



## HKUSTx: ELEC1200.3x A System View of Communications: From Signals to Packets (Part 3)



Bookmarks

- ▶ Pre-course Materials
- ▶ Topic 1: Course Overview
- ▶ Topic 2: The Link Layer
- ▶ Topic 3: The Network Layer
- ▶ Topic 4: Routing
- ▶ Topic 5: The Transport Layer
- ▶ Topic 6: Reliable Transfer Protocols

Topic 7: The Application Layer &gt; 7.4 Lab 4 - Application Layer &gt; Lab 4 - Overall Objectives



Bookmark

## LAB 4 - OVERALL OBJECTIVES


In this lab, we will learn the Network and Application Layer protocols. We will study the Hypertext Transfer Protocol (HTTP) by using a network protocol analyzer called **Wireshark**. In particular, we will complete five tasks.

1. In task 1, we will study the basics about how to read information about HTTP messages received and sent by your computer using **Wireshark**.
2. In task 2, we will study the GET message sent by your computer and the response from the server when your browser downloads a simple one-line html file.
3. In task 3, we will study the HTTP conditional GET and the server response when your web browser performs object caching.
4. In task 4, we will study the HTTP messages when you download a HTML file with embedded objects.


### Overview

## ▼ Topic 7: The Application Layer


### 7.1 Application Layer

Week 4 Quiz due Feb 15, 2016 at 15:30 UTC 


### 7.2 Hypertext Transfer Protocol (HTTP)

Week 4 Quiz due Feb 15, 2016 at 15:30 UTC 

### 7.3 Domain Name System (DNS)

Week 4 Quiz due Feb 15, 2016 at 15:30 UTC 

### 7.4 Lab 4 - Application Layer

Lab due Feb 15, 2016 at 15:30 UTC 

## ► Topic 8: Course Review

## ► MATLAB download and tutorials

A packet sniffer is a basic tool for observing the messages exchanged between entities over a network. Once you run the packet sniffer on your computer, it captures (“sniffs”) messages being sent from or received by your computer. It stores and displays the contents of the various protocol fields in these captured messages. The packet sniffer itself is passive, i.e. it never sends packets itself, nor are any received packets explicitly addressed to the packet sniffer. Instead, the packet sniffer captures copies of the packets that are sent from or received by the applications and protocols executing on your machine.

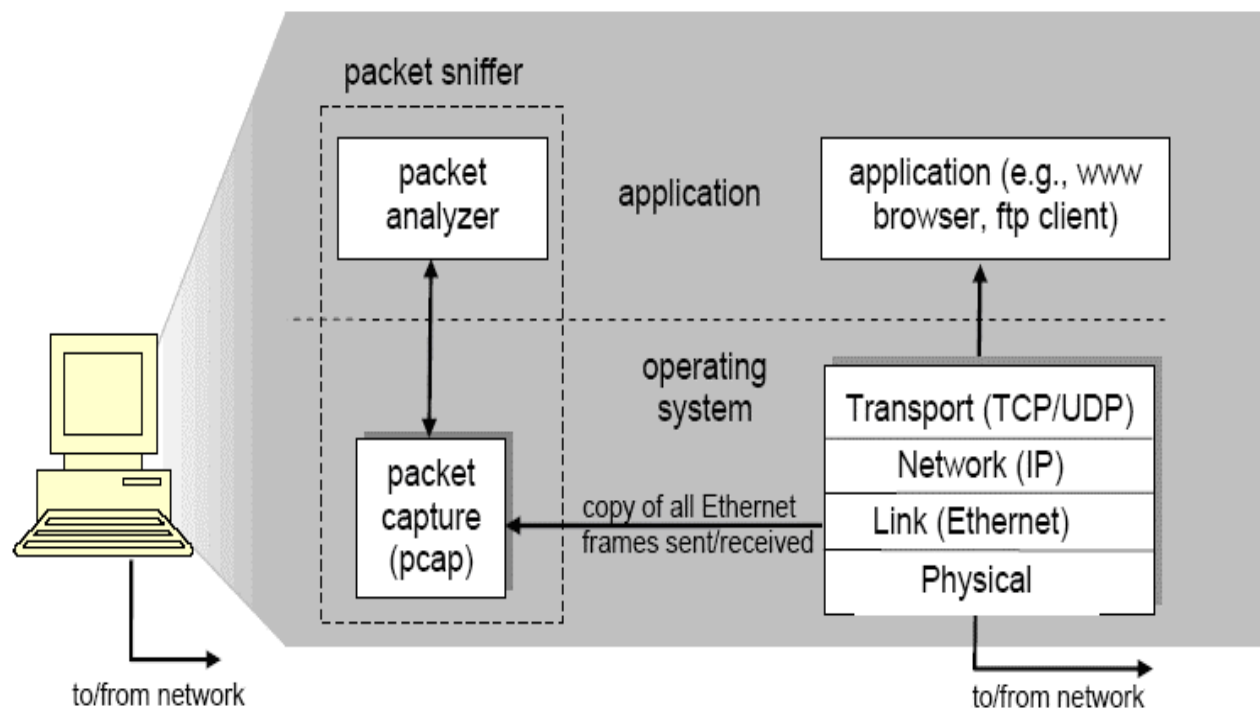


Figure 1: Basic structure of Packet sniffer

Figure 1 shows the basic structure of a packet sniffer. The parts in the right of Figure 1 are the Internet protocols and applications (e.g. web browser, ftp client) that normally run on your computer. The packet sniffer, which is highlighted by the dashed rectangle, is an addition to the usual software in your computer. It is composed of two parts: the packet capture library and the packet analyzer.

The packet capture library receives a copy of every link-layer frame that is sent from or received by your computer. Recall from the lecture that messages exchange by higher layer protocols such as HTTP, FTP, DNS, TCP, UDP or IP all are eventually encapsulated in link-layer frames that are transmitted over physical media, such as an Ethernet cable. In Fig. 1, the assumed physical medium is an Ethernet connection. Packets from all upper layer protocols are eventually encapsulated within an Ethernet frame. Assuming that your computer is only using one link-layer connection, capturing all link-layer frames thus gives you all messages sent from or received by all protocols and applications executing in your computer.

The packet analyzer 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 the protocols. For example, suppose we are interested in displaying the various fields in messages exchange by the HTTP protocol in Fig. 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 contain strings like “GET,” “POST” or “HEAD.”



© edX Inc. All rights reserved except where noted. EdX, Open edX and the edX and Open EdX logos are registered trademarks or trademarks of edX Inc.

