Wireshark Einführung

Stand: Sommersemester 2020

Inhalt

Netzanalyse-Software Wireshark

Funktionalität

Wireshark ist ein weitverbreitetes Programm zur Analyse von aufgezeichnetem Datenverkehr (Traces) in Rechnernetzen. Die Open Source Software ermöglicht einen detaillierten Einblick in viele Protokolle und eignet sich auch zur Fehlersuche. Wireshark bietet unter anderem die folgenden Funktionen:

- Dekodierung und Anzeige von vielen verschiedenen Protokollen
- Online-Aufzeichnung von Datenverkehr und Offline-Analyse von aufgezeichneten Traces
- Protokoll-spezifische Diagnose-Werkzeuge und Statistiken

Das Programm ist für Windows, Linux, MacOS und viele weitere Betriebssysteme frei verfügbar.

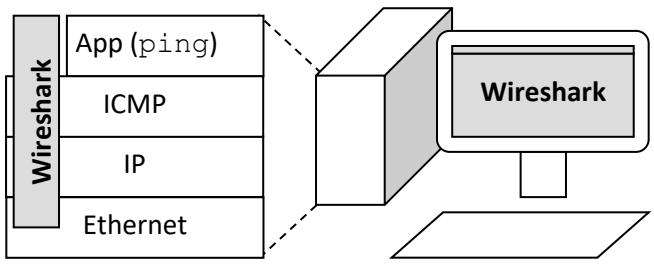
Auf der Homepage von Wireshark wird die Verwendung detailliert erläutert und dokumentiert:

URI	Inhalt
https://www.wireshark.org	Homepage des Open Source Projekts
https://www.wireshark.org/docs/	Dokumentation (z.B. Video Tutorials)
https://www.wireshark.org/docs/wsug_html_chunked/	User Guide

Aufzeichnen

Die Software Wireshark kann Datenverkehr auf Netzadaptern eines Rechners aufzeichnen. Dazu wird die Kommunikation über eine Packet Capture (PCAP) Programmierschnittstelle mitgeschnitten, verarbeitet und ggf. auch gespeichert. Bei einem Rechner mit Ethernet Netzadapter werden dann alle vom Rechner versendeten bzw. empfangenen Ethernet Rahmen aufgezeichnet. Wireshark kann dabei parallel zu Anwendungen verwendet werden, deren Kommunikation analysiert werden soll.

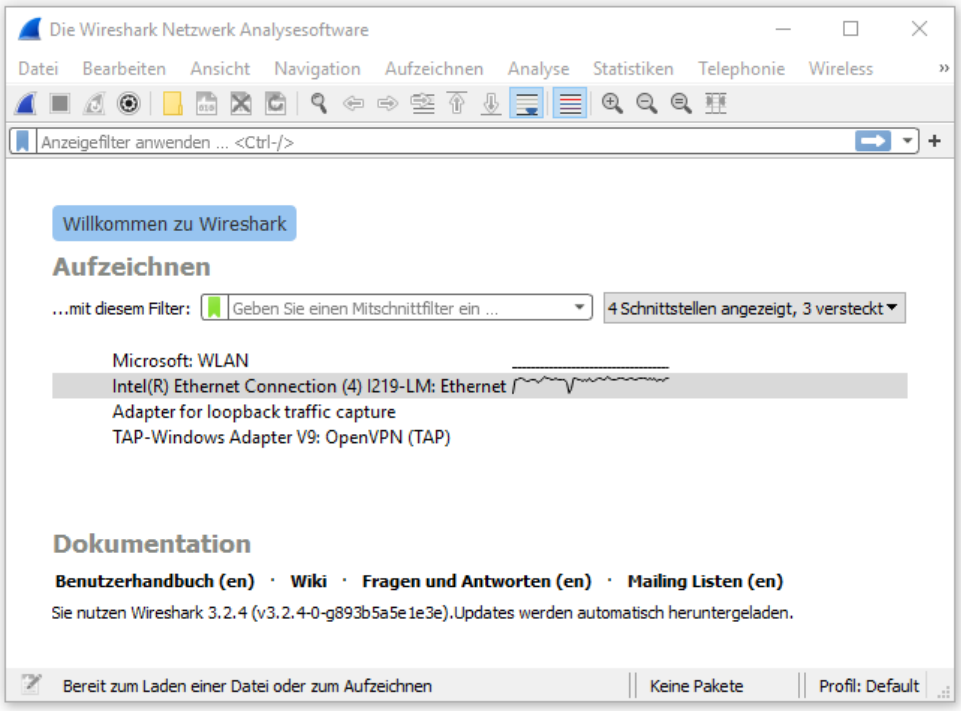
Beispielsweise können mit Wireshark die ICMP Nachrichten aufgezeichnet werden, die von der Anwendung `ping` gesendet und empfangen werden:



Protokollstapel am Beispiel von `ping`

Benutzeroberfläche

Der Startbildschirm der Wireshark Anwendung unter Windows ist in der folgenden Abbildung illustriert:

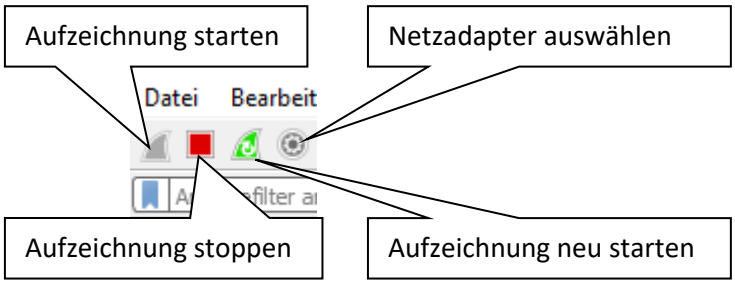


Es ist beim Start der Anwendung möglich, unter den verschiedenen Netzadaptern des Rechners denjenigen auszuwählen, auf der die gesendeten und empfangenen Rahmen (Frames) aufgezeichnet werden sollen. Wireshark zeichnet dann alle Rahmen auf, die auf diesem Netzadapter gesendet bzw. empfangen wurden, d.h. den kompletten Datenverkehr des Rechners über diesen Netzadapter. Für die Aufzeichnung gelten jedoch einige Einschränkungen:

- Unter Windows ist es mit Wireshark ggf. nicht möglich, lokale Kommunikation innerhalb des Rechners aufzuzeichnen (Loopback Schnittstelle).
- Die Aufzeichnung umfasst Rahmen in der Sicherungsschicht; eine Aufzeichnung analoger Signale in der Bitübertragungsschicht ist nicht möglich.
- Bei Ethernet wird die Frame Check Sequence (FCS) ggf. nicht aufgezeichnet; die Prüfsumme wird in der Regel bereits in der Netzwerkkarte überprüft und ist dann für Wireshark nicht zugänglich.

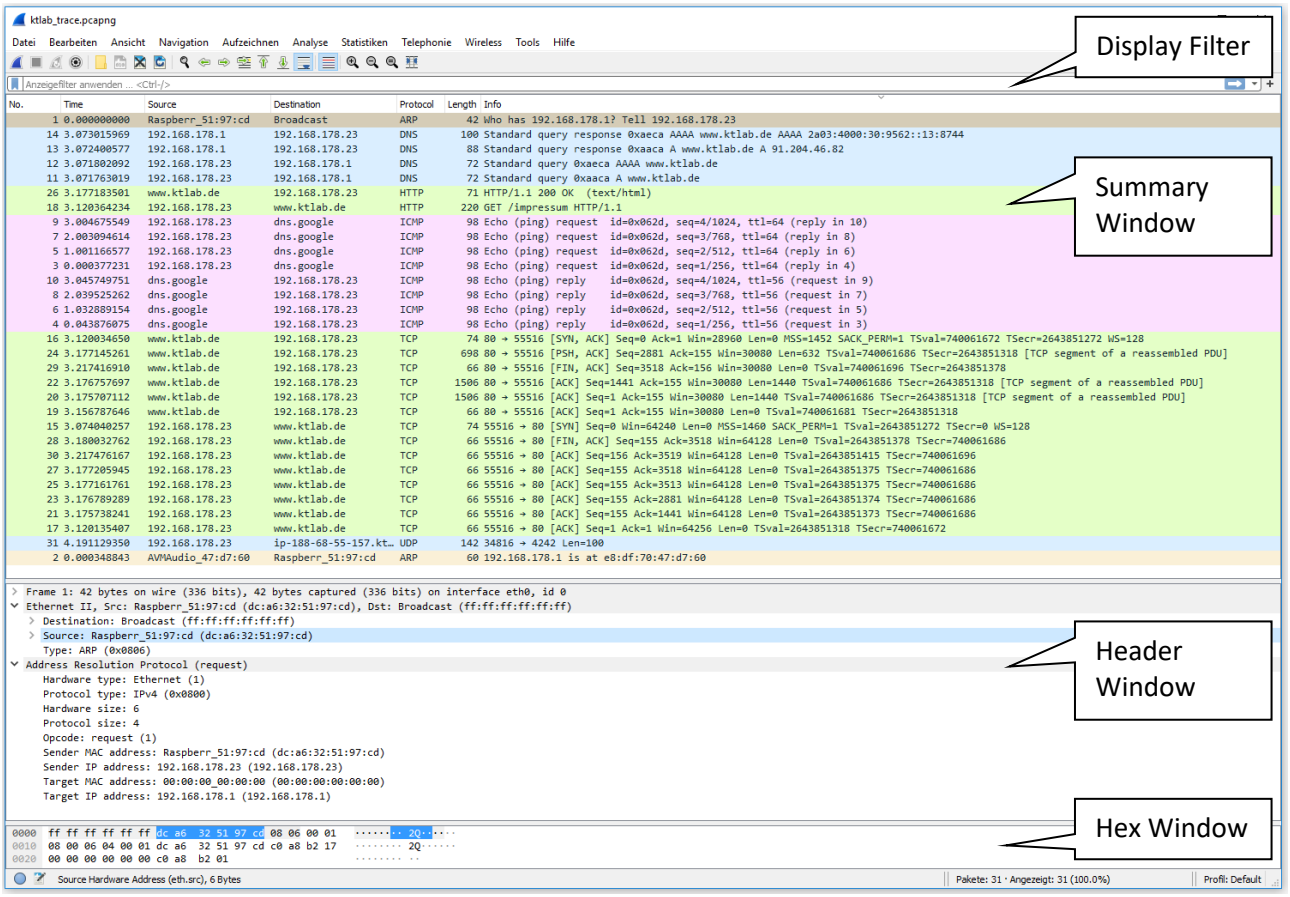
Bei der Verwendung von Wireshark ist es wichtig, dass der richtige Adapter ausgewählt wird, d.h. zum Beispiel der Netzadapter für die Ethernet-Netzwerkkarte bei Anbindung an ein Ethernet LAN. Eine Auswahl anderer vorhandener Netzadapter liefert ggf. nicht das erwartete Ergebnis.

Es ist möglich, den Netzadapter nach Programmstart zu ändern, wenn keine Aufzeichnung läuft. Die Aufzeichnung kann u.a. durch die Icons unterhalb der Menüleiste gestartet bzw. gestoppt werden.



Während einer laufenden Aufzeichnung wird der komplette Datenverkehr des Rechners mitgeschnitten und gespeichert. Bei einer lang laufenden Aufzeichnung kann die Datenmenge erheblich ansteigen und die Liste der aufgezeichneten Rahmen wird unübersichtlich. Es kann daher sinnvoll sein, die Aufzeichnung nach Abschluss eines Experiments zu stoppen und anschließend für die nächste Messung neu zu starten.

Aufgezeichneter Datenverkehr wird in Wireshark in verschiedenen Darstellungen angezeigt.



Standardmäßig sind in Wireshark drei verschiedene Ansichten verfügbar:

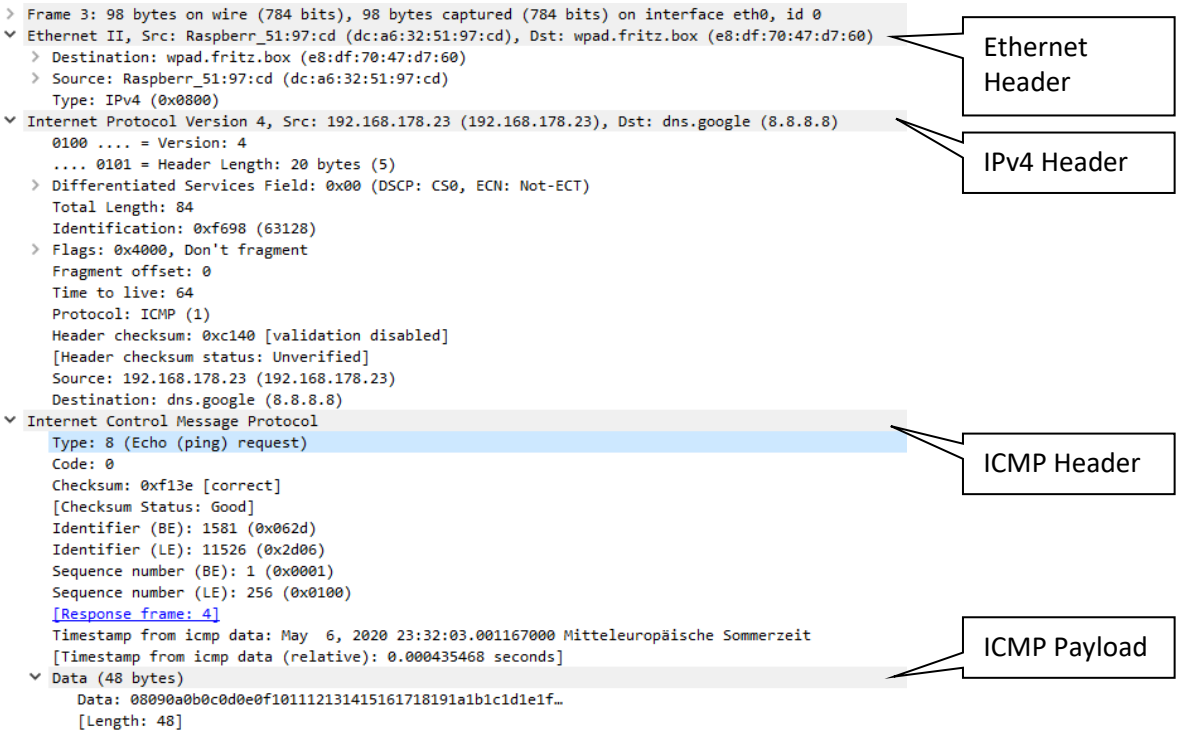
- Paketliste (**Summary Window**): Das obere Fenster stellt als Gesamtansicht zeilenweise alle aufgezeichneten Rahmen dar, inklusive der Nummer und dem Aufzeichnungszeitpunkt. Standardmäßig erfolgt die Auflistung in chronologischer Reihenfolge und es werden weitere Datenfelder angezeigt, wie beispielsweise die Quell-IP-Adresse und die Ziel-IP-Adresse, die Anzahl aufgezeichneter Bytes sowie erkannte Protokolle. Anpassungen dieser Darstellung sind möglich.
- Paketdetails (**Header Window**): Das mittlere Fenster zeigt eine Aufschlüsselung alle Datenfelder in den aufgezeichneten Rahmen bzw. Paketen, sofern die entsprechenden Protokolle erkannt wurden. Je nach Protokoll sind ggf. auch weitere Zusatzinformationen verfügbar.
- Paketbytes (**Hex Window**): Im unteren Fenster kann der Rahmen-Inhalt in Hexadezimaldarstellung analysiert werden. Dabei werden ggf. ausgewählte Datenfelder hervorgehoben. Diese Information wird in der Regel nur für eine detaillierte Analyse einzelner Datenfelder benötigt.

Im dargestellten Screenshot wird eine abgespeicherte Verkehrsaufzeichnung analysiert. Für einen ARP Rahmen werden im Beispiel die Datenfelder in Ethernet und ARP dargestellt.

Im Header Window werden, sofern möglich, die im Rahmen enthaltenen Protokolle dekodiert. Dabei wird die niedrigste Protokollschicht (Schicht 1 und 2) oben im Fenster dargestellt. Bei Aufzeichnung von Ethernet-Datenverkehr ist das oberste dekodierbare Protokoll „Ethernet II“. Es folgen ggf. weitere im Rahmen enthaltene Protokolle höherer Schichten, wie z.B. ARP.

Im Header Window von Wireshark wird der erkannte Protokollstapel also von **oben nach unten beginnend mit der niedrigsten Schicht visualisiert**.

Diese Darstellung wird auch in einem weiteren Beispiel verdeutlicht, in dem eine über Ethernet und IPv4 übertragene ICMP Nachricht analysiert wird:



Wenn weitere Details zu bestimmten Protokollen verfügbar sind, wird dies im Header Window durch das **Symbol >** angezeigt. Durch Mausklick kann die entsprechende Ansicht dann erweitert werden. Im Beispiel ist z.B. die Information zur Aufzeichnung von „Frame 3“ in der ersten Zeile ausgeblendet.

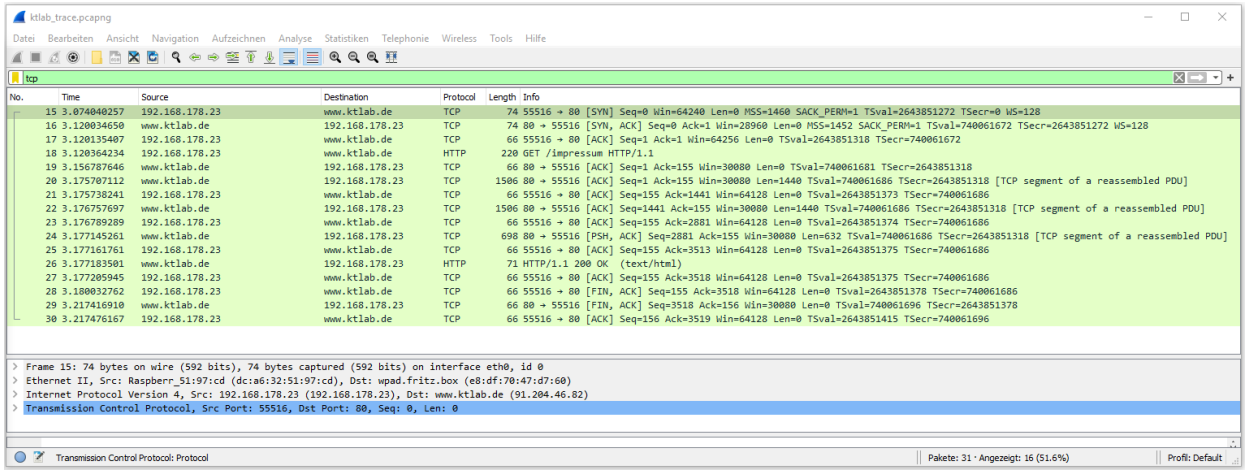
In **eckigen Klammern** werden ggf. von Wireshark ermittelte Zusatzinformationen ausgegeben, die nicht aufgezeichnet wurden, sondern durch nachträgliche Analyse des Datenverkehrs ermittelt wurden.

Die Ethernet Frame Check Sequence (FCS) am Rahmenende wird durch Wireshark nicht aufgezeichnet und wird daher auch nicht dargestellt. Angaben in Wireshark über die Menge aufgezeichneter Bytes von Ethernet Rahmen **beinhalten daher nicht die 4 byte lange Ethernet FCS**.

Wireshark Display Filter

Die Anwendung Wireshark bietet eine Vielzahl an Möglichkeiten, mit Filtern Teilmengen aus dem aufgezeichneten Datenverkehr zu selektieren. Das wichtigste Hilfsmittel hierfür sind **Display Filter**. Wenn ein Display Filter verwendet wird, stellt Wireshark aus allen insgesamt aufgezeichneten Rahmen nur diejenigen dar, die auf einen durch den Benutzer vorgegebenen Filterausdruck passen.

Display Filter können in der Bedienoberfläche unterhalb der Menüleiste eingegeben werden, sowohl während einer laufenden Aufzeichnung als auch nach dem Abstoppen. Gültige Filterausdrücke werden mit einem grünen Hintergrund visualisiert. Bei Syntax-Fehlern hat das Eingabefeld dagegen einen roten Hintergrund. Dies wird nachfolgend für einen Filter veranschaulicht, der nur TCP Segmente anzeigt:



Wireshark-Filter werden erst angewendet, wenn dies vom Benutzer explizit gefordert wird, z.B. durch Drücken der **Eingabe-Taste (Return-Taste**). Dies gilt auch nach Änderungen von Filter-Ausdrücken.

In der Praxis ist zur Vermeidung von Verwirrung wichtig, dass der unterhalb der Menüzeile angezeigte Display-Filter auch tatsächlich angewendet wird. Es bietet sich daher an, nach Änderungen von Filter-Ausdrücken immer die Eingabe-Taste (Return) zu betätigen.

Während einer Analyse von aufgezeichnetem Datenverkehr ist es ggf. sinnvoll, Display Filter einzusetzen, um die relevanten Teile einer Aufzeichnung zu selektieren. Weitere Informationen zu Display Filtern können unter anderem auf den folgenden Web-Seiten gefunden werden:

URI
https://www.wireshark.org/docs/wsug_html_chunked/ChWorkBuildDisplayFilterSection.html
https://www.wireshark.org/docs/man-pages/wireshark-filter.html
https://wiki.wireshark.org/DisplayFilters
https://www.wireshark.org/docs/dfref/

Wireshark verwendet eine eigene Sprache zur Erstellung von Filterausdrücken. In Display Filtern kann man auf alle Datenfelder in den von Wireshark unterstützen Protokollen zugreifen. Neben einfachen Wertevergleichen sind auch sehr komplexe Ausdrücke möglich.

Nachfolgend sind einige nützliche **Datenfelder** in verschiedenen Protokollen aufgelistet. Eine vollständige Liste aller unterstützen Protokolle und die Definition der jeweiligen Attribute findet sich in der Dokumentation von Wireshark.

Ethernet		**IPv4**		**Protokolle**	
`eth.src`	Quelladresse	`ip.src`	Quelladresse	`arp`	ARP
`eth.dst`	Zieladresse	`ip.dst`	Zieladresse	`icmp`	ICMP
`eth.addr`	Quelle oder Ziel	`ip.addr`	Quelle oder Ziel	`tcp`	TCP
`eth.type`	EtherType	`ip.len`	Paket-Länge	`udp`	UDP

Jedes Datenfeld kann selbst als Filterausdruck verwendet werden. Wenn ein Filterausdruck nur ein Attribut enthält, d.h. keinen Vergleich, dann werden nur Rahmen bzw. Pakete angezeigt, in denen dieses Datenfeld bzw. das entsprechende Protokoll vorhanden ist.

Beispiele:

- `arp` Zeigt nur Rahmen an, die das ARP Protokoll enthalten
- `icmp` Zeigt nur Rahmen an, die eine ICMP Nachrichten enthalten

Display Filter können verschiedene **Vergleichsoperatoren** einsetzen, die sowohl in C-ähnlichem Syntax wie auch als englische Abkürzung eingegeben werden können:

Vergleich		**Operator**
`==`	`eq`	Gleich
`!=`	`ne`	Ungleich
`>`	`gt`	Größer
`<`	`lt`	Kleiner
`>=`	`ge`	Größer-gleich
`<=`	`le`	Kleiner-gleich

Diese Liste ist nicht vollständig, es gibt beispielsweise noch weitere Operatoren für Zeichenketten oder Bitvergleiche. Ausdrücke können auch durch Klammern `()` zusammengefasst werden. In der Praxis sind insbesondere die **Operatoren für Gleichheit** (`==` oder `eq`) bzw. für **Ungleichheit** (`!=` oder `ne`) wichtig.

Beispiele:

- `eth.type == 0x0800` Rahmen, die Ethernet mit EtherType 0x0800 enthalten
- `eth.type eq 0x0800` (äquivalent)
- `ip.dst == 10.0.0.1` Rahmen, die ein IPv4 Paket mit Ziel-Adresse 10.0.0.1 enthalten
- `ip.dst eq 10.0.0.1` (äquivalent)
- `ip.src != 10.0.0.2` Rahmen, die nicht von IPv4 Adresse 10.0.0.2 kommen
- `ip.src ne 10.0.0.2` (äquivalent)

Werte können unter anderem als Dezimalzahl oder als Hexadezimalzahl (z.B. `0x0800`) angegeben werden.

Ethernet MAC Adressen können sowohl mit Bindestrich als auch mit Doppelpunkt angegeben werden; beides hat die gleiche Bedeutung. Hexadezimalziffern können groß oder klein geschrieben werden.

Beispiele:

- `eth.dst == FF-FF-FF-FF-FF-FF` Ethernet Rahmen an die Broadcast Adresse
- `eth.dst == FF:FF:FF:FF:FF:FF` (äquivalent)

IP Adressen können nicht nur als einzelne Adresse spezifiziert werden, sondern es kann auch ein gesamtes Sub-Netz in Classless Inter Domain Routing (CIDR) Notation angegeben werden.

Beispiele:

- `ip.dst == 10.0.0.3` Pakete an die IPv4 Ziel-Adresse 10.0.0.3
- `ip.dst == 10.0.0.0/24` Pakete an Sub-Netz 10.0.0.0 mit Netzmaske 255.255.255.0
- `ip.src eq 10.0.0.0/16` Pakete von Sub-Netz 10.0.0.0 mit Netzmaske 255.255.0.0

Filterausdrücke können durch **logische Ausdrücke** kombiniert werden. Auch für die Verkettung gibt es sowohl eine C-ähnliche Schreibweise wie auch entsprechende englische Begriffe:

Logik		Operator
`&&`	`and`	Und
`||`	`or`	Oder
`!`	`not`	Nicht
`^^`	`xor`	Exklusiv-Oder

Beispiele:

- `icmp && ip.dst == 10.0.0.4` ICMP Nachrichten an die IPv4 Ziel-Adresse 10.0.0.4
- `tcp or udp` UDP oder TCP als Transportprotokoll

Einige in Filter verfügbare Attribute stehen für mehrere Datenfelder. Zum Beispiel vergleicht `eth.addr` mit der Quell-MAC-Adresse oder mit der Ziel-MAC-Adresse. In gleicher Weise vergleicht `ip.addr` die Quell-IP-Adresse oder die Ziel-IP-Adresse in IPv4. Filter auf diese Attribute entsprechen daher einer logischen Oder-Verknüpfung.

Beispiele:

- `eth.addr == 01-80-C2-00-00-00` ist gleichbedeutend mit
  `(eth.src == 01-80-C2-00-00-00 || eth.dst == 01-80-C2-00-00-00)`
- `ip.addr == 10.0.0.5` ist gleichbedeutend mit
  `(ip.src == 10.0.0.5 || ip.dst == 10.0.0.5)`
- `!(ip.addr == 10.0.0.6)` ist gleichbedeutend mit
  `!(ip.src == 10.0.0.6 || ip.dst == 10.0.0.6)`
- `ip.addr != 10.0.0.7` ist gleichbedeutend mit
  `(ip.src != 10.0.0.7 || ip.dst != 10.0.0.7)`

Neben Display Filtern ist es in Wireshark auch möglich, schon während der Aufzeichnung Verkehr mit einem Capture Filter herauszufiltern, womit die für die Aufzeichnung zu speichernde Datenmenge reduziert werden kann. Der Syntax der Filterausdrücke kann sich von Display Filtern unterscheiden. So lange keine langen Aufzeichnungen durchgeführt werden, ist es in der Regel ausreichend, Display Filter zu verwenden.