

## 2<sup>nd</sup> Assignment: “Ethereal”

### 1. 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. This can be done in simulated scenarios or in a “real” network environment such as 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 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 your computer. The packet sniffer, shown within the dashed rectangle in Figure 1 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. Recall from the discussion in the class 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.

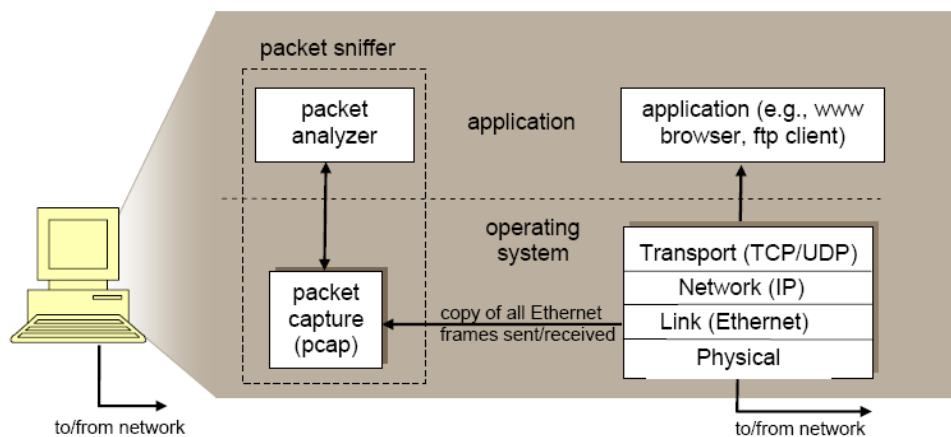


Figure 1: Packet sniffer structure

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.” We will be using the **Ethereal**<sup>1</sup> packet sniffer 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 this assignment – it is stable, has a large user base and well-documented support that includes:

- (<http://www.ethereal.com/docs/user-guide/>), a user-guide
- (<http://www.ethereal.com/ethereal.1.html>), man pages
- (<http://www.ethereal.com/faq.html>), and a detailed FAQ

**Ethereal** also has rich functionality that includes the capability to analyze more than 500 protocols, and 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. Incidentally, some people pronounce the name Ethereal as “etherreal,” while others pronounce it “e-thir-E-al,” as in the English word ethereal, which means ghostly or insubstantial. The name’s origin comes from the Ethernet protocol.

## 2. How/where to find Ethereal

For this assignment you can either work from the labs using any of the available systems or by downloading/installing Ethereal on your personal computer. If you decide to work from the Labs then you can run the GUI version of the program by typing “ethereal &” from the command line and then select to work in **unprivileged mode**.

On the other hand, if you prefer to work from home you will need to have access to a computer that supports both Ethereal and the *libpcap* packet capture library. 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:

---

<sup>1</sup> <http://www.ethereal.com>

- If needed, download and install the *libpcap* software. Pointers to the *libpcap* software are provided from the Ethereal download pages. For 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.

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

### 3. Running Ethereal

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

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 are 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 Ethereal 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 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**<sup>2</sup>, 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.

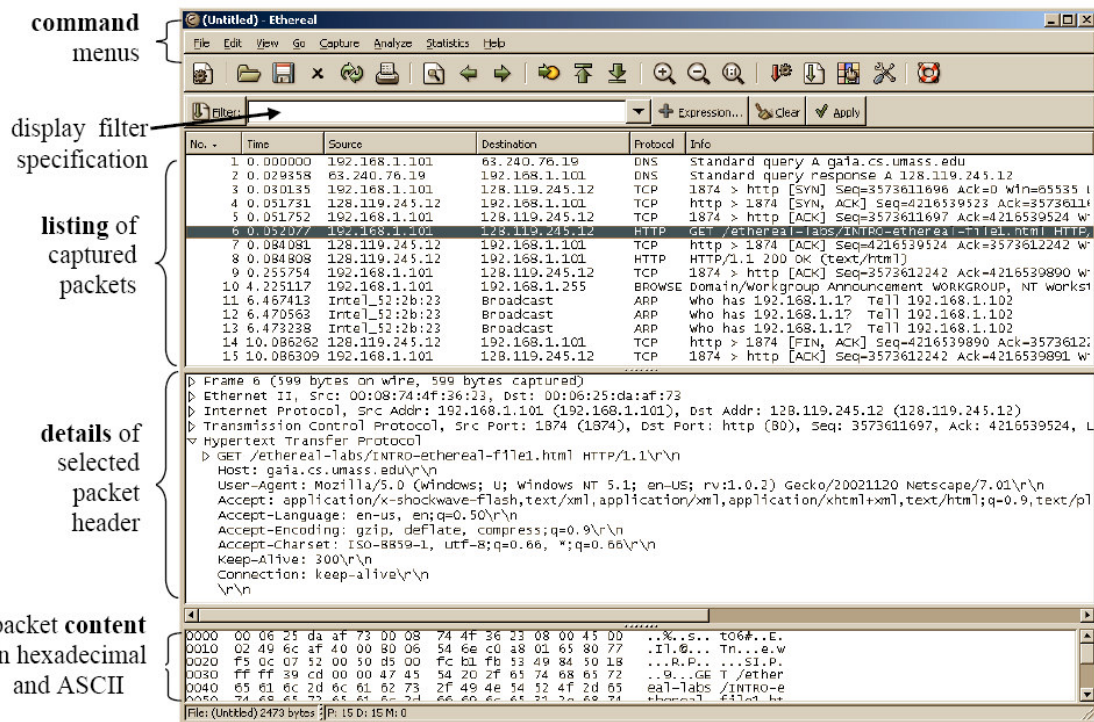


Figure 2: Ethereal GUI

## 4. Deliverables

### (a) Designing Ethereal Filters

**Ethereal** and **Tethereal** share a powerful filter engine that helps remove the noise from a packet trace and lets you see only the packets that interest you. If a packet meets the requirements expressed in your filter, then it is displayed in the list of packets. Display filters let you compare the fields within a protocol against a specific value, compare fields against fields, and check the existence of specified fields or protocols. For this part you have to design filters that select (hence, display) only that packets that:

- i. Use TCP protocol
- ii. Use UDP protocol

<sup>2</sup> This holds for the MS Windows version of Ethereal. In Linux GUI the “packet display filter” is located at the bottom of the window.

- iii. Use ICMP
- iv. Have the MAC (Ethernet) broadcast address as there destination
- v. Have an IP address with the first byte (e.g, 74.13.21.2) closer to the first two digits of your SID <sup>3</sup>
- vi. Have an IP address with the first byte (e.g, 74.13.21.2) closer to the first two digits of your SID but are not TCP
- vii. Have an IP address with the first byte (e.g, 74.13.21.2) closer to the first two digits of your SID but are neither TCP nor UDP

About how to compose filters refer to:

- <http://www.ethereal.com/docs/man-pages/ethereal-filter.4.htm> <sup>4</sup>

To get the required statistics:

- Command menus [See Figure 2] → Statistics → Summary

For each filter you have to turn in:

- The syntax of the filter
- The number of displayed packets after you activate the filter (this holds for all the remaining questions)
- Time elapsed between the first and the last packet
- Average packet size
- Total number of bytes
- Average bytes per second.
  - i. Write a formula that will give you the average bytes per second
  - ii. Use the data from the statistics to verify that the two results (that is, the value you get from the Statistics and the value you get from your formula) are the same.

## (b) Statistics – Flows – Endpoints

For this part you have to:

- i. Open from the command menus → Statistics → Protocol Hierarchy window.
  - a. Write a small discussion about the information provided by this feature of Ethereal.
  - b. Also give a brief description of each protocol that you see and at which OSI layer it operates
- ii. Open from the command menus → Statistics → Conversations.
  - a. Write a small discussion about the information provided by this feature of Ethereal.
  - b. Find the MAC layer (Layer-2 OSI) ‘conversation’ with the maximum number of packets. What is the percentage of packets transferred from A to B and

---

<sup>3</sup> That is, you first find the IP address (from those present in the file) that is closer to your SID (only the first two digits) and hardcode it in your filter.

<sup>4</sup> You don’t have to read everything, the knowledge needed for this assignment are only the very basics

- what from B to A (compare to total number of packets exchange by A and B) .
- c. Repeat b, but this time for the IP layer.
- d. How many are the IP Conversation and how many are the TCP conversations<sup>5</sup>. Which layer (IP or TCP) has the most conversations and why? How are the IP and how the TCP conversation defined?
- iii. Open from the command menus → Statistics → Endpoints.
  - a. Write a small discussion about the information provided by this feature of Ethereal.
  - b. Find the MAC layer (Layer-2 OSI) endpoint with the maximum number of packets. What do Tx and Rx stand for?
  - c. Repeat b, but this time for the IP layer.

### (c) I/O Graphs

- Open from the command menus → Statistics → IO Graphs.
  - i. Write a small discussion about the information provided by this feature of Ethereal.
  - ii. Plot on the same graph
    - a. All the packets (black color)
    - b. Only the TCP (red color)
    - c. Only the UDP (blue color)
    - d. Repeat a,b and c but now the y-axis of your graph will be in packets/tick rather than bytes/tick. Can you see any difference? Why? Describe your results.
  - iii. Select from the “Conversation Window” the flow (numbered as they appear on the window) that is closer to the last digit of your SID. Using the IO Graph feature of Ethereal, for that flow you have selected plot on the same graph:
    - a. All the packets belonging to that flow (black color)
    - b. All the packets from A to B (red color)
    - c. All the packets from B to A (blue color)
    - d. Repeat a,b and c but now the y-axis of your graph will be in packets/tick rather than bytes/tick. Can you see any difference? Why? Describe your results.
  - iv. Write a filter in order to keep only the Ethernet frames that have as destination the broadcast address (MAC address FF:FF:FF:FF:FF:FF). Which is the higher layer protocol that generated this traffic (provide its name and a brief description). Using the IO Graph feature of Ethereal plot these frames. Provide the plot with a small description of your results along with an explanation of the nature of the waveform.

---

<sup>5</sup> TCP conversations are offered refer to as TCP flows in the networking literature.

## 5. ReadMe

For 4.a and 4.b provide an answer to each question as well as comments or anything unusual that you might have observed. For 4.c provide the plots that you have produced <sup>6</sup> as well as any filters that you might have design and an answer to each of the questions asked. Also note that your assignments should read like a report and not a collage of plots and numbers. The quality of your reports will be graded.

**Trace:** You will all have to use the same trace. The trace can be found on-line at:  
`acmserver.cs.ucr.edu/~anirban/5min-run4.eth`

**Turn-in:** You have to turn in your written report by 05/21/06 11:59 PM.

---

**BEST OF LUCK!**

---

### References:

<http://gaia.cs.umass.edu/ethereal-labs/>  
<http://www.ethereal.com/>

---

<sup>6</sup> Use the 'print screen' button to capture the figure.