

**Wireshark filters**

Wireshark’s most powerful feature is it vast array of filters. There over 242000 fields in 3000 protocols that let you drill down to the exact traffic you want to see. These filters and its powerful filter engine helps remove the noise from a packet trace and only see the packets of interest.

Display filters allow us to compare fields within a protocol against a specific value, compare fields against fields and check the existence os specific fields or protocols.

Bellow you can find a small list of the most common protocols and fields when filtering traffic with Wireshark.

**Ethernet**

* eth.addr — address
* eth.dst — destination
* eth.ig — IG bit
* eth.len — length
* eth.lg — LG bit
* eth.src — source
* eth.trailer — trailer
* eth.type — type

**ARP — Address resolution Protocol**

* arp.dst.hw\_mac -Target MAC address
* arp.dst.proto\_ipv4 — Target IP address
* arp.hw.size — Hardware size
* arp.hw.type — Hardware type
* arp.opcode — Opcode
* arp.proto.size — Protocol size
* arp.proto.type — Protocol type
* arp.src.hw\_mac — Sender MAC address
* arp.src.proto\_ipv4 — Sender IP address

**IPv4 — Internet Protocol version 4**

* ip.addr — Source or Destination Address
* ip.checksum — Header checksum
* ip.checksum\_bad — Bad
* ip.checksum\_good — Good
* ip.dsfield — Diffrentiated Services Field
* ip.dsfield.ce — ECN-CE
* ip.dsfield.dscp — Diferrentiated Services Codepoint
* ip.dsfield.ect — ECT Capable Transport
* ip.dst — Destination
* ip.dst\_host — Destination host
* ip.flags — Flags
* ip.flags.df — Don’t fragment
* ip.flags.mf — More fragments
* ip.flags.rb — Reserved bits
* ip.frag\_offset — Fragment offset
* ip.fragment — IPv4 Fragment
* ip.fragment.error -Defragmentation error
* ip.fragment.multipletails — Multiple tail fragment found
* ip.fragment.overlap — Fragment overlap
* ip.fragment.overlap.conflict — Confliting data in fragment overlap
* ip.fragment.toolongfragment — Fragment too long
* ip.fragments — IPv4 fragments
* ip.hdr\_len — Header length
* ip.host — Source or Destination Host
* ip.id — Identification
* ip.len — Total length
* ip.proto — Protocol
* ip.reassembled\_in — Reassembled IPv4 in frame
* ip.src — Source
* ip.src\_host — Source host
* ip.tos — Type of Service
* ip.tos.cost — Cost
* ip.tos.delay — Delay
* ip.tos.precedence — Precedence
* ip.tos.reliability — Reliability
* ip.tos.throughput — Throughtput
* ip.ttl — Time to live
* ip.version — Version

**IPv6 — Internet Protocol version 6**

* ipv6.addr — Source or Destination Address
* ipv6.class — Traffic class
* ipv6.dst — Destination
* ipv6.dst\_host — Destination host
* ipv6.dst\_opt — Destination Option
* ipv6.flow — Flow label
* ipv6.fragment IPv6 Fragment
* ipv6.fragment.error — Defragmentation Error
* ipv6.fragment.id — Identification
* ipv6.fragment.more — More Fragment
* ipv6.fragment.multipletails — Multiple tail fragment found
* ipv6.fragment.offset — Offset
* ipv6.fragment.overlap — Fragment overlap
* ipv6.fragment.overlap.conflict -Confliting data in fragment overlap
* ipv6.fragment.toolongfragment — Fragment too long
* ipv6.fragments — IPv6 Fragments
* ipv6.hlim — Hop limit
* ipv6.hop\_opt — Hop-by-hop option
* ipv6.host — Source or destination host
* ipv6.mipv6\_home\_address — Home address
* ipv6.mipv6\_length — option length
* ipv6.mipv6\_type — Option type
* ipv6.nxt — Next header
* ipv6.opt.pad1 — Pad1
* ipv6.opt.padn — PadN
* ipv6.plen — Payload lenght
* ipv6.reassembled\_in — Reassembled in Frame
* ipv6.routing\_hdr — Routing header type
* ipv6.routing\_hdr.addr — Address
* ipv6.routing\_hdr.left — Segments left
* ipv6.routing\_hdr.type — Type
* ipv6.src — Source
* ipv6.src\_host — Source host
* Ipv6.version — Version

**TCP — Transfer Control Protocol**

* tcp.ack — Anknowlegment number
* tcp.checksum — Checksum
* tcp.checksum\_bad — Bad checksum
* tcp.checksum\_good — Good checksum
* tcp.continuation\_to — This is a contiuation to the PDU in frame
* tcp.dstport — Destination Port
* tcp.flags — Flags
* tcp.flags.ack — Acknolegment
* tcp.flags.cwr — Congestion Window reduced
* tcp.flags.ecn — ECN-Echo
* tcp.flags.fin — Fin
* tcp.flags.push — Push
* tcp.flags.reset — Reset
* tcp.flags.syn — Syn
* tcp.flags.urg — Urgent
* tcp.hdr\_len — Header length
* tcp.len — TCP Sgment Len
* tcp.nxtseq — Next sequent number
* tcp.options — TCP options
* tcp.options.cc — TCP CC Options
* tcp.options.ccecho — TCP CC Echo Option
* tcp.options.ccnew — TCP CC New option
* tcp.options.echo — TCP echo option
* tcp.options.echo\_reply — TCP Echo Reply option
* tcp.options.md5 — TCP MD5 option
* tcp.options.mss — TCP MSS option
* tcp.options.mss\_val — MSS value
* tcp.options.qs — TCP QS option
* tcp.options.sack — TCP Sack option
* tcp.options.sack\_le — TCP Sack Left Edge
* tcp.options.sack\_perm — TCP Sack Permitted option
* tcp.options.sack\_re — TCP Sack Right Edge
* tcp.options.time\_stamp — TCP Timestamp value
* tcp.options.wscale — TCP Window Scale option
* tcp.options.wscale\_val — TCP Window Scale Optin Value
* tcp.pdu.last\_frame — Last frame of the PDU
* tcp.pdu.size — PDU size
* tcp.pdu.time — Time until the last segment of this PDU
* tcp.port — Source or Destination Port
* tcp.reassembled\_in — Reassembled PDU in frame
* tcp.segment — TCP segment
* tcp.segment.error — Reassembling error
* tcp.segment.multipletails — Multiple tail segment found
* tcp.segment.overlap — Segment ovelap
* tcp.segment.overlap.conflict — Conflicting data in segment overlap
* tcp.segment.toolongfragment — Segment too long
* tcp.segments -Reassembled TCP segments
* tcp.seq — Sequence number
* tcp.srcport — Source port
* tcp.time\_delta — Time sence previous frame in the TCP stream
* tcp.time\_relative — Time since first frame in the TCP stream
* tcp.urgent\_pointer — Urgent pointer
* tcp.window\_size — Window size value

**UDP — User Datagram Protocol**

* udp.checksum — Checksum
* udp.checksum\_bad — Bad checksum
* udp.checksum\_good — Good checksum
* udp.dstport — Destination Port
* udp.length — Length
* udp.port — Source or Destination Port
* udp.srcport — Source Port

**ICMP — Internet Control Message Protocol**

* icmp.checksum — Checksum
* icmp.checksum\_bad — Bad checksum
* icmp.code — Code
* icmp.ident — Identifier
* icmp.mtu — MTU
* icmp.redir\_gw — Gateway address
* icmp.seq — sequence Number
* icmp.type — Type

**ICMPv6 — Internet Control Message Protocol version 6**

* icmpv6.all\_comp — Component
* icmpv6.checksum — Checksum
* icmpv6.checksum\_bad — Bad Checksum
* icmpv6.code — Code
* icmpv6.comp — Component
* icmpv6.haad.ha\_addrs — Home Agent Address
* icmpv6.identifier — Identifier
* icmpv6.option — ICMPv6 Option
* icmpv6.option.cga — CGA
* icmpv6.option.length — Length
* icmpv6.option.name\_type.fqdn — FQDN
* icmpv6.option.name\_x501 — DER Encoer X.501 name
* icmpv6.option.rsa.key\_hash — Key hash
* icmpv6.option.type — Type
* icmpv6.ra.cur\_hop\_limit — Cur hop limit
* icmpv6.ra.reachable\_time — Reachable time
* icmpv6.ra.retrans\_timer — Retrans timer
* icmpv6.ra.router\_lifetime — Router lifetime
* icmpv6.recursive\_dns\_serv — Recursive DNS Server
* Icmpv6.type- Type

**HTTP — Hypertext Transfer Protocol**

* http.accept — Accept
* http.accept\_encoding — Accept encoding
* http.accept\_language — Accept language
* http.authbasic — Credentials
* http.authorization — Authorization
* http.cache\_control — Cache control
* http.connection — Connection
* http.content\_encoding — Content encoding
* http.content\_length — Content length
* http.content\_type — Content type
* http.cookie — Cookei
* http.date — Date
* http.host — Host
* http.last\_modified — Last modified
* http.location — Location
* http.notification — Notification
* http.proxy\_authenticate- Proxy authenticate
* http.proxy\_authorization — Proxy authorization
* http.proxy\_connect\_host — Proxu connect hostname
* http.proxy\_connect\_port — Proxy connect port
* http.referer — Referer
* http.request — Request
* http.request.method — Request method
* http.request.uri — Request URI
* http.request.version — Request version
* http.response — Response
* http.response.code — Status code
* http.server — Server
* http.set\_cookie — Set Cookie
* http.transfer\_encoding — transfer encoding
* http.user\_agent — user agent
* http.www\_authenticate — WWW-Authenticate
* http.x\_forwarded\_for — X-Forwarded For

**Operators**

Protocols and fields can be checked for existence in the filter box. Fields can also be compared against values. The comparison operators can be expressed either through English-like abbreviations or through C-like symbols:

* eq or ==
* ne or !=
* gt or >
* lt or <
* ge or >=
* le or <=

**Logic**

Tests can be combined using logical expressions that are expressible in C-like syntax or with English-like abbreviations:

* and or && Logical AND
* or or || Logical OR
* xor or ^^ Logical XOR
* not or ! Logical NOT
* [n] […] Substring operator