ÇUKUROVA UNIVERSITY
FACULTY OF ENGINEERING
COMPUTER ENGINEERING DEPARTMENT

EXERCISE SHEET 1 - OSI reference model

Task 1

a. What is the basic idea of the OSI reference model?

b. What does one understand (according to OSI) by an "open" network?

Task 2

a. Explain the principle of layering for communication tasks!

b. What are the advantages and disadvantages?

Task 3

Which are the 7 layers of the OSI reference model (English and Turkish names) and which functionalities are assigned to them?

Task 4

Which are the 5 layers of the TCP/IP model (English and Turkish names) and which functionalities are assigned to them? What are the advantages and disadvantages of TCP/IP model compared to OSI refence model?

Task 5

Understand and go through the following Wireshark tutorial at home. Discuss your interpretations and conclusions during the lab hour.

Packet Capture & Traffic Analysis with Wireshark

This lab introduces packet capture (packet sniffing) and network traffic analysis with the **Wireshark** tool.

Packet Capture (Packet Sniffing)

A **packet sniffer** is an application which can capture and analyse network traffic which is passing through a system's Network Interface Card (NIC). The sniffer sets the card to **promiscuous mode** which means all traffic is read, whether it is addressed to that machine or not.

Packet Analysis

Wireshark is an open source cross-platform packet capture and analysis tool, with versions for Windows and Linux. The GUI window gives a detailed breakdown of the network protocol stack for each packet, colorising packet details based on protocol, as well as having functionality to filter and search the traffic, and pick out TCP streams. Wireshark can also save packet data to files for offline analysis and export/import packet captures to/from other tools. Statistics can also be generated for packet capture files.

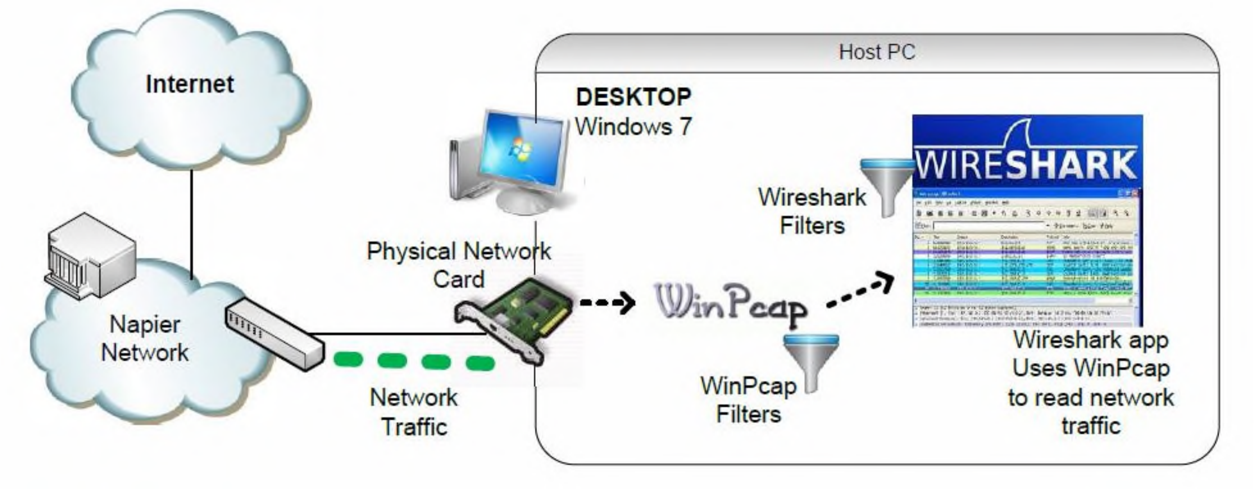
Wireshark can be used for **network troubleshooting**, to **investigate security issues**, and to **analyse and understand network protocols**. The packet sniffer can exploit information passed in plaintext, i.e. not encrypted. Examples of **protocols** which pass information in plaintext are **Telnet, FTP, SNMP, POP, and HTTP**.

Wireshark is a GUI based network capture tool. There is another command line-based version of the packet capture utility, called **tcpdump**. **Tcpdump** provides many similar features as Wireshark, but is console-based. It can be a good alternative if only command line access is available, and also uses less resources as it has no GUI to generate.

Using Wireshark to Capture and Analyse Traffic

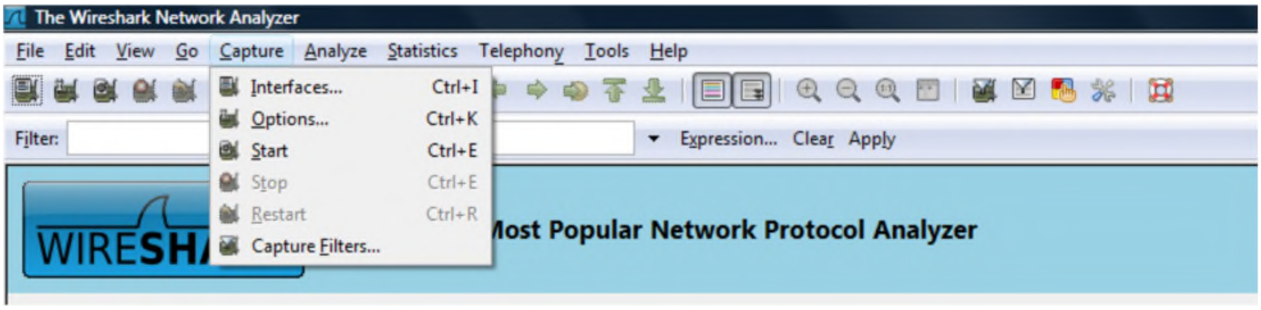
In this exercise, the fundamentals of the **Wireshark Packet Sniffer and Protocol Analyser** tool will be introduced. Then Wireshark will be used to perform basic protocol analysis on TCP/IP network traffic.

Using Wireshark to Capture Traffic

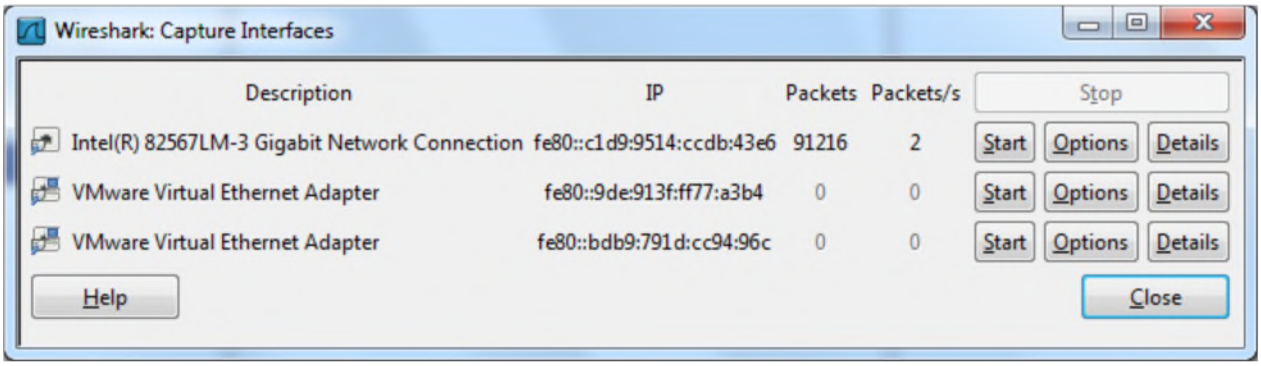


Select a Network Interface to Capture Packets through.

Start the Wireshark application. When Wireshark is first run, a default, or blank window is shown. To list the available network interfaces, select the **Capture->Interfaces** menu option.


The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Telephony Tools Help
Interfaces... Ctrl+I
Options... Ctrl+K
Start Ctrl+E
Stop Ctrl+E
Restart Ctrl+R
Capture Filters...
Filter:
Expression... Clear Apply
WIRESHA
Most Popular Network Protocol Analyzer

Wireshark should display a popup window such as the one shown in Figure 2. To capture network traffic click the **Start** button for the network interface you want to capture traffic on. Windows can have a long list of virtual interfaces, before the Ethernet Network Interface Card (NIC).

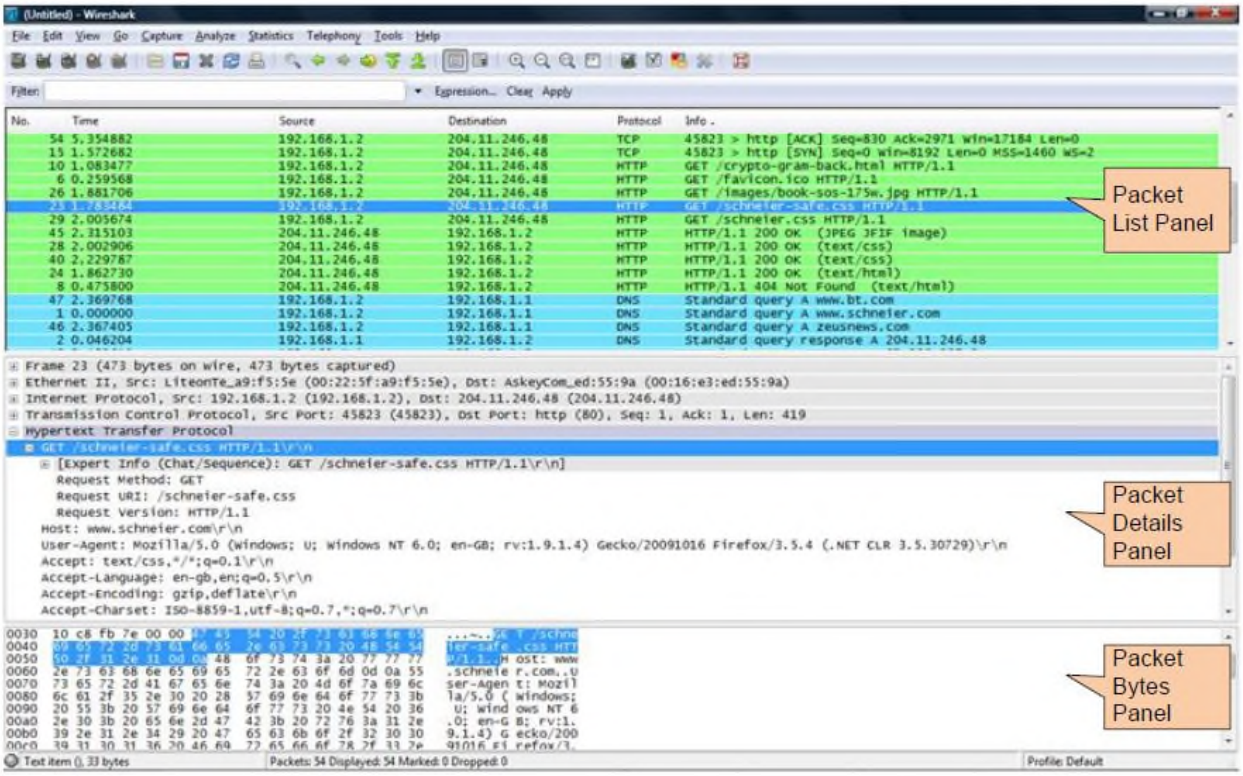


Wireshark: Capture Interfaces

Description	IP	Packets	Packets/s	Stop		
Intel(R) 82567LM-3 Gigabit Network Connection	fe80::c1d9:9514:ccdb:43e6	91216	2	Start	Options	Details
VMware Virtual Ethernet Adapter	fe80::9de:913f:ff77:a3b4	0	0	Start	Options	Details
VMware Virtual Ethernet Adapter	fe80::bdb9:791d:cc94:96c	0	0	Start	Options	Details

Figure 2 - Wireshark Interfaces Window

Generate some network traffic with a Web Browser, such as Internet Explorer or Chrome. Your Wireshark window should show the packets, and now look something like.



(Untitled) - Wireshark
File Edit View Go Capture Analyze Statistics Telephony Tools Help
Filter:
Expression... Clear Apply
Packet List Panel
Packet Details Panel
Packet Bytes Panel

```
No.  Time       Source          Destination     Protocol  Info
 54 5.354882    192.168.1.2     204.11.246.48   TCP       45823 > http [ACK] Seq=830 Ack=2971 Win=17184 Len=0
 15 1.572682    192.168.1.2     204.11.246.48   TCP       45823 > http [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=2
 10 1.083477    192.168.1.2     204.11.246.48   HTTP      GET /crypto-gram-back.html HTTP/1.1
  6 0.259568    192.168.1.2     204.11.246.48   HTTP      GET /favicon.ico HTTP/1.1
 26 1.881706    192.168.1.2     204.11.246.48   HTTP      GET /images/book-sos-175w.jpg HTTP/1.1
 23 1.783464    192.168.1.2     204.11.246.48   HTTP      GET /schneier-safe.css HTTP/1.1
 29 2.005674    192.168.1.2     204.11.246.48   HTTP      GET /schneier.css HTTP/1.1
 45 2.315103    204.11.246.48   192.168.1.2     HTTP      HTTP/1.1 200 OK  (JPEG JFIF image)
 28 2.002906    204.11.246.48   192.168.1.2     HTTP      HTTP/1.1 200 OK  (text/css)
 40 2.229787    204.11.246.48   192.168.1.2     HTTP      HTTP/1.1 200 OK  (text/css)
 24 1.862730    204.11.246.48   192.168.1.2     HTTP      HTTP/1.1 200 OK  (text/html)
  8 0.475800    204.11.246.48   192.168.1.2     HTTP      HTTP/1.1 404 Not Found  (text/html)
 47 2.369768    192.168.1.2     192.168.1.1     DNS       Standard query A www.bt.com
  1 0.000000    192.168.1.2     192.168.1.1     DNS       Standard query A www.schneier.com
 46 2.367405    192.168.1.2     192.168.1.1     DNS       Standard query A zeusnews.com
  2 0.046204    192.168.1.1     192.168.1.2     DNS       Standard query response A 204.11.246.48
```

```
⊞ Frame 23 (473 bytes on wire, 473 bytes captured)
⊞ Ethernet II, Src: LiteonTe_a9:f5:5e (00:22:5f:a9:f5:5e), Dst: AskeyCom_ed:55:9a (00:16:e3:ed:55:9a)
⊞ Internet Protocol, Src: 192.168.1.2 (192.168.1.2), Dst: 204.11.246.48 (204.11.246.48)
⊞ Transmission Control Protocol, Src Port: 45823 (45823), Dst Port: http (80), Seq: 1, Ack: 1, Len: 419
⊟ Hypertext Transfer Protocol
  ⊟ GET /schneier-safe.css HTTP/1.1\r\n
    ⊞ [Expert Info (Chat/Sequence): GET /schneier-safe.css HTTP/1.1\r\n]
      Request Method: GET
      Request URI: /schneier-safe.css
      Request Version: HTTP/1.1
    Host: www.schneier.com\r\n
    User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.0; en-GB; rv:1.9.1.4) Gecko/20091016 Firefox/3.5.4 (.NET CLR 3.5.30729)\r\n
    Accept: text/css,*/*;q=0.1\r\n
    Accept-Language: en-gb,en;q=0.5\r\n
    Accept-Encoding: gzip,deflate\r\n
    Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7\r\n
```

```
0030  10 c8 fb 7e 00 00 47 45  54 20 2f 73 63 68 6e 65   ...~..GE T /schne
0040  69 65 72 2d 73 61 66 65  2e 63 73 73 20 48 54 54   ier-safe .css HTT
0050  50 2f 31 2e 31 0d 0a 48  6f 73 74 3a 20 77 77 77   P/1.1..H ost: www
0060  2e 73 63 68 6e 65 69 65  72 2e 63 6f 6d 0d 0a 55   .schneie r.com..U
0070  73 65 72 2d 41 67 65 6e  74 3a 20 4d 6f 7a 69 6c   ser-Agen t: Mozil
0080  6c 61 2f 35 2e 30 20 28  57 69 6e 64 6f 77 73 3b   la/5.0 ( Windows;
0090  20 55 3b 20 57 69 6e 64  6f 77 73 20 4e 54 20 36    U; Wind ows NT 6
00a0  2e 30 3b 20 65 6e 2d 47  42 3b 20 72 76 3a 31 2e   .0; en-G B; rv:1.
00b0  39 2e 31 2e 34 29 20 47  65 63 6b 6f 2f 32 30 30   9.1.4) G ecko/200
00c0  39 31 30 31 36 20 46 69  72 65 66 6f 78 2f 33 2e   91016 Fi refox/3.
```

```
Text item (), 33 bytes   Packets: 54 Displayed: 54 Marked: 0 Dropped: 0   Profile: Default
```

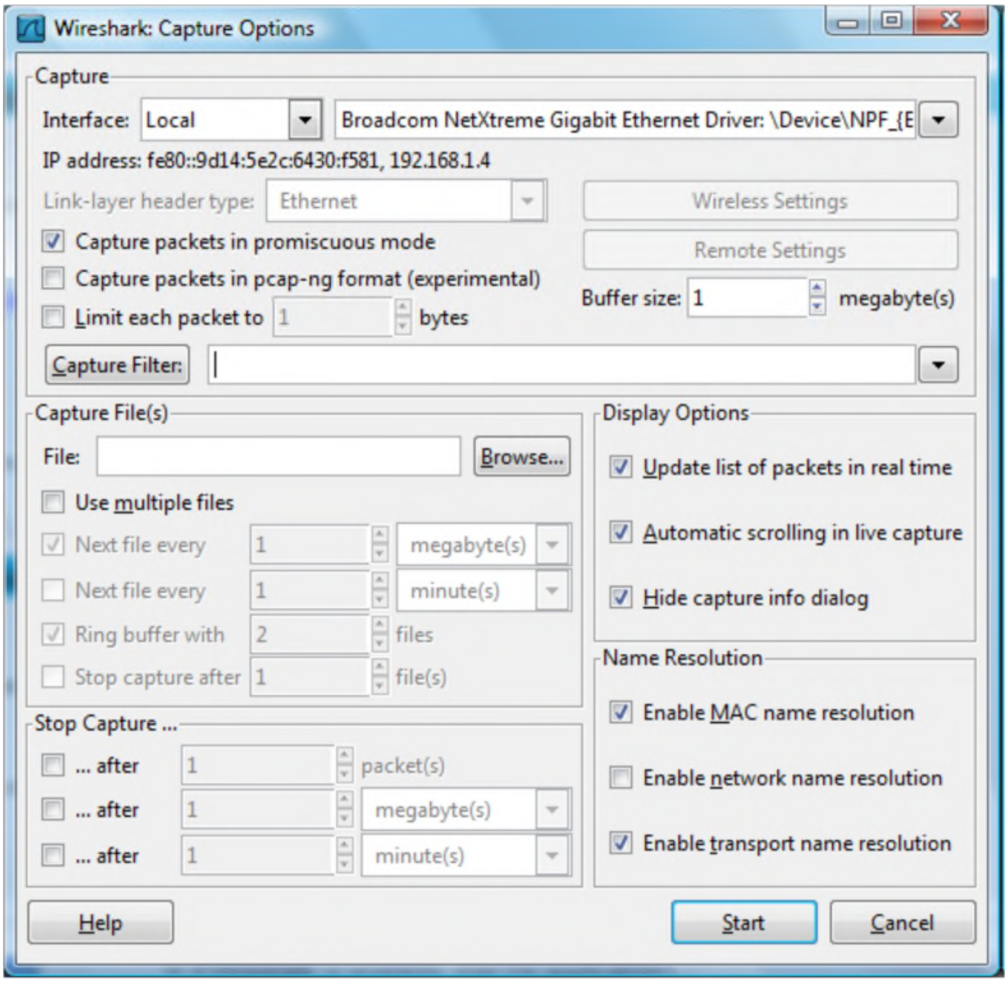
Figure 3 - Wireshark capturing traffic

To stop the capture, select the **Capture->Stop** menu option, Ctrl+E, or the Stop toolbar button. What you have created is a Packet Capture or ***'pcap'***, which you can now view and analyse using the Wireshark interface, or save to disk to analyse later.

The capture is split into 3 parts:

1. **Packet List Panel –** this is a list of packets in the current capture. It colors the packets based on the protocol type. When a packet is selected, the details are shown in the two panels below.
2. **Packet Details Panel –** this shows the details of the selected packet. It shows the different protocols making up the layers of data for this packet. Layers include Frame, Ethernet, IP, TCP/UDP/ICMP, and application protocols such as HTTP.
3. **Packet Bytes Panel –** shows the packet bytes in Hex and ASCII encodings.

To select more detailed options when starting a capture, select the **Capture->Options** menu option, or **Ctrl+K**, or the Capture Options button on the toolbar (the wrench). This should show a window such as shown in Figure 4.



Wireshark: Capture Options

Capture

Interface: Local | Broadcom NetXtreme Gigabit Ethernet Driver: \Device\NPF_{E

IP address: fe80::9d14:5e2c:6430:f581, 192.168.1.4

Link-layer header type: Ethernet

☑ Capture packets in promiscuous mode

☐ Capture packets in pcap-ng format (experimental)

☐ Limit each packet to 1 bytes

Buffer size: 1 megabyte(s)

Capture Filter:

Capture File(s)

File:

☐ Use multiple files

☑ Next file every 1 megabyte(s)

☐ Next file every 1 minute(s)

☑ Ring buffer with 2 files

☐ Stop capture after 1 file(s)

Stop Capture ...

☐ ... after 1 packet(s)

☐ ... after 1 megabyte(s)

☐ ... after 1 minute(s)

Display Options

☑ Update list of packets in real time

☑ Automatic scrolling in live capture

☑ Hide capture info dialog

Name Resolution

☑ Enable MAC name resolution

☐ Enable network name resolution

☑ Enable transport name resolution

Figure 4 - Wireshark Capture Options

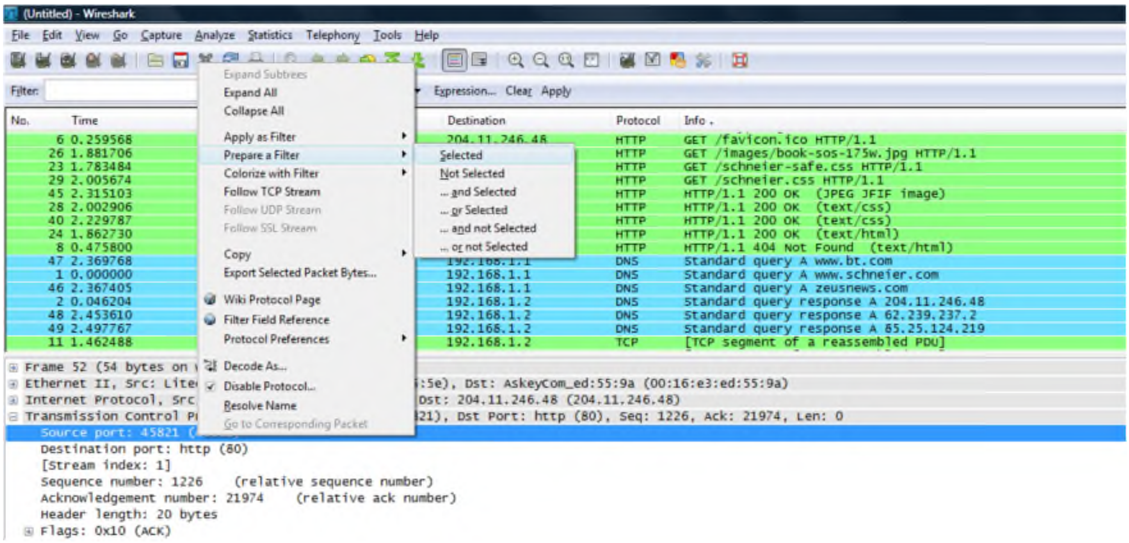
Some of the more interesting options are:

- ***Capture Options** > **Interface** -* Again the important thing is to select the correct Network Interface to capture traffic through.
- ***Capture Options** > **Capture File** –* useful to save a file of the packet capture in real time, in case of a system crash.
- ***Display Options** > **Update list of packets in real time** –* A display option, which should be checked if you want to view the capture as it happens (typically switched off to capture straight to a file, for later analysis).
- ***Name Resolution** > **MAC name resolution** –* resolves the first 3 bytes of the MAC Address, the Organization Unique Identifier (OUI), which represents the Manufacturer of the Card.
- ***Name Resolution** > **Network name resolution** –* does a DNS lookup for the IP Addresses captured, to display the network name. Set to off by default, so covert scans do not generate this DNS traffic, and tip off who's packets you are sniffing.

Make sure the **MAC name resolution** is selected. Start the capture, and generate some Web traffic again, then stop the capture.

Wireshark Display Filters

Right click on the **Source Port** field in the **Packet Details Panel**. Select **Prepare a Filter->Selected**.

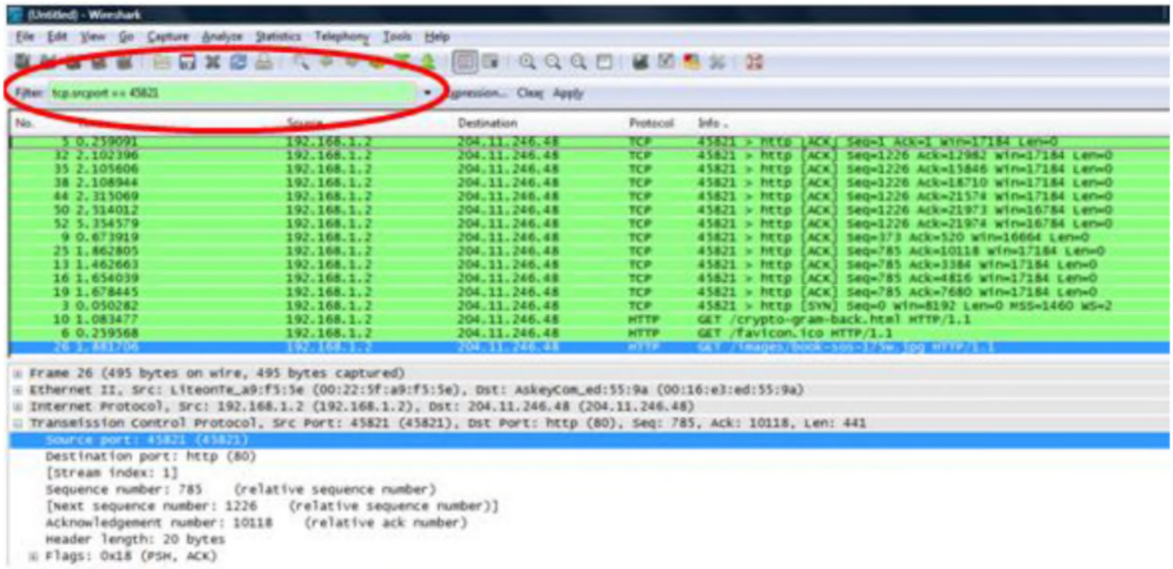


```
No.   Time        Destination      Protocol  Info
  6 0.259568      204.11.246.48    HTTP      GET /favicon.ico HTTP/1.1
 26 1.881706                       HTTP      GET /images/book-sos-175w.jpg HTTP/1.1
 23 1.783484                       HTTP      GET /schneier-safe.css HTTP/1.1
 29 2.005674                       HTTP      GET /schneier.css HTTP/1.1
 45 2.315103                       HTTP      HTTP/1.1 200 OK  (JPEG JFIF image)
 28 2.002906                       HTTP      HTTP/1.1 200 OK  (text/css)
 40 2.229787                       HTTP      HTTP/1.1 200 OK  (text/css)
 24 1.862730                       HTTP      HTTP/1.1 200 OK  (text/html)
  8 0.475800                       HTTP      HTTP/1.1 404 Not Found  (text/html)
 47 2.369768      192.168.1.1      DNS       Standard query A www.bt.com
  1 0.000000      192.168.1.1      DNS       Standard query A www.schneier.com
 46 2.367405      192.168.1.1      DNS       Standard query A zeusnews.com
  2 0.046204      192.168.1.2      DNS       Standard query response A 204.11.246.48
 48 2.453610      192.168.1.2      DNS       Standard query response A 62.239.237.2
 49 2.497767      192.168.1.2      DNS       Standard query response A 85.25.124.219
 11 1.462488      192.168.1.2      TCP       [TCP segment of a reassembled PDU]

⊞ Frame 52 (54 bytes on w
⊞ Ethernet II, Src: Lite                  :5e), Dst: AskeyCom_ed:55:9a (00:16:e3:ed:55:9a)
⊞ Internet Protocol, Src                  Dst: 204.11.246.48 (204.11.246.48)
⊟ Transmission Control Pr                 21), Dst Port: http (80), Seq: 1226, Ack: 21974, Len: 0
    Source port: 45821 (
    Destination port: http (80)
    [Stream index: 1]
    Sequence number: 1226    (relative sequence number)
    Acknowledgement number: 21974    (relative ack number)
    Header length: 20 bytes
  ⊞ Flags: 0x10 (ACK)
```

Figure 5 - Filtering on a protocol field

Wireshark automatically generates a **Display Filter**, and applies it to the capture. The filter is shown in the **Filter Bar**, below the button toolbar. Only packets captured with a Source Port of the value selected should be displayed. The window should be similar to that shown in Figure 6. This same process can be performed on most fields within Wireshark, and can be used to include or exclude traffic.



Filter: tcp.srcport == 45821

```
No.  Time         Source        Destination    Protocol  Info
  5 0.259091      192.168.1.2   204.11.246.48  TCP       45821 > http [ACK] Seq=1 Ack=1 Win=17184 Len=0
 32 2.102396      192.168.1.2   204.11.246.48  TCP       45821 > http [ACK] Seq=1226 Ack=12982 Win=17184 Len=0
 35 2.105606      192.168.1.2   204.11.246.48  TCP       45821 > http [ACK] Seq=1226 Ack=15846 Win=17184 Len=0
 38 2.108944      192.168.1.2   204.11.246.48  TCP       45821 > http [ACK] Seq=1226 Ack=18710 Win=17184 Len=0
 44 2.315069      192.168.1.2   204.11.246.48  TCP       45821 > http [ACK] Seq=1226 Ack=21574 Win=17184 Len=0
 50 2.314012      192.168.1.2   204.11.246.48  TCP       45821 > http [ACK] Seq=1226 Ack=21973 Win=16784 Len=0
 52 5.354579      192.168.1.2   204.11.246.48  TCP       45821 > http [ACK] Seq=1226 Ack=21974 Win=16784 Len=0
  9 0.673919      192.168.1.2   204.11.246.48  TCP       45821 > http [ACK] Seq=373 Ack=520 Win=16664 Len=0
 25 1.862805      192.168.1.2   204.11.246.48  TCP       45821 > http [ACK] Seq=785 Ack=10118 Win=17184 Len=0
 13 1.462663      192.168.1.2   204.11.246.48  TCP       45821 > http [ACK] Seq=785 Ack=3384 Win=17184 Len=0
 16 1.654039      192.168.1.2   204.11.246.48  TCP       45821 > http [ACK] Seq=785 Ack=4816 Win=17184 Len=0
 19 1.678445      192.168.1.2   204.11.246.48  TCP       45821 > http [ACK] Seq=785 Ack=7680 Win=17184 Len=0
  3 0.050282      192.168.1.2   204.11.246.48  TCP       45821 > http [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=2
 10 1.083477      192.168.1.2   204.11.246.48  HTTP      GET /crypto-gram-back.html HTTP/1.1
  6 0.259568      192.168.1.2   204.11.246.48  HTTP      GET /favicon.ico HTTP/1.1
 26 1.881706      192.168.1.2   204.11.246.48  HTTP      GET /images/book-sos-175w.jpg HTTP/1.1
```

```
⊞ Frame 26 (495 bytes on wire, 495 bytes captured)
⊞ Ethernet II, Src: LiteonTe_a9:f5:5e (00:22:5f:a9:f5:5e), Dst: AskeyCom_ed:55:9a (00:16:e3:ed:55:9a)
⊞ Internet Protocol, Src: 192.168.1.2 (192.168.1.2), Dst: 204.11.246.48 (204.11.246.48)
⊟ Transmission Control Protocol, Src Port: 45821 (45821), Dst Port: http (80), Seq: 785, Ack: 10118, Len: 441
    Source port: 45821 (45821)
    Destination port: http (80)
    [Stream index: 1]
    Sequence number: 785    (relative sequence number)
    [Next sequence number: 1226    (relative sequence number)]
    Acknowledgement number: 10118    (relative ack number)
    Header length: 20 bytes
  ⊞ Flags: 0x18 (PSH, ACK)
```

Figure 6 - Wireshark Display Filter