphersuit is used.

Bytes	Len	Value	Description
00	1	16	Record Content Type - Handshake Message
01-02	2	03 01	SSL version - TLSv1
03-04	2	02 05	Record Length
05	1	0b	Handshake Type - certificate
06-08	3	00 02 01	Message Length
09-0B	3	00 01 fe	Certificate Length Certificates (to be traced)

The X.509 certificate structure can be found from the ITU recommendation X.509 "The directory - Authentication Framework".

Server_Key_Exchange

Server_Hello_Done

This is an empty message indicating that the server has sent all the handshaking messages. This is needed because the server can send some optional messages after the certificate message.

Bytes	Len	Value	Description
00	1	16	Record Content Type - Handshake Message
01-02	2	03 01	SSL version - TLSv1
03-04	2	00 04	Record Length
05	1	0e	Handshake Type - Server_Hello_Done (check the last 3 bytes)

Client_Key_Exchange

The `client_key_exchange` message contains the `pre_master_secret` when RSA key exchange is used. The `pre_master_secret` is 48-byte, consists of protocol version (2 bytes) and 46 random bytes.

Bytes	Len	Value	Description
00	1	16	Record Content Type - Handshake Message
01-02	2	03 01	SSL version - TLSv1
03-04	2	00 86	Record Length
05	1	10	Handshake Type - Client_Key_Exchange
06-08	3	00 00 82	Message Length
			<code>pre_master_secret</code> (130 bytes): encrypted using server's public key extracted from the server's certificate

Change_Cipher_Spec

Bytes	Len	Value	Description
00	1	14	Record Content Type - Change_Cipher_Spec
01-02	2	03 01	SSL version - TLSv1
03-04	2	00 01	Record Length
05	1	01	??

Certificate_Verify

Change_Cipher_Spec

Unknown Handshaking Message (D4) - to check

Application_Data

Client-to-Server - the HTTP request message: GET /test.html HTTP/1.0

Server-to-Client - the HTTP response message

Alert

Comparison of TLS v1, SSL v3 and SSL v2

The TLS v1 specification stated, "TLS v1 and SSL v3 are very similar". Some of minor differences include minor changes in HMAC calculation, ciphersuit support, and pseudo-random number generation. TLS v1 can be regarded as SSL v3.1.

SSL v2 has a big security hole in the negotiation of the ciphersuit (and should not be used). The attacker can convince the client and server to use a weaker encryption than what they are capable of. This is called "ciphersuit rollback" attack.

Result:

Thus the confidentiality and Integrity using SSL was verified.

Ex. No : 7
Date:

Experiment Eavesdropping, Dictionary attacks, MITM attacks

Aim :

To experiment eavesdropping, Dictionary attacks, MITM attacks

Visual Objective:



Introduction

Password cracking is a term used to describe the penetration of a network, system, or resource with or without the use of tools to unlock a resource that has been secured with a password. Password cracking tools may seem like powerful decryptors, but in reality are little more than fast, sophisticated guessing machines.

Types of password breaking

Dictionary attack

A simple *dictionary* attack is usually the fastest way to break into a machine. A dictionary file (a text file full of dictionary words) is loaded into a cracking application, which is run against user accounts located by the application.

Brute force attack

A *brute force* attack is a very powerful form of attack, though it may often take a long time to work depending on the complexity of the password. The program will begin trying any and every combination of numbers and letters and running them against the hashed passwords.

Passwords that are composed of random letters numbers and characters are most vulnerable to this type of attack.

Hybrid attack

Another well-known form of attack is the *hybrid* attack. A hybrid attack will add numbers or symbols to the search words to successfully crack a password. Many people change their passwords by simply adding a number to the end of their current password. Therefore, this type of attack is the most versatile, while it takes longer than a standard dictionary attack it does not take as long as a brute force attack.

Cracking Process

Since a brute force attack is the most time consuming and is not likely to break any passwords that are not composed of random characters, the best plan is to use techniques that are computationally efficient compared to untargeted and unspecific techniques. By applying what is known about how users select passwords, an intruder can tremendously increase the odds in their favor of finding passwords. With the right techniques, some poor passwords can be cracked in under a second.

The real power of dictionary attacks come from understanding the ways in which most people vary names and dictionary words when attempting to create a password. By applying all the common transformations to every word in the electronic list and encrypting each result the number tested passwords multiplies rapidly. Cracking tools can often detect “clever” ways of manipulating words to hide their origin. For example, such cracking programs often subject each word to a list of rules. A *rule* could be anything, any manner in which a word might appear. Typical rules might include

- Alternate upper- and lowercase lettering.

- Spell the word forward and then backward, and then fuse the two results (for example: cannac).

- Add the number 1 to the beginning and/or end of each word.

Naturally, the more rules one applies to the words, the longer the cracking process takes. However, more rules also guarantee a higher likelihood of success.

Task 1 – Microsoft Office Password Recovery

Many applications require you to establish an ID and password that may be saved and automatically substituted for future authentication. The password will usually appear on the screen as a series of asterisks. This is fine as long as your system remembers the password for you but what if it "forgets" or you need it for use on another system. Fortunately, many utilities have been written to recover such passwords. In this task, you will use OfficeKey to recover the password for a MS word document.

Step 1: Find the folder “Lab1” on your desktop, and open it.

You will find OfficeKey and a MS document in the folder.

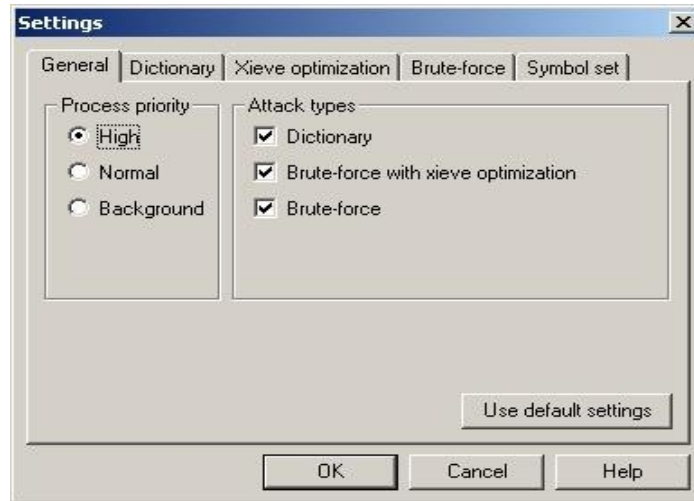
Step 2: Open the Office Key – Password Recovery tool

Step 3: Press the “Recover” button in the upper left corner, or select File → Recover



Step 4: Choose the password protected MS Office File you have saved to the Desktop.

Step 5: After running the first password auditing session, check to see if Office key has cracked the password. If the password has not been cracked press the Settings button on the upper tool bar.



Step 6: Once in the Settings menu you will be able to modify the search parameters and customize a more targeted search



Step 7: Repeat steps 3 and 4 until the password has been cracked and opens the MS Office File.

Step 8: Write down the contents of the MS word document and the password into your lab report and submit it to your TA.

Task 2 – Password Auditing (Windows platform):

The purpose of this task is to familiarize you with act of password cracking/recovery. Password cracking software uses a variety of approaches, including intelligent guessing, dictionary attacks and automation that tries every possible combination of characters. Given enough time the automated method can crack any password, but more effective passwords will last months before breaking.

When a password is entered and saved on a computer it is encrypted, the encrypted password becomes a string of characters called a “hash” and is saved to a password file. A password cannot be reverse-decrypted. So a cracking program encrypts words and characters given to it (wordlist or randomly generated strings of characters) and compares the results with hashed passwords. If the hashes match then the password has successfully been guessed or “cracked”. This process is usually performed offline against a captured password file so that being locked out of the account is not an issue, and guessing can go on continuously. Thus, revealing the passwords is simply a matter of CPU time and dictionary size

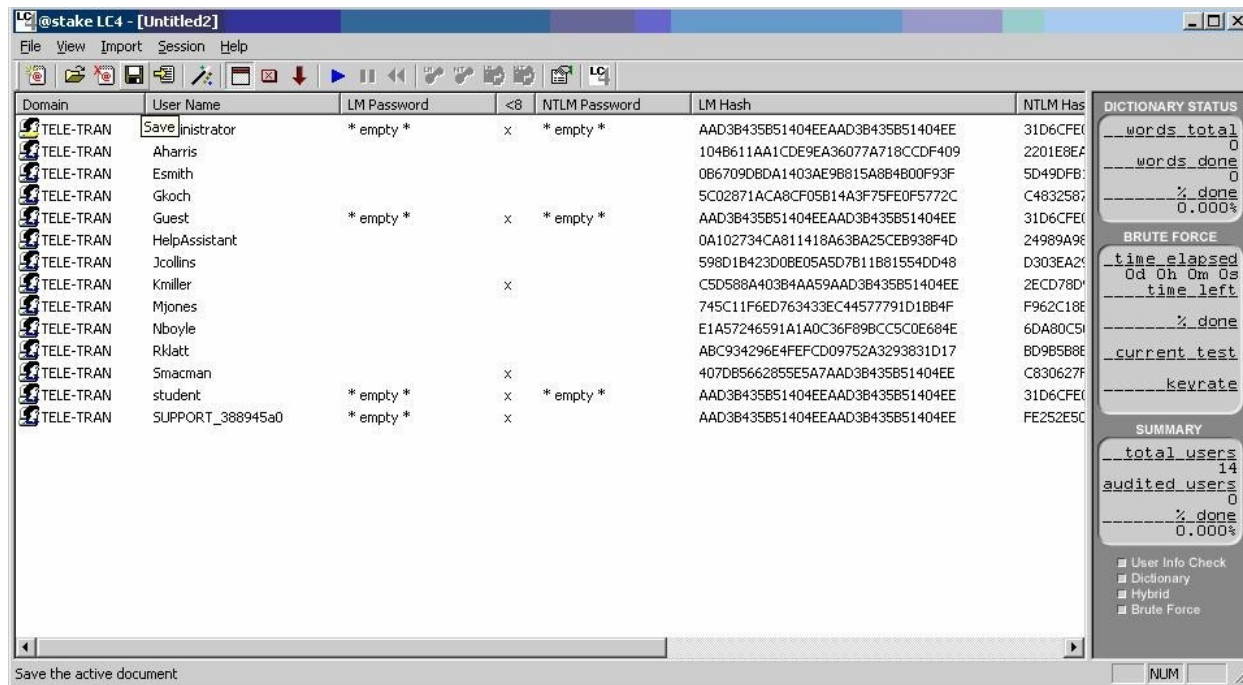
1. You obtain a *dictionary file*, which is no more than a flat file (plain text) list of words (commonly referred to as *wordlists*).
2. These words are fed through any number of programs that encrypt each word. Such encryption conforms to the DES standard.
3. Each resulting encrypted word is compared with the target password. If a match occurs, there is better than a 90 percent chance that the password was cracked.

Step 1: Go to Lab1 folder, and open LC4 to audit the passwords on your Windows system.

Select File New Session

Select Import Import from PWDUMP File (in the same folder)

Select the “Passwords” file that has been provided to you.



Objectives

This password file has been retrieved from a system that we must gain access to. To do this you must crack as many passwords as possible as quickly as possible. We have captured the user names and encrypted passwords for ten users. The user names follow a standard pattern of first initial and last name, but the passwords have no set standards. We do know that users of this system are encouraged to add numbers and other characters to the words they chose for passwords.

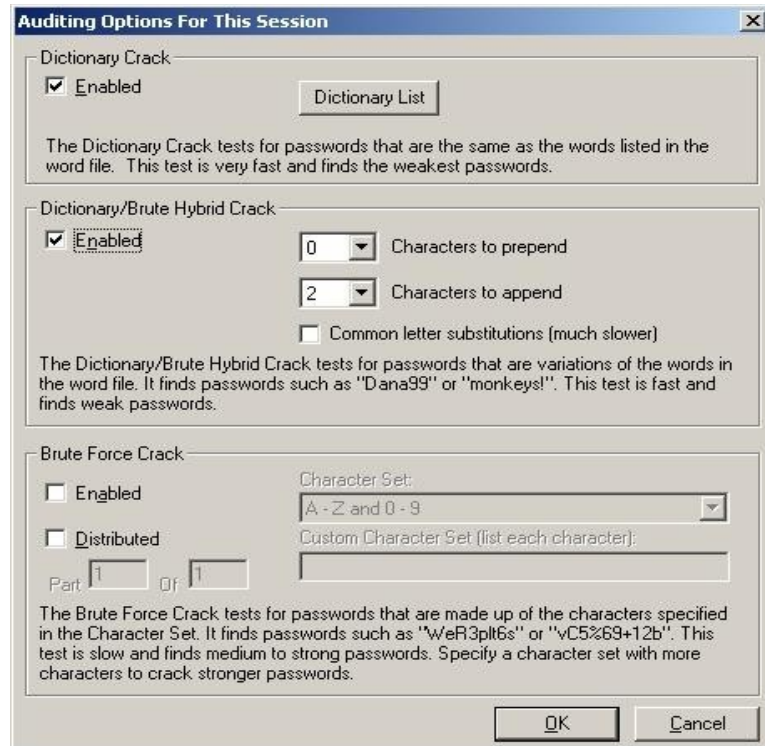
To aid you in cracking these passwords we have managed to collect some basic information about the users. This personal information may help you target your searches as to what the user's password may be.

Kmiller	Ken Miller is an avid fly fisher and his record number of catches is just under 30
Smacman	Steven MacMan has a fiancé who's name is 4 letters long and starts with a "K"
Gkoch	Gina Koch grew up with her German grandmother, who used to call

	her 'Little Precious' *
Mjones	Matt Jones was born in 1979. He compares himself to a Shakespearean character who was born via C section
Tgriffin	Tim Griffin loves funky '70's and '80s music. And songs about 'Love'
Rklatt	Ryan Klatt is a big Star Trek fan and has most likely chosen an obscure reference for his password *
Nboyle	Nancy Boyle is an a fan of the books of British writer Douglas Adams
Esmith	Edward Smith was very close to his grandfather who died in 1968. We know his grandfather's name was a less common name starting with 'L'
Jcollins	Jim Collins keeps a copy of the book "The Prince" *
Hharris	Alan Harris has a wife named Sue and a daughter named Megan Alan was married on May 3 rd . His daughter was born on August 6 th

Step 2: Select Session Session Options

Use this menu to customize your password search. Here you can add different word list for Dictionary attacks, change Hybrid attack features. Keep in mind you are working with a short dead line and more in depth searches will take longer then you have. You must use the information given to you to target your search most specifically at more likely passwords.



Step 3: Select Session Begin “Audit” or Press the blue play button on the upper toolbar to start the password search.

Step 4: After the first search has run check your progress. Have some of the passwords been cracked all the way though or have some only been partially cracked. Use what you’ve learned from this first search to target your next few searches. You will need to search the internet and use the information you have been given about each user to find words they may have used as their password.

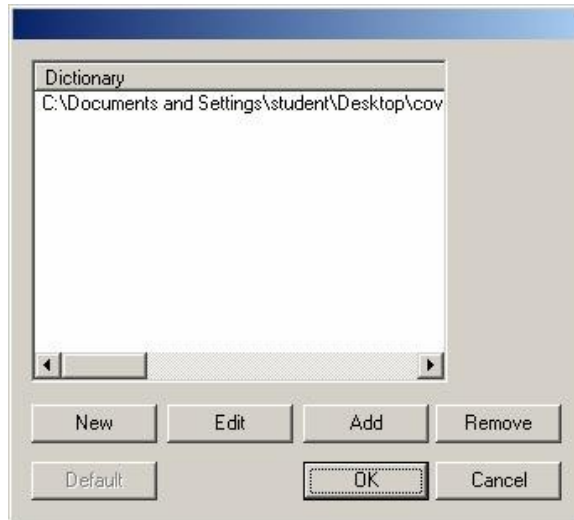
Note: The question marks in the partially cracked passwords do not necessarily represent the number of remaining undiscovered characters.

Step 5: Add words to your wordlist

Session Session Options

Press the ‘Dictionary List’ button in the Dictionary crack section. Here you can edit your current word list and add words by selecting the ‘EDIT’ button and entering each word on a new line. You can also add multiple dictionaries and wordlist.

Step 6: You may chose to conduct dictionary attacks with other wordlists. You can find additional wordlist to use here: <ftp://ftp.cerias.purdue.edu/pub/dict>



Step 7: Continue searching for possible passwords during the remainder of the lab. Repeating steps 3 and 4 each time you modify your search.

Step 8: Once you have cracked all the passwords in the file, write them down in your lab report or once the lab time has ended, submit the passwords you were able to crack.

Result :

Thus the experiment for Eavesdropping, Dictionary attacks, MITM attacks was done successfully.

Ex. No : 8
Date:

Perform an Experiment to Sniff Traffic using ARP Poisoning. Experiment to Sniff Traffic using ARP Poisoning.

AIM

Perform an Experiment to Sniff Traffic using ARP Poisoning.

Description:

ARP is the acronym for Address Resolution Protocol. It is used to convert IP address to physical addresses [MAC address] on a switch. The host sends an ARP broadcast on the network, and the recipient computer responds with its physical address [MAC Address]. The resolved IP/MAC address is then used to communicate. **ARP poisoning is sending fake MAC addresses to the switch so that it can associate the fake MAC addresses with the IP address of a genuine computer on a network and hijack the traffic.**

ARP Poisoning Countermeasures

Static ARP entries: these can be defined in the local ARP cache and the switch configured to ignore all auto ARP reply packets. The disadvantage of this method is, it's difficult to maintain on large networks. IP/MAC address mapping has to be distributed to all the computers on the network. **ARP poisoning detection software:** these systems can be used to cross check the IP/MAC address resolution and certify them if they are authenticated. Uncertified IP/MAC address resolutions can then be blocked.

Operating System Security: this measure is dependent on the operating system been used. The following are the basic techniques used by various operating systems.

- **Linux based:** 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.
- Computers communicate using networks. These networks could be on a local area network LAN or exposed to the internet. **Network Sniffers are programs that capture low-level package data that is transmitted over a network.** An attacker can analyze this information to discover valuable information such as user ids and passwords.
- In this article, we will introduce you to common network sniffing techniques and tools used to sniff networks.

What is network sniffing?

Computers communicate by broadcasting messages on a network using IP addresses. Once a message has been sent on a network, the recipient computer with the matching IP address responds with its MAC address.

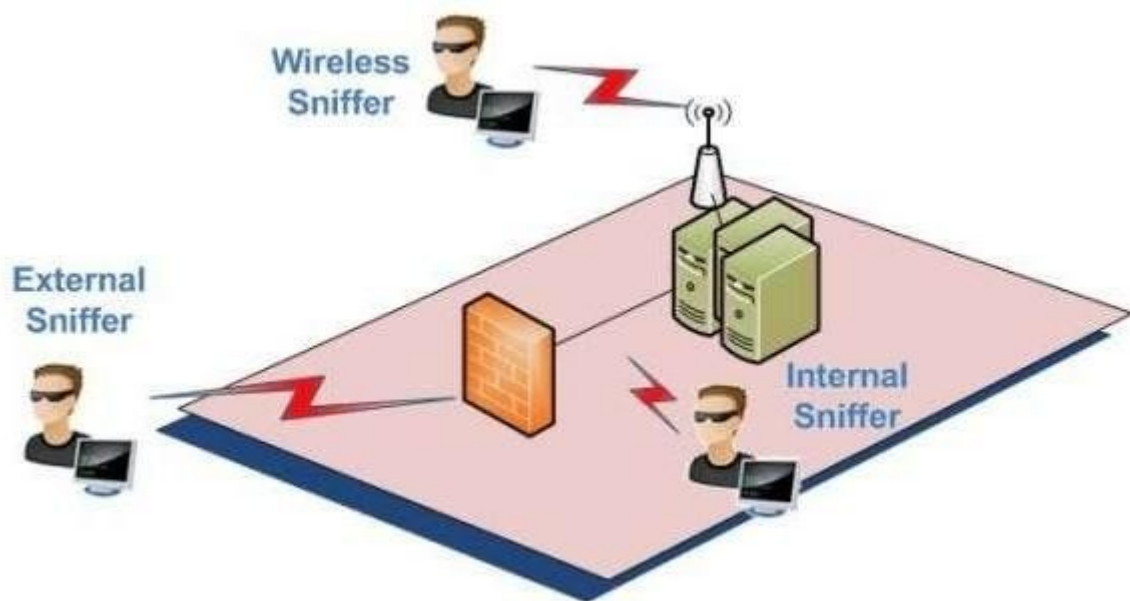
Network sniffing is the process of intercepting data packets sent over a network. This can be done by the specialized software program or hardware equipment. Sniffing can be used to;

- Capture sensitive data such as login credentials
 - Eavesdrop on chat messages
 - Capture files have been transmitted over a network
- The following are protocols that are vulnerable to sniffing

- Telnet

- Rlogin
- HTTP
- SMTP
- NNTP
- POP
- FTP
- IMAP

The above protocols are vulnerable if login details are sent in plain text



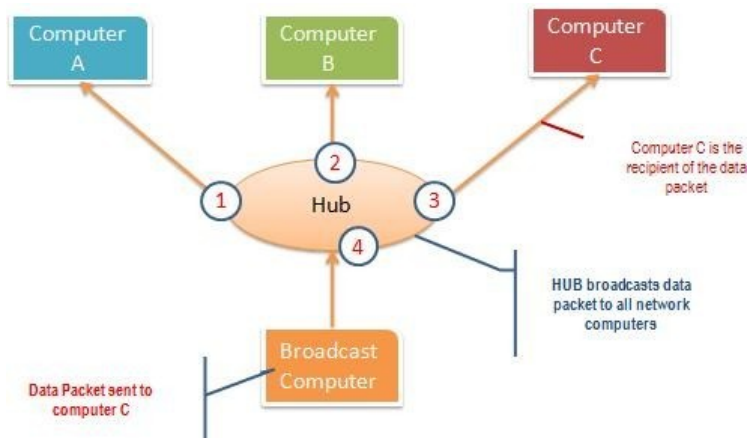
Passive and Active Sniffing

Before we look at passive and active sniffing, let's look at two major devices used to network computers; hubs and switches.

A hub works by sending broadcast messages to all output ports on it except the one that has sent the broadcast. The recipient computer responds to the broadcast message if the IP address

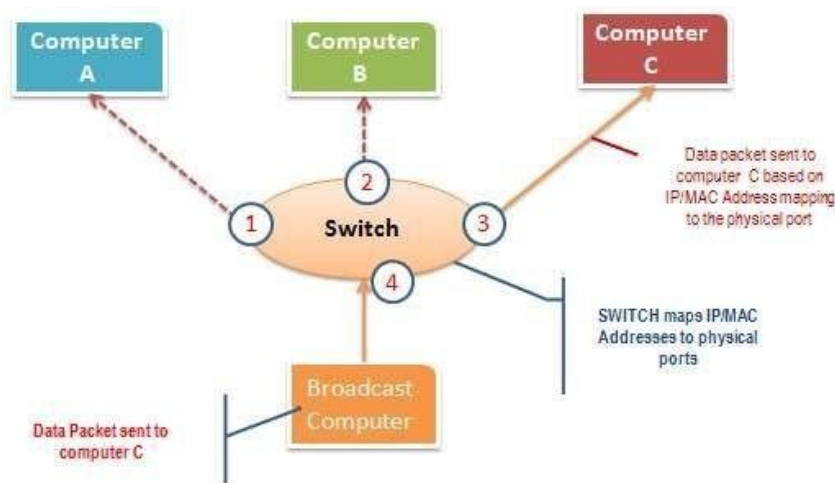
matches. This means when using a hub, all the computers on a network can see the broadcast message. It operates at the physical layer (layer 1) of the OSI Model.

The diagram below illustrates how the hub works.



A switch works differently; it maps IP/MAC addresses to physical ports on it. Broadcast messages are sent to the physical ports that match the IP/MAC address configurations for the recipient computer. This means broadcast messages are only seen by the recipient computer. Switches operate at the data link layer (layer 2) and network layer (layer 3).

The diagram below illustrates how the switch works.



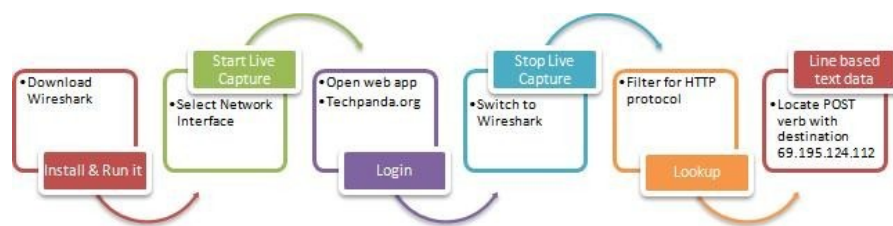
Passive sniffing is intercepting packages transmitted over a network that uses a hub. It is called passive sniffing because it is difficult to detect. It is also easy to perform as the hub sends

broadcast messages 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.

Sniffing the network using Wireshark

The illustration below shows you the steps that you will carry out to complete this exercise without confusion



Download Wireshark from this link <http://www.wireshark.org/download.html>

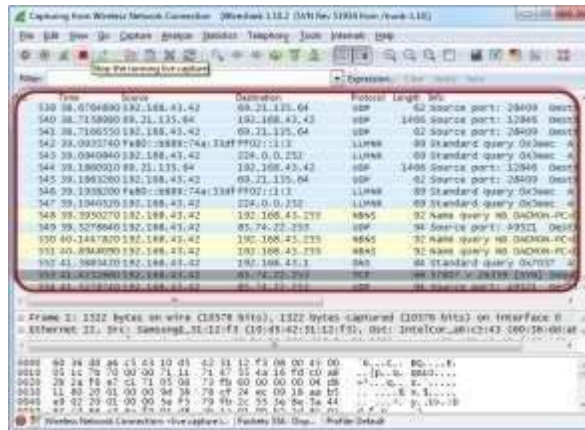
- Open Wireshark
- You will get the following screen



- Select the network interface you want to sniff. Note for this demonstration, we are using a wireless network connection. If you are on a local area network, then you should select the

local area network interface.

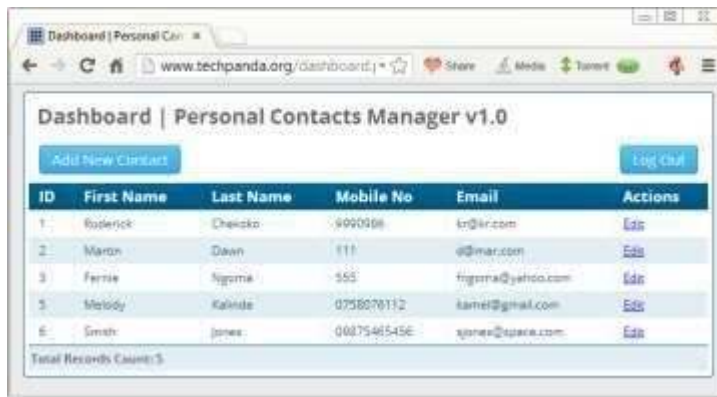
- Click on start button as shown above



- Open your web browser and type in <http://www.techpanda.org/>



- The login email is admin@google.com and the password is **Password2010**
- Click on submit button
- A successful logon should give you the following dashboard



- Go back to Wireshark and stop the live capture



- Filter for HTTP protocol results only using the filter textbox



No.	Time	Source	Destination	Protocol	Length	Info
172	10.4308270	192.168.43.42	69.195.124.112	HTTP	433	GET / HTTP/1.1
188	11.4490510	69.195.124.112	192.168.43.42	HTTP	1188	HTTP/1.1 200 OK (text/html)
329	23.7367370	192.168.43.42	192.168.43.42	HTTP	233	HTTP/1.1 200 OK (text/html)
362	25.4481440	192.168.43.42	192.168.43.42	HTTP	362	GET /subscribe?host_int=74 HTTP/1.1
724	26.8228240	192.168.43.42	69.195.124.112	HTTP	724	POST /index.php HTTP/1.1
1254	27.1204930	69.195.124.112	192.168.43.42	HTTP	1254	HTTP/1.1 302 Moved Temporarily
567	27.7534960	192.168.43.42	69.195.124.112	HTTP	567	GET /dashboard.php HTTP/1.1
362	28.1167100	192.168.43.42	192.168.43.42	HTTP	362	[TCP Retransmission] GET / HTTP/1.1
1322	28.7180900	69.195.124.112	192.168.43.42	HTTP	1322	HTTP/1.1 200 OK (text/html)

Look for POST verb under Info column

- Locate the Info column and look for entries with the HTTP verb POST and click on it
- Just below the log entries, there is a panel with a summary of captured data. Look for the summary that says Line-based text data: application/x-www-form-urlencoded

Filter: http

No.	Time	Source	Destination	Protocol	Length	Info
172	10.4308270	192.168.43.42	69.195.124.112	HTTP	433	GET / HTTP/1.1
188	11.4490510	69.195.124.112	192.168.43.42	HTTP	1188	HTTP/1.1 200 OK (text/html)
329	23.7367370	192.168.43.42	192.168.43.42	HTTP	233	HTTP/1.1 200 OK (text/html)
362	25.4481440	192.168.43.42	192.168.43.42	HTTP	362	GET /subscribe?host_int=74 HTTP/1.1
724	26.8228240	192.168.43.42	69.195.124.112	HTTP	724	POST /index.php HTTP/1.1
1254	27.1204930	69.195.124.112	192.168.43.42	HTTP	1254	HTTP/1.1 302 Moved Temporarily
567	27.7534960	192.168.43.42	69.195.124.112	HTTP	567	GET /dashboard.php HTTP/1.1
362	28.1167100	192.168.43.42	192.168.43.42	HTTP	362	[TCP Retransmission] GET / HTTP/1.1
1322	28.7180900	69.195.124.112	192.168.43.42	HTTP	1322	HTTP/1.1 200 OK (text/html)

Line-based text data: application/x-www-form-urlencoded

email=admin@google.com&password=password&remember_me=remember+me

All POST variables have been captured in plaintext

- You should be able to view the plaintext values of all the POST variables submitted to the server via HTTP protocol.

Result:

Thus the experiment to Sniff Traffic using ARP Poisoning was performed.

Ex. No : 9

Demonstration of Intrusion Detection System(IDS)

Date:

AIM:

To demonstrate 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 is important 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 “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 folder “C:\Snort\bin” folder. (at the Prompt, 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

```
Administrator: C:\Windows\system32\cmd.exe
Total Memory Allocated: 0
=====
Snort exiting
C:\Snort\bin>snort -W

--> Snort! <*-
o"~>~
,,,~
ean
Version 2.9.6.0-WIN32 GRE (Build 47)
By Martin Roesch & The Snort Team: http://www.snort.org/snort/snort-team

Copyright (C) 2014 Cisco and/or its affiliates. All rights reserved.
Copyright (C) 1998-2013 Sourcefire, Inc., et al.
Using PCRE version: 8.10 2010-06-25
Using ZLIB version: 1.2.3

Index      Physical Address      IP Address      Device Name      Description
-----
1          00:00:00:00:00:00      0000:0000:fe80:0000:0000:0000:78d2:6299 \Device\
NPF_{45DAC1EF-70A2-4C33-B712-AE311620EB7A} VMware Virtual Ethernet Adapter
2          00:00:00:00:00:00      0000:0000:fe80:0000:0000:0000:bca3:2f66 \Device\
NPF_{C355D233-3D77-484F-A344-65626159980E} VMware Virtual Ethernet Adapter
3          00:00:00:00:00:00      0000:0000:fe80:0000:0000:0000:ada3:46c9 \Device\
NPF_{3264BC0F-4BF2-49C5-B5D9-A12EFE40F17C} Microsoft

C:\Snort\bin>
```

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.

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

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)

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

Example:

example snort

Change the RULE_PATH variable to the path of rules folder.

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

path to rules

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

```
C:\Snort\lib\snort_dynamicccpreprocessor
```

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

Change the path of the “dynamicengine” variable value in the “snort.conf” file..

Example:

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

Add the paths for “include classification.config” and “include reference.config” files.

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

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

Remove the comment (#) on the line to allow ICMP rules, if it is commented with a #.

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

You can also remove the comment of ICMP-info rules comment, if it is

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

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
```

Comment (add a #) the whitelist \$WHITE_LIST_PATH/white_list.rules and the blacklist

Change the nested_ip inner , \ to nested_ip inner #, \

Comment out (#) following lines:

```
#preprocessor normalize_ip4
```

```
#preprocessor normalize_tcp: ips ecn stream
```

```
#preprocessor normalize_icmp4
```

```
#preprocessor normalize_ip6
```

```
#preprocessor normalize_icmp6
```

Save the “snort.conf” file.

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)

If a 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
```

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 –


```
Administrator: C:\Windows\system32\cmd.exe - snort -A console -i3 -c c:\Snort\etc\snort.conf -l c:\Snort\var\log
Rules Engine: SF_SNORT_DETECTION_ENGINE Version 2.1 <Build 1>
Preprocessor Object: SF_SSLPP Version 1.1 <Build 4>
Preprocessor Object: SF_SSH Version 1.1 <Build 3>
Preprocessor Object: SF_SMTP Version 1.1 <Build 9>
Preprocessor Object: SF_SIP Version 1.1 <Build 1>
Preprocessor Object: SF_SDF Version 1.1 <Build 1>
Preprocessor Object: SF_REPUTATION Version 1.1 <Build 1>
Preprocessor Object: SF_POP Version 1.0 <Build 1>
Preprocessor Object: SF_MODBUS Version 1.1 <Build 1>
Preprocessor Object: SF_IMAP Version 1.0 <Build 1>
Preprocessor Object: SF_GTP Version 1.1 <Build 1>
Preprocessor Object: SF_FTPTELNET Version 1.2 <Build 13>
Preprocessor Object: SF_DNS Version 1.1 <Build 4>
Preprocessor Object: SF_DNP3 Version 1.1 <Build 1>
Preprocessor Object: SF_DCERPC2 Version 1.0 <Build 3>
Commencing packet processing (pid=2164)
03/29-23:53:16.033913 [**] [120:3:1] <http_inspect> NO CONTENT-LENGTH OR TRANSFER-ENCODING IN HTTP RESPONSE [**] [Classification: Unknown Traffic] [Priority: 3]
1 <TCP> 192.168.1.1:80 -> 192.168.1.20:56506
03/29-23:53:16.035372 [**] [120:3:1] <http_inspect> NO CONTENT-LENGTH OR TRANSFER-ENCODING IN HTTP RESPONSE [**] [Classification: Unknown Traffic] [Priority: 3]
1 <TCP> 192.168.1.1:80 -> 192.168.1.20:56507
03/29-23:53:16.036479 [**] [120:3:1] <http_inspect> NO CONTENT-LENGTH OR TRANSFER-ENCODING IN HTTP RESPONSE [**] [Classification: Unknown Traffic] [Priority: 3]
1 <TCP> 192.168.1.1:80 -> 192.168.1.20:56508
03/29-23:53:16.037093 [**] [120:3:1] <http_inspect> NO CONTENT-LENGTH OR TRANSFER-ENCODING IN HTTP RESPONSE [**] [Classification: Unknown Traffic] [Priority: 3]
1 <TCP> 192.168.1.1:80 -> 192.168.1.20:56509
03/29-23:53:16.142921 [**] [120:3:1] <http_inspect> NO CONTENT-LENGTH OR TRANSFER-ENCODING IN HTTP RESPONSE [**] [Classification: Unknown Traffic] [Priority: 3]
1 <TCP> 192.168.1.1:80 -> 192.168.1.20:302
03/29-23:53:16.194409 [**] [120:3:1] <http_inspect> NO CONTENT-LENGTH OR TRANSFER-ENCODING IN HTTP RESPONSE [**] [Classification: Unknown Traffic] [Priority: 3]
1 <TCP> 192.168.1.1:80 -> 192.168.1.20:56510
03/29-23:53:16.677078 [**] [120:3:1] <http_inspect> NO CONTENT-LENGTH OR TRANSFER-ENCODING IN HTTP RESPONSE [**] [Classification: Unknown Traffic] [Priority: 3]
1 <TCP> 192.168.1.1:80 -> 192.168.1.20:56512
03/29-23:53:16.808301 [**] [120:3:1] <http_inspect> NO CONTENT-LENGTH OR TRANSFER-ENCODING IN HTTP RESPONSE [**] [Classification: Unknown Traffic] [Priority: 3]
1 <TCP> 192.168.1.1:80 -> 192.168.1.20:56513
03/29-23:53:16.944237 [**] [120:3:1] <http_inspect> NO CONTENT-LENGTH OR TRANSFER-ENCODING IN HTTP RESPONSE [**] [Classification: Unknown Traffic] [Priority: 3]
1 <TCP> 192.168.1.1:80 -> 192.168.1.20:56514
03/29-23:53:16.948012 [**] [120:3:1] <http_inspect> NO CONTENT-LENGTH OR TRANSFER-ENCODING IN HTTP RESPONSE [**] [Classification: Unknown Traffic] [Priority: 3]
1 <TCP> 192.168.1.1:80 -> 192.168.1.20:56515
03/29-23:53:16.953992 [**] [120:3:1] <http_inspect> NO CONTENT-LENGTH OR TRANSFER-ENCODING IN HTTP RESPONSE [**] [Classification: Unknown Traffic] [Priority: 3]
1 <TCP> 192.168.1.1:80 -> 192.168.1.20:56516
03/29-23:53:16.967744 [**] [120:3:1] <http_inspect> NO CONTENT-LENGTH OR TRANSFER-ENCODING IN HTTP RESPONSE [**] [Classification: Unknown Traffic] [Priority: 3]
1 <TCP> 192.168.1.1:80 -> 192.168.1.20:56517
03/29-23:53:16.982649 [**] [120:3:1] <http_inspect> NO CONTENT-LENGTH OR TRANSFER-ENCODING IN HTTP RESPONSE [**] [Classification: Unknown Traffic] [Priority: 3]
1 <TCP> 192.168.1.1:80 -> 192.168.1.20:56518
```

RESULT:

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

Ex. No : 10

Date:

Network Monitoring Tools

Aim :

To explore about Network monitoring tools

Network monitoring is an essential part of network management. It involves using various tools to monitor a system network and determine slowness and weak connections, among other issues. Knowing more about these tools can help you understand them better and use the right ones that suit your requirements. In this article, we define what network monitoring tools are, provide details about various tools and discuss about some tips that can help you choose the right tool for your requirements.

What Are Network Monitoring Tools?

Network monitoring tools are software that you can use to evaluate network connections. These software programs can help you monitor a network connection and identify network issues, which may include failing network components, slow connection speed, network outage or unidentifiable connections. Network management and monitoring tools can also help you resolve these issues or establish solutions that prevent specific issues from occurring in the future.

Network Monitoring Tools

Here are eight monitoring tools along with their descriptions and features:

1. SolarWinds Network Performance Monitor

SolarWinds Network Performance Monitor is a multi-vendor monitoring tool. It allows users to monitor multiple vendors' networks at the same time. It also provides network insights for thorough visibility into the health of the networks. Some prominent features include network availability monitoring, intelligent network mapping, critical path visualisation, performance analysis and advanced alerting. SolarWinds also allows users to track VPN tunnel status. It prompts when a VPN

tunnel is available to help users ensure a stable connection between sites. SolarWinds provides a seven-day free trial, after which users can choose a preferred subscription plan.

2. Auvik

Auvik is a network monitoring and management tool. It offers a quick implementation process that helps users to set up the tool easily. It also has a clean user interface that makes it easy to navigate and use. The tool provides in-depth network visibility that enables faster troubleshooting for network issues. Users can automate network visibility using Auvik. It provides real-time updates on network issues and configuration changes.

3. Datadog Network Monitoring

Datadog Network Monitoring offers services for on-premises devices and cloud networks. A highlighting feature of this tool is the visualisations. It offers various graphical representations of all the network connections on a system. It also allows users to track key metrics like network latency, connection churn and transmission control protocol (TCP) retransmits. Users can monitor the health of a network connection at different endpoints at the application, IP address, port or process ID layers. Other prominent features include automated log collection and user interface monitoring.

4. Paessler PRTG Network Monitor

Paessler's network connection monitoring tool provides a clean user interface and network visibility on multiple devices. Users can track the health of different connection types like local area networks (LAN), wide area network (WAN), servers, websites, applications and services. The tools also integrate with various technologies, which makes it easier to use it for different types of applications. It provides distribute monitoring, allowing users to track network connections on devices in different locations. The tool also provides apps for mobile platforms that can help users to track network health on mobile phones.

5. ManageEngine OpManager

ManageEngine OpManager is a good network monitoring and managing tool for users that prefer in-depth view of network health and issues. This tool provides over 2000 network performance monitors that allow users to track and monitor their connections and perform detailed analyses on issues. It also

provides over 200 dashboard widgets that can help users customise their dashboard to their own suitability. Other features include CPU, memory and disk utilisation monitoring on local and virtual machines. It also allows setting network performance threshold and notifies the user in case of a violation.

6. Domotz

Domotz is an expansive tool that provides a list of features for monitoring network connections. It allows users to customise their network monitoring preferences. Users can write scripts to retrieve the data they wish to evaluate. It also allows connection to open ports on remote devices while ensuring network security. Users can also scan and monitor network connections globally. Domotz also allows to backup and restore network configuration for switches, firewalls and access points and alerts when there is a change in the configuration.

7. Checkmk

Checkmk is a tool that allows users to automate it completely. You can customise its operations and enable it to perform tasks automatically. It also identifies network and security components without the user requiring manual set up. For example, the tool can identify a firewall even if the user has not set it up. Its Agent Bakery feature enables users to manage agents and automate agent updating. This reduces manual effort to monitor network connections. The tool also includes over 2000 plug-ins for enhancing network monitoring.

8. Progress Whatsup Gold

Progress Whatsup Gold is a basic network monitoring software. It provides a minimal user interface with essential features like device monitoring, application monitoring, analysing network traffic and managing configurations. The tool allows users to monitor cloud devices, inspect suspicious connections, automate configuration backups and identify, and resolve bandwidth issues.

Other Tools For Network Monitoring

Here are three additional tools for network monitoring:

- Fortra Intermapper: This tool enables users to monitor network connections using network maps, allowing them to get a holistic view of all the connections. It also provides various colour codes for different network status, along with real-time notifications through text, email and sound.
- Nagios Core: Nagios Core is a monitoring engine that works as the primary application for all Nagios projects, including the Nagios Network Analyser. It integrates with other Nagios applications and provides users with features like a visual dashboard, custom application monitoring, automated alert system, advanced user management and network security monitoring.
- Zabbix: Zabbix provides a thorough network monitoring solution with features like server monitoring, cloud monitoring, application monitoring and service monitoring. The tool also includes features like metric collection, business monitoring and root cause analyses of network issues, and allows users to establish a threshold for connection anomalies.

Tips To Choose A Network Monitoring And Management Tool

Here are some useful tips that you can consider while selecting a tool for network monitoring:

Understand the requirements

Understanding why you require network monitoring software is important in the process. Define what feature you want and for what purpose. This can help you identify the right tool for your use. It may also help you choose the correct subscription plan on paid tools.

Browse multiple tools

Once you identify the requirements, consider browsing multiple tools. Visit the websites of the tools and look for the features you require. Spend time studying the features and understand how they can be useful to your requirements. You can also identify a few tools and compare their features to each other.

Consider the budget

Some tools may be free to use, while some may require you to purchase a subscription plan. Paid tools typically offer a free trial period of up to 30 days. Once you identify which tool you may like to use,

see if it is free or requires payment. If it is a paid tool, try exploring its features and efficiency during the trial period. Consider keeping a backup tool in case the tool that you choose does not fit your usage.

Result:

Thus the network monitoring tools was explored

Ex. No : 11

Study to configure Firewall, VPN

Date:

AIM:

To study the features of firewall in providing network security and to set Firewall Security in windows.

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.

When we first connect to some network, we are prompted to select a network location. This feature is known as Network Location Awareness(NLA). This feature enables us to assign a network profile to the connection based on the location. Different network profiles contain different collections of firewall rules. In Windows 7, different network profiles can be configured on different interfaces. For example, our wired interface can have different profile than our wireless interface. There are three different network profiles available:

- Public
- Home/Work - private network
- Domain - used within a domain

Configuring Windows Firewall

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



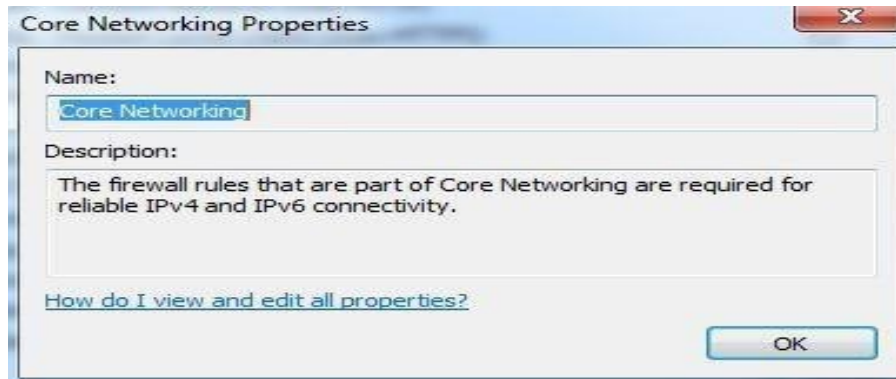
Firewall.

By default, Windows Firewall is enabled for both private (home or work) and public networks. 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 through Windows Firewall" option.



Exceptions

To change settings in this window we have to click the "Change settings" button. As you can see, here we have a list of predefined programs and features that can be allowed to communicate on private or public networks. For example, notice that the Core Networking feature is allowed on both private and public networks, while the File and Printer Sharing is only allowed on private networks. We can also see the details of the items in the list by selecting it and then clicking the Details button.



Details

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.

Add a Program

Here we have to browse to the executable of our program and then click the Add button. Notice that we can also choose location types on which this program will be allowed to communicate by clicking on the "Network location types" button.

Network Locations

Many applications will automatically configure proper exceptions in Windows Firewall when we run them. For example, if we enable streaming from Media Player, it will automatically configure firewall settings to allow streaming. The same thing is if we enable Remote Desktop feature from the system properties window. By enabling Remote Desktop feature we actually create an exception in Windows Firewall.

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

Firewall Customization

Note that we can modify settings for each type of network location (private or public). Interesting thing here is that we can block all incoming connections, including those in the list of allowed programs.

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 Firewall will not work.

Firewall Service



Windows Event Log	This service ...	Started	Automatic
Windows Firewall	Windows Fi...	Started	Automatic
Windows Font Cache S...	Optimizes p...	Started	Automatic (D...
Windows Image Acqui	Provides im		Manual

In our case the service is running. If we stop it, we will get a warning that we should turn on our Windows Firewall.



Warning

Remember that with Windows Firewall we can only configure basic firewall settings, and this is enough for most day-to-day users. However, we can't configure exceptions based on ports in Windows Firewall any more. For that we have to use Windows Firewall with Advanced Security.

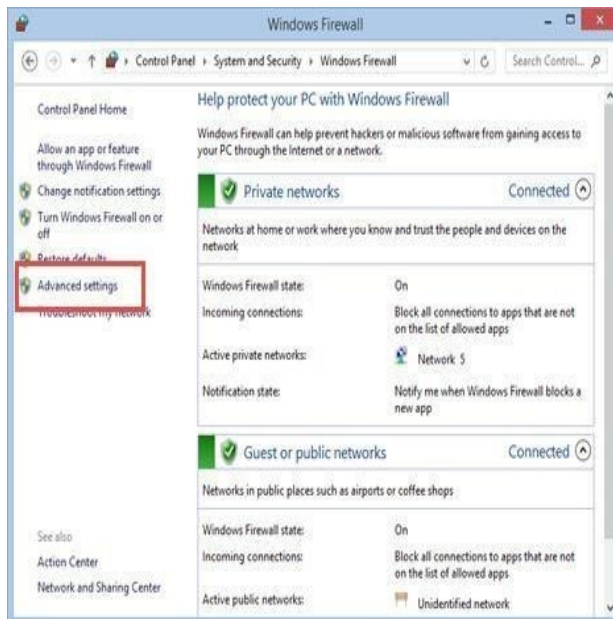
How to Start & Use the Windows Firewall with Advanced Security

The *Windows Firewall with Advanced Security* is a tool which gives you detailed control over the rules that are applied by the *Windows Firewall*. You can view all the rules that are used by the *Windows Firewall*, change their properties, create new rules or disable existing ones. In this tutorial we will share how to open the *Windows Firewall with Advanced Security*, how to find your way around it and talk about the types of rules that are available and what kind of traffic they filter.

How to Access the Windows Firewall with Advanced Security

You have several alternatives to opening the *Windows Firewall with Advanced Security*:

One is to open the standard Windows Firewall window, by going to "*Control Panel -> System and Security -> Windows Firewall*". Then, click or tap *Advanced settings*.



In Windows 7, another method is to search for the word *firewall* in the *Start Menu* search box and click the "*Windows Firewall with Advanced Security*" result.

In Windows 8.1, *Windows Firewall with Advanced Security* is not returned in search results and you need to use the first method shared above for opening it.

The *Windows Firewall with Advanced Security* looks and works the same both in Windows 7 and Windows 8.1. To continue our tutorial, we will use screenshots that were made in Windows 8.1.

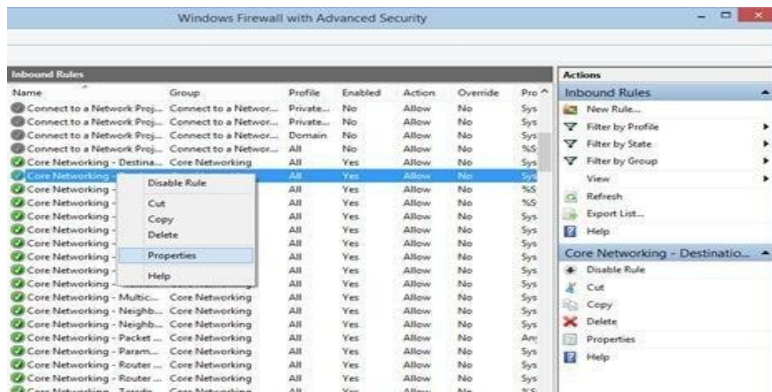


What Are The Inbound & Outbound Rules?

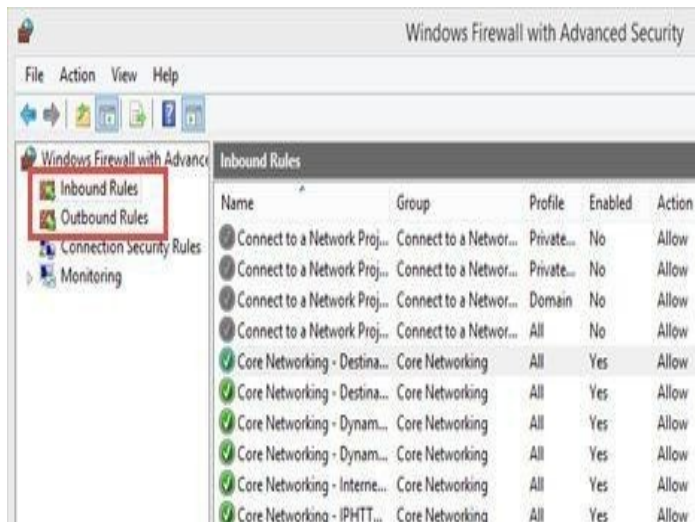
In order to provide the security you need, the *Windows Firewall* has a standard set of inbound and outbound rules, which are enabled depending on the location of the network you are connected to.

Inbound rules are applied to the traffic that is coming from the network and the Internet to your computer or device. Outbound rules apply to the traffic from your computer to the network or the Internet.

These rules can be configured so that they are specific to: computers, users, programs, services, ports or protocols. You can also specify to which type of network adapter (e.g. wireless, cable, virtual private network) or user profile it is applied to.

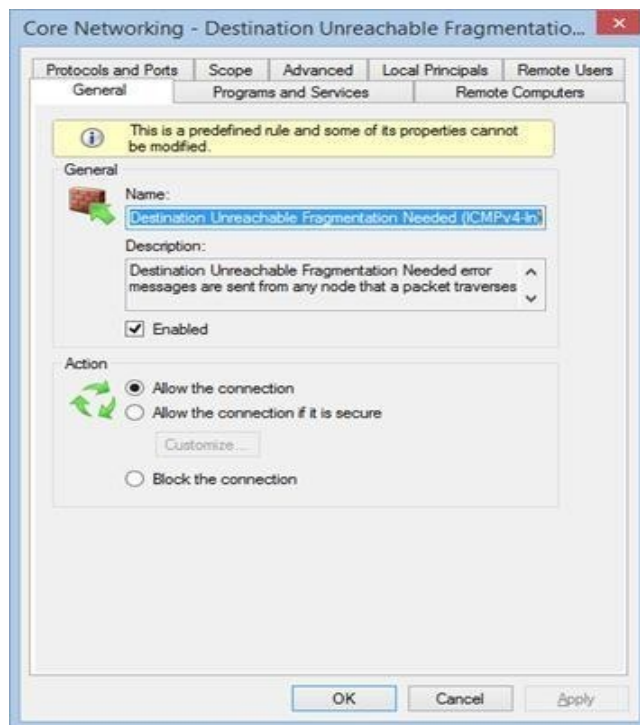


In the *Windows Firewall with Advanced Security*, you can access all rules and edit their properties. All you have to do is click or tap the appropriate unit in the left-side panel.



The rules used by the *Windows Firewall* can be enabled or disabled. The ones which are enabled or active are marked with a green check-box in the *Name* column. The ones that are disabled are marked with a gray check-box.

If you want to know more about a specific rule and learn its properties, right click on it and select *Properties* or select it and press *Properties* in the column on right, which lists the actions that are available for your selection.

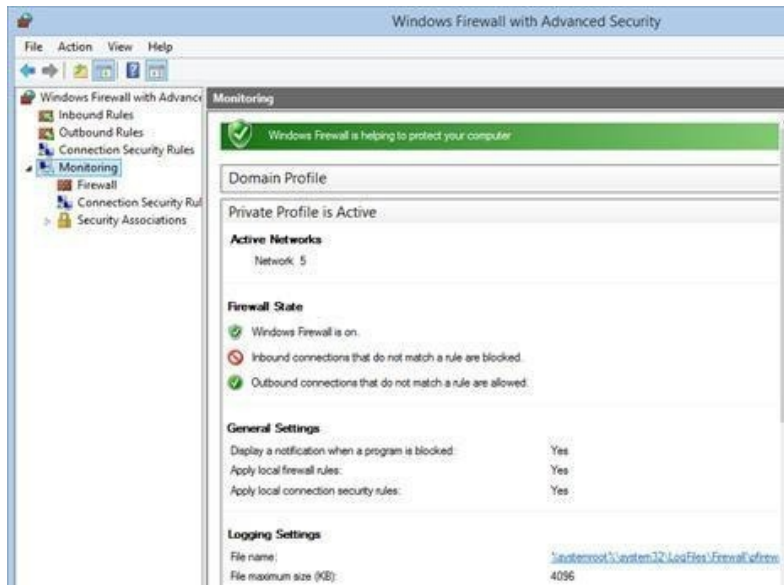


What Are The Connection Security Rules?

Connection security rules are used to secure traffic between two computers while it crosses the network. One example would be a rule which defines that connections between two specific computers must be encrypted.

Unlike the inbound or outbound rules, which are applied only to one computer, connection security rules require that both computers have the same rules defined and enabled.

If you want to see if there are any such rules on your computer, click or tap "*Connection Security Rules*" on the panel on the left. By default, there are no such rules defined on Windows computers and devices. They are generally used in business environments and such rules are set by the network administrator.



What Does the Windows Firewall with Advanced Security Monitor?

The *Windows Firewall with Advanced Security* includes some monitoring features as well. In the *Monitoring* section you can find the following information: the firewall rules that are active (both inbound and outbound), the connection security rules that are active and whether there are any active security associations.

You should note that the *Monitoring* section shows only the active rules for the current network location.

used to determine the operating system running on the host machine. Another feature is "boot-time filtering". This feature ensures that the firewall is working at the same time when the network interface becomes active, which was not the case in previous versions of Windows.

When we first connect to some network, we are prompted to select a network location. This feature is known as Network Location Awareness (NLA). This feature enables us to assign a network profile to the connection based on the location. Different network profiles contain different collections of firewall rules. In Windows 7, different network profiles can be configured on different interfaces. For example, our wired interface can have different profile than our wireless interface. There are three different network profiles available:

- Public
- Home/Work - private network
- Domain - used within a domain

We choose those locations when we connect to a network. We can always change the location in the Network and Sharing Center, in Control Panel. The Domain profile can be automatically assigned by the NLA service when we log on to an Active Directory domain. Note that we must have administrative rights in order to configure firewall in Windows 7.

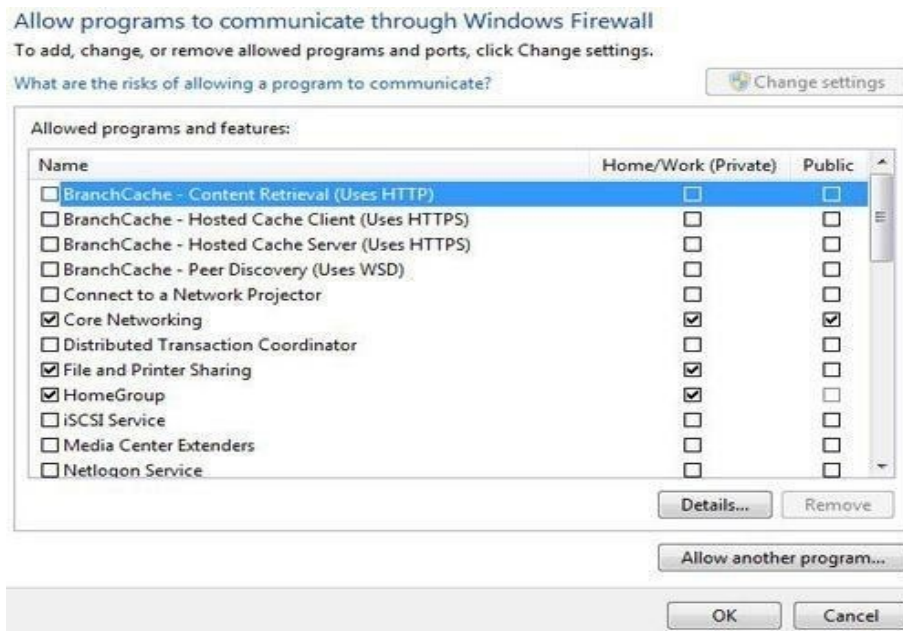
2.1.1 Configuring Windows Firewall

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



Windows Firewall.

By default, Windows Firewall is enabled for both private (home or work) and public networks. 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 through Windows Firewall" option.



Exceptions

To change settings in this window we have to click the "Change settings" button. As you can see, here we have a list of predefined programs and features that can be allowed to communicate on private or public networks. For example, notice that the Core Networking feature is allowed on both private and public networks, while the File and Printer Sharing is only allowed on private networks. We can also see the details of the items in the list by selecting it and then clicking the Details button.

Details

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.

Add a Program

Here we have to browse to the executable of our program and then click the Add button. Notice that we can also choose location types on which this program will be allowed to communicate by clicking on the "Network location types" button.



Network Locations

Many applications will automatically configure proper exceptions in Windows Firewall when we run them. For example, if we enable streaming from Media Player, it will automatically configure firewall settings to allow streaming. The same thing is if we enable Remote Desktop feature from the system properties window. By enabling Remote Desktop feature we actually create an exception in Windows Firewall.

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



Firewall Customization

Note that we can modify settings for each type of network location (private or public). Interesting thing here is that we can block all incoming connections, including those in the list of allowed programs.

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 Firewall will not work.

Windows Event Collector	This service ...	Started	Automatic
Windows Event Log	This service ...	Started	Automatic
Windows Firewall	Windows Fi...	Started	Automatic
Windows Font Cache S...	Optimizes p...	Started	Automatic (D...
Windows Image Acqui	Provides im		Manual

Firewall Service

In our case the service is running. If we stop it, we will get a warning that we should turn on our Windows Firewall.



Warning

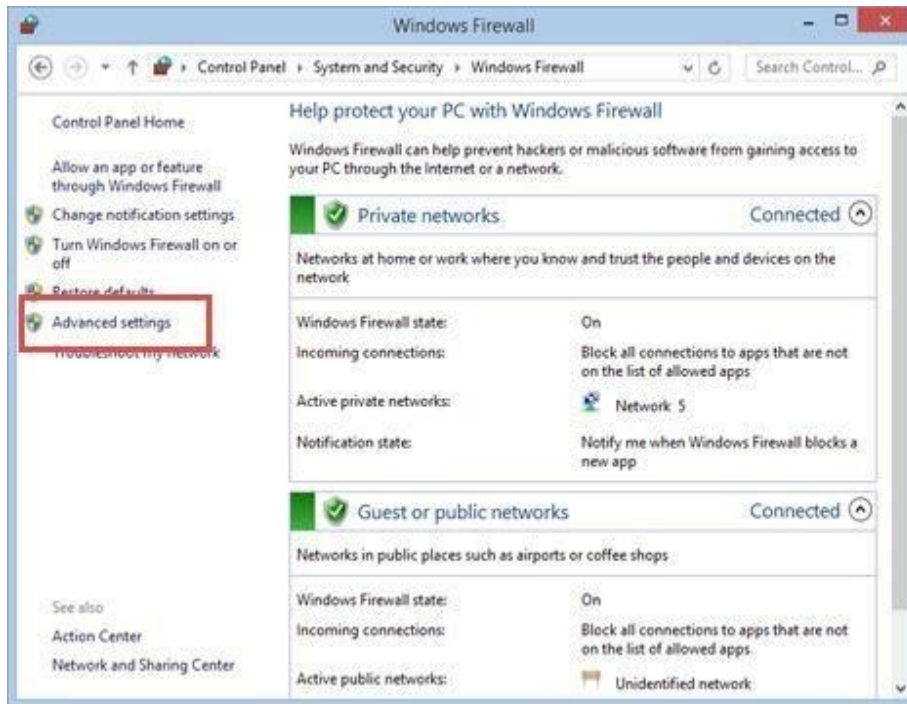
Remember that with Windows Firewall we can only configure basic firewall settings, and this is enough for most day-to-day users. However, we can't configure exceptions based on ports in Windows Firewall any more. For that we have to use Windows Firewall with Advanced Security.

How to Start & Use the Windows Firewall with Advanced Security

The *Windows Firewall with Advanced Security* is a tool which gives you detailed control over the rules that are applied by the *Windows Firewall*. You can view all the rules that are used by the *Windows Firewall*, change their properties, create new rules or disable existing ones. In this tutorial we will share how to open the *Windows Firewall with Advanced Security*, how to find your way around it and talk about the types of rules that are available and what kind of traffic they filter. How to Access the Windows Firewall with Advanced Security

You have several alternatives to opening the *Windows Firewall with Advanced Security*:

One is to open the standard Windows Firewall window, by going to "*Control Panel -> System and Security -> Windows Firewall*". Then, click or tap *Advanced settings*.

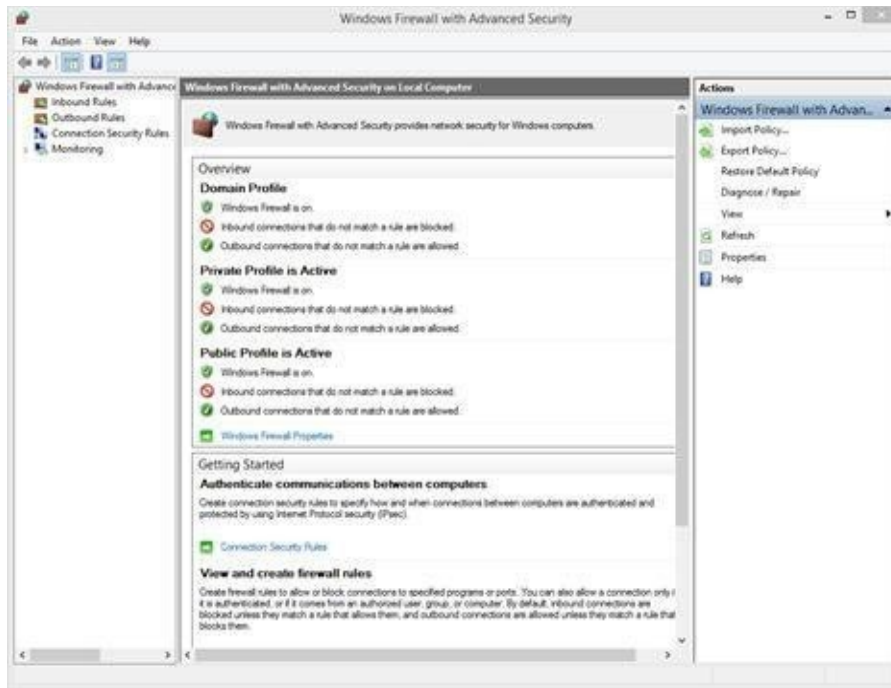


In Windows 7, another method is to search for the word *firewall* in the *Start Menu* search box and click the "*Windows Firewall with Advanced Security*" result.



In Windows 8.1, *Windows Firewall with Advanced Security* is not returned in search results and you need to use the first method shared above for opening it.

The *Windows Firewall with Advanced Security* looks and works the same both in Windows 7 and Windows 8.1. To continue our tutorial, we will use screenshots that were made in Windows 8.1.



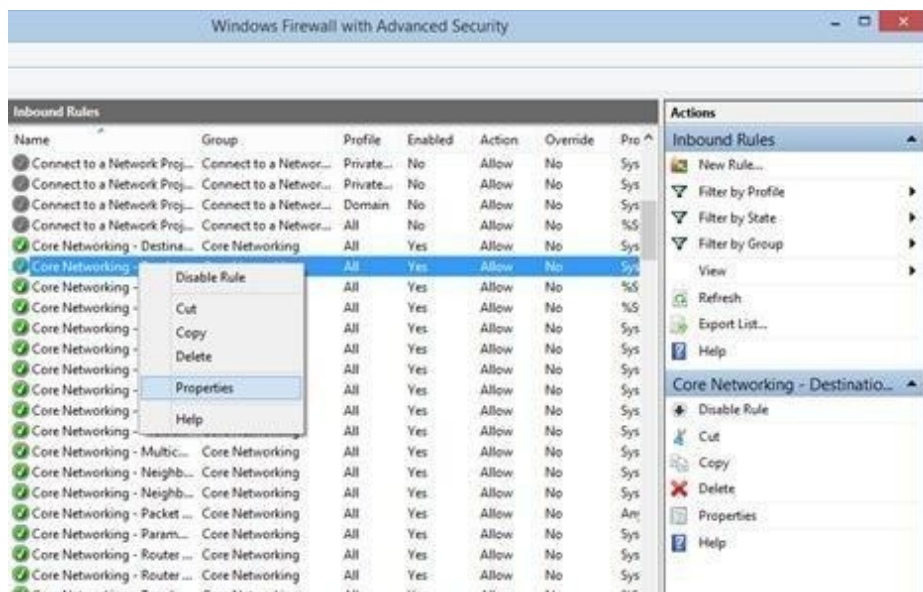
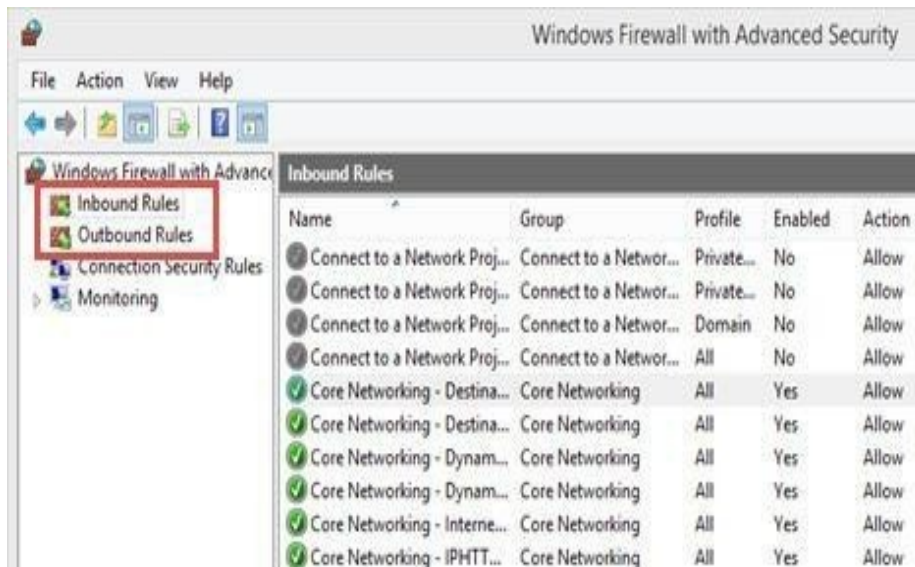
What Are The Inbound & Outbound Rules?

In order to provide the security you need, the *Windows Firewall* has a standard set of inbound and outbound rules, which are enabled depending on the location of the network you are connected to.

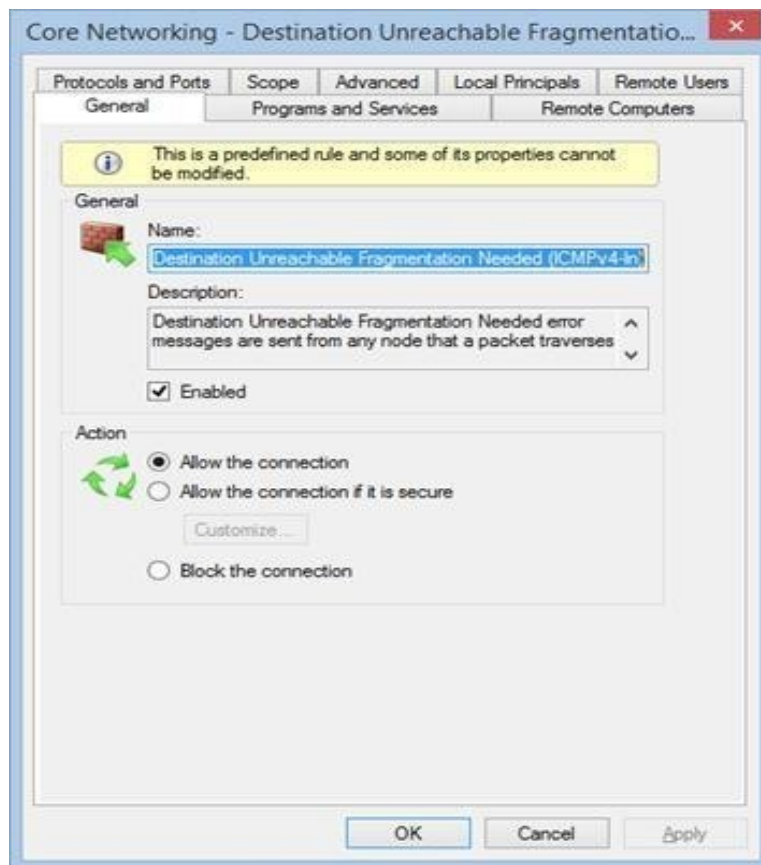
Inbound rules are applied to the traffic that is coming from the network and the Internet to your computer or device. Outbound rules apply to the traffic from your computer to the network or the Internet.

These rules can be configured so that they are specific to: computers, users, programs, services, ports or protocols. You can also specify to which type of network adapter (e.g. wireless, cable, virtual private network) or user profile it is applied to.

In the *Windows Firewall with Advanced Security*, you can access all rules and edit their properties. All you have to do is click or tap the appropriate unit in the left-side panel.



The rules used by the *Windows Firewall* can be enabled or disabled. The ones which are enabled or active are marked with a green check-box in the *Name* column. The ones that are disabled are marked with a gray check-box. If you want to know more about a specific rule and learn its properties, right click on it and select *Properties* or select it and press *Properties* in the column on right, which lists the actions that are available for your selection.



2.1.1.1 What Are The Connection Security Rules?

Connection security rules are used to secure traffic between two computers while it crosses the network. One example would be a rule which defines that connections between two specific computers must be encrypted.

Unlike the inbound or outbound rules, which are applied only to one computer, connection security rules require that both computers have the same rules defined and enabled.

If you want to see if there are any such rules on your computer, click or tap *"Connection Security Rules"* on the panel on the left. By default, there are no such rules defined on Windows computers and devices. They are generally used in business environments and such rules are set by the network administrator.



2.1.1.2 What Does the Windows Firewall with Advanced Security Monitor?

The *Windows Firewall with Advanced Security* includes some monitoring features as well. In the *Monitoring* section you can find the following information: the firewall rules that are active (both inbound and outbound), the connection security rules that are active and whether there are any active security associations.



You should note that the *Monitoring* section shows only the active rules for the current network location.

Result:

study of the features of firewall in providing network security and to set Firewall Security in windows .