**Capture Network Traffic**

**Using Wireshark**

**Dr. Lixin Wang**

**1. What is Wireshark?**

It is a **packet sniffer** which is a basic tool for observing the packets exchanged between network entities. As its 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 packets. It can also be used to analyze captured packets.

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. 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.

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”.



**Figure 1: Packet sniffer structure**

**2. The Wireshark interface**



**Figure 2: Wireshark Graphical User Interface**

**3. How to use the display filter in Wireshark?**

An online tutorial

<https://www.ictshore.com/wireshark/wireshark-filter-tutorial/>

**4. If the display filter “ftp-data” is applied on the Wireshark interface, each packet in the list is an FTP data packet and belongs to a file contained in the capture.**

**Questions:**

1. How many files are contained in the capture or a PCAP file?
2. Given an FTP data packet in the list, which file does it belong to?
3. Can we recover the files based on the FTP data packets in the list?

**5. How to recover a file from the captured FTP data packets?**

* Filter ftp-data
* **Statistics -> conversations**
* On the TCP tab, check only “limit to display filter”
* Select one of the FTP sessions
* Click the button “**follow stream**”
* The stream number is the FTP session number of a file
* Change “show and save data **as** **raw**” (default is ASCII)
* Then click “save as” and save it as a **JPG** file