

WIRESHARK Newsletter Februar 2025

Liebe Kunden und Wireshark Freunde

Dieser Wireshark Newsletter von Leutert NetServices informiert Sie unregelmässig über Neuerungen im Zusammenhang mit dem Open Source Analyzer Wireshark und weiteren Netzwerkanalyse-Produkten.

Schlagzeilen

News:

- Neue Funktionen ab **Wireshark Version 4**
- **Stratoshark**, der neue Bruder von **Wireshark**
- Das erste **PacketFest '25 in Zürich**

Tipps, Tricks & Traces:

- Kostengünstige, mobile **10 Gbit/s Aufzeichnungshardware**
- **Wireshark Profil** für **Encapsulated Frames**
- **Cool Commands**, kurze Video Clips zu den Themen **Wireshark**, **Cisco** usw.

Kurse & Events:

- Neuer **Wireshark** Kurs **The Packet Factor** bei **AnyWeb Training**
- Aktuelle **Kursdaten**



Neue Funktionen ab Wireshark Version 4

Display- zu Capture-Filter Übersetzer

- **Display- und Capture-Filter** haben aus historischen Gründen einen **unterschiedlichen Befehlssatz**

Capture Filter Syntax:

```
ether host 00:11:95:b7:e0:3e
ether src 00:11:95:b7:e0:3e
ether dst 00:11:95:b7:e0:3e
```

```
host 192.168.178.1
src host 192.168.178.1
dst host 192.168.178.1
```

```
net 192.168.178.0 mask 255.255.255.0
net 192.168.178/24
```

```
arp
not ip
tcp
udp port 138
```

Operators: not, and, or

<http://wiki.wireshark.org/CaptureFilters>

Display Filter Syntax:

```
eth.addr == 00:11:95:b7:e0:3e
eth.src == 00:11:95:b7:e0:3e
eth.dst == 01:00:5e:00:00:09
```

```
ip.addr == 192.168.178.1
ip.src == 192.168.178.1
ip.dst == 192.168.178.1
```

```
ip.addr == 192.168.178.0/24
ip.host contains "192.168.178"
```

```
arp
not ip
tcp
udp.port == 138
```

Operators: not, and, or

<http://wiki.wireshark.org/DisplayFilters>

- **Kursteilnehmer** fragen oft, ob es einen **Display- zu Capture-Filter Übersetzer** gibt? **Ja! Ab V4.4**

Neue Funktionen ab Wireshark Version 4

Display- zu Capture-Filter Übersetzer

- Beispiel Display Filter: (ip.addr == 130.177.80.201) && (tcp.dstport == 445)

No.	Time	Source	Destination	Protocol	Length	Info
16	5.294082	130.177.80.201	130.177.152.23	SMB	134	Trans2 Request, QUERY_PATH_INFO, Query File Basic Info, Pa
18	5.300639	130.177.80.201	130.177.152.23	SMB	156	Trans2 Request, QUERY_PATH_INFO, Query File Basic Info, Pa

- Gehe zu → Edit → Copy → Display filter as pcap filter

Anmerkung: Wenn dieses Feld grau hinterlegt ist, gibt es keinen entsprechenden Capture Filter

- Gehe zu → Capture → Options Interface wählen und Filter mit rechtem Mausklick einfügen

Neue Funktionen ab Wireshark Version 4

Automatic Profile Switching

- **Wireshark Profile** sind eine sehr nützliche Funktion, sie ermöglichen die Darstellung zu personalisieren indem die Anzeige mit eigenen **Spalten, Filter-Knöpfen, Farben** usw. an die Bedürfnisse angepasst werden kann.
- Sinnvoll ist z.B. ein eigenes **Profil für jedes der verschiedenen Protokolle**.
- **Profile** können **importiert und exportiert** werden, und lassen so auf einfache Weise auf weitere Computer verteilen (linke Maustaste im Wireshark Feld rechts unten).
- Im **Internet** gibt es zahlreiche **vorgefertigte Profile**, die bei Bedarf runtergeladen werden können.
- Mein Kollege **Walter Hofstetter** hat eine übersichtliche **Liste** dieser Profile erstellt; daraus sind die Eigenschaften der verschiedenen Profile ersichtlich.
- Dies ermöglicht, die Eignung eines Profils zu beurteilen, **ohne dieses installieren zu müssen**. Die Liste ist als PDF [hier](#) abrufbar.

Wireshark bietet eine neue Funktion: Automatic Profile Switching

- Diese ermöglicht das **automatische Aktivieren eines Profils** beim Öffnen eines Trace Files.
- Als **Zusatz zum Profil** wird ein entsprechender **Filter** konfiguriert, damit wird beim Öffnen eines Trace Files das entsprechende Profil aktiviert.

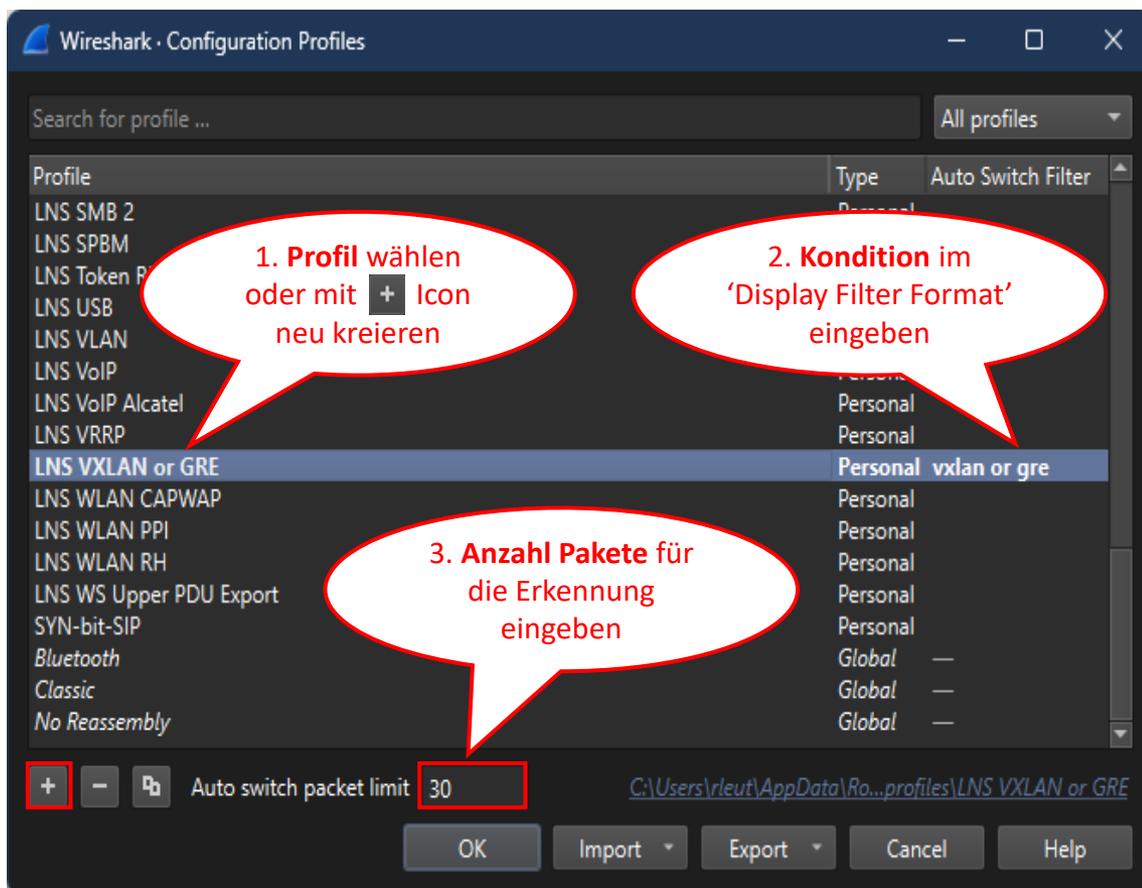
<input type="checkbox"/> Default	LNS MGCP
Bluetooth	LNS MPLS
Classic	LNS Multicast with VLAN
LNS BICC ISUP	LNS No Reassembly
LNS Diameter	LNS OpenFlow1.0
LNS DOCSIS	LNS OSPF
LNS Dual Interfaces	LNS QUIC
• LNS Ethernet	LNS RTP
LNS Ethernet DNS	LNS SCTP
LNS Ethernet ESP	LNS SIP
LNS Ethernet ICMP DNS	LNS Skype
LNS Ethernet Netflow	LNS SMB
LNS Ethernet SNMP	LNS SMB 2
LNS Ethernet TCP	LNS SPBM
LNS Ethernet TCP Diagram	LNS Token Ring
LNS Ethernet UDP	LNS USB
LNS GRE Tunnel	LNS VLAN
LNS H.323	LNS VoIP
LNS H225	LNS VoIP Alcatel
LNS HTTP	LNS VRRP
LNS IPv6	LNS VXLAN or GRE
LNS IS-IS	LNS WLAN CAPWAP
LNS LDAP	LNS WLAN PPI
LNS M3UA	LNS WLAN RH

Meine Wireshark Profile

Neue Funktionen ab Wireshark Version 4

Automatic Profile Switching

- Diese Funktion aktiviert beim Öffnen eines Trace Files **automatisch ein entsprechendes Profil**.
- Zum Einrichten: **Wireshark** rechts unten im Feld **Profile** → **Rechte Maus Taste** → **Manage Profiles**



Die Protokolle **VXLAN** und **GRE-Tunnel** enthalten beide **äußere und innere Adressfelder**. (Wie so ein Profil erstellt wird, ist ab Seite 17 erklärt.)

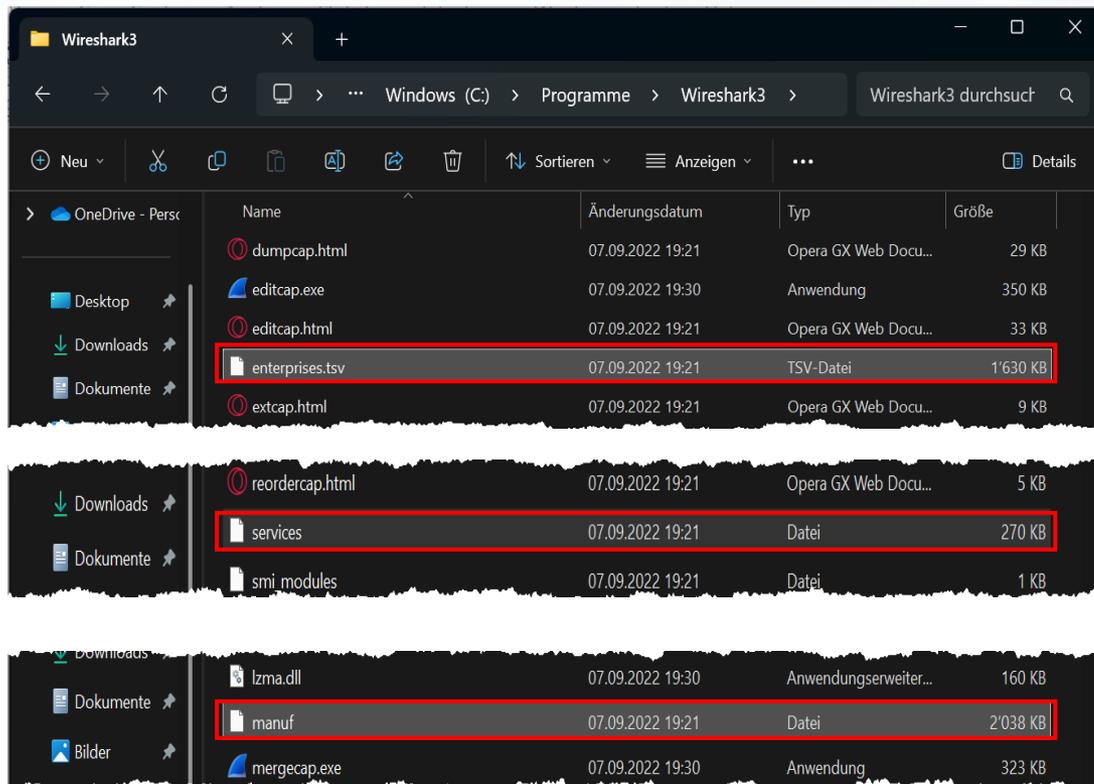
Zum Einrichten lässt sich als **Kondition** ein beliebiger **Display Filter** eingeben. Nach diesem sucht **Wireshark** beim Öffnen eines Trace Files in der eingegebenen Anzahl von Paketen.

Dieses Profil wird sich automatisch aktivieren, wenn in den ersten 30 Paketen das **VXLAN-** oder **GRE-Protokoll** erkannt wird.

Ein Profil kann mit dem  Icon kopiert, oder mit dem  Icon gelöscht werden.

Neue Funktionen ab Wireshark Version 4

Um die **Aufstartzeit** von Wireshark zu verbessern, wurden untenstehende Dateien kompiliert, im **Wireshark-Code integriert** und sind deshalb im Wireshark Programm Ordner **nicht mehr editierbar**.



Bis Wireshark Version 3:

Die Datei **enterprises.tsv** enthielt die „SMI Network Management Private Enterprise Codes“, die von Wireshark bei der Namensauflösung beispielsweise in den Protokollen **Diameter** und **NetFlow** verwendet wurde.

Wireshark verwendete die Datei **services**, um Portnummern in Namen zu übersetzen.
z.B. **Port 443 TCP/UDP → HTTPS**

Die Datei **manuf** wurde von Wireshark zur Übersetzung von MAC-Adressen zu Herstellernamen verwendet.
z.B. **00:01:D7 → F5Networks**

Ab Version 4: Wireshark zeigt die Standard **manuf** Liste neu unter → **Tools** → **MAC Address Blocks**

Unter diesem [Link](#) zeigt **Laura Chappell** wie weiterhin eine eigene **manuf** Datei erstellt werden kann.

Stratoshark

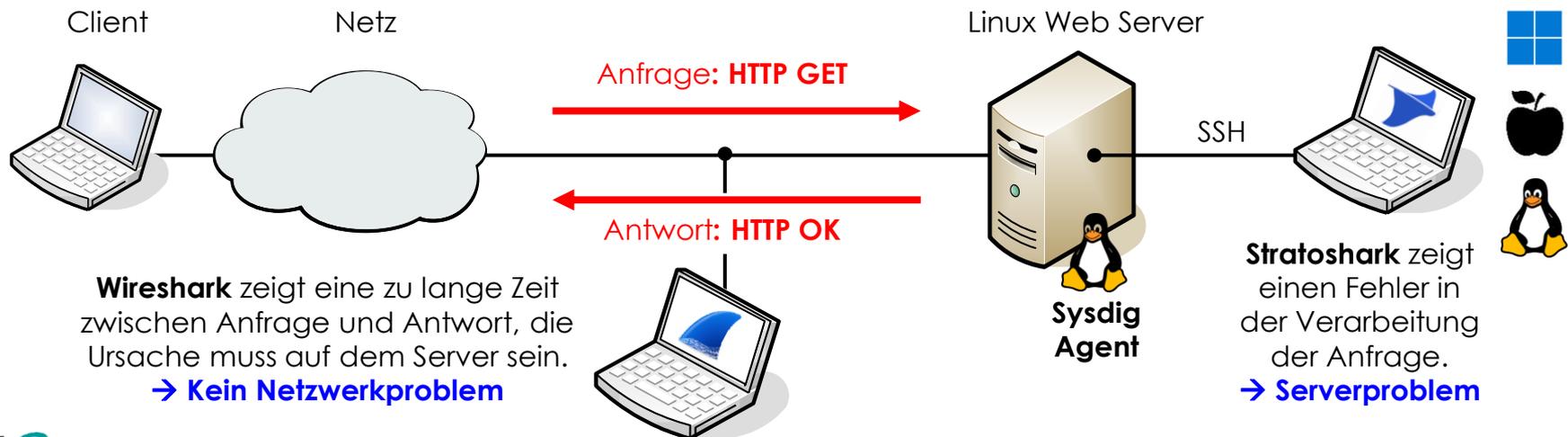
Stratoshark ist ein neues **Open Source Tool**, welches die Aktivitäten, z.B. **innerhalb eines Servers** überwachen und analysieren kann.

Die Software zeichnet **Befehle und Transaktionen** zwischen den **Applikationen** und zum **Betriebssystem** (aktuell nur für LINUX), sogenannte **System Calls**, auf.

Dies hilft Applikationsentwicklern und Systembetreuern, die **System-internen Abläufe** mitzuverfolgen, **Fehler** und **Sicherheitslücken erkennen und beheben zu können**.



Stratoshark ist die ideale Ergänzung zu **Wireshark** für die Eingrenzung von Problemursachen.
Beispiel: Ein Benutzer klagt über lange Antwortzeiten des Web-Servers: **Netzwerk oder Server?**



Stratoshark

Stratoshark wurde entwickelt von der Firma [sysdig](#), Marktführer im Bereich **Monitoring von Cloud and Containers**.

Sysdig ist auch der Sponsor von **Wireshark** und **Gerald Combs**, der Gründer von Wireshark, ist dort seit 2022 der **Director of Open Source**. ([siehe Newsletter 2023](#))



Genial ! Stratoshark verwendet dasselbe GUI wie Wireshark, dadurch werden sich Wireshark Benutzer schnell zurechtfinden. Es sind bereits zahlreiche Video-Clips als Einführung verfügbar; das von **Ross Bagurdes** mag ich besonders <https://youtu.be/Uz97DZmwRSM>

```

evnt.type == "connect"
Time      | Event name | Dir | Proc Name | PID | TID | FD | FD Name | Arguments
---|---|---|---|---|---|---|---|---
33. 0.889018108 connect < peira 9863 9889 5 10.0.0.122:35588->10.96.0.10:53 res=0 tuple=10.0.0.122:35588->10.96.0.10:53 fd=5 (<4up>10.0.0.122:35588->10.96.0.10:53)
29. 5.890254477 connect < peira 9863 9889 6 10.0.0.122:38428->10.96.0.10:53 res=0 tuple=10.0.0.122:38428->10.96.0.10:53 fd=6 (<4up>10.0.0.122:38428->10.96.0.10:53)
34. 0.889577795 connect < peira 9863 9877 5 10.0.0.122:41658->10.104.171.164:80 res=115 (EINPROGRESS) tuple=10.0.0.122:41658->10.104.171.164:80 fd=5 (<4t>10.0.0.122:41658->10.104.171.164:80)
29. 5.891178653 connect < peira 9863 9883 5 10.0.0.122:41654->10.104.171.164:80 res=115 (EINPROGRESS) tuple=10.0.0.122:41654->10.104.171.164:80 fd=5 (<4t>10.0.0.122:41654->10.104.171.164:80)
29. 5.890254488 connect < peira 9863 9878 5 10.0.0.122:49447->10.96.0.10:53 res=0 tuple=10.0.0.122:49447->10.96.0.10:53 fd=5 (<4up>10.0.0.122:49447->10.96.0.10:53)
33. 0.889018158 connect < peira 9863 9888 6 10.0.0.122:55967->10.96.0.10:53 res=0 tuple=10.0.0.122:55967->10.96.0.10:53 fd=5 (<4up>10.0.0.122:55967->10.96.0.10:53)
12. 1.740326513 connect < peira 10366 10389 3 10.0.0.199:32856->10.106.200.181:80 res=115 (EINPROGRESS) tuple=10.0.0.199:32856->10.106.200.181:80 fd=3 (<4t>10.0.0.199:32856->10.106.200.181:80)
33. 6.741282441 connect < peira 10366 10386 3 10.0.0.199:32862->10.106.200.181:80 res=115 (EINPROGRESS) tuple=10.0.0.199:32862->10.106.200.181:80 fd=3 (<4t>10.0.0.199:32862->10.106.200.181:80)
30. 1.557471743 connect < peira 10157 10195 6 10.0.0.199:33402->10.96.0.10:53 res=0 tuple=10.0.0.199:33402->10.96.0.10:53 fd=6 (<4up>10.0.0.199:33402->10.96.0.10:53)
31. 1.639431119 connect < peira 10157 10185 6 10.0.0.199:37400->10.96.0.10:53 res=0 tuple=10.0.0.199:37400->10.96.0.10:53 fd=6 (<4up>10.0.0.199:37400->10.96.0.10:53)
11. 1.639434705 connect < peira 10283 10342 6 10.0.0.199:39403->10.96.0.10:53 res=0 tuple=10.0.0.199:39403->10.96.0.10:53 fd=6 (<4up>10.0.0.199:39403->10.96.0.10:53)
11. 1.640011117 connect < peira 10283 10334 5 10.0.0.199:40171->10.104.171.164:80 res=115 (EINPROGRESS) tuple=10.0.0.199:40171->10.104.171.164:80 fd=5 (<4t>10.0.0.199:40171->10.104.171.164:80)
32. 6.641836788 connect < peira 10283 10331 5 10.0.0.199:40186->10.104.171.164:80 res=115 (EINPROGRESS) tuple=10.0.0.199:40186->10.104.171.164:80 fd=5 (<4t>10.0.0.199:40186->10.104.171.164:80)
33. 6.740710595 connect < peira 10366 10388 3 10.0.0.199:42554->10.96.0.10:53 res=0 tuple=10.0.0.199:42554->10.96.0.10:53 fd=3 (<4up>10.0.0.199:42554->10.96.0.10:53)
30. 1.557100802 connect < peira 10157 10185 5 10.0.0.199:42000->10.96.0.10:53 res=0 tuple=10.0.0.199:42000->10.96.0.10:53 fd=5 (<4up>10.0.0.199:42000->10.96.0.10:53)
32. 1.739000258 connect < peira 10366 10387 6 10.0.0.199:44910->10.96.0.10:53 res=0 tuple=10.0.0.199:44910->10.96.0.10:53 fd=6 (<4up>10.0.0.199:44910->10.96.0.10:53)
12. 1.739669902 connect < peira 10366 10388 3 10.0.0.199:47189->10.96.0.10:53 res=0 tuple=10.0.0.199:47189->10.96.0.10:53 fd=3 (<4up>10.0.0.199:47189->10.96.0.10:53)
11. 1.639642948 connect < peira 10283 10333 5 10.0.0.199:49235->10.96.0.10:53 res=0 tuple=10.0.0.199:49235->10.96.0.10:53 fd=5 (<4up>10.0.0.199:49235->10.96.0.10:53)
32. 6.641800261 connect < peira 10283 10342 5 10.0.0.199:54827->10.96.0.10:53 res=0 tuple=10.0.0.199:54827->10.96.0.10:53 fd=5 (<4up>10.0.0.199:54827->10.96.0.10:53)
10. 1.655081089 connect < peira 10157 10197 5 10.0.0.199:55640->10.107.194.50:80 res=115 (EINPROGRESS) tuple=10.0.0.199:55640->10.107.194.50:80 fd=5 (<4t>10.0.0.199:55640->10.107.194.50:80)
31. 6.560621362 connect < peira 10157 10195 5 10.0.0.199:55668->10.107.194.50:80 res=115 (EINPROGRESS) tuple=10.0.0.199:55668->10.107.194.50:80 fd=5 (<4t>10.0.0.199:55668->10.107.194.50:80)
31. 6.559428185 connect < peira 10157 10228 5 10.0.0.199:56696->10.96.0.10:53 res=0 tuple=10.0.0.199:56696->10.96.0.10:53 fd=5 (<4up>10.0.0.199:56696->10.96.0.10:53)
33. 6.740720005 connect < peira 10366 10387 6 10.0.0.199:60315->10.96.0.10:53 res=0 tuple=10.0.0.199:60315->10.96.0.10:53 fd=6 (<4up>10.0.0.199:60315->10.96.0.10:53)
  
```

```

System Event 16980: 61 bytes
Sysdig Event
Event Information
Event Source: syscall
Latency: 9263
Latency (s): 0
Latency (ns): 9263
Human-Readable Latency: 92us
Direction: <
Type: connect
Category: net
CPU Number: 0
Arguments: res=115 (EINPROGRESS) tuple=10.0.0.122:45686->10.96.174.75:80 fd=5 (<4t>10.0.0.122:45686->10.96.174.75:80)
Information: res=115 (EINPROGRESS) tuple=10.0.0.122:45686->10.96.174.75:80 fd=5 (<4t>10.0.0.122:45686->10.96.174.75:80)
Return Value: EINPROGRESS
Raw Return Value: -115
Failed: True
Is I/O: False
Is Read: False
Is Write: False
Is Wait: False
Error Count: 1
Network Error Count: 1
  
```

Stratoshark öffnet **Event Files** mit den Extensions **.scap** (von sysdig) und **.pcapng** (Wireshark)

Hier zeigt **Stratoshark** (Windows Version) einen **HTTP 404 Error**, aufgezeichnet auf einem **Linux Web Server**.

Das File **404Error.scap** ist als Demo verfügbar auf [GitHub](#)

Wireshark kombiniert mit Stratoshark

Wireshark_http2demo.pcapng

No.	Time	Source	Destination	Protocol	Length	Cum.Bytes	Info
29	16:06:28.509405	172.28.184.41	195.181.175.40	TCP	74	74	38028 → 80 [SYN] Seq=3605851958 Win=642
30	16:06:28.535264	195.181.175.40	172.28.184.41	TCP	74	148	80 → 38028 [SYN, ACK] Seq=3168821410 Ac
31	16:06:28.535281	172.28.184.41	195.181.175.40	TCP	66	214	38028 → 80 [ACK] Seq=3605851959 Ack=316
32	16:06:28.535345	172.28.184.41	195.181.175.40	HTTP	195	409	GET / HTTP/1.1
33	16:06:28.565546	195.181.175.40	172.28.184.41	TCP	66	475	80 → 38028 [ACK] Seq=3168821411 Ack=360
34	16:06:28.565592	195.181.175.40	172.28.184.41	TCP	1514	1989	[TCP Previous segment not captured] 80
35	16:06:28.565598	172.28.184.41	195.181.175.40	TCP	78	2067	[TCP Dup ACK 31#1] 38028 → 80 [ACK] Seq

Frame 32: 195 bytes on wire (1560 bits), 195 by
 Ethernet II, Src: 16:40:4a:eb:23:75 (16:40:4a:eb:23:75), Dst: 08:00:27:00:00:00
 Internet Protocol Version 4, Src: 172.28.184.41, Dst: 195.181.175.40
 Transmission Control Protocol, Src Port: 38028, Dst Port: 80
 Hypertext Transfer Protocol

GET / HTTP/1.1
 User-Agent: Wget/1.21.1
 Accept: */*
 Accept-Encoding: identity
 Host: www.http2demo.io
 Connection:

Zugriff auf eine Webseite, gleichzeitig aufgezeichnet mit **Wireshark** und **Stratoshark** (Danke Walter Hofstetter)



Wireshark zeigt den **Frame HTTP Get** als Zugriff auf eine Webseite.

Stratoshark_htt2demo.pcapng

Time	Event name	Dir	Proc Name	PID	TID	FD	FD Name	Container Name	Arguments	Info
16:06:28.535325974	write	<	wget	2421515	2421515	2	/dev/pts/0	host	res=11...	write
16:06:28.535331066	select	>	wget	2421515	2421515			host		select
16:06:28.535332969	select	>	wget	2421515	2421515			host	res=1	select
16:06:28.535333933	write	>	wget	2421515	2421515	3	172.28.184.4...	host	fd=3(<...	write, fd=3
16:06:28.535351215	write	<	wget	2421515	2421515	3	172.28.184.4...	host	res=12...	write
16:06:28.535359624	ioctl	<	wget	2421515	2421515	0	/dev/pts/0	host	fd=0(<...	ioctl
16:06:28.535360813	ioctl	<	wget	2421515	2421515	0	/dev/pts/0	host	res=0	ioctl
16:06:28.535361809	getppgrp	>	wget	2421515	2421515			host		getppgrp
16:06:28.535362188	getppgrp	<	wget	2421515	2421515			host		getppgrp
16:06:28.535364593	write	>	wget	2421515	2421515	2	/dev/pts/0	host	fd=2(<...	write, fd=2

System Event 15473: 118 bytes
 Sysdig Event
 Event Information
 Event Arguments
 Process Information
 Process Ancestors
 File Descriptor Information
 Container Information
 User Information
 Group Information

P...GET / HTTP/1.1
 User-Agent: Wget/1.21.1
 Accept: */*
 Accept-Encoding: identity
 Host:



Stratoshark zeigt denselben **HTTP Get** als **Event** mit IP-Adressen, Process ID, Thread ID und vielen weiteren Details.

Stratoshark Detail Fenster

The screenshot shows the Stratoshark application window. At the top, there is a menu bar with options: File, Edit, View, Go, Capture, Analyze, Statistics, Tools, Help. Below the menu is a toolbar with various icons. The main area displays a table of network events. Two rows are highlighted in red:

No.	Time	Event name	Dir	Proc Name	PID	TID	FD	FD Name
15473	16:06:28.535351215	write	<	wget	2421515	2421515	3	172.28.184.41
15474	16:06:28.535359624	ioctl	>	wget	2421515	2421515	0	/dev/pts/0

Below the table, the details for event 15473 are expanded. The 'File Descriptor Information' section is highlighted with a red box and contains the following details:

```

System Event 15473: 118 bytes
Sysdig Event
Event Information
Event Arguments
Process Information
Process Ancestors
File Descriptor Information
  FD Number: 3
  FD Type: ipv4
  FD Type Char: 4
  FD Name: 172.28.184.41:38028->195.181.175.40:80
  FD Client Address (IPv4): 172.28.184.41
  FD Server Address (IPv4): 195.181.175.40
  FD Local Address (IPv4): 172.28.184.41
  FD Remote Address (IPv4): 195.181.175.40
  FD Client Port: 38028
  FD Server Port: 80
  FD Local Port: 38028
  FD Remote Port: 80
  FD IP Protocol: tcp
  FD Socket Family: ip
  FD Server: False
  FD ID: 24215153
  FD Container Name: :172.28.184.41:38028->195.181.175.40:80
  FD Connected: True
  FD Name Changed: False
  FD Device: 0
  FD Major Device: 0
  FD Minor Device: 0
  FD Inode Number: 0
  FD Upper Layer: False
  FD Lower Layer: False
Container Information
User Information
  User ID: 1000
  User Name: waltherh
  Home Directory: /home/waltherh
  Shell: /bin/bash
  Login User ID: 1000
  Login User Name: waltherh
Group Information
  Group ID: 1001
  
```



In der Übersicht zeigt **Stratoshark** anstelle von Paketen pro Zeile einen **System Event**.

Die Zeilen im **Detail Fenster** können aufgeklappt werden und zeigen einen erstaunlichen Detaillierungsgrad.



Obschon noch im **Pre-Release** Stadium, zeigt **Stratoshark** bereits sein riesiges Potenzial.

Vielen Dank an **Gerald Combs**, **Loris Degioanni** und ihr Team für dieses grossartige Tool.

PacketFest '25 in Zürich

News für die ntop- und Wireshark-Community: Am **8. und 9. Mai** findet das erste **PacketFest** statt.

PacketFest ist eine zweitägige technische Veranstaltung, bei der die ntop- und Wireshark-Community zusammenkommt, um Themen wie Netzwerküberwachung, Troubleshooting, Cybersicherheit und Open-Source-Technologien zu diskutieren.

Das Hauptziel der Veranstaltung besteht darin, zu zeigen, wie diese aktuellen Herausforderungen mit Hilfe bevorzugter Tools effektiv angegangen werden können.

Eine wertvolle Möglichkeit mit Gleichgesinnten in Kontakt zu treten, Erfahrungen auszutauschen und die **neuesten Funktionen von ntop und Wireshark** zu erfahren.



Walter Hofstetter: Paketguru der ersten Stunde, wird zeigen, wie mit Wireshark **Cyber Threats** erkannt werden können.

Rolf Leutert: Ich werde Erkenntnisse aus **40 Jahren Paketanalyse** schildern und neue Wireshark Funktionen zeigen.

Es erwarten Sie zahlreiche weitere interessante Themen und Anwenderberichte.
Der Gründer von Wireshark, **Gerald Combs**, wird zugeschaltet folgenden Beitrag präsentieren
“Wiresharchaeology: How it started and where we're headed”

PacketFest '25 in Zürich

Dank des Hauptsponsors [ntop](#) können die Teilnahmekosten minimal gehalten werden.

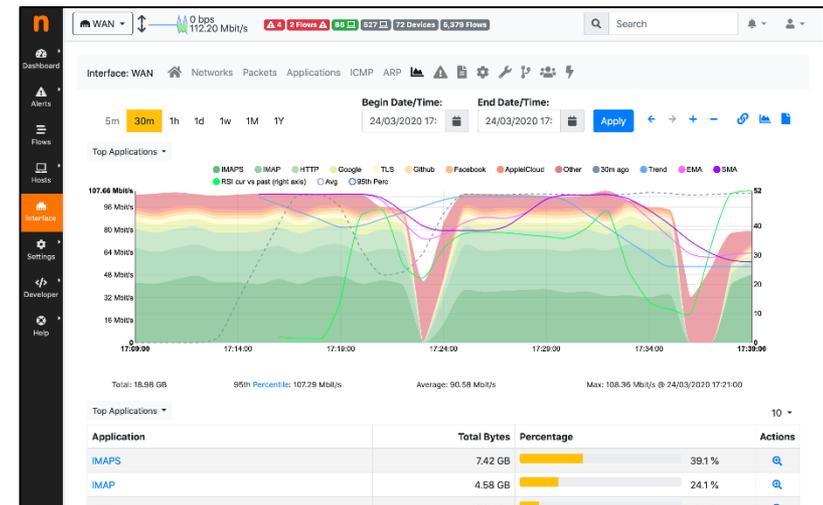
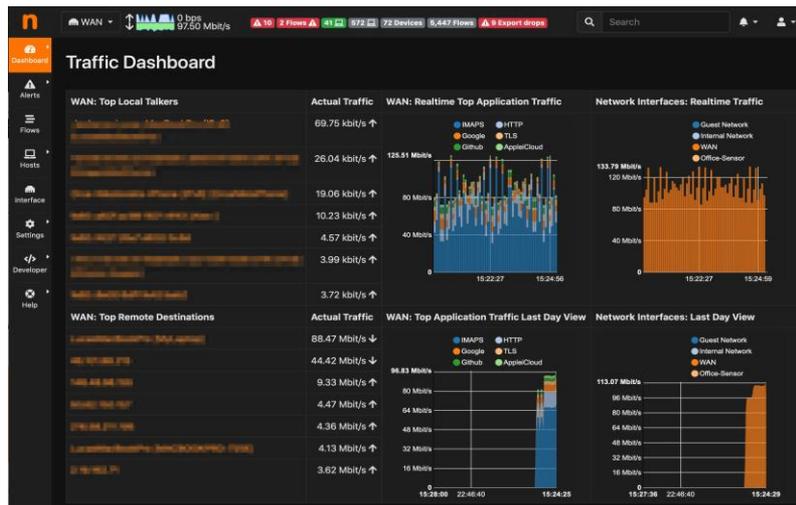
Bei schneller [Anmeldung](#) bis 28. Feb. **CHF 125.-**, ab 1. März **CHF 150.-** (inkl. Mittagessen und Pausenverpflegung).

Vor dem PacketFest findet **am 7. Mai** die **Pre-Conference** mit einem **ntopng Training** statt. Die Anzahl Teilnehmer ist beschränkt. Die Teilnahme ist in den Kosten vom PacketFest **eingeschlossen**.

Das **Open-Source-Tool ntopng** der Firma [ntop](#) wird für die webbasierte Überwachung des Netzwerkverkehrs sowie der Sicherheitsaspekte eingesetzt und bietet u.a. folgende Funktionen:

- Passives Erfassen des Netzwerkverkehrs
- Sammeln von Netzwerkflüssen (NetFlow, sFlow usw.)
- Überwachen von ausgewählten Netzwerkgeräten
- Überwachen der Netzwerkinfrastruktur über SNMP

ntopng zeigt nicht nur Verkehrsstatistiken, sondern analysiert auch den Verkehr, zieht Schlussfolgerungen zum beobachteten Verkehrstyp und meldet Überschreitungen von Cybersicherheitsmetriken.



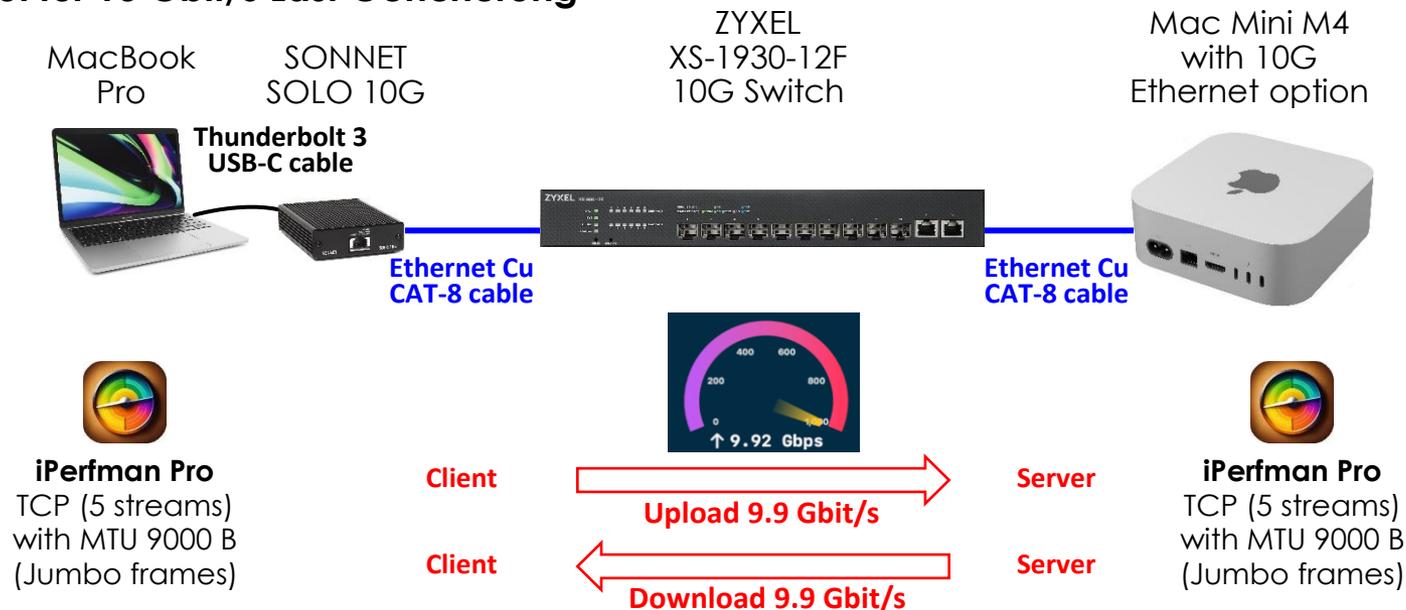


Tipps, Tricks & Traces

Kostengünstige, mobile 10 Gbit/s Aufzeichnungshardware

- Erst seit wenigen Monaten ist der neue **Mac Mini M4** von Apple auf dem Markt und setzt beim **Preis-Leistungs-Verhältnis neue Massstäbe** (CHF 900.- mit 512GB SSD und 10 Gbit Ethernet Option)
- Ein Kursteilnehmer brachte mich auf die Idee zu untersuchen, ob sich damit eine kostengünstige, **mobile Capture Unit für Wireshark** mit bis zu 10 Gbit/s realisieren lässt.
- Die erste Herausforderung war es, eine Testanordnung zu erstellen, die in der Lage ist die volle Datenrate von **10 Gbit/s zu generieren**.
- Ein erster Versuch mit einem künstlich erzeugten Switch **Spanning-Tree-Loop** ergab nur ca. 2 Gbit/s.
- Mit **iPerfman Pro** (eine GUI-Version des bekannten Tools **iPerf3**) und optimierten TCP-Parametern konnte Vollast auf dem 10 Gbit Ethernet generiert werden.

Testlayout für 10 Gbit/s Last Generierung





Tipps, Tricks & Traces

Kostengünstige, mobile 10 Gbit/s Aufzeichnungshardware

- Das Ziel war es zu ermitteln, **wieviele Prozent** der 9.9 Gbit/s der **Mac Mini** aufzuzeichnen vermag.
- Aus verschiedenen Gründen (wegen zusätzlicher CPU-Last, Ethernet Offloading-Funktionen usw.) ist es in der Praxis **nicht empfohlen**, Wireshark auf dem Sender oder Empfänger aufzeichnen zu lassen.
- Zur Vereinfachung des Testlayouts und in Ermangelung eines zweiten Mac Mini wurde hier (entgegen der Empfehlung) **trotzdem mit Wireshark auf dem Mac Mini aufgezeichnet**, gleichzeitig zum laufenden **iPerfman Pro**.
- Die Resultate waren erstaunlich: **Kein Frameverlust während der gesamten Aufzeichnung!**

Testlayout für 10 Gbit/s Aufzeichnung

Mac Mini M4 with 10G Ethernet option



Ethernet Cu CAT-8 cable

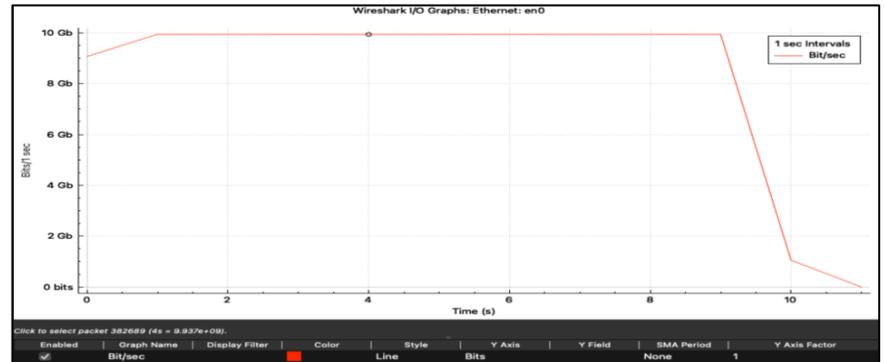
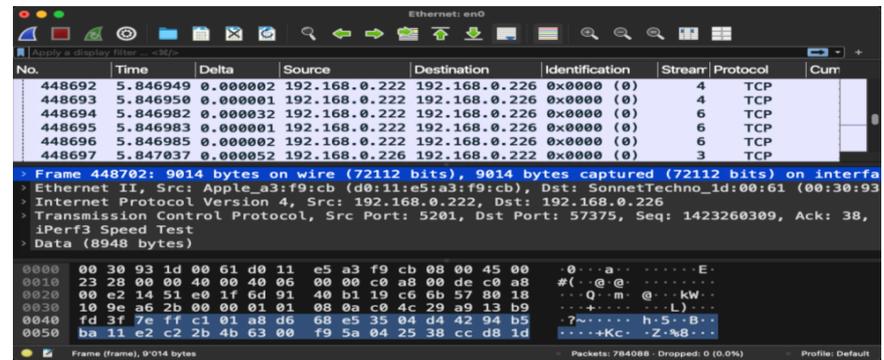


iPerfman Pro TCP (5 streams) with MTU 9000 B (Jumbo frames)

No dropped Frames!



9.9 Gigabit/s Throughput





Tipps, Tricks & Traces

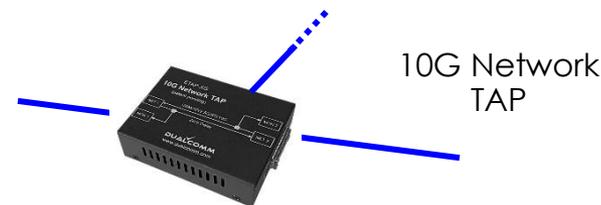
Kostengünstige, mobile 10 Gbit/s Aufzeichnungshardware

- Der Mac Mini braucht zur Bedienung externes Zubehör wie **Monitor, Keyboard und Maus**.
- Für eine portable Variante sind jedoch **möglichst wenig Zusatzgeräte** erwünscht.
- Ein **remote Zugriff** auf den Mac Mini von einem **MacBook** oder **Windows PC** ist dabei ideal.
- Eine andere Variante mit einem **iPad mit Keyboard** und USB-C Kabel ist ebenfalls möglich.
- In der Praxis würde der Mac Mini an den **Mirror Port eines Switches** oder an einen **10G TAP** angeschlossen.
- Mit entsprechenden Konvertern kann die Konfiguration auch für **10G Fiber** eingesetzt werden.

Network Access



or



Capture Unit

Mac Mini M4 with 10G Ethernet option



10G Ethernet Cu CAT-8 cable

MAC Remote Control



or





Kostengünstige, mobile 10 Gbit/s Aufzeichnungshardware

Zusammenfassung

- In der verwendeten Konfiguration zeigten die Aufzeichnungen **keinen Paketverlust** bei **vollem 10 Gbit Durchsatz**. Die guten gemessenen Resultate sind **nur ein Snapshot** in einer künstlich aufgebauten Umgebung, geben jedoch einen ersten Eindruck über die Leistungsfähigkeit des **Mac Mini als Capture Unit**.
- Kommerziell erhältliche 10G Capture Appliances bieten noch **zusätzliche Funktionen wie graphische Darstellungen/Statistiken und mehr**. Dies jedoch zu einem mindestens 5-fachen Preis gegenüber der Mac Mini Variante. Diese Kosten lassen sich bei nur gelegentlichem Gebrauch oft nicht rechtfertigen.
- Leider hatte ich bis jetzt noch keine Gelegenheit, diese Konfiguration mit dem Mac Mini in einem **Live-Netzwerk praktisch zu testen**. Ich werde dies jedoch beim nächsten Troubleshooting-Einsatz bei einem Kunden mit 10 Gigabit Netzwerk nachholen.
- Ein 10 Gigabit Netzwerk ist in der Regel so dimensioniert, dass eine volle Auslastung nur selten erreicht wird (und sonst müsste es auf 40 Gigabit aufgerüstet werden). Deshalb kann angenommen werden, dass die Mac Mini Variante auch in einem 10 Gigabit Live-Netzwerk gute Resultate liefern kann.
- Im Weiteren gilt es zu beachten, dass ein 10G Ethernet (wie auch 1G Ethernet) immer **Full-Duplex** unterstützt, d.h. der maximal nutzbare Durchsatz beträgt 10 Gbit pro Richtung, total also **20 Gbit/s**. Dies würde natürlich den Einsatzbereich des Mac Mini übersteigen.



Tipps, Tricks & Traces

Erstellen eines VXLAN Profils

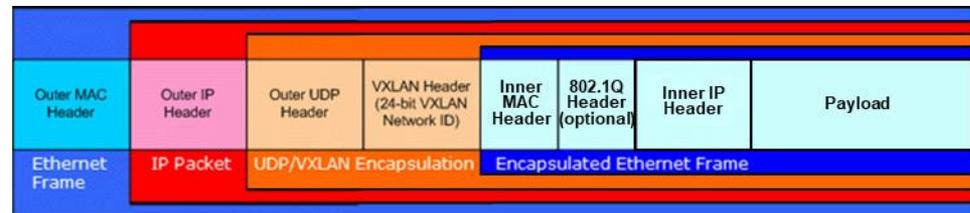
- Zahlreiche Protokolle sind **enkapsuliert**, ein Ethernet Frame wird zum Transport in ein **weiteres Protokoll gepackt**.
- Diese Pakete enthalten **mehrere MAC- und IP-Adressen**, die **des Original Frames** und die **des Tunnel-Protokolls**.
- Die Felder werden auch als **Innere (Inner) und Äussere (Outer) Header** bezeichnet.
- **Welche dieser Adressen zeigt nun Wireshark** beim Öffnen eines solchen Trace Files?
- Per Default zeigt **Wireshark die Adressen des Original Frames** (Inner Header), und nicht die des Tunnel-Protokolls.

→ Mit einem entsprechenden Profil, **lassen sich alle Adressfelder anzeigen**.

Encapsulated or Tunneled Protocols:

- **GRE tunnel** (Generic Routing Encapsulation)
- **CAPWAP tunnel** (Control And Provisioning of Wireless Access Points)
- **ICMP messages** (Internet Control Message Protocol)
- **IP in IP** (i.e., IPv4 in IPv6)
- **VPN** (Virtual Private Network)
- **VXLAN** (Virtual Extensible LAN)

VXLAN encapsulated frame with **Inner** and **Outer** MAC and IP headers





Tipps, Tricks & Traces

Erstellen eines VXLAN Profils

Trace Table:

No.	Time	Delta	Outer IP Src	Outer IP Dst	VNI	Inner IP Src	Inner IP Dst	Outer TTL	Inner TTL	Length	Protocol	Info
1	0.000	0.000	192.168.20.1	239.0.0.10	10010			16		110	ARP	Who has 192.168.10.153? Tell 192.168.10.153
2	0.000	0.000	192.168.20.2	192.168.20.1	10010			16		110	ARP	192.168.10.153 is at 00:50:56:b2:92:62
3	0.000	0.000						60		60	ARP	Who has 192.168.20.2? Tell 192.168.20.2
4	0.000	0.000						60		60	ARP	192.168.20.2 is at 00:50:56:b2:f6:01
5	0.000	0.000	192.168.20.1	192.168.20.2	10010	192.168.10.151	192.168.10.153	16	64	148	ICMP	Echo (ping) request id=0x245e, seq=1/2
6	0.000	0.000	192.168.20.2	192.168.20.1	10010	192.168.10.153	192.168.10.151	16	64	148	ICMP	Echo (ping) reply id=0x245e, seq=1/2
7	1.001	1.000	192.168.20.1	192.168.20.2	10010	192.168.10.151	192.168.10.153	16	64	148	ICMP	Echo (ping) request id=0x245e, seq=2/5
8	1.001	0.000	192.168.20.2	192.168.20.1	10010	192.168.10.153	192.168.10.151	16	64	148	ICMP	Echo (ping) reply id=0x245e, seq=2/5
9	2.000	0.998	192.168.20.1	192.168.20.2								
10	2.000	0.000	192.168.20.2	192.168.20.1								
11	2.999	0.998	192.168.20.1	192.168.20.2								
12	2.999	0.000	192.168.20.2	192.168.20.1								
13	3.999	0.999	192.168.20.1	192.168.20.2								
14	3.999	0.000	192.168.20.2	192.168.20.1								
15	4.999	0.999	192.168.20.1	192.168.20.2								
16	4.999	0.000	192.168.20.2	192.168.20.1								

Wireshark Preferences - Columns Table:

Displayed	Title	Type	Fields	Field Occurrence	Resolved	Width	Alignment
<input checked="" type="checkbox"/>	No.	Number			<input checked="" type="checkbox"/>	72	Left
<input checked="" type="checkbox"/>	Time	Time (format as specified)			<input checked="" type="checkbox"/>	68	Default
<input checked="" type="checkbox"/>	Delta	Delta time displayed			<input checked="" type="checkbox"/>	106	Center
<input checked="" type="checkbox"/>	Outer IP Src	Custom	ip.src	1	<input checked="" type="checkbox"/>	180	Default
<input checked="" type="checkbox"/>	Outer IP Dst	Custom	ip.dst	1	<input checked="" type="checkbox"/>	169	Default
<input checked="" type="checkbox"/>	VNI	Custom	vxlan.vni	0	<input checked="" type="checkbox"/>	85	Center
<input checked="" type="checkbox"/>	Inner IP Src	Custom	ip.src	2	<input checked="" type="checkbox"/>	187	Default
<input checked="" type="checkbox"/>	Inner IP Dst	Custom	ip.dst	2	<input checked="" type="checkbox"/>	180	Default
<input checked="" type="checkbox"/>	Outer TTL	Custom	ip.ttl	1	<input checked="" type="checkbox"/>	105	Center
<input checked="" type="checkbox"/>	Inner TTL	Custom	ip.ttl	2	<input checked="" type="checkbox"/>	102	Center
<input checked="" type="checkbox"/>	Length	Packet length (bytes)			<input checked="" type="checkbox"/>	77	Center
<input checked="" type="checkbox"/>	Protocol	Protocol			<input checked="" type="checkbox"/>	92	Default
<input checked="" type="checkbox"/>	Info	Information			<input checked="" type="checkbox"/>	524	Default

Callout 1: Namen und Position der Spalten

Callout 2: Im Field Occurrence lassen sich Äussere = 1 und Innere = 2 Felder auswählen



Cool Commands

- Auf der Webseite von [AnyWeb Training](#) in Zürich werden ab **12. März 2025** kurze Video-Clips mit **Cool Commands** aufgeschaltet.
- Da werden auf kurze und prägnante Weise neue und praktische Befehle aus den Bereichen **Wireshark, Cisco, Python** und **Linux** erklärt.
- Den Termin notieren oder sich für den AnyWeb Training Newsletter [registrieren](#).

The Packet Factor – neuer **Wireshark Kurs** zum Thema Network Security

- Mein langjähriger Kollege **Walter Hofstetter** ist Autor und Leiter dieses Kurses und eine Koryphäe auf dem Gebiet **Netzwerk-Analyse und -Security**.
- Der Kurs **The Packet Factor** zeigt, wie Wireshark im Bereich **Intrusion Detection and Prevention** auf Paketebene zur Erkennung von Netzwerk-Manipulationen und -Attacken eingesetzt werden kann.
- Es gibt in dieser Kombination und diesem Detaillierungsgrad aktuell **keinen vergleichbaren Kurs** auf dem Markt. Mehr Details und Anmeldung bei [AnyWeb Training](#) in Zürich

→ **Walter**, wann entwickeln wir den ersten **Stratoshark Kurs?**



Unsere Wireshark-Protokoll-Kurse & andere Events

- **PacketFest '25 in Zürich** **8.-9.05.2025** [→ Zur Anmeldung PacketFest '25](#)
 Pädagogische Hochschule Zürich **→ Garantierte Durchführung**
- **Öffentliche Kurse in der Schweiz**
 AnyWeb Training in Zürich [→ Zur Anmeldung bei AnyWeb](#)
TCP/IP Analyse mit Wireshark **11.-13.03.2025** **→ Garantierte Durchführung**

Studerus in Schwerzenbach [→ Zur Anmeldung bei Studerus](#)
VoIP Analyse mit Wireshark **20.03.2025**

- **Öffentliche Kurse in Österreich** [→ Zur Anmeldung bei ARROW](#)
 Bei Arrow ECS GmbH in Wien
- **Öffentliche Kurse in Deutschland** [→ Zur Anmeldung bei ALSO](#)
 Remote Kurse bei ALSO
- **Übersicht aller öffentlichen Kurse** [→ Kurse Leutert NetServices](#)

Unsere Spezialität sind **Firmenkurse** oder **Tech-Sessions** nach ihren Wünschen zu den Themen:

- **Einführung Netzwerkanalyse, Wireshark Tipps & Tricks, TCP/IP, QUIC, WLAN, VoIP und IPv6**
- [YouTube Webinar](#) (1h) **Troubleshooting WLANs mit Wireshark** aufgezeichnet durch [onway](#).

Hier unser [Newsletter Archiv](#). Es würde uns freuen, Sie in einem unserer Kurse begrüßen zu können.

Have fun and enjoy sniffing, Rolf Leutert