Wireshark Cheat Sheet

A comprehensive cheat sheet for using Wireshark, covering essential features, filters, and techniques for network analysis and security monitoring.

Wireshark Basics

Getting Started

Purpose: Wireshark is a network protocol analyzer that captures and analyzes network traffic in real-time.

Download: Get it from www.wireshark.org.

Interface: Familiarize yourself with the main window, including the capture filter bar, packet list pane, packet details pane, and packet bytes pane.

Capture Interface: Select the correct network interface from the capture options to start capturing traffic.

Capture Filters: Use capture filters to limit the traffic captured to only what you need (e.g., `tcp port 80`).

Stop Capture: Use the stop button (red square) to halt the packet capture process.

Save Capture: Save captured packets in a `.pcap` or `.pcapng` file for later analysis.

Common Interface Elements

Packet List Pane:	Displays a summary of each captured packet.
Packet Details Pane:	Shows detailed information about the selected packet's protocol layers and fields.
Packet Bytes Pane:	Displays the raw data of the packet in hexadecimal and ASCII format.
Filter Toolbar:	Allows you to apply display filters to focus on specific traffic.
Statistics Menu:	Provides various statistical summaries of the captured traffic.
Go Menu:	Allows navigation of captured packets.

Display Filters

Basic Filters

- `ip.addr == 192.168.1.1` - Filter by IP address.
- `tcp.port == 80` - Filter by TCP port.
- `http` - Show only HTTP traffic.
- `dns` - Show only DNS traffic.
- `icmp` - Show only ICMP traffic.
- `arp` - Show only ARP traffic.

Advanced Filters

- `ip.src == 192.168.1.1 and ip.dst == 10.0.0.1` - Filter by source and destination IP addresses.
- `tcp.flags.syn == 1 and tcp.flags.ack == 0` - Filter for TCP SYN packets (used for connection initiation).
- `http.request.method == "GET"` - Filter HTTP GET requests.
- `http.response.code == 404` - Filter HTTP 404 errors.
- `tcp.stream eq 5` - Follow TCP stream number 5.
- `frame.len > 1000` - Packets larger than 1000 bytes.

Filter Operators

`==`	Equal to
`!=`	Not equal to
`>`	Greater than
`<`	Less than
`>=`	Greater than or equal to
`<=`	Less than or equal to

Capture Filters (BPF)

Basic Syntax

Capture filters use Berkeley Packet Filter (BPF) syntax and are applied *before* traffic is captured.
They can significantly reduce the amount of data to be analyzed.

- `host 192.168.1.1` - Capture traffic to or from the host 192.168.1.1.
- `net 192.168.1.0/24` - Capture traffic within the 192.168.1.0/24 network.
- `port 80` - Capture traffic on port 80.
- `tcp` - Capture only TCP traffic.
- `udp` - Capture only UDP traffic.

Combining Filters

`and`, `&&`	Combine filters, both conditions must be true.
`or`, `||`	Combine filters, either condition can be true.
`not`, `!`	Negate a filter.
`host 192.168.1.1 and port 80`	Capture traffic to/from 192.168.1.1 on port 80.
`net 10.0.0.0/24 or port 53`	Capture traffic on the 10.0.0.0/24 network or port 53.
`not arp`	Capture everything except ARP traffic.

Advanced Features

Following Streams

Follow TCP Stream: Right-click on a TCP packet and select "Follow" -> "TCP Stream" to see the entire conversation.

Follow UDP Stream: Similar to TCP, but for UDP packets.

This is useful for reassembling data transmitted over a connection, such as HTTP requests and responses.

Analyzing Statistics

Statistics Menu: Use the Statistics menu to generate reports on captured traffic.

Conversations: Analyze traffic between different endpoints.

Endpoints: Show a list of all endpoints in the capture.

Protocol Hierarchy: See the distribution of traffic by protocol.

IO Graphs: Visualize traffic patterns over time.

Security Analysis

Detecting Anomalies: Look for unusual traffic patterns, large packet sizes, or connections to unknown hosts.

Identifying Malware: Examine traffic for known malware signatures or communication patterns.

Analyzing Encrypted Traffic: While you can't see the content, you can analyze the metadata (IP addresses, ports, TLS versions) of encrypted traffic.

VLAN Tagging: Use `vlan.id == <VLAN ID>` to filter specific VLANs.