Final report

Qiyue Zhou

30119330

**Introduction**

Wireshark is a free and open-source packet analyzer. It is used for network troubleshooting, analysis, software and communications protocol development, and education. Originally named Ethereal, the project was renamed Wireshark in May 2006 due to trademark issues.

Wireshark is cross-platform, using the Qt widget toolkit in current releases to implement its user interface, and using pcap to capture packets; it runs on Linux, macOS, BSD, Solaris, some other Unix-like operating systems, and Microsoft Windows. There is also a terminal-based (non-GUI) version called TShark. Wireshark, and the other programs distributed with it such as TShark, are free software, released under the terms of the GNU General Public License.

Wireshark is very similar to tcpdump, but has a graphical front-end, plus some integrated sorting and filtering options.

Wireshark lets the user put network interface controllers into promiscuous mode (if supported by the network interface controller), so they can see all the traffic visible on that interface including unicast traffic not sent to that network interface controller's MAC address. However, when capturing with a packet analyzer in promiscuous mode on a port on a network switch, not all traffic through the switch is necessarily sent to the port where the capture is done, so capturing in promiscuous mode is not necessarily sufficient to see all network traffic. Port mirroring or various network taps extend capture to any point on the network. Simple passive taps are extremely resistant to tampering[citation needed].

On GNU/Linux, BSD, and macOS, with libpcap 1.0.0 or later, Wireshark 1.4 and later can also put wireless network interface controllers into monitor mode.

If a remote machine captures packets and sends the captured packets to a machine running Wireshark using the TZSP protocol or the protocol used by OmniPeek, Wireshark dissects those packets, so it can analyze packets captured on a remote machine at the time that they are captured.

**Features of Wireshark**

Wireshark is a data capturing program that "understands" the structure (encapsulation) of different networking protocols. It can parse and display the fields, along with their meanings as specified by different networking protocols. Wireshark uses pcap to capture packets, so it can only capture packets on the types of networks that pcap supports.

Data can be captured "from the wire" from a live network connection or read from a file of already-captured packets.

Live data can be read from different types of networks, including Ethernet, IEEE 802.11, PPP, and loopback.

Captured network data can be browsed via a GUI, or via the terminal (command line) version of the utility, TShark.

Captured files can be programmatically edited or converted via command-line switches to the "editcap" program.

Data display can be refined using a display filter.

Plug-ins can be created for dissecting new protocols.

VoIP calls in the captured traffic can be detected. If encoded in a compatible encoding, the media flow can even be played.

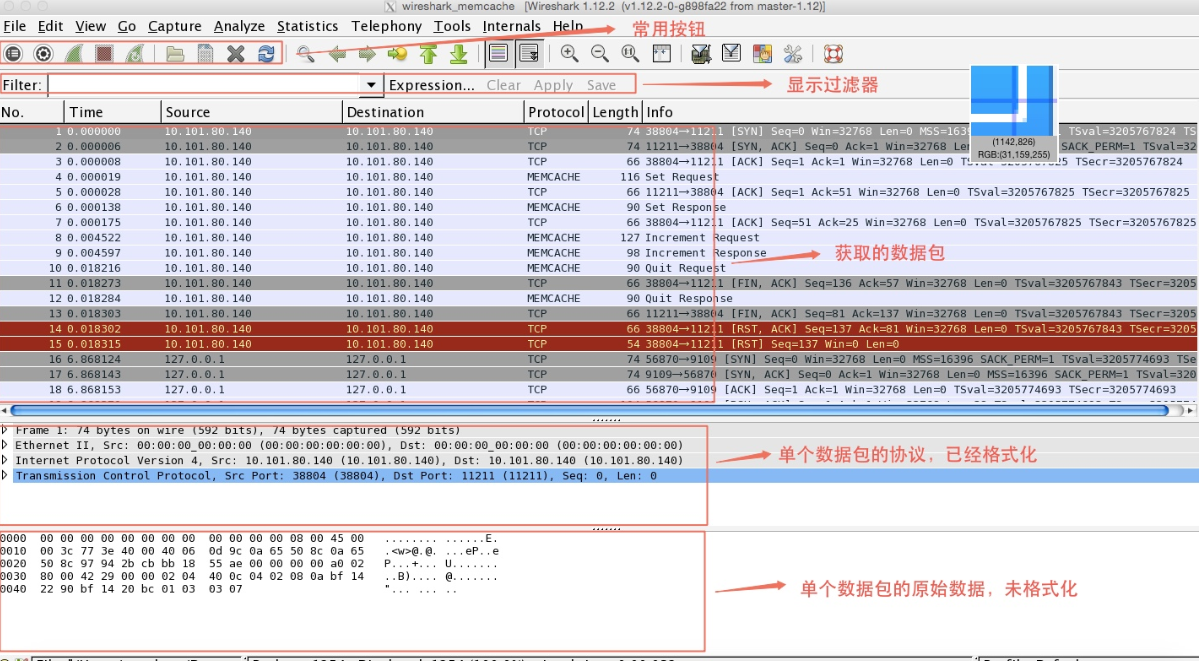
Raw USB traffic can be captured.

Wireless connections can also be filtered as long as they traverse the monitored Ethernet.

Various settings, timers, and filters can be set to provide the facility of filtering the output of the captured traffic.

Wireshark's native network trace file format is the libpcap format supported by libpcap and WinPcap, so it can exchange captured network traces with other applications that use the same format, including tcpdump and CA NetMaster. It can also read captures from other network analyzers, such as snoop, Network General's Sniffer, and Microsoft Network Monitor.

**Operation**



**Explanation:**

The functions of the commonly used buttons from left to right are:

1. List the available interfaces.

2. Some options that need to be set when capturing packets. Generally, the last setting result is retained.

3. Start a new packet capture.

4. Pause packet capture.

5. Continue this packet capture.

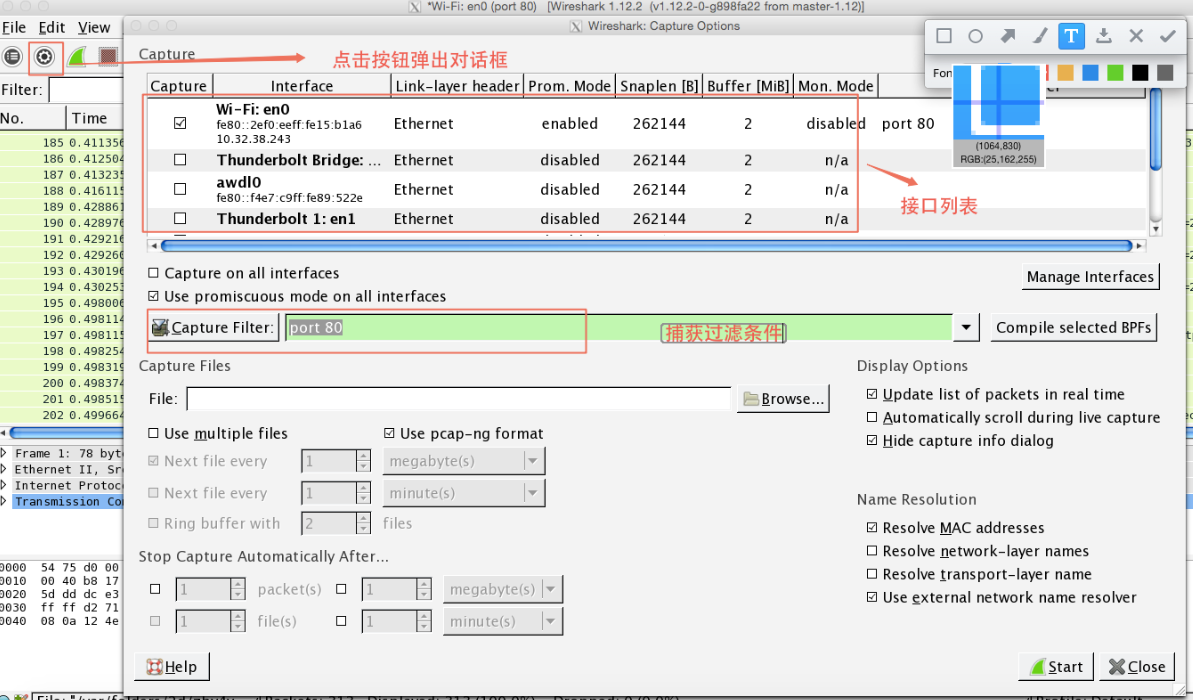
6. Open the capture file. You can open the file after capturing and saving. You can open not only files saved by the wireshark software, but also files saved by tcpdump with the -w parameter.

7. Save the file. Save the results of this packet capture or analysis.

8. Close the open file. After the file is closed, it will switch to the initial interface.

9. reload capture files.

**Set data scraping options**



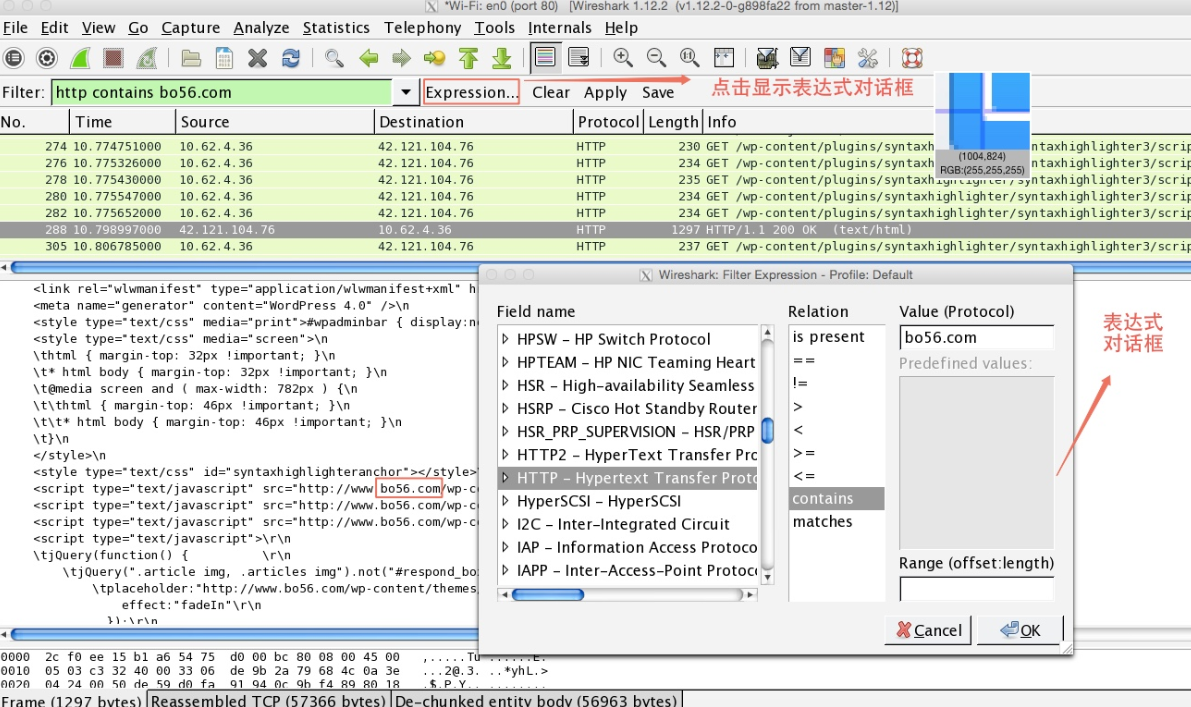
First, select the interface that needs to listen for data packets. The interface list area lists all available interfaces. If the check box in front of the interface is checked, it means that the interface is listening to capture packets.

Second, set the promiscuous mode. Setting promiscuous mode is to set the network card to promiscuous mode. If you do not set promiscuous mode, your computer can only get the data packets sent to your computer and the data packets going out of your computer. If promiscuous mode is set, you can capture all packets on the LAN. If the check box in front of "Use promiscuous mode on all interfaces" is selected in the window, the promiscuous mode is used for all interfaces. If you want to set it separately, you can double-click the interface in the interface list, and the following dialog box will pop up. Then select or remove the check box in front of "Capture packets in promiscuous mode". Then click the ok button.

Again, set the capture filter. There will be a "Capture Filter" item in the main setting dialog box that pops up by clicking the setting button and in the dialog box that is displayed by double-clicking the interface list. In the text box we can set the capture filter. For example, if we only capture http-related packets, we can set the capture condition to "port 80". Because http uses port 80.

Finally, after all the settings are completed, click the "Start" button in the main settings window to start capturing data. After the data is captured, you can click the "Save" button in the frequently used buttons to save the data.

**Use display filters**



The display filter is applied to the capture file to tell wireshark to display only those packets that meet the filtering conditions. Display filters are more commonly used than capture filters. It can be used to filter unwanted packets, but it does not delete the data. If you want to restore it, just delete the filter.

The filter expression dialog, Wireshark makes it easy to set filter expressions. Click the "Expression" button to open this dialog box.

The dialog is divided into three parts: left, middle, and right. On the left are all the protocol domains that can be used. On the right are the condition values ​​related to the protocol domain. In the middle is the relationship between the protocol domain and the condition value. Filter expressions are useful for beginners. As shown in the figure above, the effect of the expression we created is to display all the packets containing the keyword "bo56.com" in the http protocol package.

Field name description:

This list shows all supported protocols. After clicking the triangle symbol in the front, the filterable fields of this agreement can be listed. When selecting any item in the "Field name" list, you only need to enter the protocol domain you want, and the corresponding protocol domain option will be automatically located.

Relation description:

is present If the selected protocol domain exists, the related data packets are displayed.

contains determines whether a protocol, field, or fragment contains a value

matches Determines whether a protocol or string matches a given Perl expression.

Value (Protocol):

Enter the appropriate values ​​here. If the selected protocol domain and this value satisfy the relationship specified in Relation, the relevant data packet is displayed.

Predefined values:

Some protocol fields contain predefined values, somewhat similar to enumerated types in C. If the protocol domain you choose contains such values, you can choose from this list.

Function description:

The filter language also has the following functions:

upper (string-field)-convert a string to uppercase

lower (string-field)-convert string to lower case

upper ((and lower ((are useful when dealing with case-sensitive string comparisons. For example:

upper (ncp.nds\_stream\_name) contains “BO56.COM”

lower (mount.dump.hostname) == ”BO56.COM”

If you are familiar with this rule, you will find it more efficient to enter expressions manually. At that time, when the expression was manually entered in the flter text box, if there was a problem with the input syntax, the background color of the text box would become red. At this time, you can continue to type or modify. After knowing that the expression in the text box is correct, the background color of the text box will turn green again.

**Conclusion**

Anyways, Wireshark is a very powerful tool for detecting the Internet on your computer if you can master that it will helpful for your computer network process.