also known as wireshark

# Ethereal: Getting Started

CSCI 3171: Network Computing

## Introduction

One's understanding of network protocols can often be greatly deepened by "seeing protocols in action" and by "playing around with protocols" – observing the sequence of messages exchanged between two protocol entities, delving down into the details of protocol operation, and causing protocols to perform certain actions and then observing these actions and their consequences.

In this lab you will be examining a *pre-captured* networking session. This means that the packets to and from a computer have been recorded and saved in a file. By examining these files, you'll observe the network protocols interacting and exchanging messages with protocol entities executing elsewhere in the Internet.

The basic tool for observing the messages exchanged between executing protocol entities is called a **packet sniffer**. As the name suggests, a packet sniffer captures ("sniffs") messages 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 messages. 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. Similarly, received packets are never explicitly addressed to the packet sniffer. Instead, a packet sniffer receives a *copy* of packets that are sent/received from/by application and protocols executing on your machine.

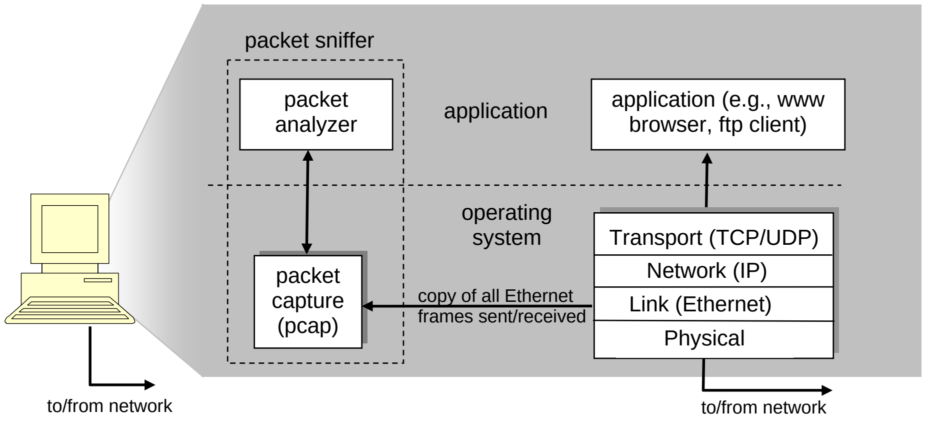


**Figure 1:** Packet sniffer structure

Figure 1 shows the structure of a packet sniffer. At the right of Figure 1 are the protocols (in this case, Internet protocols) and applications (such as a web browser or ftp client) that normally run on a computer. The packet sniffer, shown within the dashed rectangle in Figure 1 is an addition to the usual software on a computer, and consists of two parts:

1. The **packet capture library** receives a copy of every link-layer frame that is sent from or received by your computer. Recall from the discussion from section 1.7.2 in the text (Figure 1.18[1]) that 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 all messages sent/received from/by all protocols and applications executing in your computer.

2. 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 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 1. The packet analyzer understands the format of Ethernet frames, and so can identify the IP datagram within an Ethernet frame. It also understands the IP datagram format, so that it can extract the TCP segment within the IP datagram. Finally, it understands the TCP segment structure, so it can extract the HTTP message contained in the TCP segment. Finally, it understands the HTTP protocol and so, for example, knows that the first bytes of an HTTP message will contain the string "GET," "POST," or "HEAD," as shown in Figure 2.8 in the text.

We will be using the Ethereal packet sniffer [http://www.ethereal.com] 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, Ethereal is a packet analyzer that uses a packet capture library in your computer). Ethereal is a free network protocol analyzer that runs on Windows, Linux/Unix, and Mac computers. It's an ideal packet analyzer for our labs – it is stable, has a large user base and well-documented support that includes:

- a user-guide (http://www.ethereal.com/docs/user-guide/)
- man pages (http://www.ethereal.com/ethereal.1.html )
- a detailed FAQ (http://www.ethereal.com/faq.html )
- rich functionality that includes the capability to analyze more than 500 protocols
- a well-designed user interface.

It operates in computers using Ethernet to connect to the Internet, as well as so-called point-to-point protocols such as PPP.

[1] Figure numbers refer to figures in the 3rd edition of our text.

# Using Ethereal in the Department

For security reasons you need to run ethereal off *torch.cs.dal.ca*.

**Using Ethereal from a Windows machine:**

1. Open an X-windows session (double-click on the X-win32 shortcut).
2. Define host “torch.cs.dal.ca” as an xhost.
3. Double-click on “putty”.
4. Type the command “ssh username@torch.cs.dal.ca”. Enter password.
5. Type command: “Export DISPLAY=hostname:0.0”
   Where hostname is the name of the machine you are connecting from.
   Note that this cannot be localhost:0.0 as the icons will not display properly.
6. Type command “/opt/ethereal/bin/ethereal” and hit enter!

**Using Ethereal from a MAC machine:**

1. Run X11.
2. Type command “ssh –X username@torch.cs.dal.ca” Enter password.
3. Type command “/opt/ethereal/bin/ethereal” and hit enter.

**Using Ethereal from a Unix machine:**

1. Type command: “Export DISPLAY=hostname:0.0”
   Where hostname is the name of the machine you are connecting from.
   Note that this cannot be localhost:0.0 as the icons will not display properly.
2. Type command “/opt/ethereal/bin/ethereal” and hit enter.

# Using Ethereal on Your Own Computer

In order to run Ethereal, you will need to have access to a computer that supports both Ethereal and the *libpcap* packet capture library. Windows XP has libcap pre-installed. If the *libpcap* software is not installed within your operating system, you will need to install *libpcap* or have it installed for you in order to use Ethereal. See http://www.ethereal.com/download.html for a list of supported operating systems and download sites.

Download and install the Ethereal and (if needed) *libpcap* software:

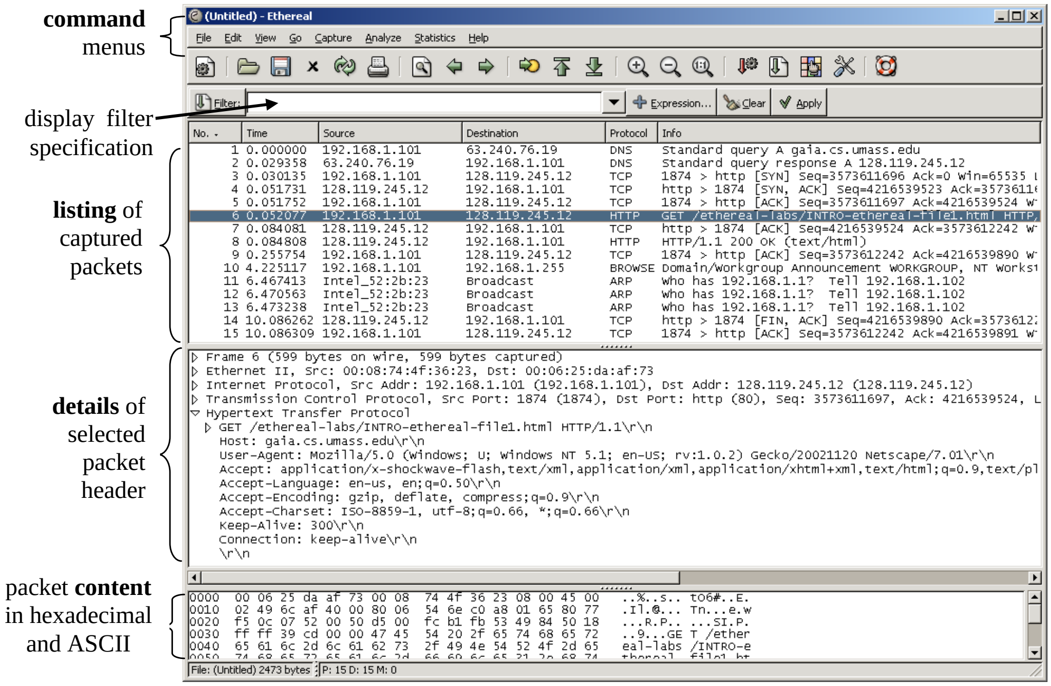
- If needed, download and install the *libpcap* software. Pointers to the *libpcap* software are provided from the Ethereal download pages. For older Windows machines, the *libpcap* software is known as *WinPCap*, and can be found at http://winpcap.polito.it/ See FAQ question #2 at http://winpcap.polito.it/ To determine whether or not *WinPCap* is already installed on your machine.

- Go to http://www.ethereal.com and download and install the Ethereal binary for your computer.
- Download the Ethereal user guide. You will most likely only need Chapters 1 and 3.

The Ethereal FAQ has a number of helpful hints and interesting tidbits of information, particularly if you have trouble installing or running Ethereal.

## Running Ethereal

When you run the Ethereal program, the Ethereal graphical user interface shown in Figure 2 will de displayed. Initially, no data will be displayed in the various windows.


command menus
display filter specification
listing of captured packets
details of selected packet header
packet content in hexadecimal and ASCII
(Untitled) - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter:
Expression...
Clear
Apply
File: (Untitled) 2473 bytes
P: 15 D: 15 M: 0

```
No. .  Time       Source           Destination      Protocol Info
     1 0.000000   192.168.1.101    63.240.76.19     DNS      Standard query A gaia.cs.umass.edu
     2 0.029358   63.240.76.19     192.168.1.101    DNS      Standard query response A 128.119.245.12
     3 0.030135   192.168.1.101    128.119.245.12   TCP      1874 > http [SYN] Seq=3573611696 Ack=0 Win=65535 L
     4 0.051731   128.119.245.12   192.168.1.101    TCP      http > 1874 [SYN, ACK] Seq=4216539523 Ack=357361
     5 0.051752   192.168.1.101    128.119.245.12   TCP      1874 > http [ACK] Seq=3573611697 Ack=4216539524 W
     6 0.052077   192.168.1.101    128.119.245.12   HTTP     GET /ethereal-labs/INTRO-ethereal-file1.html HTTP,
     7 0.084081   128.119.245.12   192.168.1.101    TCP      http > 1874 [ACK] Seq=4216539524 Ack=3573612242 W
     8 0.084808   128.119.245.12   192.168.1.101    HTTP     HTTP/1.1 200 OK (text/html)
     9 0.255754   192.168.1.101    128.119.245.12   TCP      1874 > http [ACK] Seq=3573612242 Ack=4216539890 W
    10 4.225117   192.168.1.101    192.168.1.255    BROWSE   Domain/Workgroup Announcement WORKGROUP, NT Works
    11 6.467413   Intel_52:2b:23   Broadcast        ARP      Who has 192.168.1.1?  Tell 192.168.1.102
    12 6.470563   Intel_52:2b:23   Broadcast        ARP      Who has 192.168.1.1?  Tell 192.168.1.102
    13 6.473238   Intel_52:2b:23   Broadcast        ARP      Who has 192.168.1.1?  Tell 192.168.1.102
    14 10.086262  128.119.245.12   192.168.1.101    TCP      http > 1874 [FIN, ACK] Seq=4216539890 Ack=357361
    15 10.086309  192.168.1.101    128.119.245.12   TCP      1874 > http [ACK] Seq=3573612242 Ack=4216539891 W
```

```
▷ Frame 6 (599 bytes on wire, 599 bytes captured)
▷ Ethernet II, Src: 00:08:74:4f:36:23, Dst: 00:06:25:da:af:73
▷ Internet Protocol, Src Addr: 192.168.1.101 (192.168.1.101), Dst Addr: 128.119.245.12 (128.119.245.12)
▷ Transmission Control Protocol, Src Port: 1874 (1874), Dst Port: http (80), Seq: 3573611697, Ack: 4216539524, L
▽ Hypertext Transfer Protocol
  ▷ GET /ethereal-labs/INTRO-ethereal-file1.html HTTP/1.1\r\n
    Host: gaia.cs.umass.edu\r\n
    User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.0.2) Gecko/20021120 Netscape/7.01\r\n
    Accept: application/x-shockwave-flash,text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/pl
    Accept-Language: en-us, en;q=0.50\r\n
    Accept-Encoding: gzip, deflate, compress;q=0.9\r\n
    Accept-Charset: ISO-8859-1, utf-8;q=0.66, *;q=0.66\r\n
    Keep-Alive: 300\r\n
    Connection: keep-alive\r\n
    \r\n
```

```
0000   00 06 25 da af 73 00 08   74 4f 36 23 08 00 45 00   ..%..s.. tO6#..E.
0010   02 49 6c af 40 00 80 06   54 6e c0 a8 01 65 80 77   .Il.@... Tn...e.w
0020   f5 0c 07 52 00 50 d5 00   fc b1 fb 53 49 84 50 18   ...R.P.. ...SI.P.
0030   ff ff 39 cd 00 00 47 45   54 20 2f 65 74 68 65 72   ..9...GE T /ether
0040   65 61 6c 2d 6c 61 62 73   2f 49 4e 54 52 4f 2d 65   eal-labs /INTRO-e
```

**Figure 2:** Ethereal Graphical User Interface

The Ethereal 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 menu. The File menu allows you to save captured packet data or open a file containing previously captured packet data, and exit the Ethereal application.

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

- The **packet-header details window** provides details about the packet selected (highlighted) in the packet listing window. (To select a packet in the packet listing window, place the cursor over the packet's one-line summary in the packet listing window and click with the left mouse button.). These details include information about the Ethernet frame and IP datagram that contains this packet. The amount of Ethernet and IP-layer detail displayed can be expanded or minimized by clicking on the right-pointing or down-pointing arrowhead to the left of the Ethernet frame or IP datagram line in the packet details window. If the packet has been carried over TCP or UDP, TCP or UDP details will also be displayed, which can similarly be expanded or minimized. Finally, details about the highest level protocol that sent or received this packet are also provided.

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

- Towards the top of the Ethereal 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 Ethereal hide (not display) packets except those that correspond to HTTP messages.

## Taking Ethereal for a Test Run

The best way to learn about any new piece of software is to try it out! Do the following

1. Start up your favorite web browser, which will display your selected homepage.

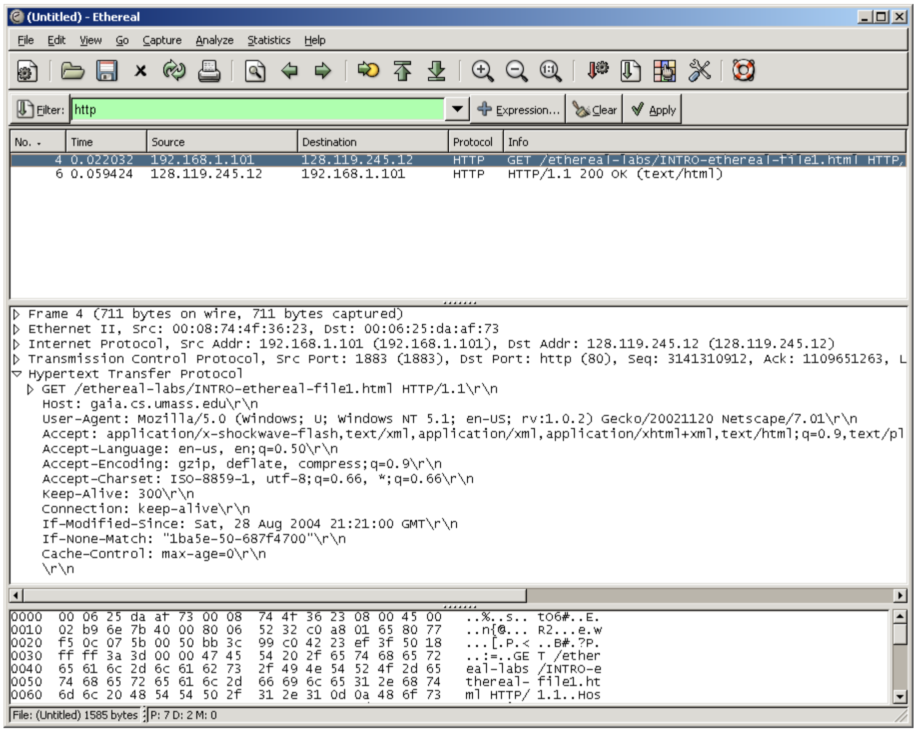
2. Start up the Ethereal software. You will initially see a window similar to that shown in Figure 2, except that no packet data will be displayed in the packet-listing, packet-header, or packet-contents window.

3. To open a captured session, select the File pull down menu and select *Open*.

4. Type in "http" (without the quotes, and in lower case – all protocol names are in lower case in Ethereal) into the display filter specification window at the top of the main Ethereal window. Then select *Apply* (to the right of where you entered "http"). This will cause only HTTP message to be displayed in the packet-listing window.

5. Select the first http message shown in the packet-listing window. This should be an HTTP GET. When you select the HTTP GET message, the Ethernet frame, IP datagram, TCP segment, and HTTP message header information will be displayed in the packet-header window[2]. By clicking on right-pointing and down-pointing arrowsheads to the left side of the packet details window, *minimize* the amount of Frame, Ethernet, Internet Protocol, and Transmission Control Protocol information displayed. *Maximize* the amount information displayed about the HTTP protocol. Your Ethereal display should now look roughly as shown in Figure 5 but will many more packets shown.

6. Exit Ethereal



Filter: http

```
No. -  Time       Source          Destination     Protocol  Info
     4 0.022032   192.168.1.101   128.119.245.12  HTTP      GET /ethereal-labs/INTRO-ethereal-file1.html HTTP,
     6 0.059424   128.119.245.12  192.168.1.101   HTTP      HTTP/1.1 200 OK (text/html)
```

```
▷ Frame 4 (711 bytes on wire, 711 bytes captured)
▷ Ethernet II, Src: 00:08:74:4f:36:23, Dst: 00:06:25:da:af:73
▷ Internet Protocol, Src Addr: 192.168.1.101 (192.168.1.101), Dst Addr: 128.119.245.12 (128.119.245.12)
▷ Transmission Control Protocol, Src Port: 1883 (1883), Dst Port: http (80), Seq: 3141310912, Ack: 1109651263, L
▽ Hypertext Transfer Protocol
  ▷ GET /ethereal-labs/INTRO-ethereal-file1.html HTTP/1.1\r\n
    Host: gaia.cs.umass.edu\r\n
    User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.0.2) Gecko/20021120 Netscape/7.01\r\n
    Accept: application/x-shockwave-flash,text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/pl
    Accept-Language: en-us, en;q=0.50\r\n
    Accept-Encoding: gzip, deflate, compress;q=0.9\r\n
    Accept-Charset: ISO-8859-1, utf-8;q=0.66, *;q=0.66\r\n
    Keep-Alive: 300\r\n
    Connection: keep-alive\r\n
    If-Modified-Since: Sat, 28 Aug 2004 21:21:00 GMT\r\n
    If-None-Match: "1ba5e-50-687f4700"\r\n
    Cache-Control: max-age=0\r\n
    \r\n
```

```
0000  00 06 25 da af 73 00 08  74 4f 36 23 08 00 45 00   ..%..s.. tO6#..E.
0010  02 b9 6e 7b 40 00 80 06  52 32 c0 a8 01 65 80 77   ..n{@... R2...e.w
0020  f5 0c 07 5b 00 50 bb 3c  99 c0 42 23 ef 3f 50 18   ...[.P.< ..B#.?P.
0030  ff ff 3a 3d 00 00 47 45  54 20 2f 65 74 68 65 72   ..:=..GE T /ether
0040  65 61 6c 2d 6c 61 62 73  2f 49 4e 54 52 4f 2d 65   eal-labs /INTRO-e
0050  74 68 65 72 65 61 6c 2d  66 69 6c 65 31 2e 68 74   thereal- file1.ht
0060  6d 6c 20 48 54 54 50 2f  31 2e 31 0d 0a 48 6f 73   ml HTTP/ 1.1..Hos
```

File: (Untitled) 1585 bytes | P: 7 D: 2 M: 0

**Figure 5:** Ethereal display after step 9

## Acknowledgements

This lab has been adapted from the work of by J.F. Kurose, K.W. Ross © 2005.