Introduction to Wireshark (Analysis)

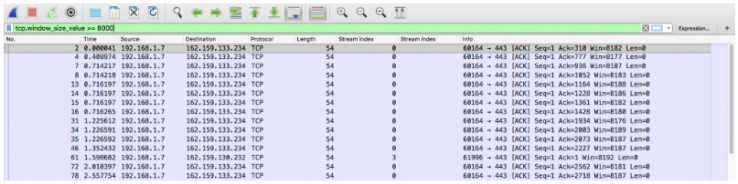
Knowing how to use Wireshark features to enhance network traffic analysis differs from having the skills to analyze network traffic. The latter can only come from real-world experience, practice and research, and a deep understanding of networks, such as the information covered at the start of this course. There is a wealth of network- analysis-related material online, and many more sample PCAP files with which you can practice.

This section will cover the former – how to take advantage of major Wireshark features to improve and assist in in-depth manual analysis. We will explore with applying display filters, using packet list columns, following protocol streams, viewing different traffic statistics.

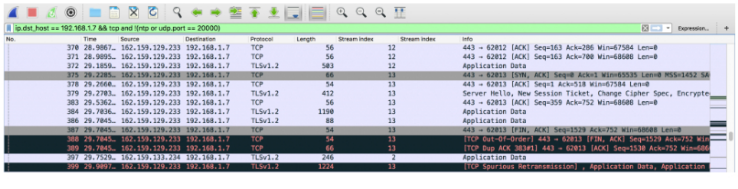
APPLYING DISPLAY FILTERS

As described in the previous sections, the display filter controls which packets are shown in the packet list. This significantly improves the ease of traffic analysis, as important traffic can be separated from the general noise in the network. To filter by the presence of a protocol or header field in a packet, the expression should specify only the protocol or header field, e.g.

- **udp** to display only UDP packets, and
- **http.request** to display only HTTP requests



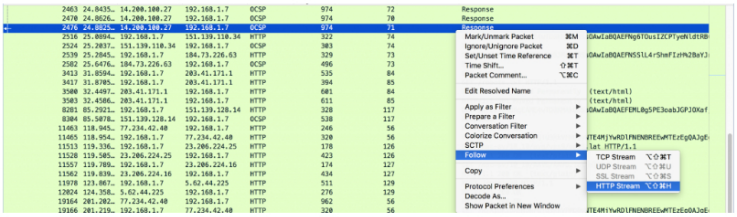
To filter by the values of header fields, you can specify the header field, then a comparison operator, and then the value that it should match. For example, `tcp.port == 80` displays packets that have a source or destination port of 80 (HTTP), and `tcp.window_size_value >= 8000` displays TCP packets with a window size of 8000 bytes or over.



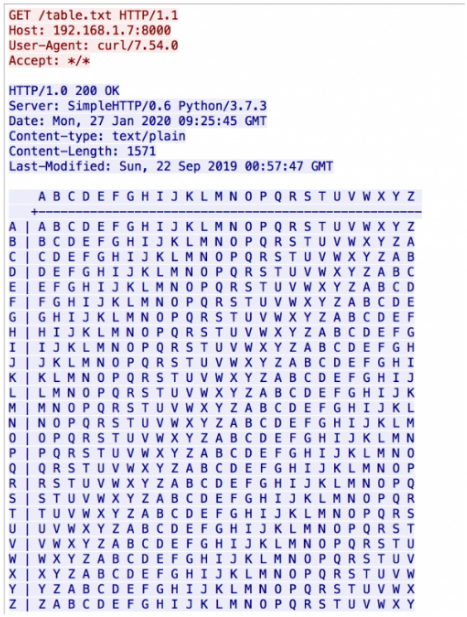
Multiple filter statements can be chained by using logical operators, including `and` (`&&`) and (`or`/`||`). For example, to display TCP packets addressed to 192.168.1.7, you can use `ip.dst_host == 192.168.1.7 && tcp`, and to display either NTP traffic or UDP traffic from/to port 20000, you can use `ntp or udp.port == 20000`. The **not** (!) operator excludes specific packets from being displayed, such as `not ftp` to exclude FTP packets from being displayed in the packet list. Finally, brackets can be used to group statements together.

FOLLOWING STREAMS & CUSTOMISING COLUMNS

Imagine you want to analyze an HTTP communication between a web server and a host. You are interested in having a holistic view of the dozens of HTTP requests and responses. In this case, looking at each individual packet will not be of much profit to us – this is where Wireshark's protocol stream following feature shines.

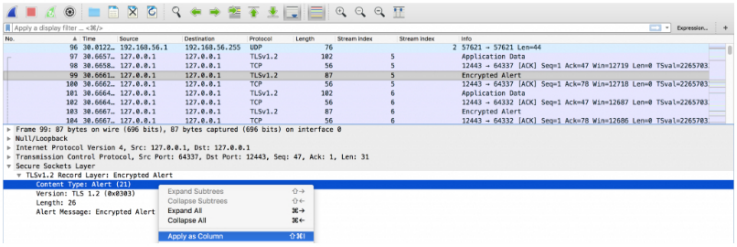


To follow a stream, right-click on a packet within the stream, and click **`Follow > TCP/UDP/SSL/HTTP Stream`**. Inapplicable streams should be greyed out in the pop-up menu. Wireshark should automatically apply a display filter for the stream and open a new window displaying the contents of the packets in the stream.



As you can see from the above image of a simple HTTP file download, HTTP requests should be highlighted in red and HTTP responses in blue. You can easily see the request header and the response header and file content. Compared to looking at the request and response packets individually, looking at the stream saves time and effort as the communication is displayed all in one window.

In addition to following streams, you can view specific packet information in the packet list by customizing the columns in the packet list.



To add a packet header value as a column, right-click the header field and select Apply as Column. In the above image, the SSL content type is applied as a column. The benefit of adding certain header values as a column is the ability for quick visual identification of packets having a specific value that we are interested in.

VIEWING CAPTURE STATISTICS

Wireshark collects different statistics about the traffic in the capture file – such as the percentage proportions of protocols, the number of bytes transmitted to different hosts, the IP addresses of all the hosts that has appeared in the capture. They are helpful in certain scenarios, such as finding potential exfiltration vectors and identifying the exfiltrating host based on network usage.

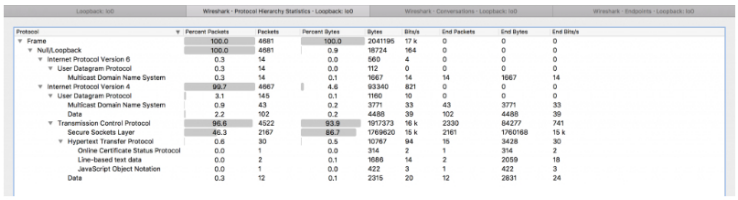
This section will discuss three of the statistics windows:

- Protocol Hierarchy
- Conversations,
- and Endpoints

PROTOCOL HIERARCHY

The Protocol Hierarchy window displays the percentages of the number of packets or bytes in a protocol conversation against the entire traffic. The protocols are organized in layers from Layer 2 to Layer 7. The below image shows:

- 99.7% of the traffic captured were IPv4 packets
- 96.6% being TCP, and
- 46.3% being SSL
- It also shows that there were 2167 SSL packets, totaling around 177 KB.



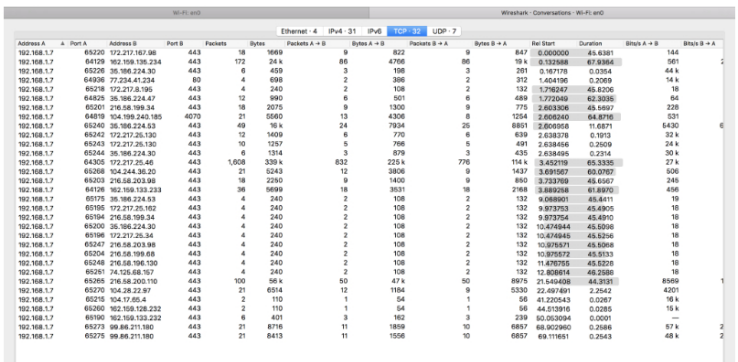
To check a specific protocol traffic in the packet list, right-click on a protocol and select **`Apply as Filter > Selected/Not Selected`** to have Wireshark automatically create and apply a display filter showing/excluding the specified protocol traffic.

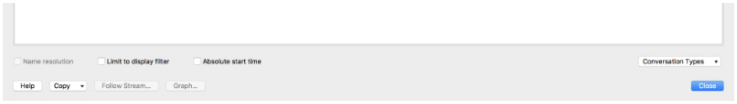
The Protocol Hierarchy window can also identify data exfiltration through unusual or unused protocols. If you notice a very small portion of FTP traffic in a large network that doesn't use FTP, it might be worth it to check out the FTP traffic and make sure the traffic is legitimate. Quite a few of network analysis challenges in CTFs dealing with data exfiltration can be solved by identifying unusual protocols from the Protocol Hierarchy window.

CONVERSATIONS

The Conversations window also provides a wealth of information on the traffic, including which hosts communicated which hosts, on which ports, and with a total of how many bytes and packets in the conversation. This window is great for identifying the different MAC or IP addresses that a host has communicated with, and the volume of traffic between them.

Similarly to the Protocol Hierarchy window, the Conversations window can be very helpful in investigating data exfiltration attempts, as it can identify the attacker by IP address. If a host has been transmitting many packets and bytes to an unidentified IP address without receiving many packets in return, an exfiltration could have been occurring.



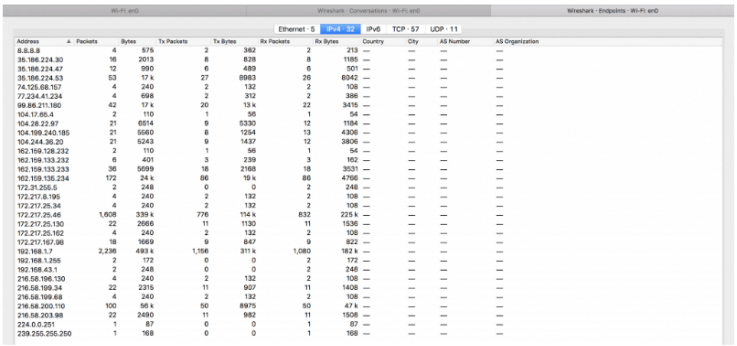


In the above image, as shown by the first line, host 192.168.56.17 has been connecting to 172.217.167.98's HTTPS server (this is actually Google) with port 65220 and has sent and received 9 packets. Similar to the Protocol Hierarchy window, you can right-click on a line, select **`Apply as Filter > Selected/Not Selected`** and choose the direction of traffic to apply a filter for the line.

ENDPOINTS

Lastly, the Endpoints window shows all of the different hosts that appear in the capture and the amount of packets/bytes they sent and received. This window is useful in sorting hosts by their network activity, by either transmission or receiving volume, or by both.

If a host has been receiving much more traffic than they have been transmitting, the host is probably downloading a large file. On the other hand, if the host has been transmitting more than they have been receiving, the host is probably uploading files or backing up to remote storage, such as the cloud.



In the above diagram, host 192.168.1.7 has sent 1156 packets, totaling 311 KB, and has received 1080 packets, totaling 182 KB. You can switch to other protocols, such as TCP to have traffic sorted by ports and Ethernet to have traffic sorted by MAC addresses. Same with all other statistics windows, you can right-click on a host to apply it as a filter.

Quizzes

Lab) Network Traffic Analysis

Previous Topic
Back to Lesson
Next Topic