

**Course Code :** CSL604**Course Name :** System Security Lab**Class :** TE-CO**Batch :** Computer Engineering**Roll no :** 18CO63**Name :** SHAIKH TAUSEEF MUSHTAQUE ALI**Experiment : 01**

Aim : Design and Implementation of a product cipher using Substitution and Transposition ciphers

Code :

import string

```
k=int(input("ENTER A KEY VALUE:"))
d=str(input("ENTER A STRING: "))
ct = []
alphabets = string.ascii_uppercase
for j in d:
    b=j.upper()
    if b in alphabets and j.islower():
        e=(alphabets.index(b)+k)%26
        ct.append(alphabets[e].lower())
    elif b in alphabets and j.isupper():
        a=(alphabets.index(b)+k)%26
        ct.append(alphabets[a].upper())
    else:
        ct.append(" ")
matrix = [[False for i in range(len(ct))]]
for j in range(k)]  
  
print("Cipher Text: ",*ct)
j=0
for i in range(len(ct)):
    matrix[j][i]=ct[i]
    if j == k - 1:
        flag = False
    elif j == 0:
        flag = True
    if flag == True:
        j = j + 1
    else:
        j = j - 1
answer=[]
for key in range(k):
    for text in range(len(ct)):
        if matrix[key][text]!=False:
            answer.append(matrix[key][text])
```



ANJUMAN-I-ISLAM'S KALSEKAR TECHNICAL CAMPUS

School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)

```
print("Encrypted Text: ", *answer)
```

Output :

```
[mastmac@MASTMACs-Mac-mini code % python3 cipher.py
ENTER A KEY VALUE:4
ENTER A STRING: TAUSEEF
Cipher Text: X E Y W I I J
Encrypted Text: X J E I Y I W
mastmac@MASTMACs-Mac-mini code %]
```

Conclusion:

A product cipher combines two or more transformations in a manner intending that the resulting cipher is more secure than the individual components to make it resistant to cryptanalysis. Implemented product cipher using Substitution and Transposition ciphers



School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)

Course Code : CSL604**Course Name : System Security Lab****Class : TE-CO****Batch : Computer Engineering****Roll no : 18CO63****Name : SHAIKH TAUSEEF MUSHTAQUE ALI****Experiment : 02**

Aim : Implementation and analysis of Playfair cipher.

Code :

```
print("\n\t\t PLAYFAIR CIPHER \n")
```

```
key=input("ENTER KEY: ")  
key=key.replace(" ", "")  
key=key.upper()  
def matrix(x,y,initial):  
    return [[initial for i in range(x)] for j in range(y)]
```

```
result=list()  
for c in key:  
    if c not in result:  
        if c=='J':  
            result.append('I')  
        else:  
            result.append(c)  
flag=0  
for i in range(65,91):  
    if chr(i) not in result:  
        if i==73 and chr(74) not in result:  
            result.append("I")  
            flag=1  
        elif flag==0 and i==73 or i==74:  
            pass  
        else:  
            result.append(chr(i))
```

```
k=0  
my_matrix=matrix(5,5,0)  
for i in range(0,5):  
    for j in range(0,5):  
        my_matrix[i][j]=result[k]  
    k+=1
```

```
def locindex(c):  
    loc=list()  
    if c=='J':  
        c='I'
```



```
for i,j in enumerate(my_matrix):
    for k,l in enumerate(j):
        if c==l:
            loc.append(i)
            loc.append(k)
            return loc

def encrypt():
    msg=str(input("\nENTER MESSAGE: "))
    msg=msg.upper()
    msg=msg.replace(" ", "")
    i=0
    for s in range(0,len(msg)+1,2):
        if s<len(msg)-1:
            if msg[s]==msg[s+1]:
                msg=msg[:s+1]+X+msg[s+1:]
    if len(msg)%2!=0:
        msg=msg[:]+'X'
    print("\nCIPHER TEXT: ",end=' ')
    while i<len(msg):
        loc=list()
        loc=locindex(msg[i])
        loc1=list()
        loc1=locindex(msg[i+1])
        if loc[1]==loc1[1]:
            print("{} {}".format(my_matrix[(loc[0]+1)%5][loc[1]],my_matrix[(loc1[0]+1)%5][loc1[1]]),end=' ')
            elif loc[0]==loc1[0]:
                print("{} {}".format(my_matrix[loc[0]][(loc[1]+1)%5],my_matrix[loc1[0]][(loc1[1]+1)%5]),end=' ')
            else:
                print("{} {}".format(my_matrix[loc[0]][loc1[1]],my_matrix[loc1[0]][loc[1]]),end=' ')
            i=i+2
        print("")

def decrypt():
    msg=str(input("\nEnter CIPHER TEXT: "))
    msg=msg.upper()
    msg=msg.replace(" ", "")
    print("\nPLAIN TEXT: ",end=' ')
    i=0
    while i<len(msg):
        loc=list()
        loc=locindex(msg[i])
        loc1=list()
```



School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)

```
loc1=locindex(msg[i+1])
if loc[1]==loc1[1]:
    print("{{}{}".format(my_matrix[(loc[0]-1)%5][loc[1]],my_matrix[(loc1[0]-1)%5][loc1[1]]),end=' ')
elif loc[0]==loc1[0]:
    print("{{}{}".format(my_matrix[loc[0]][(loc[1]-1)%5],my_matrix[loc1[0]][(loc1[1]-1)%5]),end=' ')
else:
    print("{{}{}".format(my_matrix[loc[0]][loc1[1]],my_matrix[loc1[0]][loc[1]]),end=' ')
i=i+2

print("")

while(1):
    print("\nCHOOSE AN OPTION: \n")
    choice=int(input(" 1.ENCRYPTION \n 2.DECRYPTION \n 3.EXIT \n\n"))
    if choice==1:
        encrypt()
    elif choice==2:
        decrypt()
    elif choice==3:
        print("\n EXITING PLAYFAIR CIPHER... \n")
        exit()
    else:
        print("\nINVALID OPTION! CHOOSE CORRECT OPTION \n")
```



School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)

Output:

```
EXP02 — zsh — 66x41
mastmac@MASTMACs-Mac-mini EXP02 % python3 EXP02_PLAYFAIRCIPHER.py

PLAYFAIR CIPHER

ENTER KEY: 3

CHOOSE AN OPTION:

1. ENCRYPTION
2. DECRYPTION
3. EXIT

1

ENTER MESSAGE: hello

CIPHER TEXT: IF NV MK

CHOOSE AN OPTION:

1. ENCRYPTION
2. DECRYPTION
3. EXIT

2

ENTER CIPHER TEXT: ifnvmk

PLAIN TEXT: HE LX LO

CHOOSE AN OPTION:

1. ENCRYPTION
2. DECRYPTION
3. EXIT

3

EXITING PLAYFAIR CIPHER...

mastmac@MASTMACs-Mac-mini EXP02 %
```

Conclusion:

A playfair cipher unlike traditional cipher we encrypt a pair of alphabets(digraphs) instead of a single alphabet.
--

**Course Code : CSL604****Course Name : System Security Lab****Class : TE-CO****Batch : Computer Engineering****Roll no : 18CO63****Name : SHAIKH TAUSEEF MUSHTAQUE ALI****Experiment : 03****Aim :** Implementation and analysis of RSA cryptosystem and Digital signature scheme using RSA.**Code :**

```
import hashlib
import random
import math
print(".....RSA Encryption Technique.....")
pt=input("Enter the text to be encrypted:")
code = hashlib.sha1(pt.encode())
code = code.hexdigest()
plain = pt.replace(" ","")
if plain.isalpha():
    pta=pt.lower()
    ptn=[ord(i)%97 for i in pta]
elif pt.isdigit():
    ptn=int(pt)

#n=int(input("Enter a composite prime number(n)"))
primes = []
for x in range(1,1001):
    for y in range(2,x):
        if x%y==0:
            break
    else:
        primes.append(x)

p, q = random.choice(primes), random.choice(primes)
phi=(p-1)*(q-1)
n = p * q
e= 0
for i in range(2,26):
    if math.gcd(i,phi)==1:
        e=i
        break
def modInverse(a,m):
    for x in range(1, m):
        if (((a%m) * (x%m)) % m == 1):
            return x
    return -1
d = modInverse(e,phi)
print(d)
if d!= -1:
    if plain.isalpha():
```



School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)

```
ct= [(i**e)%n for i in ptn]
print("Encrypted value::",*ct)
dt= [(i**d)%n for i in ct]
else:
    ct = (ptn**e) % n
    dt = (ct**d) % n
else:
    print("Encryption is not Possible!!!!!!")
if plain.isalpha() and pt.islower():
    dt = "".join([chr(int(i)+97) for i in dt]).replace("\x81"," ")
elif plain.isalpha() and pt.isupper():
    dt = "".join([chr(65+int(i)) for i in dt]).replace("a"," ")
else:
    dt= str(dt)

hashvalue = hashlib.sha1(dt.encode())
hashvalue = hashvalue.hexdigest()
if code==hashvalue:
    print("Message Integrity is maintained!!!")
    print("Decrypted value::",dt)
else:
    print("Corrupted message!!!")
```

Output:

```
mastmac@MASTMACs-Mac-mini code % python3 en.py
.....RSA Encryption Technique.....
Enter the text to be encrypted:TAUSEEF
251597
Encrypted value:: 372884 0 255499 206996 1024 1024 3125
Message Integrity is maintained!!!
Decrypted value:: TAUSEEF
mastmac@MASTMACs-Mac-mini code %
```

Conclusion:

Implemented and analyzed RSA cryptosystem and Digital signature scheme using RSA.



School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)

Course Code : CSL604

Course Name : System Security Lab

Class : TE-CO

Batch : Computer Engineering

Roll no : 18CO63

Name : SHAIKH TAUSEEF MUSHTAQUE ALI

Experiment : 04

Aim : For varying message sizes, test integrity of message using MD-5, SHA-1, and analyse the performance of the two protocols. Use crypt APIs.

Output :

```
users@Tauseef:~$ touch experi3.txt
```

```
users@Tauseef:~$ echo "Tauseef" >> experi3.txt
```

```
users@Tauseef:~$ md5sum experi3.txt  
d1e87d22dd21e5a63a7578e055e750f8 experi3.txt
```

```
users@Tauseef:~$ sha1sum experi3.txt  
2dd4a57416b419a354798122e23e34282435dc43 experi3.txt
```

```
users@Tauseef:~$ md5sum experi3.txt > experi3.md5 && md5sum -c experi3.md5  
experi3.txt: OK
```

```
users@Tauseef:~$ sha1sum experi3.txt > experi3.sha1 && sha1sum -c experi3.sha1  
experi3.txt: OK
```

```
users@Tauseef:~$ echo "Pathan" >> experi3.txt
```

```
users@Tauseef:~$ md5sum -c experi3.md5  
experi3.txt: FAILED  
md5sum: WARNING: 1 computed checksum did NOT match
```

```
users@Tauseef:~$ sha1sum -c experi3.sha1  
experi3.txt: FAILED  
sha1sum: WARNING: 1 computed checksum did NOT match
```

Conclusion:

Tested message integrity using MD-5, SHA-1, and analyzed the performance of the two protocols using crypt APIs for varying message sizes.

**Course Code : CSL604****Course Name : System Security Lab****Class : TE-CO****Batch : Computer Engineering****Roll no : 18CO63****Name : SHAIKH TAUSEEF MUSHTAQUE ALI****Experiment: 05**

Aim : Study the use of network reconnaissance tools like WHOIS, dig, traceroute, nslookup to gather information about networks and domain registrars.

Output:

```
mastmac@MASTMACs-Mac-mini ~ % traceroute -m 10 phalicious-shield-aboutus.netlify.app
traceroute: Warning: phalicious-shield-aboutus.netlify.app has multiple addresses; using 52.220.193.16
traceroute to phalicious-shield-aboutus.netlify.app (52.220.193.16), 10 hops max, 52 byte packets
 1  192.168.0.1 (192.168.0.1)  1.031 ms  0.322 ms  0.405 ms
 2  xiaqiang (192.168.31.1)  0.733 ms  0.699 ms  0.629 ms
 3  1.186.179.1.dvois.com (1.186.179.1)  1.489 ms  2.412 ms  1.967 ms
 4  114.79.129.97.dvois.com (114.79.129.97)  1.912 ms  1.498 ms  2.022 ms
 5  * * *
 6  * * *
 7  * * *
 8  * * *
 9  * * *
10  * * *
mastmac@MASTMACs-Mac-mini ~ %
```

```
mastmac@MASTMACs-Mac-mini ~ % nslookup phalicious-shield-aboutus.netlify.ap
p
Server:      192.168.0.1
Address:     192.168.0.1#53

Non-authoritative answer:
Name:  phalicious-shield-aboutus.netlify.app
Address: 104.248.158.121
Name:  phalicious-shield-aboutus.netlify.app
Address: 167.99.78.230
```



School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)

```
● ● ● mastmac ~ % whois phalicious-shield-aboutus.netlify.app
% IANA WHOIS server
% for more information on IANA, visit http://www.iana.org
% This query returned 1 object

refer:      whois.nic.google

domain:     APP

organisation: Charleston Road Registry Inc.
address:    1600 Amphitheatre Parkway Mountain View, CA 94043
address:    United States

contact:    administrative
name:       TLD Admin
organisation: Google Inc.
address:    111 8th Avenue
address:    New York, NY 10011
address:    United States
phone:      +1 404 978 8419
fax-no:     +1 650 492 5631
e-mail:    iana-contact@google.com

contact:    technical
name:       TLD Engineering
organisation: Google Inc
address:    76 9th Avenue, 4th Floor
address:    New York, NY 10011
address:    United States
phone:      +1 404 978 8419
fax-no:     +1 650 492 5631
e-mail:    crr-tech@google.com

nserver:    NS-TLD1.CHARLESTONROADREGISTRY.COM 2001:4860:4802:32:0:0:69 216.239.32.105
nserver:    NS-TLD2.CHARLESTONROADREGISTRY.COM 2001:4860:4802:34:0:0:69 216.239.34.105
nserver:    NS-TLD3.CHARLESTONROADREGISTRY.COM 2001:4860:4802:36:0:0:69 216.239.36.105
nserver:    NS-TLD4.CHARLESTONROADREGISTRY.COM 2001:4860:4802:38:0:0:69 216.239.38.105
nserver:    NS-TLD5.CHARLESTONROADREGISTRY.COM 2001:4860:4805:0:0:0:69 216.239.60.105
ds-rdata:   23684 8 2 3a5cc8a31e02c94aba6461912fabb7e9f5e34957bb6114a55a864d96aec31836

whois:      whois.nic.google

status:     ACTIVE
remarks:   Registration information: https://www.registry.google

created:   2015-06-25
changed:   2020-04-20
source:    IANA

# whois.nic.google

Nameserver not found.
>>> Last update of WHOIS database: 2021-05-19T12:08:38Z <<<
```



```
mastmac@MASTMACs-Mac-mini ~ % dig phalicious-shield-aboutus.netlify.app

; <>> DiG 9.10.6 <>> phalicious-shield-aboutus.netlify.app
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 34982
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 4, ADDITIONAL: 9

;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;phalicious-shield-aboutus.netlify.app. IN A

;; ANSWER SECTION:
phalicious-shield-aboutus.netlify.app. 20 IN A 167.99.78.230
phalicious-shield-aboutus.netlify.app. 20 IN A 206.189.46.168

;; AUTHORITY SECTION:
netlify.app. 4673 IN NS dns2.p01.nsone.net.
netlify.app. 4673 IN NS dns3.p01.nsone.net.
netlify.app. 4673 IN NS dns4.p01.nsone.net.
netlify.app. 4673 IN NS dns1.p01.nsone.net.

;; ADDITIONAL SECTION:
dns1.p01.nsone.net. 57656 IN A 198.51.44.1
dns2.p01.nsone.net. 94896 IN A 198.51.45.1
dns3.p01.nsone.net. 94896 IN A 198.51.44.65
dns4.p01.nsone.net. 94896 IN A 198.51.45.65
dns1.p01.nsone.net. 94896 IN AAAA 2620:4d:4000:6259:7:1:0:1
dns2.p01.nsone.net. 94896 IN AAAA 2a00:edc0:6259:7:1::2
dns3.p01.nsone.net. 94896 IN AAAA 2620:4d:4000:6259:7:1:0:3
dns4.p01.nsone.net. 94896 IN AAAA 2a00:edc0:6259:7:1::4

;; Query time: 32 msec
;; SERVER: 192.168.0.1#53(192.168.0.1)
;; WHEN: Wed May 19 17:40:40 IST 2021
;; MSG SIZE rcvd: 363
```

Conclusion:

Network reconnaissance tools like WHOIS, dig, traceroute and nslookup are studied to gather information about networks and domain registrars.



Course Code : CSL604

Course Name : System Security Lab

Class : TE-CO

Batch : Computer Engineering

Roll no : 18CO63

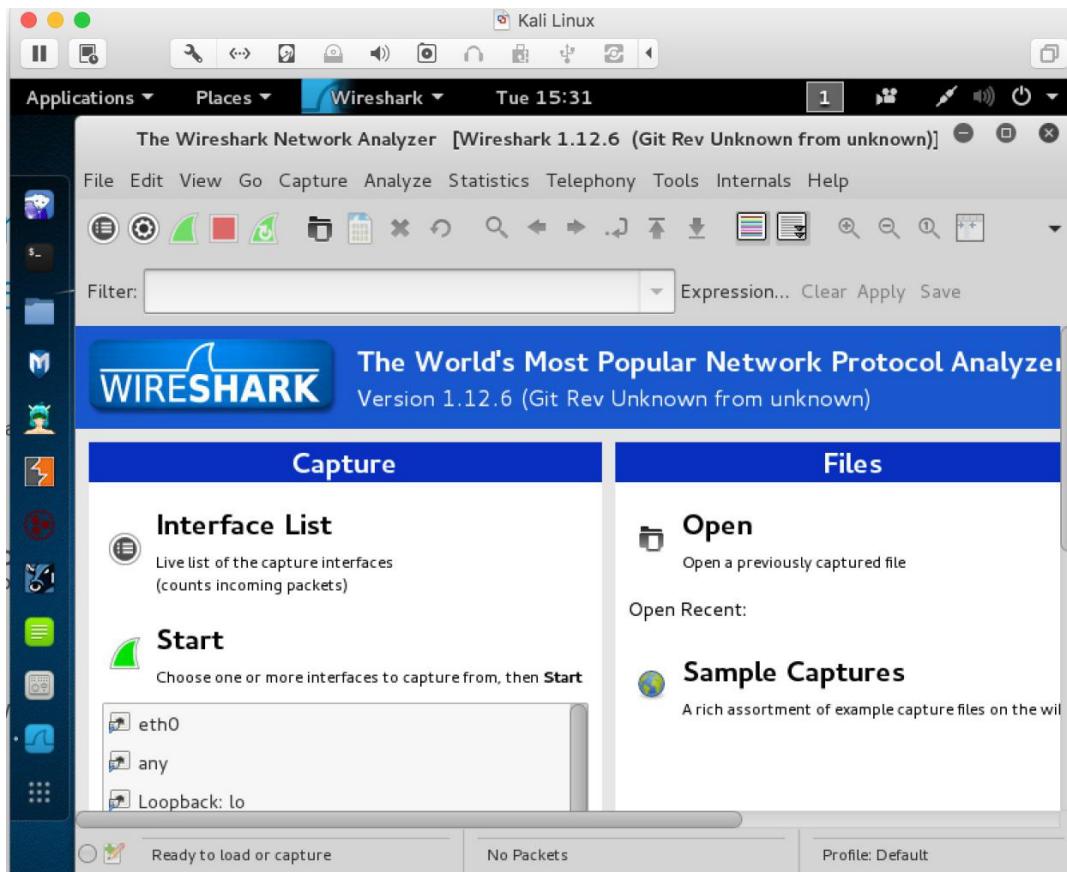
Name : SHAIKH TAUSEEF MUSHTAQUE ALI

Experiment : 06

Aim : Study of packet sniffer tools : wireshark, 1. Download and install wireshark and capture icmp, tcp, and http packets in promiscuous mode.

Theory:**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



ANJUMAN-I-ISLAM'S KALSEKAR TECHNICAL CAMPUS

School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)

Background

TCP/IP Network Stack

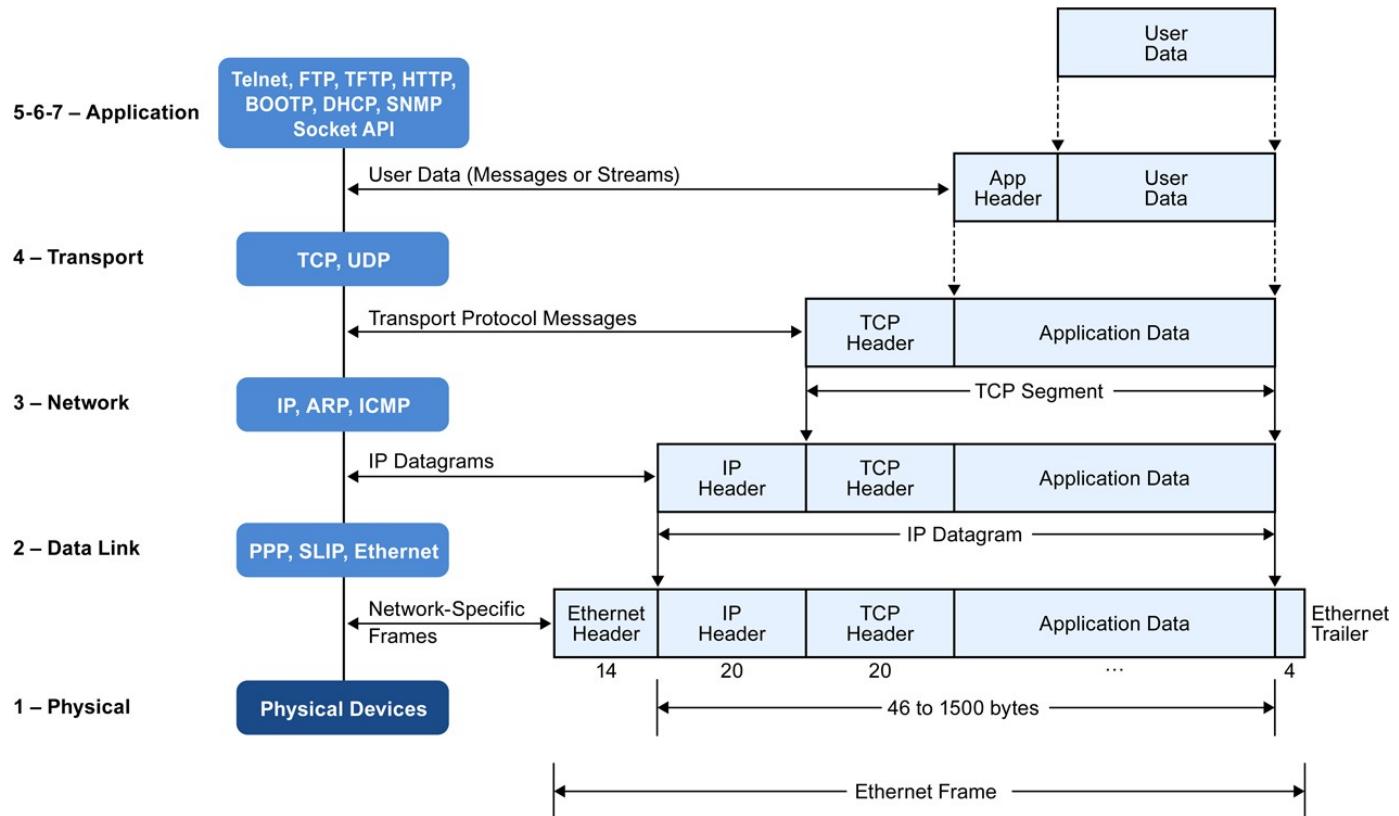


Figure 2: Encapsulation of Data in the TCP/IP Network Stack

In the CSC 4190 Introduction to Computer Networking, 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



ANJUMAN-I-ISLAM'S KALSEKAR TECHNICAL CAMPUS

School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)

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. A 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.

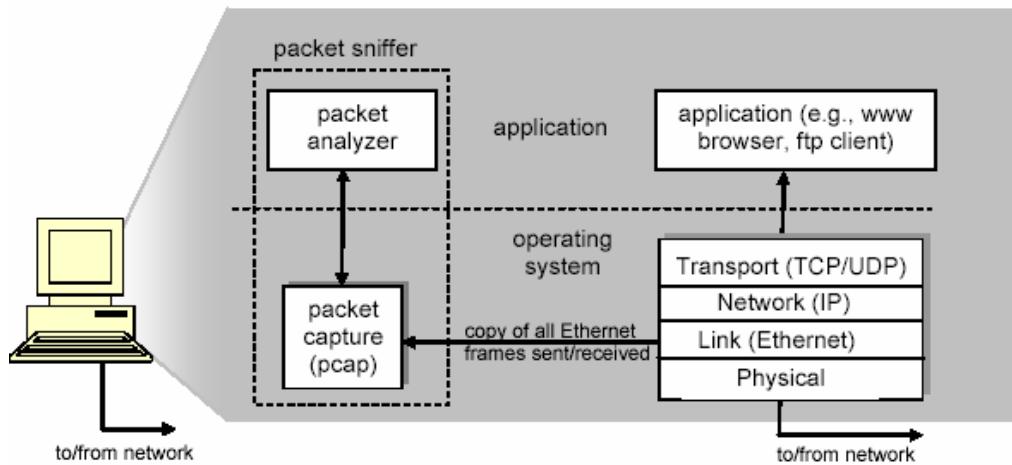


Figure 3: Packet Sniffer Structure

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,



ANJUMAN-I-ISLAM'S KALSEKAR TECHNICAL CAMPUS

School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)

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.



ANJUMAN-I-ISLAM'S KALSEKAR TECHNICAL CAMPUS

School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)

Getting Wireshark

The Kali 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>

The screenshot shows a web browser displaying the official Wireshark download page at <https://www.wireshark.org/download.html>. The page features the Wireshark logo and navigation links. The main content area is titled "Download Wireshark" and indicates that the current stable release is 2.0.1. It lists several download options under the "Stable Release (2.0.1)" section, including Windows installers (64-bit and 32-bit), a Windows PortableApps® (32-bit) option, and OS X download links for Intel 64-bit (.dmg) and 32-bit (.dmg) versions, along with a Source Code link. Below this, there are sections for "Old Stable Release (1.12.9)" and "Documentation". At the bottom, a "Having Problems?" link is available.

Figure 4: 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.



ANJUMAN-I-ISLAM'S KALSEKAR TECHNICAL CAMPUS

School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)

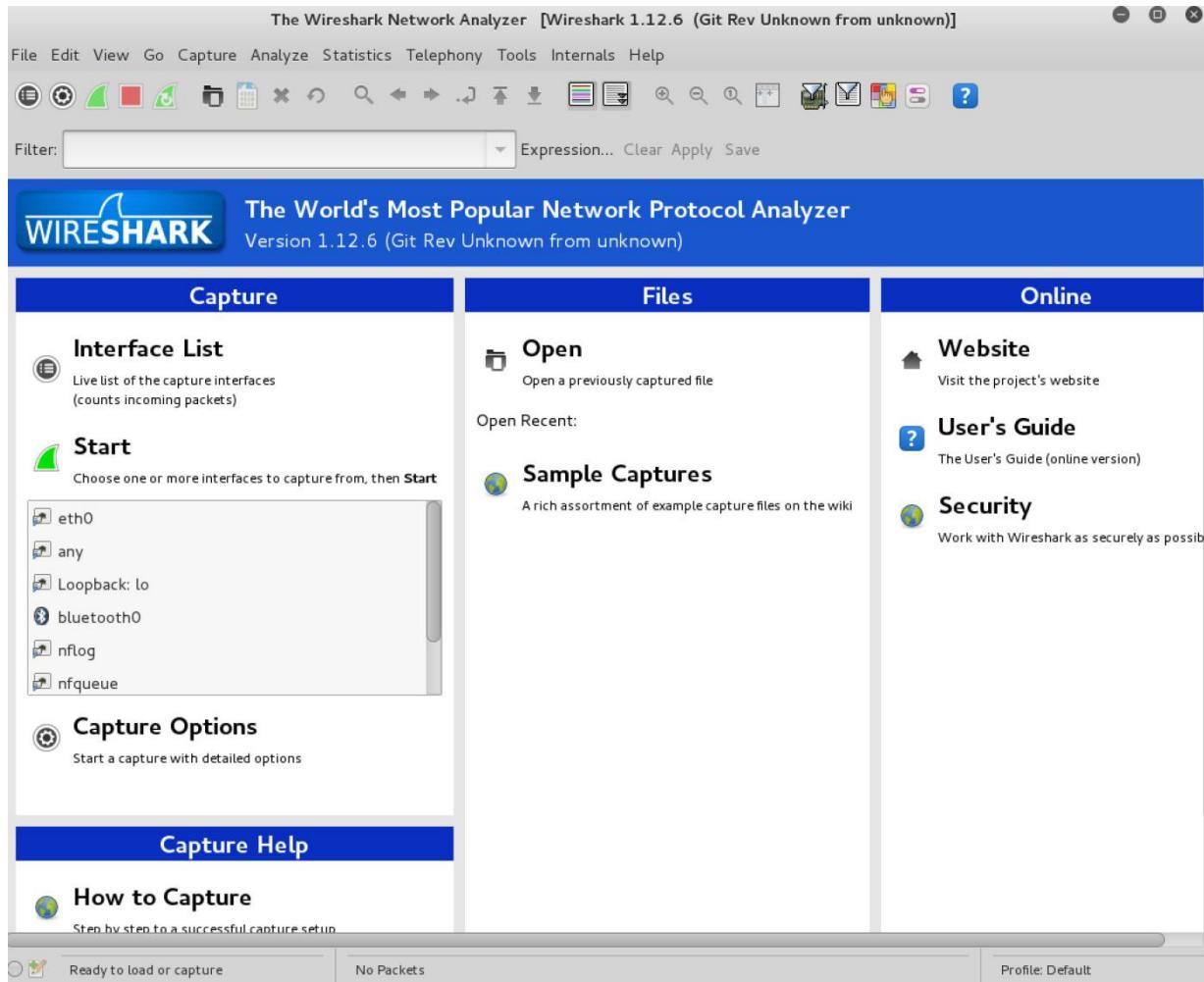


Figure 5: Initial Graphic User Interface of Wireshark

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.



ANJUMAN-I-ISLAM'S KALSEKAR TECHNICAL CAMPUS

School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)

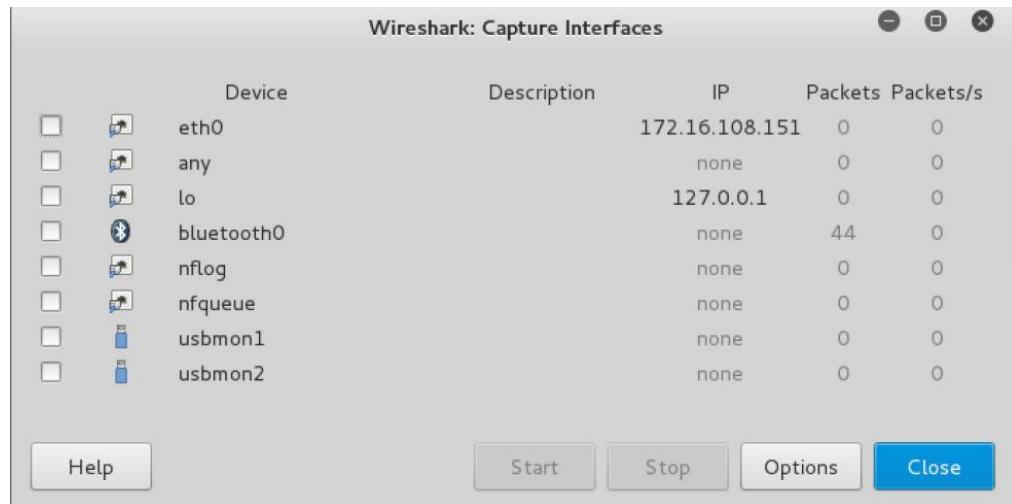


Figure 6: Capture Interfaces in Wireshark

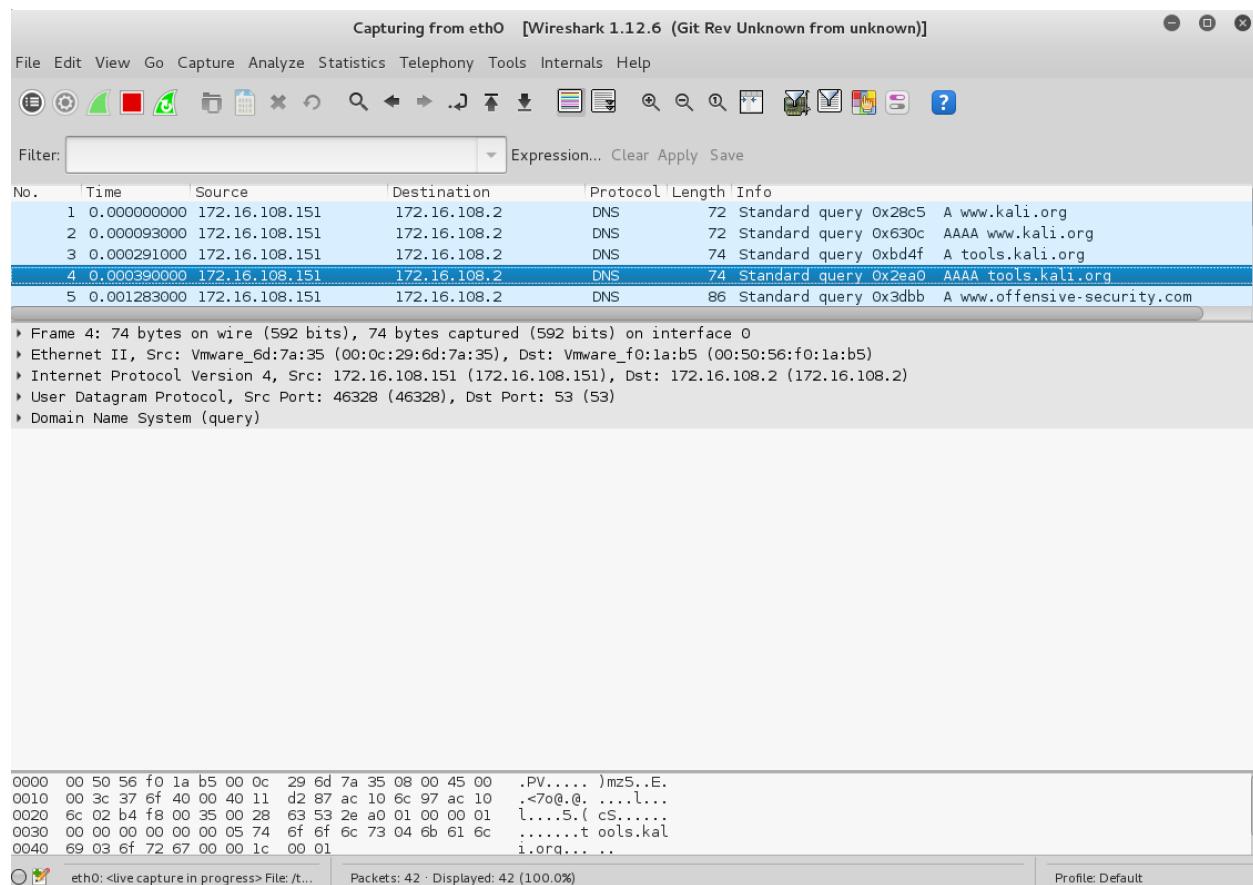


Figure 7: Capturing Packets in Wireshark

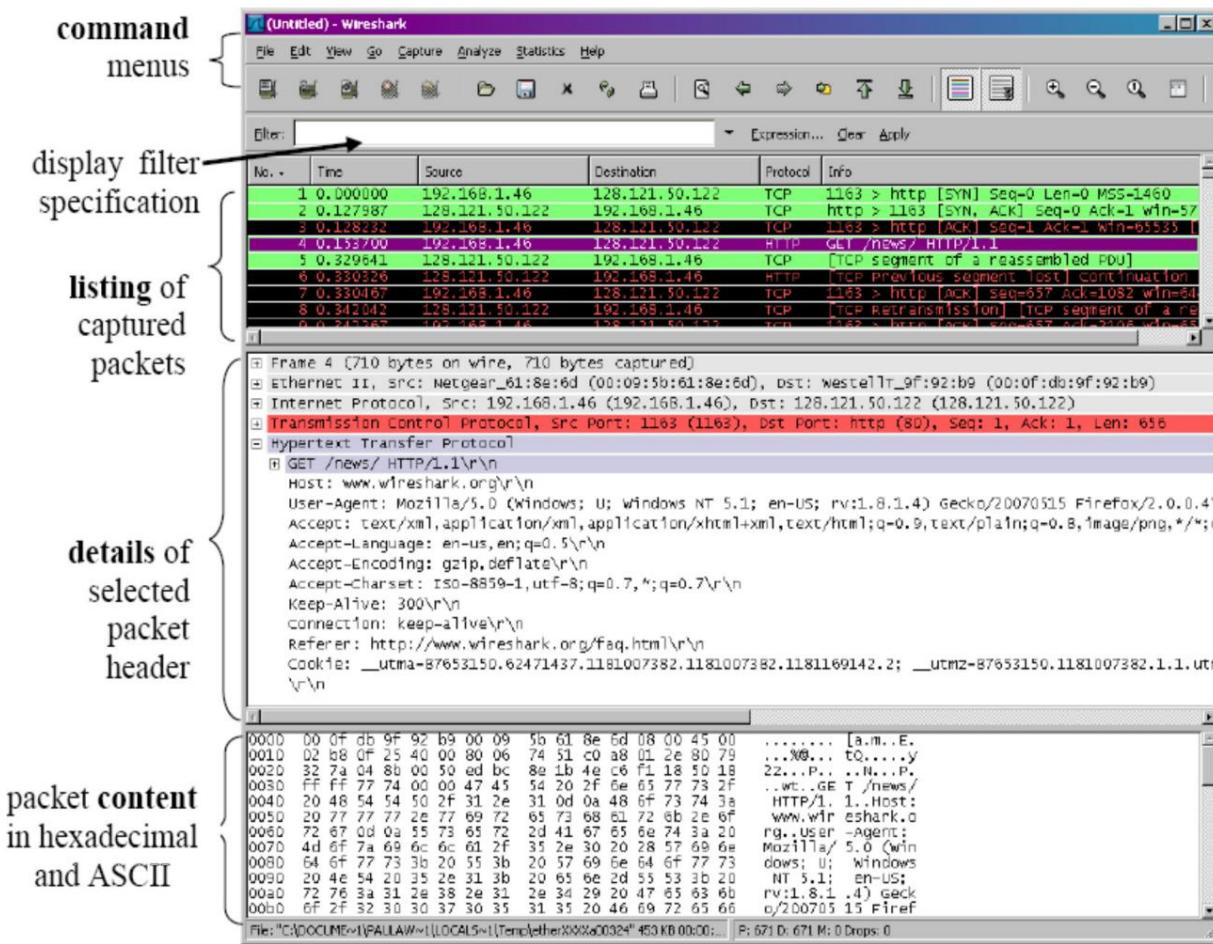


Figure 8: Wireshark Graphical User Interface on Microsoft Windows

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



ANJUMAN-I-ISLAM'S KALSEKAR TECHNICAL CAMPUS

School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)

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.



ANJUMAN-I-ISLAM'S KALSEKAR TECHNICAL CAMPUS

School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)

Capturing from eth0 [Wireshark 1.12.6 (Git Rev Unknown from unknown)]

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

Filter: Expression... Clear Apply Save

No.	Time	Source	Destination	Protocol	Length	Info
7833	131.7408250	172.16.108.152	192.122.185.45	TLSv1.2	95	Application Data
7834	131.7410640	192.122.185.45	172.16.108.152	TCP	60	443>45312 [ACK] Seq=202574 Ack=5826 Win=64240 Len=0
7835	131.7495260	192.122.185.45	172.16.108.152	TLSv1.2	95	Application Data
7836	131.7495330	172.16.108.152	192.122.185.45	TCP	54	45312>443 [ACK] Seq=5826 Ack=202615 Win=65535 Len=0
7837	135.2604700	172.16.108.152	50.31.164.175	TCP	54	[TCP Keep-Alive] 33587>443 [ACK] Seq=2186 Ack=3183 Win=42408 Len=0
7838	135.2606360	50.31.164.175	172.16.108.152	TCP	60	[TCP Keep-Alive ACK] 443>33587 [ACK] Seq=3183 Ack=2187 Win=64240 Len=0
7839	135.6393630	172.16.108.1	172.16.108.255	DB-LSP-D	223	Dropbox LAN sync Discovery Protocol
7840	145.2774900	172.16.108.152	50.31.164.175	TCP	54	[TCP Keep-Alive] 33587>443 [ACK] Seq=2186 Ack=3183 Win=42408 Len=0
7841	145.2776770	50.31.164.175	172.16.108.152	TCP	60	[TCP Keep-Alive ACK] 443>33587 [ACK] Seq=3183 Ack=2187 Win=64240 Len=0
7842	146.3646130	50.31.164.175	172.16.108.152	TCP	60	443>33587 [RST, ACK] Seq=3183 Ack=2187 Win=64240 Len=0
7843	165.6896550	172.16.108.1	172.16.108.255	DB-LSP-D	223	Dropbox LAN sync Discovery Protocol

* Frame 7119: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
* Ethernet II, Src: VMware_f0:1a:b5 (00:50:56:f0:1a:b5), Dst: VMware_6d:7a:35 (00:0c:29:6d:7a:35)
* Internet Protocol Version 4, Src: 23.61.75.27 (23.61.75.27), Dst: 172.16.108.152 (172.16.108.152)
* Transmission Control Protocol, Src Port: 80 (80), Dst Port: 44266 (44266), Seq: 3547, Ack: 883, Len: 0

0000 00 0c 29 6d 7a 35 00 50 56 f0 1a b5 08 00 45 00 ...mz5.P V....E.
0010 00 28 b0 c7 00 00 80 06 0f 08 17 3d 4b 1b ac 10=K...
0020 6c 98 00 50 ac ea e4 43 ca e9 a4 a9 3f cd 50 10 i.P...C?P.
0030 fa fo f9 03 00 00 00 00 00 00 00 00 00 00

eth0: <live capture in progress> File: /t... Packets: 7843 - Displayed: 7843 {100.0%} Profile: Default



ANJUMAN-I-ISLAM'S KALSEKAR TECHNICAL CAMPUS

School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)

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:

Capturing from eth0 [Wireshark 1.12.6 (Git Rev Unknown from unknown)]

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

Filter: http Expression... Clear Apply Save

No.	Time	Source	Destination	Protocol	Length	Info
4124	25.63389500	172.16.108.152	141.217.1.160	HTTP	508	GET /promos/1376/programs-min_1.png HTTP/1.1
4126	25.63399600	[172.16.108.152]	[141.217.1.160]	HTTP	513	GET /promos/1376/apply-students-2015.jpg HTTP/1.1
4160	25.66536200	141.217.1.160	172.16.108.152	HTTP	287	HTTP/1.1 200 OK (text/javascript)
4173	25.67091200	172.16.108.152	141.217.1.160	HTTP	528	GET /promos/1376/winter-registration2015-3section_1.jpg HTTP/1.1
4187	25.67268800	141.217.1.160	172.16.108.152	HTTP	1247	HTTP/1.1 200 OK (PNG)
4208	25.67323100	172.16.108.152	141.217.1.160	HTTP	512	GET /promos/1380/flu-shot-wayne-edu.jpg HTTP/1.1
4223	25.68279200	141.217.1.160	172.16.108.152	HTTP	1284	HTTP/1.1 200 OK (GIFB9a)
4233	25.68285500	141.217.1.160	172.16.108.152	HTTP	402	HTTP/1.1 200 OK (PNG)
4235	25.68306500	172.16.108.152	141.217.1.160	HTTP	508	GET /images/news/van-jones-news.jpg HTTP/1.1
4236	25.68308400	172.16.108.152	141.217.1.160	HTTP	518	GET /resources/images/footer/give-to-wsu.gif HTTP/1.1
4250	25.68478900	141.217.1.160	172.16.108.152	HTTP	1402	HTTP/1.1 200 OK (PNG)

Frame 4126: 513 bytes on wire (4104 bits), 513 bytes captured (4104 bits) on interface 0
Ethernet II, Src: Vmware_6d:7a:35 (00:0c:29:6d:7a:35), Dst: Vmware_f0:1a:b5 (00:50:56:f0:1a:b5)
Internet Protocol Version 4, Src: 172.16.108.152 (172.16.108.152), Dst: 141.217.1.160 (141.217.1.160)
Transmission Control Protocol, Src Port: 52099 (52099), Dst Port: 80 (80), Seq: 1, Ack: 1, Len: 459
Hypertext Transfer Protocol

0000 00 50 56 f0 1a b5 00 0c 29 6d 7a 35 08 00 45 00 .PV.....)mz5..E.
0010 01 f3 32 ce 40 00 40 05 5e 15 ac 10 6c 98 8d d9 ..2.0.0. ^....
0020 01 a0 cb 83 00 50 f1 b7 ba 26 14 6a 7a c8 50 18P. .&jz.P.
0030 72 10 c3 8a 00 90 47 45 54 20 7f 72 6f 6d 6f r.....CE T /promo
0040 72 11 31 33 37 36 2f 61 70 70 6c 79 2d 73 74 75 s/1376/a pply-stu
0050 64 65 66 74 73 2d 32 30 31 35 2e 64 70 67 2d 48 dents-20 15_.jpg H
0060 65 66 74 73 2d 32 30 31 35 2e 64 70 67 2d 48 TTP/1.1 200 OK H
0070 61 79 6e 25 24 25 64 75 0d 0e 25 3d 3e 2d 41 .Hxx...v
0080 67 65 6e 74 3a 20 4d e1 7a 69 6c 6c 61 2f 35 2e gent: Mozilla/5.
0090 30 20 28 58 31 31 3b 20 4c 69 6e 75 79 20 69 36 0 (X11: Linux 16
00a0 38 36 3b 20 72 76 3a 33 31 2e 30 29 20 47 65 63 06: rv:3 1.0) Gec
00b0 6b 2f 32 30 31 30 31 20 46 69 72 65 ko/2010 101 Fire
00c0 66 6f 78 2f 33 31 2e 30 20 49 63 65 77 65 61 73 fox/31.0 Iceweas
00d0 65 6c 2f 33 31 2e 38 2e 30 0d 04 41 63 63 65 70 el/31.8. 0..Accp
00e0 74 3a 20 69 6d 61 67 65 2f 70 6e 67 2c 69 6d 61 t: image /png.ima
00f0 67 65 2f 2a 3b 71 3d 30 2e 38 2c 2a 2f 2a 71 ge/*:q=0 .B./*:q

eth0: <live capture in progress> File: /... Packets: 5085 - Displayed: 60 (1.2%) Profile: Default

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.



ANJUMAN-I-ISLAM'S KALSEKAR TECHNICAL CAMPUS

School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)

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:

Frame 4019: 467 bytes on wire (3736 bits), 467 bytes captured (3736 bits) on interface 0
Ethernet II, Src: VMware_6d:7a:35 (00:0c:29:6d:7a:35), Dst: VMware_f0:1a:b5 (00:50:56:f0:1a:b5)
Internet Protocol Version 4, Src: 172.16.108.152 (172.16.108.152), Dst: 141.217.1.160 (141.217.1.160)
Transmission Control Protocol, Src Port: 52089 (52089), Dst Port: 80 (80), Seq: 1, Ack: 1, Len: 413
Hypertext Transfer Protocol

0000 00 50 56 f0 1a b5 00 0c 29 6d 7a 35 08 00 45 00 .PV.....)mz5..E.
0010 01 c5 37 81 40 00 40 06 59 90 ac 10 6c 98 8d d9 ..7.@. @. Y...l...
0020 01 a0 cb 79 00 50 d9 ba b7 c9 7e 11 bf b9 50 18 ..y.R.P.
0030 72 10 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ..y.R.P.
0040 72 10 00 00 00 00 00 00 00 00 00 00 00 00 00 00 /1.1-.He st: www.
0050 77 61 79 66 65 26 65 64 75 0d 0a 55 73 65 72 2d wayne.ed u..User.
0060 41 67 65 66 74 3a 20 4d 6f 7a 69 6c 61 2f 35 Agent: Mozilla/5
0070 2a 30 20 28 58 31 31 3b 20 4c 69 6a 75 78 20 69 686; rv: 31.0) Ge
0080 36 38 36 3b 20 72 76 3a 33 31 2e 30 29 20 47 65 ck/2010 010 Fir
0090 63 6b 6f 2f 32 30 31 30 30 31 30 20 46 69 72 efox/31.0 Icewea
00a0 65 66 6f 78 2f 33 31 2e 30 29 49 63 65 77 65 61 sel/31.8 .0..acce
00b0 73 65 6c 2f 33 31 2e 29 30 0d 04 41 63 63 65 pt_text/2010.0p
00c0 70 6c 69 63 61 74 3a 6f 6e 7f 78 68 74 62 25 libinsights/1.0/htm
00d0 79 6d 6c 62 61 70 70 6c 69 63 61 74 69 6f 6e 2f xml_appl ication/
00f0 78 6d 6c 3b 71 3d 30 2e 39 22 24 2a 3b 71 3d xm:qo= 9,*/*;q=0
0100 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
File: "/tmp/wireshark_pcapng_ether0.2... Packets: 5487 - Displayed: 1 (0.0%) - Dropped: 0 (0.0%) Profile: Default

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

Frame 4015: 210 bytes on wire (1680 bits), 210 bytes captured (1680 bits) on interface 0
Ethernet II, Src: VMware_f0:1a:b5 (00:50:56:f0:1a:b5), Dst: VMware_6d:7a:35 (00:0c:29:6d:7a:35)
Internet Protocol Version 4, Src: 172.16.108.2 (172.16.108.2), Dst: 172.16.108.152 (172.16.108.152)
User Datagram Protocol, Src Port: 53 (53), Dst Port: 40430 (40430)
Domain Name System (response)

0000 00 0c 29 6d 7a 35 00 50 56 f0 1a b5 08 00 45 00 ..)mz5.P V.....E.
0010 00 c4 bd 93 00 00 80 11 4b da ac 10 6c 02 ac 10K...l...
0020 6c 98 00 35 9d ee 00 b0 12 50 6c 5a 81 80 00 01 l...._.PLZ....
0030 00 02 00 02 00 03 03 77 77 05 77 61 79 6e 65,w www.
0040 03 65 64 75 00 00 01 01 c0 0c 00 05 00 01 00 .edu.....
0050 00 00 05 00 06 08 77 68 76 32 70 72 6f 64 02 63wh v2prod.c
0060 63 c0 10 c0 00 00 01 00 00 00 00 00 00 00 04 8d c....+.....
0070 6e 73 32 00 10 c0 00 00 02 00 01 00 00 00 00 00 00 ..S.....
0080 6e 73 32 c0 10 c0 00 00 02 00 01 00 00 00 00 00 00 ns2.....
0090 05 02 6e 73 c0 10 c0 57 00 01 00 01 00 00 00 05 ns3.....g.....
00a0 00 04 8d d9 9a a2 c0 55 00 01 00 01 00 00 00 05U.....
00b0 00 04 8d d9 9a aa c0 55 00 1c 00 01 00 00 00 05U.....
00c0 00 10 26 06 97 00 00 00 f0 0e 00 00 00 00 00 00 ..&.....
00d0 00 02

File: "/tmp/wireshark_pcapng_ether0.2... Packets: 5487 - Displayed: 260 (4.7%) - Dropped: 0 (0.0%) Profile: Default

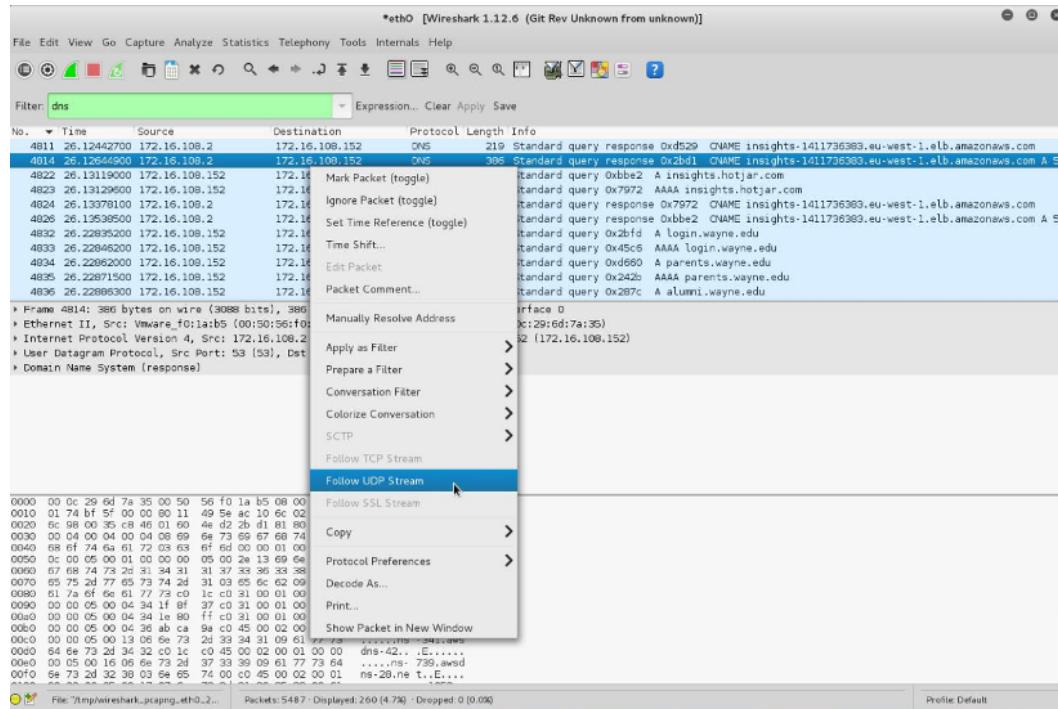


ANJUMAN-I-ISLAM'S KALSEKAR TECHNICAL CAMPUS

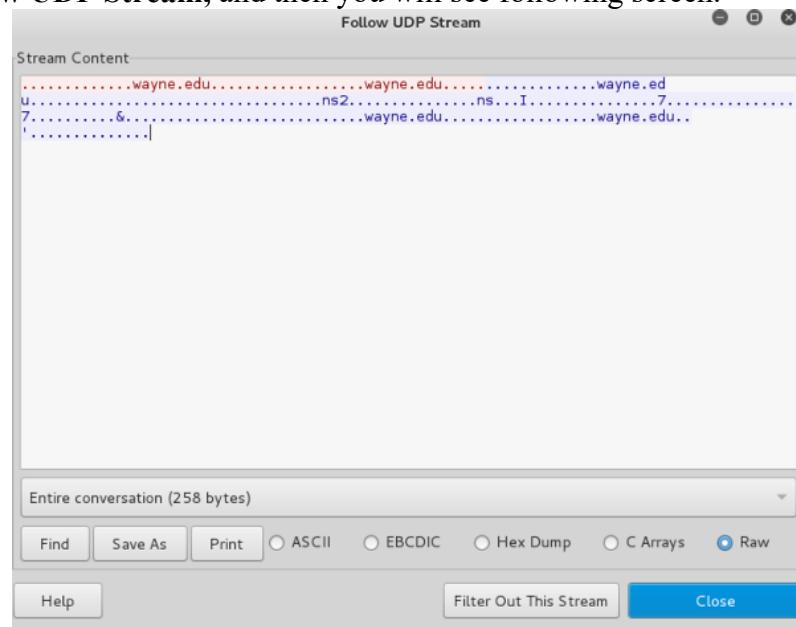
School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)

9. Let's try now to find out what are those packets contain by following one of the 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:



Click on **Follow UDP Stream**, and then you will see following screen.



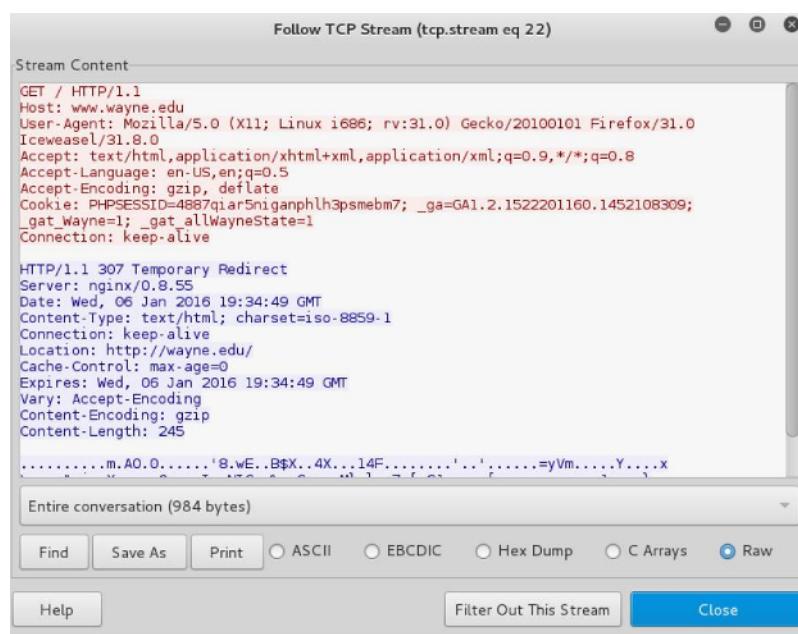
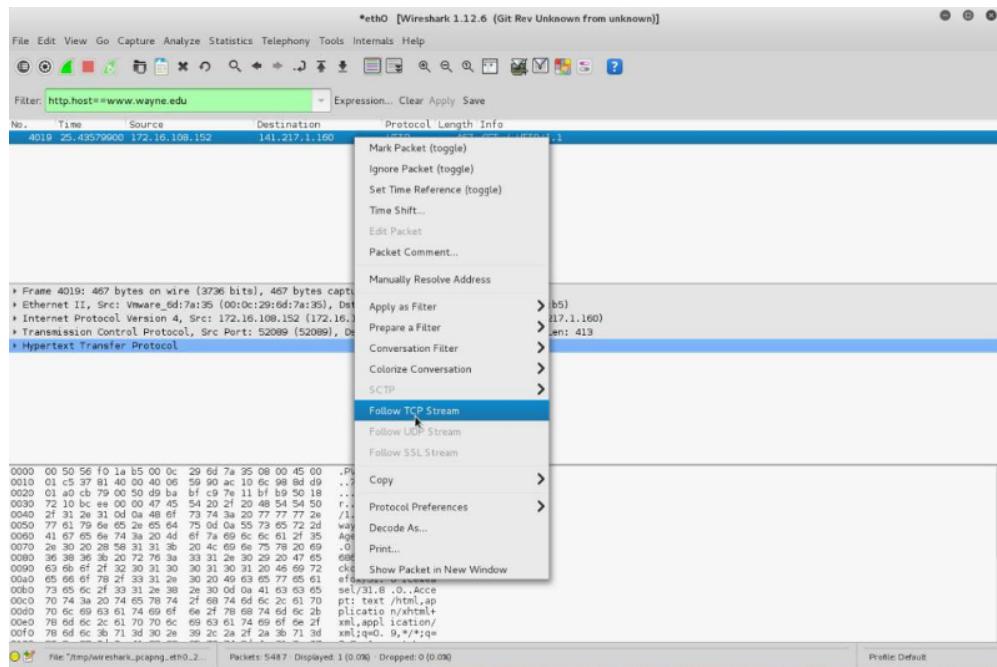


ANJUMAN-I-ISLAM'S KALSEKAR TECHNICAL CAMPUS

School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)

10. If we close this window and change the filter back to “http.host==www.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.



Conclusion : Captured icmp, tcp and http packets in promiscuous mode by packet sniffer tool wireshark.



Course Code : CSL604

Course Name : System Security Lab

Class : TE-CO

Batch : Computer Engineering

Roll no : 18CO63

Name : SHAIKH TAUSEEF MUSHTAQUE ALI

Experiment : 07

Aim : Download and install nmap. Use it with different options to scan open ports, perform OS fingerprinting, do a ping scan, tcp port scan, udp port scan, xmas scan etc.

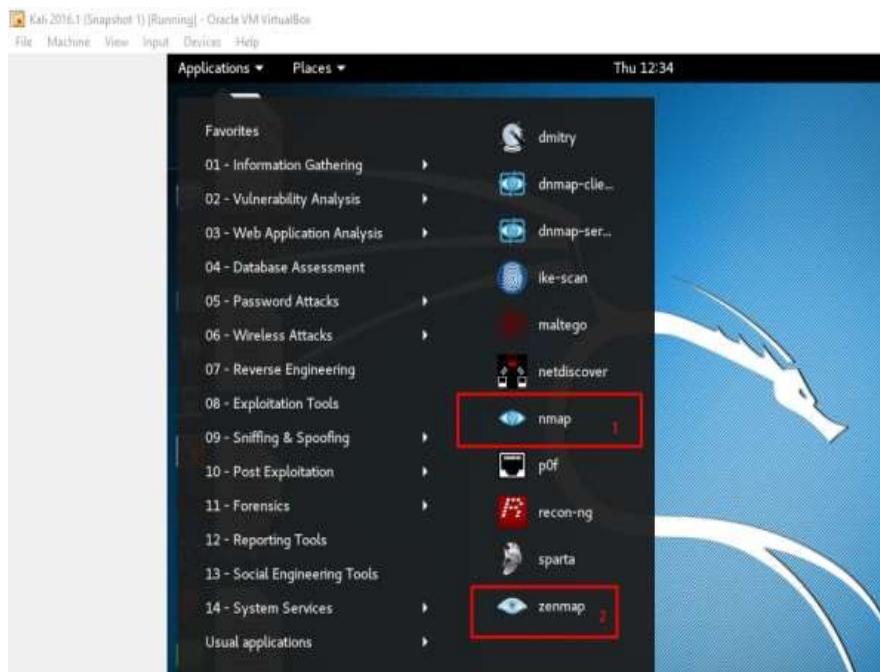
Theory :**NMAP and ZenMAP**

NMAP and ZenMAP are useful tools for the scanning phase of Ethical Hacking in Kali Linux. NMAP and ZenMAP are practically the same tool, however NMAP uses command line while ZenMAP has a GUI.

NMAP is a free utility tool for network discovery and security auditing. Many systems and network administrators also find it useful for tasks such as network inventory, managing service upgrade schedules, and monitoring host or service uptime.

NMAP uses raw IP packets in novel ways to determine which hosts are available on the network, what services (application name and version) those hosts are offering, which operating systems (and OS versions) they are running, what type of packet filters/firewalls are in use, etc.

Now, let's go step by step and learn how to use NMAP and ZenMAP.



Step 1: To open, go to Applications -> 01-Information Gathering -> nmap or zenmap.

Step 2: The next step is to detect the OS type/version of the target host. Based on the help indicated by NMAP, the parameter of OS type/version detection is variable “-O”. For more information, use this link: <https://nmap.org/book/man-os-detection.html>



School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)

The command that we will use is:

```
nmap -O 192.168.1.101
```

The following screenshot shows where you need to type the above command to see the Nmap output:

The screenshot shows the Zenmap interface. In the 'Command' field, the user has typed `nmap -O 192.168.1.101`. The 'OS' tab is selected in the hosts list, showing the target `192.168.1.101`. The 'Nmap Output' tab displays the scan results, which include a large list of open TCP ports (e.g., 22/tcp, 23/tcp, 25/tcp, 53/tcp, 80/tcp, etc.) and their services. Below the port list, OS detection details are shown, including the MAC address (`08:00:27:D1:33:60`), device type (`general purpose`), running system (`Linux 2.6.X`), OS CPE (`cpe:/o:linux:linux_kernel:2.6`), OS details (`Linux 2.6.9 - 2.6.33`), and network distance (`1 hop`). A note at the bottom states that OS detection was performed and encourages reporting incorrect results. The scan completed in 17.55 seconds.

Step 3: Next, open the TCP and UDP ports. To scan all the TCP ports based on NMAP, use the following command:

```
nmap -p 1-65535 -T4 192.168.1.101
```

Where the parameter “`-p`” indicates all the TCP ports that have to be scanned. In this case, we are scanning all the ports and “`-T4`” is the speed of scanning at which NMAP has to run. Following are the results. In green are all the TCP open ports and in red are all the closed ports. However, NMAP does not show as the list is too long.



School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)

Target: 192.168.1.101 Profile: Scan Cancel

Command: nmap -p 1-65535 -T4 192.168.1.101

Hosts Services Nmap Output Ports / Hosts Topology Host Details Scans

OS Host 192.168.1.101

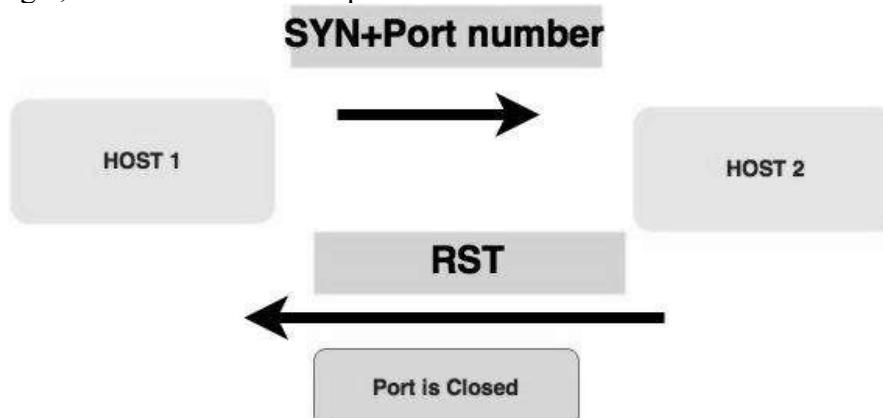
Starting Nmap 7.12 (https://nmap.org) at 2016-09-16 18:04 Central European Daylight Time
Nmap scan report for 192.168.1.101
Host is up (0.000010s latency).
Not shown: 65505 closed ports ←

PORT	STATE	SERVICE
21/tcp	open	ftp
22/tcp	open	ssh
23/tcp	open	telnet
25/tcp	open	smtp
53/tcp	open	domain
80/tcp	open	http
111/tcp	open	rpcbind
139/tcp	open	netbios-ssn
445/tcp	open	microsoft-ds
512/tcp	open	exec
513/tcp	open	login
514/tcp	open	shell
1099/tcp	open	rmiregistry
1524/tcp	open	ingreslock
2049/tcp	open	nfs
2121/tcp	open	ccproxy-ftp
3306/tcp	open	mysql
3632/tcp	open	distccd
5432/tcp	open	postgresql
5900/tcp	open	vnc
6000/tcp	open	X11
6667/tcp	open	irc
6697/tcp	open	unknown
8009/tcp	open	ajp13
8180/tcp	open	unknown
8787/tcp	open	unknown
48285/tcp	open	unknown
51161/tcp	open	unknown

Filter Hosts

StealthScan

Stealth scan or SYN is also known as **half-open scan**, as it doesn't complete the TCP three-way handshake. A hacker sends a SYN packet to the target; if a SYN/ACK frame is received back, then it's assumed the target would complete the connect and the port is listening. If an RST is received back from the target, then it is assumed the port isn't active or is closed.

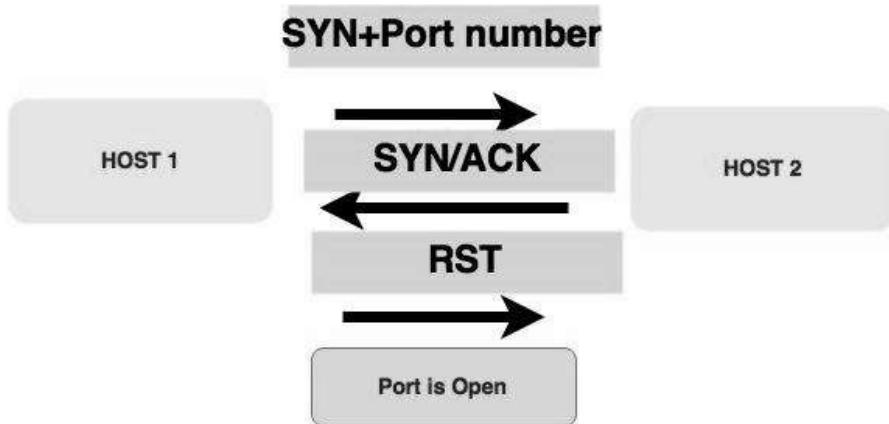




ANJUMAN-I-ISLAM'S KALSEKAR TECHNICAL CAMPUS

School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)



Now to see the SYN scan in practice, use the parameter `-sS` in NMAP. Following is the full command –

```
nmap -sS -T4 192.168.1.101
```

The following screenshot shows how to use this command:



School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)

The screenshot shows the Zenmap interface with the following details:

- Scan Menu:** Scan, Tools, Profile, Help.
- Target:** 192.168.1.101
- Profile:** (empty)
- Command:** nmap -sS -p 1-6500 192.168.1.101
- Hosts Tab:** Shows OS and Host information for 192.168.1.101.
- Nmap Output Tab:** Displays the scan results:

```
Starting Nmap 7.12 ( https://nmap.org ) at 2016-09-16
22:34 Central European Daylight Time
Nmap scan report for 192.168.1.101
Host is up (0.00030s latency).
Not shown: 6479 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
512/tcp   open  exec
513/tcp   open  login
514/tcp   open  shell
1099/tcp  open  rmiregistry
1524/tcp  open  ingreslock
2049/tcp  open  nfs
2121/tcp  open  ccproxy-ftp
3306/tcp  open  mysql
3632/tcp  open  distccd
5432/tcp  open  postgresql
5900/tcp  open  vnc
6000/tcp  open  X11
MAC Address: 08:00:27:D1:33:60 (Oracle VirtualBox
virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 16.38
seconds
```

Conclusion:

Used different options to scan open ports, perform OS fingerprinting, do a ping scan, tcp port scan, udp port scan, xmas scan etc. by downloading nmap.



ANJUMAN-I-ISLAM'S KALSEKAR TECHNICAL CAMPUS

School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)

Course Code : CSL604

Course Name : System Security Lab

Class : TE-CO

Batch : Computer Engineering

Roll no : 18CO63

Name : SHAIKH TAUSEEF MUSHTAQUE ALI

Experiment : 08

Aim : Setting up personal Firewall using iptables.

Theory :

iptables is a firewall program for **Linux**. It will monitor traffic from and to your server using **tables**.

These tables contain **sets of rules**, called **chains**, that will filter incoming and outgoing data packets.

[optin-monster-shortcode id="fv4lqeko3gylvecpszws"]

When a **packet** matches a **rule**, it is given a **target**, which can be another chain or one of these special values:

- **ACCEPT** – will allow the packet to pass through.
- **DROP** – will not let the packet pass through.
- **RETURN** – stops the packet from traversing through a chain and tell it to go back to the previous chain.

In this iptables tutorial, we are going to work with one of the default tables, called **filter**. It consists of three chains:

- **INPUT** – controls incoming packets to the server.
- **FORWARD** – filters incoming packets that will be forwarded somewhere else.
- **OUTPUT** – filter packets that are going out from your server.

Before we begin this guide, make sure you have SSH **root** or **sudo** access to your machine that runs on Ubuntu 16.04 or up. You can establish the connection through **PuTTY** (Windows) or terminal shell (Linux, macOS). If you own Hostinger VPS, you can get the SSH login details on the Servers tab of hPanel.

iptables rules only apply to **ipv4**. If you want to set up a firewall for the **ipv6** protocol, you will need to use **ip6tables** instead.

How to Install and Use Iptables Linux Firewall



ANJUMAN-I-ISLAM'S KALSEKAR TECHNICAL CAMPUS

School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)

We will divide this iptables tutorial into three steps. First, you will learn how to install the tool on Ubuntu. Secondly, we are going to show you how to define the rules. Lastly, we will guide you to make persistent changes in iptables.

Step 1 — Installing Iptables

Iptables comes pre-installed in most Linux distributions. However, if you don't have it in Ubuntu/Debian system by default, follow the steps below:

1. Connect to your server via SSH. If you don't know, you can read our [SSH tutorial](#).
2. Execute the following command one by one:

```
sudo apt-get update
```

```
sudo apt-get install iptables
```

3. Check the status of your current iptables configuration by running:

```
sudo iptables -L -v
```

4. Here, the -L option is used to list all the rules, and -v is for showing the info in a more detailed format. Below is the example output:

```
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
```

```
pkts bytes target  prot opt in out source destination
```

```
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
```

```
pkts bytes target  prot opt in out source destination
```

```
Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
```

```
pkts bytes target  prot opt in out source destination
```

You will now have the Linux firewall installed. At this point, you can notice that all chains are set to **ACCEPT** and have no rules. This is not secure since any packet can come through without filtering.



ANJUMAN-I-ISLAM'S KALSEKAR TECHNICAL CAMPUS

School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)

Don't worry. We'll tell you how to define rules on the next step of our iptables tutorial.

Step 2 – Defining Chain Rules

Defining a rule means appending it to the chain. To do this, you need to insert the **-A** option (**Append**) right after the iptables command, like so:

```
sudo iptables -A
```

It will alert iptables that you are adding new rules to a chain. Then, you can combine the command with other options, such as:

- **-i (interface)** — the network interface whose traffic you want to filter, such as eth0, lo, ppp0, etc.
- **-p (protocol)** — the network protocol where your filtering process takes place. It can be either **tcp**, **udp**, **udplite**, **icmp**, **sctp**, **icmpv6**, and so on. Alternatively, you can type **all** to choose every protocol.
- **-s (source)** — the address from which traffic comes from. You can add a hostname or IP address.
- **-dport (destination port)** — the destination port number of a protocol, such as **22 (SSH)**, **443 (https)**, etc.
- **-j (target)** — the target name (**ACCEPT**, **DROP**, **RETURN**). You need to insert this every time you make a new rule.

If you want to use all of them, you must write the command in this order:

```
sudo iptables -A <chain> -i <interface> -p <protocol (tcp/udp)> -s <source> --dport <port no.> -j <target>
```

Once you understand the basic syntax, you can start configuring the firewall to give more security to your server. For this iptables tutorial, we are going to use the **INPUT** chain as an example.

Enabling Traffic on Localhost

To allow traffic on localhost, type this command:

```
sudo iptables -A INPUT -i lo -j ACCEPT
```

For this iptables tutorial, we use **lo** or **loopback** interface. It is utilized for all communications on the localhost. The command above will make sure that the connections between a database and a web application on the same machine are working properly.



ANJUMAN-I-ISLAM'S KALSEKAR TECHNICAL CAMPUS

School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)

Enabling Connections on HTTP, SSH, and SSL Port

Next, we want **http** (port **80**), **https** (port **443**), and **ssh** (port **22**) connections to work as usual. To do this, we need to specify the protocol (**-p**) and the corresponding port (**-dport**). You can execute these commands one by one:

```
sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT
```

```
sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT
```

```
sudo iptables -A INPUT -p tcp --dport 443 -j ACCEPT
```

It's time to check if the rules have been appended in iptables:

```
sudo iptables -L -v
```

It should return with the results below which means all TCP protocol connections from the specified ports will be accepted:

0	0	ACCEPT	tcp	--	any	any	anywhere	anywhere	tcp dpt:ssh
0	0	ACCEPT	tcp	--	any	any	anywhere	anywhere	tcp dpt:http
0	0	ACCEPT	tcp	--	any	any	anywhere	anywhere	tcp dpt:https

Filtering Packets Based on Source

Iptables allows you to filter packets based on an IP address or a range of IP addresses. You need to specify it after the **-s** option. For example, to accept packets from **192.168.1.3**, the command would be:

```
sudo iptables -A INPUT -s 192.168.1.3 -j ACCEPT
```

You can also reject packets from a specific IP address by replacing the **ACCEPT** target with **DROP**.

```
sudo iptables -A INPUT -s 192.168.1.3 -j DROP
```

If you want to drop packets from a range of IP addresses, you have to use the **-m** option and **iprange** module. Then, specify the IP address range with **--src-range**. Remember, a hyphen should separate the range of ip addresses without space, like this:

```
sudo iptables -A INPUT -m iprange --src-range 192.168.1.100-192.168.1.200 -j DROP
```

Dropping all Other Traffic



ANJUMAN-I-ISLAM'S KALSEKAR TECHNICAL CAMPUS

School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)

It is crucial to use the **DROP** target for all other traffic after defining **-dport** rules. This will prevent an unauthorized connection from accessing the server via other open ports. To achieve this, simply type:

```
sudo iptables -A INPUT -j DROP
```

Now, the connection outside the specified port will be dropped.

Deleting Rules

If you want to remove all rules and start with a clean slate, you can use the **-F** option (**flush**):

```
sudo iptables -F
```

This command erases all current rules. However, to delete a specific rule, you must use the **-D** option.

First, you need to see all the available rules by entering the following command:

```
sudo iptables -L --line-numbers
```

You will get a list of rules with numbers:

```
Chain INPUT (policy ACCEPT)
```

num	target	prot	opt	source	destination
-----	--------	------	-----	--------	-------------

1	ACCEPT	all	--	192.168.0.4	anywhere
2	ACCEPT	tcp	--	anywhere	anywhere tcp dpt:https
3	ACCEPT	tcp	--	anywhere	anywhere tcp dpt:http
4	ACCEPT	tcp	--	anywhere	anywhere tcp dpt:ssh

To delete a rule, insert the corresponding chain and the number from the list. Let's say for this iptables tutorial, we want to get rid of **rule number three** of the **INPUT** chain. The command should be:

```
sudo iptables -D INPUT 3
```

Step 3 – Persisting Changes

The iptables rules that we have created are saved in memory. That means we have to redefine them on reboot. To make these changes persistent after restarting the server, you can use this command:

```
sudo /sbin/iptables-save
```



ANJUMAN-I-ISLAM'S KALSEKAR TECHNICAL CAMPUS

School of Engineering & Technology

Affiliated to : University of Mumbai, Recognised by : DTE (Maharashtra) & Approved by : AICTE (New Delhi)

It will save the current rules on the system configuration file, which will be used to reconfigure the tables every time the server reboots.

Note that you should always run this command every time you make changes to the rules. For example, if you want to disable iptables, you need to execute these two lines:

```
sudo iptables -F
```

```
sudo /sbin/iptables-save
```

You will see the following results:

```
root@vps42707999:~# sudo /sbin/iptables-save
# Generated by iptables-save v1.6.1 on Tue Nov 19 11:38:18 2019
*nat
:PREROUTING ACCEPT [645426:37524813]
:INPUT ACCEPT [558190:28836818]
:OUTPUT ACCEPT [263056:18809892]
:POSTROUTING ACCEPT [263056:18809892]
:OUTPUT_direct - [0:0]
:POSTROUTING_ZONES - [0:0]
:POSTROUTING_ZONES_SOURCE - [0:0]
:POSTROUTING_direct - [0:0]
:POST_public - [0:0]
:POST_public_allow - [0:0]
```

Conclusion:

Personal Firewall using iptables was set up.