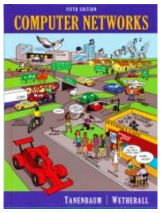
Lab Exercise – Ethernet



Objective

To explore the details of Ethernet frames. Ethernet is a popular link layer protocol. Modern computers connect to Ethernet switches rather than use classic Ethernet .

The trace file is here: http://scisweb.ulster.ac.uk/~kevin/com320/labs/wireshark/trace-ethernet.pcap

Requirements

Wireshark: This lab uses the Wireshark software tool to capture and examine a packet trace. A packet trace is a record of traffic at a location on the network, as if a snapshot was taken of all the bits that passed across a particular wire. The packet trace records a timestamp for each packet, along with the bits that make up the packet, from the lower-layer headers to the higher-layer contents.

A quick help guide to Wireshark display filters is here: http://openmaniak.com/wireshark_filters.php

1. **Launching Wireshark**
   You can type ***Wireshark*** in the run box of main Windows 8 start screen. It should load but there can be a problem with the new lab configuration for Wireshark and npf driver. Therefore if this is not working….. then please do the next step.

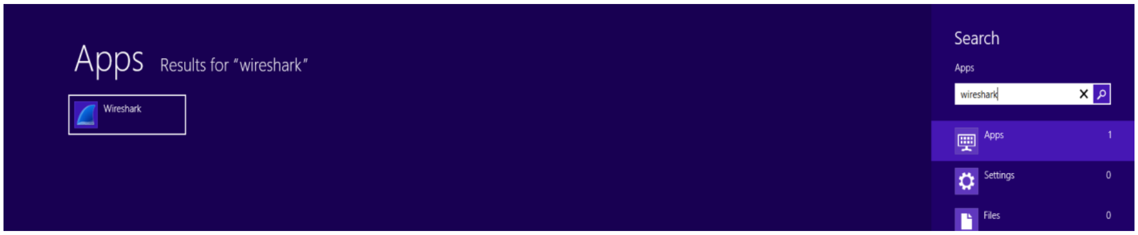
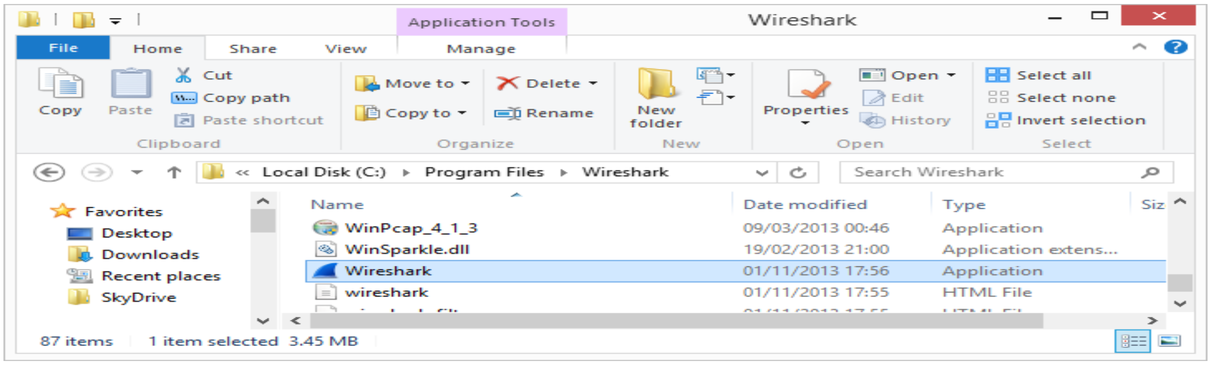


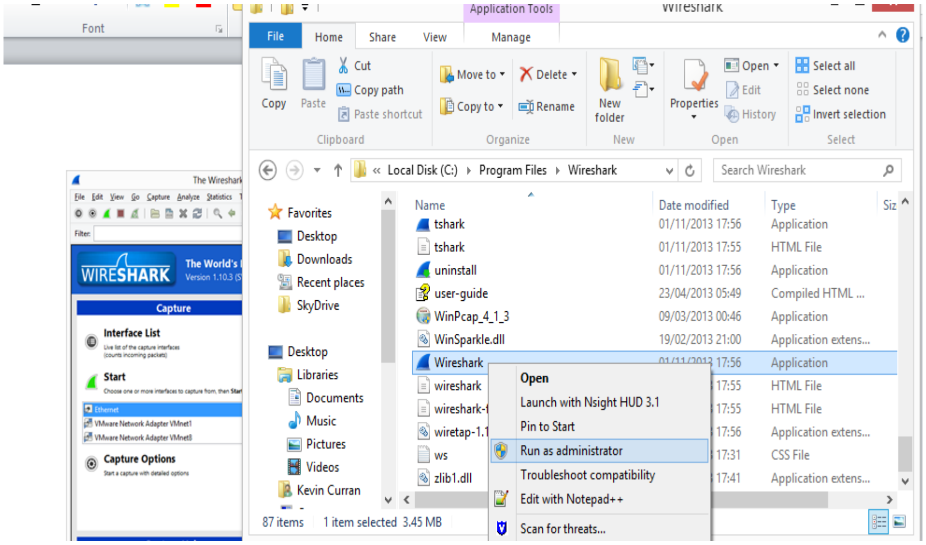
Figure 1: Wireshark in lab

2. Launch Wireshark as follows. Click the desktop icon on the main windows screen and use the file explorer to browse to C:\local Disk (C)\Program Files\Wireshark



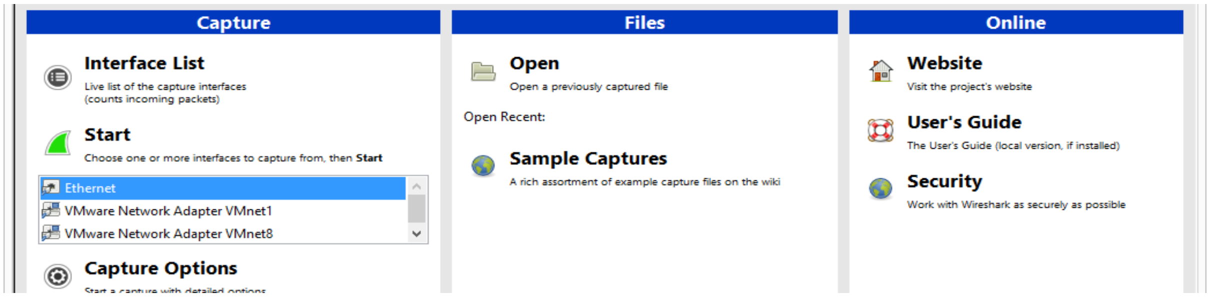
Name	Date modified	Type
WinPcap_4_1_3	09/03/2013 00:46	Application
WinSparkle.dll	19/02/2013 21:00	Application extens...
Wireshark	01/11/2013 17:56	Application
wireshark	01/11/2013 17:55	HTML File

3. Next, RIGHT CLICK on Wireshark as **"Run as administrator".**



Name	Date modified	Type
tshark	01/11/2013 17:56	Application
tshark	01/11/2013 17:55	HTML File
uninstall	01/11/2013 17:56	Application
user-guide	23/04/2013 05:49	Compiled HTML ...
WinPcap_4_1_3	09/03/2013 00:46	Application
WinSparkle.dll	19/02/2013 21:00	Application extens...
Wireshark	01/11/2013 17:56	Application
wireshark	17:55	HTML File
wireshark-(	17:55	HTML File
wiretap-1.1	17:56	Application extens...
ws	17:31	CSS File
zlib1.dll	17:41	Application extens...

4. You will then get a startup screen, as shown next:



Interface List
Live list of the capture interfaces (counts incoming packets)

Start
Choose one or more interfaces to capture from, then **Start**

Capture Options
Start a capture with detailed options

Open
Open a previously captured file

Open Recent:

Sample Captures
A rich assortment of example capture files on the wiki

Website
Visit the project's website

User's Guide
The User's Guide (local version, if installed)

Security
Work with Wireshark as securely as possible

Figure 2: Initial Wireshark Screen

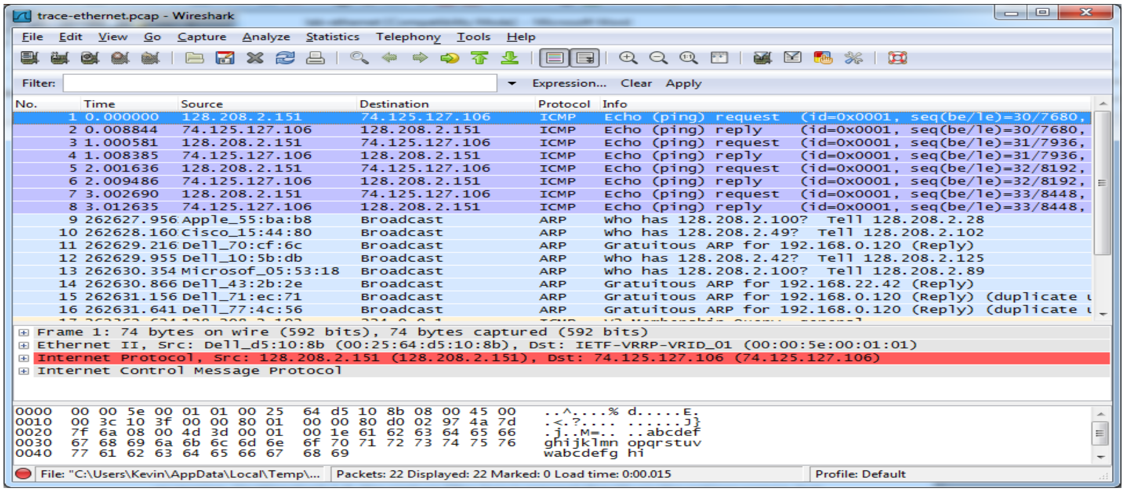
5. Take a look at the upper left hand side of the screen – you'll see an "Interface list". This is the list of network interfaces on your computer. Once you choose an interface, Wireshark will capture all packets on that interface. Click on the network card on the particular machine you are working on. In the example above, it is the **Ethernet Driver** to start packet capture (i.e., for Wireshark to begin capturing all packets being sent to/from that interface), a screen like the one below will be displayed, showing information about the packets being captured. Once you start packet capture, you can stop it by using the Capture pull down menu and selecting Stop.

Step 1: Capture a Trace

*Proceed as follows to capture a trace of ping packets. Right click and choose **"Save Link As"** and down the following trace file -*http://scisweb.ulster.ac.uk/~kevin/com320/labs/wireshark/trace-ethernet.pcap

1. *Open the following trace file from the location you downloaded to e.g. Local Disk (C):\downloads*

2. *You should see a screen similar to the following:*



```
No.   Time          Source              Destination       Protocol  Info
    1 0.000000      128.208.2.151       74.125.127.106    ICMP      Echo (ping) request  (id=0x0001, seq(be/le)=30/7680,
    2 0.008844      74.125.127.106      128.208.2.151     ICMP      Echo (ping) reply    (id=0x0001, seq(be/le)=30/7680,
    3 1.000581      128.208.2.151       74.125.127.106    ICMP      Echo (ping) request  (id=0x0001, seq(be/le)=31/7936,
    4 1.008385      74.125.127.106      128.208.2.151     ICMP      Echo (ping) reply    (id=0x0001, seq(be/le)=31/7936,
    5 2.001636      128.208.2.151       74.125.127.106    ICMP      Echo (ping) request  (id=0x0001, seq(be/le)=32/8192,
    6 2.009486      74.125.127.106      128.208.2.151     ICMP      Echo (ping) reply    (id=0x0001, seq(be/le)=32/8192,
    7 3.002690      128.208.2.151       74.125.127.106    ICMP      Echo (ping) request  (id=0x0001, seq(be/le)=33/8448,
    8 3.012635      74.125.127.106      128.208.2.151     ICMP      Echo (ping) reply    (id=0x0001, seq(be/le)=33/8448,
    9 262627.956 Apple_55:ba:b8         Broadcast         ARP       Who has 128.208.2.100?  Tell 128.208.2.28
   10 262628.160 Cisco_15:44:80         Broadcast         ARP       Who has 128.208.2.49?  Tell 128.208.2.102
   11 262629.216 Dell_70:cf:6c          Broadcast         ARP       Gratuitous ARP for 192.168.0.120 (Reply)
   12 262629.955 Dell_10:5b:db          Broadcast         ARP       Who has 128.208.2.42?  Tell 128.208.2.125
   13 262630.354 Microsof_05:53:18      Broadcast         ARP       Who has 128.208.2.100?  Tell 128.208.2.89
   14 262630.866 Dell_43:2b:2e          Broadcast         ARP       Gratuitous ARP for 192.168.22.42 (Reply)
   15 262631.156 Dell_71:ec:71          Broadcast         ARP       Gratuitous ARP for 192.168.0.120 (Reply) (duplicate u
   16 262631.641 Dell_77:4c:56          Broadcast         ARP       Gratuitous ARP for 192.168.0.120 (Reply) (duplicate u
```

```
⊞ Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits)
⊞ Ethernet II, Src: Dell_d5:10:8b (00:25:64:d5:10:8b), Dst: IETF-VRRP-VRID_01 (00:00:5e:00:01:01)
⊞ Internet Protocol, Src: 128.208.2.151 (128.208.2.151), Dst: 74.125.127.106 (74.125.127.106)
⊞ Internet Control Message Protocol
```

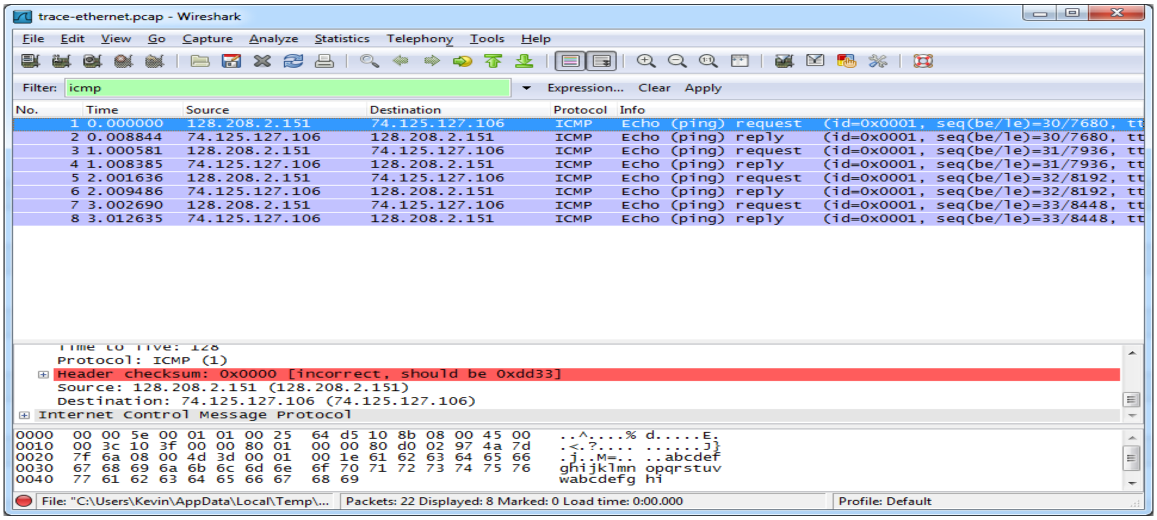
```
0000  00 00 5e 00 01 01 00 25  64 d5 10 8b 08 00 45 00   ..^....% d.....E.
0010  00 3c 10 3f 00 00 80 01  00 00 80 d0 02 97 4a 7d   .<.?.... ......J}
0020  7f 6a 08 00 4d 3d 00 01  00 1e 61 62 63 64 65 66   .j..M=.. ..abcdef
0030  67 68 69 6a 6b 6c 6d 6e  6f 70 71 72 73 74 75 76   ghijklmn opqrstuv
0040  77 61 62 63 64 65 66 67  68 69                     wabcdefg hi
```

File: "C:\Users\Kevin\AppData\Local\Temp\... Packets: 22 Displayed: 22 Marked: 0 Load time: 0:00.015 Profile: Default

Figure 3: Ethernet trace opening screen

3. *Type in to the filter box a filter of "`icmp`" and click apply.*

   Your capture window should be similar to the one pictured below, other than our highlighting.



Filter: icmp

```
No.   Time          Source              Destination       Protocol  Info
    1 0.000000      128.208.2.151       74.125.127.106    ICMP      Echo (ping) request  (id=0x0001, seq(be/le)=30/7680, tt
    2 0.008844      74.125.127.106      128.208.2.151     ICMP      Echo (ping) reply    (id=0x0001, seq(be/le)=30/7680, tt
    3 1.000581      128.208.2.151       74.125.127.106    ICMP      Echo (ping) request  (id=0x0001, seq(be/le)=31/7936, tt
    4 1.008385      74.125.127.106      128.208.2.151     ICMP      Echo (ping) reply    (id=0x0001, seq(be/le)=31/7936, tt
    5 2.001636      128.208.2.151       74.125.127.106    ICMP      Echo (ping) request  (id=0x0001, seq(be/le)=32/8192, tt
    6 2.009486      74.125.127.106      128.208.2.151     ICMP      Echo (ping) reply    (id=0x0001, seq(be/le)=32/8192, tt
    7 3.002690      128.208.2.151       74.125.127.106    ICMP      Echo (ping) request  (id=0x0001, seq(be/le)=33/8448, tt
    8 3.012635      74.125.127.106      128.208.2.151     ICMP      Echo (ping) reply    (id=0x0001, seq(be/le)=33/8448, tt
```

```
    Time to live: 128
    Protocol: ICMP (1)
  ⊞ Header checksum: 0x0000 [incorrect, should be 0xdd33]
    Source: 128.208.2.151 (128.208.2.151)
    Destination: 74.125.127.106 (74.125.127.106)
⊞ Internet Control Message Protocol
```

```
0000  00 00 5e 00 01 01 00 25  64 d5 10 8b 08 00 45 00   ..^....% d.....E.
0010  00 3c 10 3f 00 00 80 01  00 00 80 d0 02 97 4a 7d   .<.?.... ......J}
0020  7f 6a 08 00 4d 3d 00 01  00 1e 61 62 63 64 65 66   .j..M=.. ..abcdef
0030  67 68 69 6a 6b 6c 6d 6e  6f 70 71 72 73 74 75 76   ghijklmn opqrstuv
0040  77 61 62 63 64 65 66 67  68 69                     wabcdefg hi
```

File: "C:\Users\Kevin\AppData\Local\Temp\... Packets: 22 Displayed: 8 Marked: 0 Load time: 0:00.000 Profile: Default

Figure 4: Setting the capture options for `ICMP` traffic

Step 2: Inspect the Trace

Select any packet in the trace (in the top panel) to see details of its structure (in the middle panel) and the bytes that make up the packet (in the bottom panel). Now we can inspect the details of the packets. In the figure, we have selected the first packet in the trace. Note that we are using the term "packet" in a loose way. Each record captured by Wireshark more correctly corresponds to a single frame in Ethernet format that carries a packet as its payload; Wireshark interprets as much structure as it can.

In the middle panel, expand the Ethernet header fields (using the "+" expander or icon) to see their details. Our interest is the Ethernet header, and you may ignore the higher layer protocols (which are IP and ICMP in this case). Note the following:

- The frames in this trace are DIX Ethernet, called "Ethernet II" in Wireshark.
- There is no preamble in the fields shown in Wireshark. The preamble is a physical layer mechanism to help the NIC identify the start of a frame. It carries no useful data and is not received like other fields.
- There is a destination address and a source address. Wireshark is decoding some of these bits in the OUI (Organizationally Unique Identifier) portion of the address to tell us the vendor of the NIC, e.g., Dell for the source address.
- There is a Type field. For the ping messages, the Ethernet type is IP, meaning the Ethernet payload carries an IP packet. (There is no Length field as in the IEEE 802.3 format. Instead, the length of a DIX Ethernet frame is determined by the hardware of a receiving computer, which looks for valid frames that start with a preamble and end with a correct checksum, and passed up to higher layers along with the packet.)
- There is no Data field per se – the data starts with the IP header right after the Ethernet header.
- There is no pad. A pad will be present at the end if the frame would otherwise be less than 64 bytes, the minimum Ethernet frame size.
- There is no checksum in most traces, even though it really does exist. Typically, Ethernet hardware that is sending or receiving frames computes or checks this field and adds or strips it. Thus it is simply not visible to the OS or Wireshark in most capture setups.
- There are also no VLAN fields. If VLANs are in use, the VLAN tags are normally added and removed by switch ports so they will not be visible at host computers using the network.

Note: Answers to these questions are at the end of the lab notes.

Q1. What is the MAC address of the source of # 1 from IP address 128.208.2.151?

Q2. Click on # 12 and expand the [+] Address Resolution Protocol section in middle pane. What does Target MAC address: 00:00:00:00:00:00 mean?

Q3. Click again on # 12 and expand the [+] Address Resolution Protocol section in middle pane. Why is the protocol type listed as IP when it is not an IP packet?

Step 3: Ethernet Frame Structure

Try to understand the Ethernet frame format. Note the range of the Ethernet header and the Ethernet payload. See the frame structure below in Figure 5.

To work out sizes, observe that when you click on a protocol block in the middle panel (the block itself, not the "+" expander) then Wireshark will highlight the bytes it corresponds to in the packet in the lower panel and display the length at the bottom of the window. You may also use the overall packet size shown in the Length column or Frame detail block.

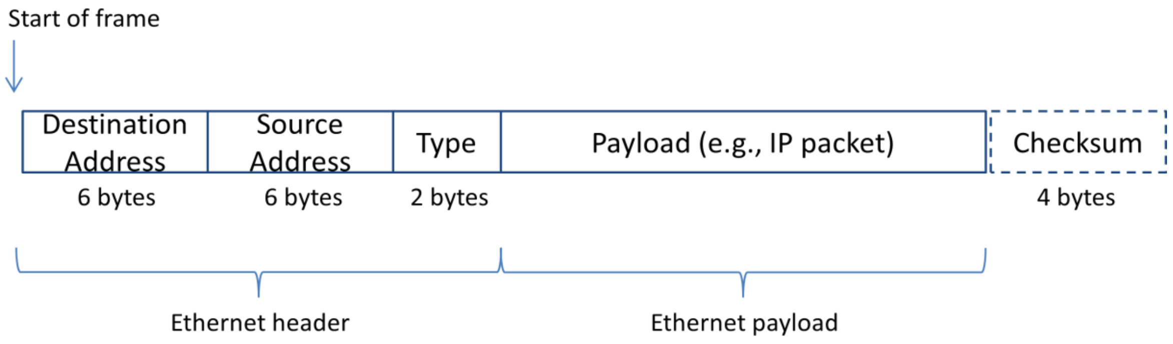


Figure 5: Structure of an Ethernet frame

There are several features to note:

- The destination address comes before the source address.
- The pad is not shown because the packets we examined (ping) are large enough that no pad is needed.
- Unlike many protocols, Ethernet has a trailer (the checksum, and pad if present) as well as a header. The checksum is handled by the hardware and not visible to Wireshark.
- The Ethernet header is 14 bytes long.

Note: Answers to these questions are at the end of the lab notes.

- Q1. Click on # 12 and expand the [+] Address Resolution Protocol section in middle pane. What does opcode (1) signify? What does opcode (2) signify?

  (note: In some Wireshark versions, opcode (1) is listed as (0x0001) & opcode (2) is listed as (0x0002)

- Q2. Click on # 12 and expand the [+] and give the hexadecimal value for the two-byte Ethernet Frame type field?

Step 4: Scope of Ethernet Addresses

Each Ethernet frame carries a source and destination address. One of these addresses is that of your computer. It is the source for frames that are sent, and the destination for frames that are received. But what is the other address? Assuming you pinged a remote Internet server, it cannot be the Ethernet address of the remote server because an Ethernet frame is only addressed to go within one LAN. Instead, it will be the Ethernet address of the router or default gateway, such as your AP in the case of 802.11. This is the device that connects your LAN to the rest of the Internet. In contrast, the IP addresses in the IP block of each packet do indicate the overall source and destination endpoints. They are your computer and the remote server.

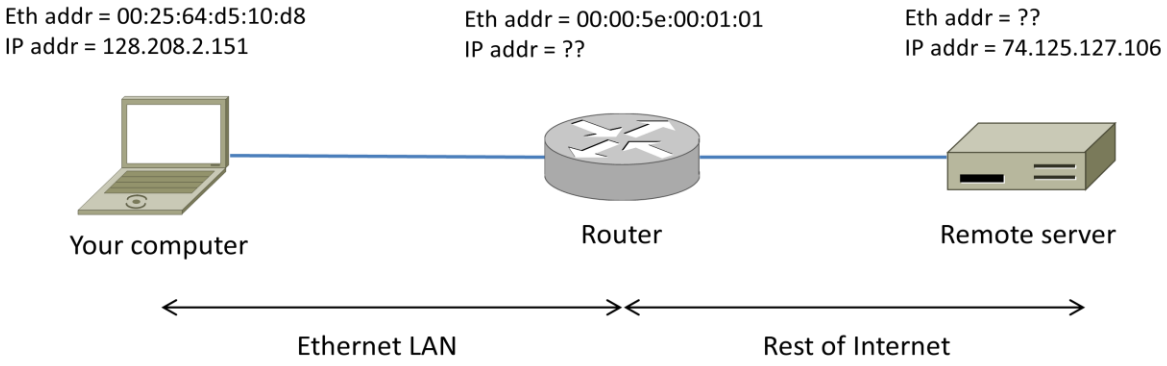


Figure 6: Ethernet and IP addresses of network devices

1. Open Wireshark and start a new capture. You may want to clear existing filters or simply close and restart Wireshark to ensure a new capture.
2. In the filter box, type the following **ip.src==youripaddress** *e.g. ip.src==193.61.191.71 (Note….You can see your IP address by running a cmd prompt and typing* **ipconfig /all)**
3. Ask your colleague for their IP address *e.g. 193.61.191.71*
4. In Windows 8, open a command line window by typing **<WINDOWS KEY>** + **R** and then type **cmd** in the *run dialog* box which should popup.
5. Type **ping theiripaddress** e.g. *ping 193.61.191.71*
6. Return to Wireshark and **stop the capture** by selecting stop in the Capture menu or the stop capture icon underneath main menu labels.

There are several features to note:

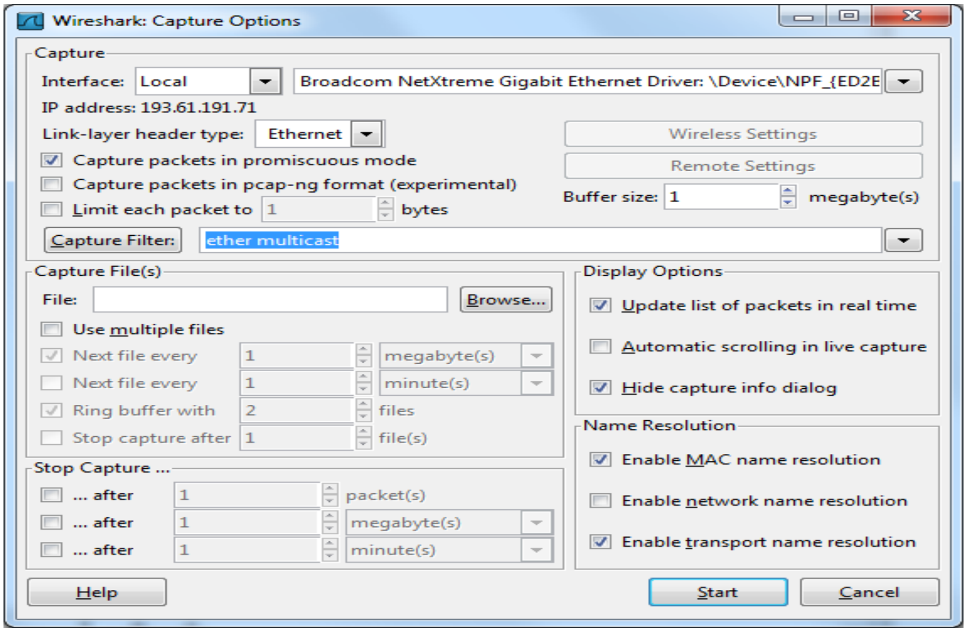
- The Ethernet and IP addresses will vary for your trace because different computers are involved, but they will have the same form, e.g. 6 bytes in hexadecimal format or 4 “dotted” bytes.
- The Ethernet & IP addresses correspond to your computer and your colleagues PC.
- Some Ethernet and IP addresses are not known from the trace alone. They are shown as “??”.

Step 5: Broadcast Frames

The trace that you gathered above captured unicast Ethernet traffic sent between a specific source and destination, e.g., your computer to the router. It is also possible to send multicast or broadcast Ethernet traffic, destined for a group of computers or all computers on the Ethernet, respectively. We can tell from the address whether it is unicast, multicast, or broadcast. Broadcast traffic is sent to a reserved Ethernet address that has all bits set to “1”. Multicast traffic is sent to addresses that have a “1” in the first bit sent on the wire; broadcast is a special case of multicast. Broadcast and multicast traffic is widely used for discovery protocols, e.g., a packet sent to everyone in an effort to find the local printer.

1. *Start a capture for broadcast and multicast Ethernet frames with a filter of* “`ether multicast`”. You do this by selecting ***Capture*** in the main menu and then selecting ***Options***. This is not to be confused with the filter box on the live capture page which will not accept the filter expression above.

2. *Wait up to 30 seconds to record background traffic, and then stop the capture. If you do not capture any packets with this filter then use the trace that we supplied.*
On most Ethernets, there is a steady chatter of background traffic as computers exchange messages to maintain network state, which is why we try to capture traffic without running any other programs. The capture filter of “`ether multicast`” will capture both multicast and broadcast Ethernet frames, but not regular unicast frames. You may have to wait a little while for these packets to be captured, but on most LANs with multiple computers you will see at least a packet every few seconds.


Wireshark: Capture Options

Capture
Interface: Local | Broadcom NetXtreme Gigabit Ethernet Driver: \Device\NPF_{ED2E
IP address: 193.61.191.71
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: ether multicast

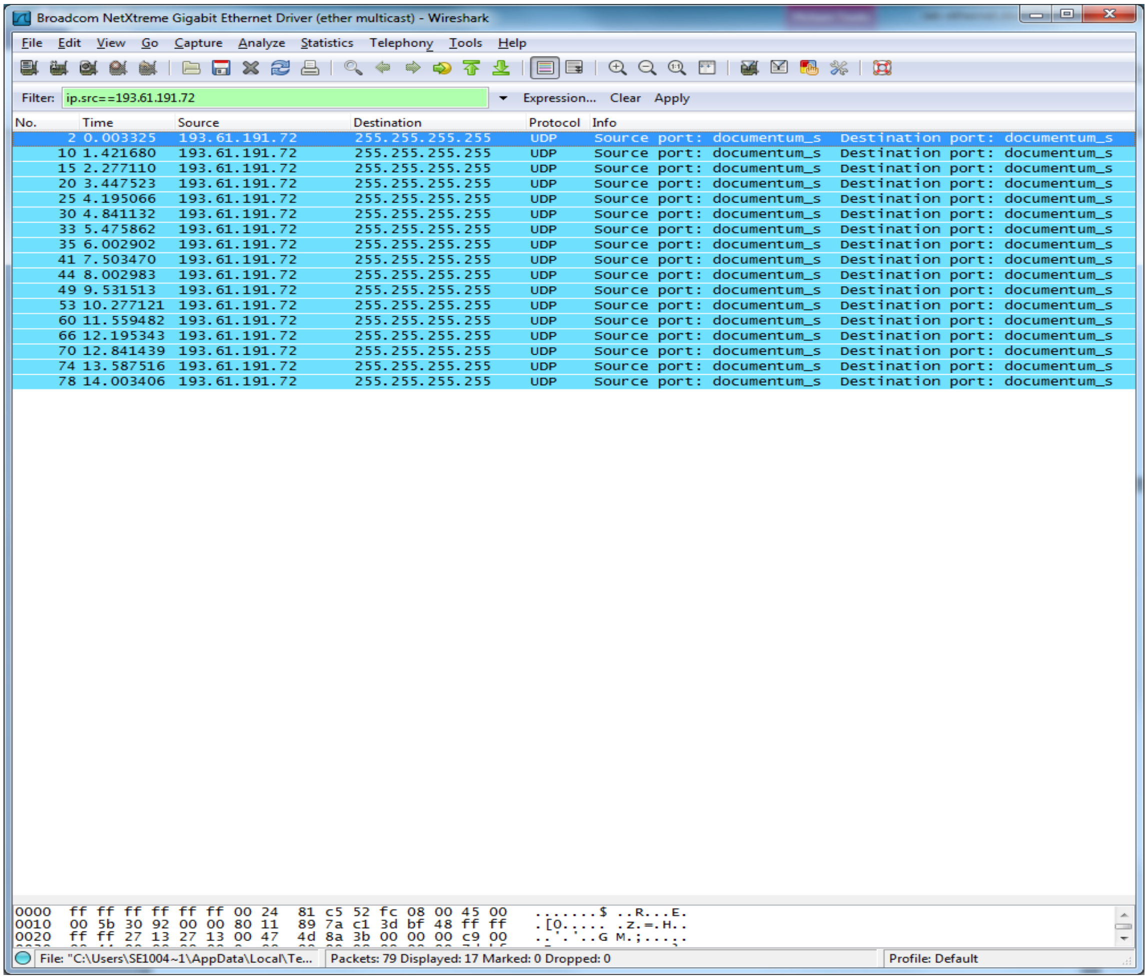
Capture File(s)
File:
☐ Use multiple files
☑ Next file every 1 megabyte(s)
☐ Next file every 1 minute(s)
☑ Ring buffer with 2 files
☐ Stop capture after 1 file(s)

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

3. *Examine the multicast and broadcast packets that you captured, looking at the details of the source and destination addresses.* Most likely one has the broadcast Ethernet address, as broadcast frames tend to be more common than multicast frames. Look at a broadcast frame to see what address is used for broadcast by Ethernet. Expand the Ethernet address fields of either broadcast or multicast frames to see which bit is set to distinguish broadcast/multicast or group traffic from unicast traffic. Your screen may look like this.


Broadcom NetXtreme Gigabit Ethernet Driver (ether multicast) - Wireshark
File Edit View Go Capture Analyze Statistics Telephony Tools Help
Filter:
Expression... Clear Apply
Profile: Default

Filter: ip.src==193.61.191.72

```
No.     Time       Source          Destination       Protocol Info
      2 0.003325   193.61.191.72   255.255.255.255   UDP      Source port: documentum_s  Destination port: documentum_s
     10 1.421680   193.61.191.72   255.255.255.255   UDP      Source port: documentum_s  Destination port: documentum_s
     15 2.277110   193.61.191.72   255.255.255.255   UDP      Source port: documentum_s  Destination port: documentum_s
     20 3.447523   193.61.191.72   255.255.255.255   UDP      Source port: documentum_s  Destination port: documentum_s
     25 4.195066   193.61.191.72   255.255.255.255   UDP      Source port: documentum_s  Destination port: documentum_s
     30 4.841132   193.61.191.72   255.255.255.255   UDP      Source port: documentum_s  Destination port: documentum_s
     33 5.475862   193.61.191.72   255.255.255.255   UDP      Source port: documentum_s  Destination port: documentum_s
     35 6.002902   193.61.191.72   255.255.255.255   UDP      Source port: documentum_s  Destination port: documentum_s
     41 7.503470   193.61.191.72   255.255.255.255   UDP      Source port: documentum_s  Destination port: documentum_s
     44 8.002983   193.61.191.72   255.255.255.255   UDP      Source port: documentum_s  Destination port: documentum_s
     49 9.531513   193.61.191.72   255.255.255.255   UDP      Source port: documentum_s  Destination port: documentum_s
     53 10.277121  193.61.191.72   255.255.255.255   UDP      Source port: documentum_s  Destination port: documentum_s
     60 11.559482  193.61.191.72   255.255.255.255   UDP      Source port: documentum_s  Destination port: documentum_s
     66 12.195343  193.61.191.72   255.255.255.255   UDP      Source port: documentum_s  Destination port: documentum_s
     70 12.841439  193.61.191.72   255.255.255.255   UDP      Source port: documentum_s  Destination port: documentum_s
     74 13.587516  193.61.191.72   255.255.255.255   UDP      Source port: documentum_s  Destination port: documentum_s
     78 14.003406  193.61.191.72   255.255.255.255   UDP      Source port: documentum_s  Destination port: documentum_s
```

```
0000  ff ff ff ff ff ff 00 24  81 c5 52 fc 08 00 45 00   .......$ ..R...E.
0010  00 5b 30 92 00 00 80 11  89 7a c1 3d bf 48 ff ff   .[0..... .z.=.H..
0020  ff ff 27 13 27 13 00 47  4d 8a 3b 00 00 00 c9 00   ..'.'..G M.;.....
```

File: "C:\Users\SE1004~1\AppData\Local\Te... Packets: 79 Displayed: 17 Marked: 0 Dropped: 0

NOTE: Before continuing, you should clear the capture filters of "ether multicast" by firstly selecting *Stop* from Capture and then again on Capture menu, selecting *options* and deleting the terms in the filter box.

4. Answer the following questions. *(Note, answers are at the end, try to work it out first….)*

    a. What is the broadcast Ethernet address, written in standard form as Wireshark displays it?

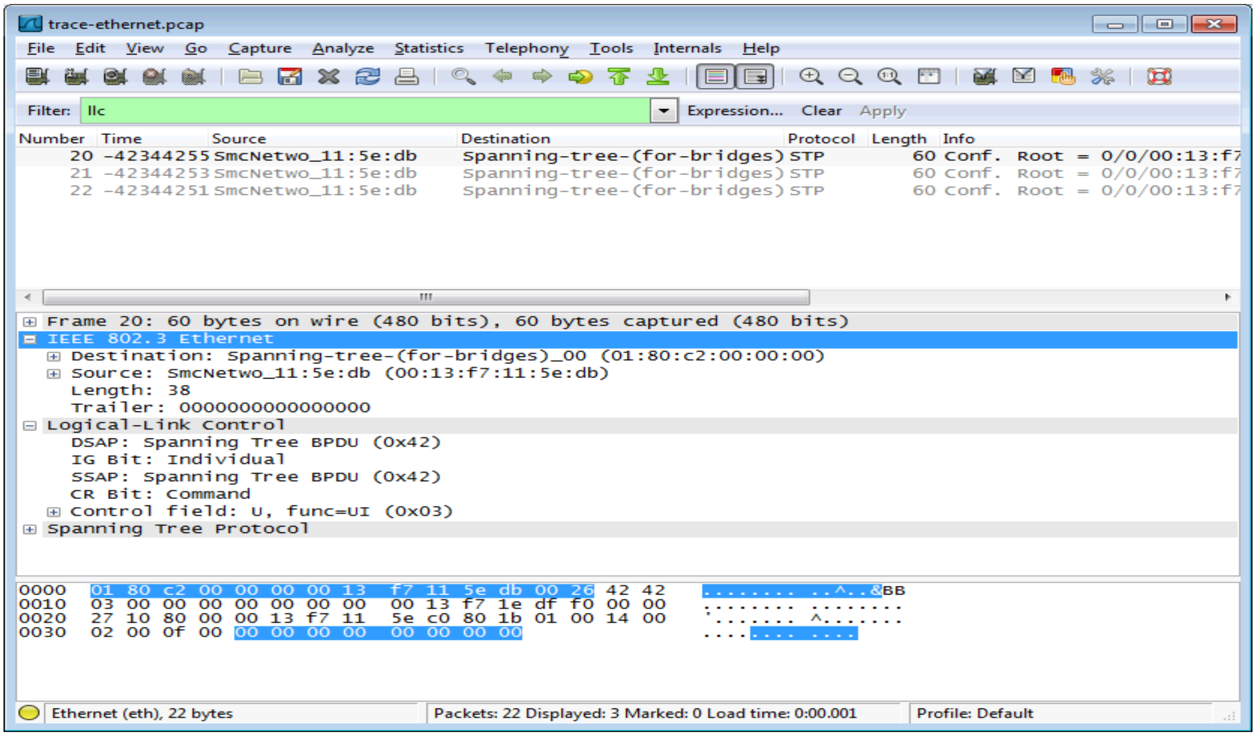
    b. Which bit of the Ethernet address is used to determine whether it is unicast or multicast/broadcast?

Step 6 - IEEE 802.3

We return again to *the trace file that you downloaded earlier from -* http://scisweb.ulster.ac.uk/~kevin/com320/labs/wireshark/trace-ethernet.pcap

You can reopen the trace file from the location you downloaded to e.g. Local Disk (C):\downloads or simply select *File menu* and *Open Recent.* The trace file should be listed there.

There are some IEEE 802.3 frames in the trace supplied. To search for IEEE 802.3 packets, enter a display filter (above the top panel of the Wireshark window) of “llc” (that was lowercase “LLC”) because the IEEE 802.3 format has the LLC protocol on top of it (and do not forget to click Apply) to apply the filter. LLC is also present on top of IEEE 802.11 wireless, but it is not present on DIX Ethernet. You should now see three packets like in the figure below.


trace-ethernet.pcap
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter: llc
Expression... Clear Apply

```
Number  Time      Source               Destination                  Protocol Length Info
    20 -42344255 SmcNetwo_11:5e:db    Spanning-tree-(for-bridges)  STP          60 Conf. Root = 0/0/00:13:f7
    21 -42344253 SmcNetwo_11:5e:db    Spanning-tree-(for-bridges)  STP          60 Conf. Root = 0/0/00:13:f7
    22 -42344251 SmcNetwo_11:5e:db    Spanning-tree-(for-bridges)  STP          60 Conf. Root = 0/0/00:13:f7
```

```
⊞ Frame 20: 60 bytes on wire (480 bits), 60 bytes captured (480 bits)
⊟ IEEE 802.3 Ethernet
  ⊞ Destination: Spanning-tree-(for-bridges)_00 (01:80:c2:00:00:00)
  ⊞ Source: SmcNetwo_11:5e:db (00:13:f7:11:5e:db)
    Length: 38
    Trailer: 0000000000000000
⊟ Logical-Link Control
    DSAP: Spanning Tree BPDU (0x42)
    IG Bit: Individual
    SSAP: Spanning Tree BPDU (0x42)
    CR Bit: Command
  ⊞ Control field: U, func=UI (0x03)
⊞ Spanning Tree Protocol
```

```
0000  01 80 c2 00 00 00 00 13  f7 11 5e db 00 26 42 42   ........ ..^..&BB
0010  03 00 00 00 00 00 00 00  00 13 f7 1e df f0 00 00   ........ ........
0020  27 10 80 00 00 13 f7 11  5e c0 80 1b 01 00 14 00   '....... ^.......
0030  02 00 0f 00 00 00 00 00  00 00 00 00               ........ ....
```

Ethernet (eth), 22 bytes | Packets: 22 Displayed: 3 Marked: 0 Load time: 0:00.001 | Profile: Default

Figure 7: IEEE 802.3 frames with Ethernet and LLC header detail

Have a look at the details of an IEEE 802.3 frame, including the LLC header for instance no 20 in the supplied trace.

The figure shows the details for our trace. Observe that the Type field is now a Length field. In our example, the frame is short enough that there is also padding of zeros identified as a Trailer or Padding.

The changes lead to a few questions for you to ponder:

1. How long are the combined IEEE 802.3 and LLC headers compared to the DIX Ethernet headers? You can use Wireshark to work this out. *Note that the Trailer/Padding and Checksum may be shown as part of the header, but they come at the end of the frame.*

2. How does the receiving computer know whether the frame is DIX Ethernet or IEEE 802.3?

3. If IEEE 802.3 has no Type field, then how is the next higher layer determined? Use Wireshark to look for the demultiplexing key.

Answers to Step 2: Inspecting the Trace

1. The MAC address is 00:25:64:d5:10:8b.

2. ARP uses the reserved MAC broadcast address 00:00:00:00:00:00 in an ARP Request to discover the IP address of 128.208.2.42. It then stores that information in a local ARP Cache allowing unlimited communication with 128.208.2.42 (for a period of time) without the need for further broadcasts. Note also that it knew the IP address but it needs a MAC address to know which actual machine it is.

3. The ARP protocol type is set to 'IP' because we are asking ARP to resolve an IP address. ARP can be used to resolve addresses for other network protocols as well.

Answers to Step 3: Ethernet Frame Structure

1. Opcode (1) or (0x0001) means ARP request and opcode (2) or (0x0002) means ARP reply.

2. The hex value for the Ethernet Frame type field is 0x0806, for ARP.

Answers to Step 5: Broadcast Addresses

1. The broadcast address is ff:ff:ff:ff:ff:ff. This is 48 bits of “all 1s”written in standard form.

2. The broadcast/multicast or “group” bit is shown by Wireshark as “.... ...1 ....” or a one in the low-order bit of the first address byte. We could also write this 01:00:00:00:00:00. This bit is actually the bit that is transmitted on the wire first because Ethernet defines the transmission order to be the “least significant bit of each byte first”.

Answers to Step 6: IEEE 802.3

1. The IEEE 802.3 header is 14 bytes, the same as DIX Ethernet. (Both also have a trailer with a checksum and padding if needed.) LLC adds another 3 bytes of headers for a total of 17 bytes of headers.

2. The DIX Ethernet Type field and IEEE 802.3 Length field are in the same position. If the value is less than 0x600 (1536) then it is interpreted as a frame length. If the value is larger than 0x600 (1536) then it is interpreted as a Type value.

3. IEEE 802.3 adds the LLC header immediately after the IEEE 802.3 header to convey the next higher layer protocol. LLC uses a single initial byte called the DSAP (destination service access point) rather than the two bytes in the Type field.