WIRE**SHARK Newsletter April 2021**

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 3.4x**
- **QUIC Protocol,** baldiger Ersatz für **TCP**?
- Neuer, preiswerter **10GB TAP**

Tipps, Tricks & Traces:

- **Files zusammenführen** mit Wireshark
- **Zeitstempel verschieben** mit Wireshark
- **Frames kürzen** oder **Felder entfernen** mit **Editcap**

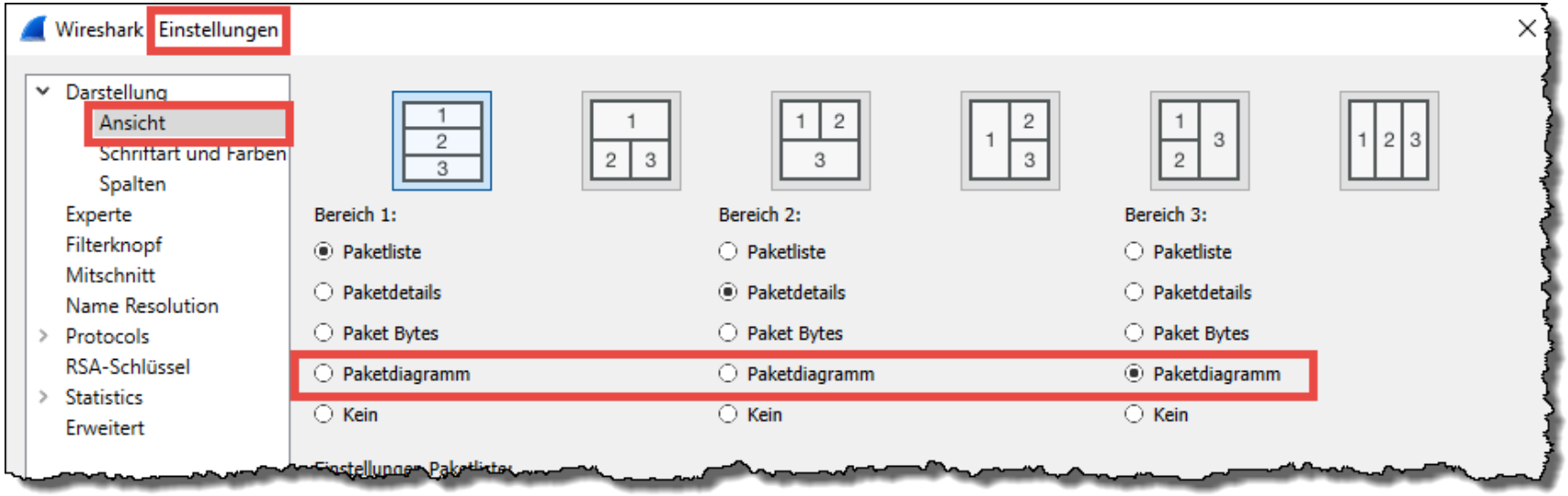
Kurse & Events:

- Aktuelle **Kursdaten**
- Kostenlose Teilnahme am **Virtuellen SharkFest'21**

Neue Funktionen ab Wireshark Version 3.4x

Paketdiagramm Bereich

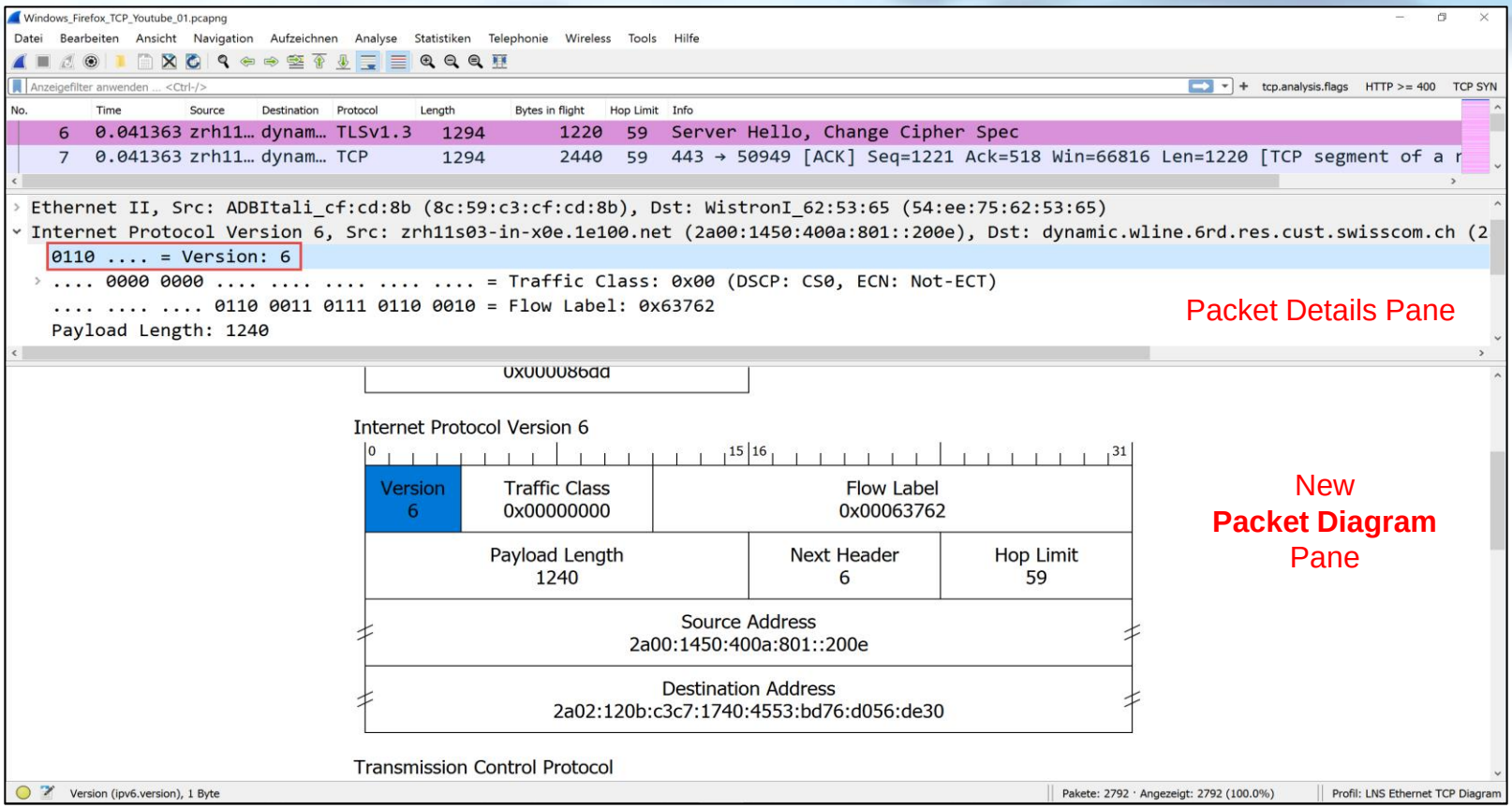
- Ein neuer Bereich zeigt grafisch den Headeraufbau des ausgewählten Paketes
- Die Aktivierung ist zu finden unter Bearbeiten → Einstellungen → Ansicht



- Da Wireshark aktuell nur maximal drei Bereiche anzeigen kann, muss ein anderes Fenster 'geopfert' werden ☹. Am ehesten kann man wohl auf den Bereich Paket Bytes verzichten.
- Diese Einstellung wird im aktiven Profil gespeichert; eine schnelle Umschaltmöglichkeit wäre deshalb eine Kopie des Profils zu erstellen, um dann einfach zwischen den Profilen hin und her wechseln zu können. Wie das geht haben sie in einem meiner Kurse gelernt ☺.

Neue Funktionen ab Wireshark Version 3.4x

Paketdiagramm Bereich



- Wird im Paketdetail ein Header angeklickt, wird das Feld im Paketdiagramm blau markiert
- Und umgekehrt: Klick auf einen Header im Paketdiagramm zeigt diesen im Paketdetail
- Rechter Mausklick im Bereich Paketdiagramm zeigt weitere Einstellungsmöglichkeiten

Quick UDP Internet Connections (QUIC)

QUIC ist ein neues Netzwerkprotokoll, das grosse Chancen hat, das über 40 Jahre alte TCP zu ersetzen. 2013 von Google präsentiert und entwickelt mit dem Ziel den Internetverkehr insgesamt zu beschleunigen, wurde es zuerst auf Google-Servern und Google-Browser CHROME implementiert.

Seit 2016 arbeitet auch die Internet Engineering Task Force (IETF) an der Weiterentwicklung und Standardisierung des QUIC-Protokolls. Das Original wurde umbenannt in gQUIC (Google QUIC) während die IETF Version unter dem Namen QUIC noch 2021 standardisiert (RFC) werden soll.

Wireshark Filter: gquic für das original Google Protocol (wenig implementiert)

Wireshark Filter: quic für die IETF Protocol-Version

Da zusammen mit QUIC auch TLS 1.3 und HTTP Version 3 eingeführt wird, wird QUIC oft mit der Bezeichnung **HTTP/3** aufgeführt. Unter diesem Namen kann das neue Protokoll bereits in vielen Browsern testweise aktiviert werden (Safari, Firefox, Microsoft Edge).

QUIC hat aktuell einen Anteil von 7% des globalen Netzwerkverkehrs, es kann jedoch erwartet werden, dass nach Fertigstellung als IETF-Standard dieser Anteil stark zunehmen wird.

Im Oktober 2020 hat Facebook seine Server und Android/IOS Apps auf QUIC umgestellt. Aktuell beträgt der QUIC-Anteil des Facebook-Datenverkehrs bereits 75%.

https://engineering.fb.com/2020/10/21/networking-traffic/how-facebook-is-bringing-quic-to-billions

→ Es ist also Zeit, sich auch aus Sicht der **Wireshark-Netzwerkanalyse** mit QUIC zu befassen.

Auf den folgenden Seiten dazu die wichtigsten Informationen im Überblick. Mehr QUIC Details und Analysen sind ab sofort in unserem TCP/IP Kurs integriert (siehe Kurshinweise auf Seite 25)

Quick UDP Internet Connections (QUIC)

TCP vs QUIC

Ziemlich genau vor 40 Jahren wurde das Transmission Control Protocol (TCP) standardisiert und spielt bis heute die wichtigste Rolle in der paketorientierten Datenübertragung.

Der TCP-Basis-Header entspricht immer noch dem ersten Standard (RFC 793) vom Sept. 1981 und wurde durch Zufügen von TCP-Optionen an die stark veränderten Übertragungsbedingungen wie höhere Bandbreiten, höhere Laufzeiten (durch grössere Distanzen) usw. angepasst.

Eine erstaunliche Beständigkeit, die in der schnelllebigen ICT einzigartig ist.
Ein Kompliment an die Entwickler und Weiterentwickler von TCP!

Trotzdem ergaben sich durch die exponentiell wachsenden Datenraten über die Jahre einige TCP-Limitationen, welche zur Entwicklungszeit nicht vorhersehbar waren.

QUIC soll schneller und sicherer als TCP werden (die Analyse wird jedoch schwieriger!)

- Beschleunigter Verbindungsaufbau durch weniger Hand-Shake-Zyklen
- Schneller Wiederaufbau bei Verbindungsunterbrüchen (WLAN!)
- Integrierte Verschlüsselung mit TLS 1.3 (nicht verhandelbar)
- Initialer Handshake bereits verschlüsselt
- Parallele Verbindungen (multiplexed Streams), fliegender UDP-Port / IP-Adresse Wechsel
- Keine Blockierung der Übertragung durch verlorene Pakete (no Head-of-line blocking)
- Verwendet UDP anstatt TCP, Acknowledges und Fluss-Steuerung mit QUIC Control Packets
- Integriertes HTTP/3 Applikationsprotokoll

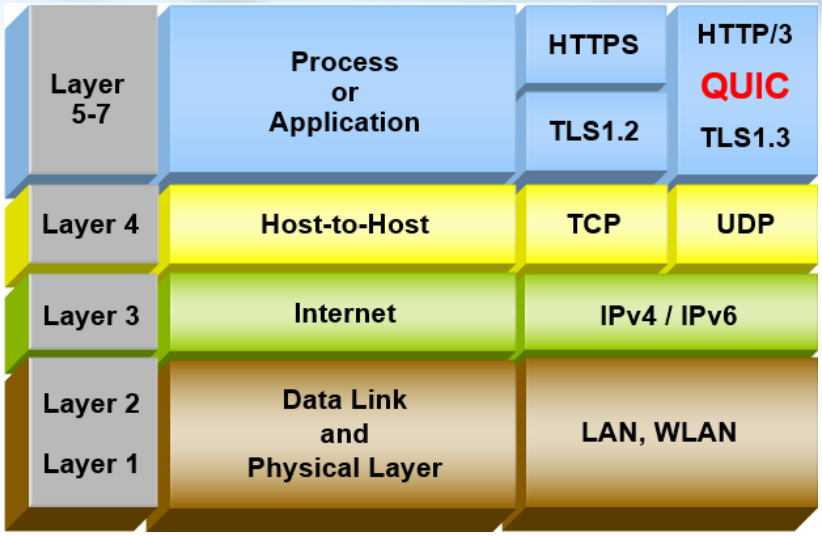
Quick UDP Internet Connections (QUIC)

TCP vs QUIC

Das Konzept des Kommunikation-Modells erlaubt den Austausch eines Layers ohne Einfluss auf die benachbarten oberen oder unteren Layer (z.B. IPv4 → IPv6).

QUIC ersetzt jedoch Teile des Application-Layers (HTTP/3), sodass die Anwendungen angepasst werden müssen, um QUIC unterstützen zu können.

D.h. im Browser auf den Clients und den Anwendungen auf den (Web-) Servern muss QUIC-Support implementiert werden.



Google (als Initiator von QUIC) unterstützt im Browser CHROME und auf allen Google-Servern QUIC als bevorzugtes Protocol.

Bei den meisten anderen Browsern kann QUIC als Experimental Option HTTP/3 aktiviert werden. Wenn die Option HTTP/3 aktiviert ist und der kontaktierte (Web-) Server für QUIC angepasst wurde, wird QUIC gegenüber TCP als Protokoll bevorzugt. (Gleich wie IPv6 gegen IPv4)

Ob ein Server QUIC unterstützt, kann mit dem Link auf Seite 9 getestet werden.

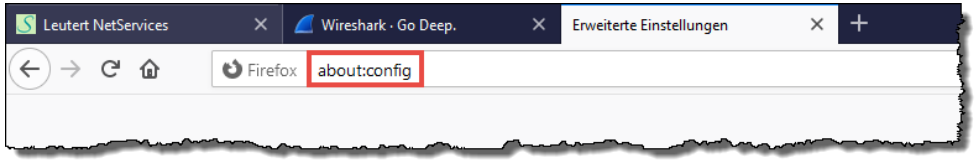
Quick UDP Internet Connections (QUIC)

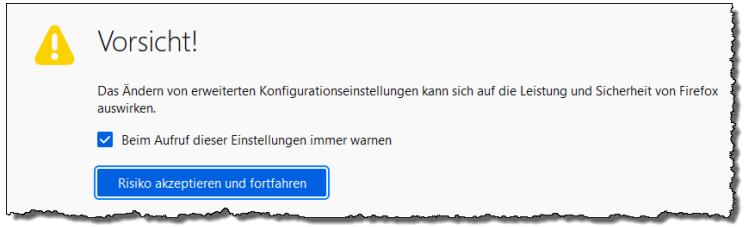
Firefox unterstützt QUIC ab Version 88.0, es muss jedoch manuell aktiviert werden.

Überprüfen der Firefox Version: rechts oben → Anwendungsmenü öffnen → Hilfe → über Firefox

Öffnen eines neuen Tabs

Eingabe: about:config





Risiko akzeptieren und fortfahren

Eingabe: network.http.http3.enabled

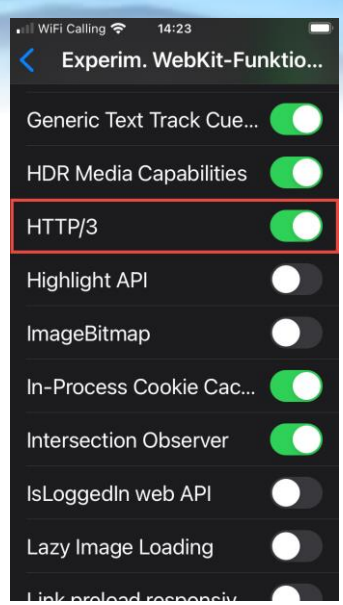
Mit Knopf rechts: false auf true wechseln

Tab schliessen

Quick UDP Internet Connections (QUIC)

Apple unterstützt QUIC ab **iOS 14** und ab **macOS Big Sur**.

Da QUIC vom IETF noch nicht als Standard (RFC) verabschiedet wurde, unterstützt Apple QUIC aktuell nur im Experimental Mode und muss manuell aktiviert werden. QUIC ist in der **Option HTTP/3** enthalten.



Ab **iOS 14** unter Einstellungen:

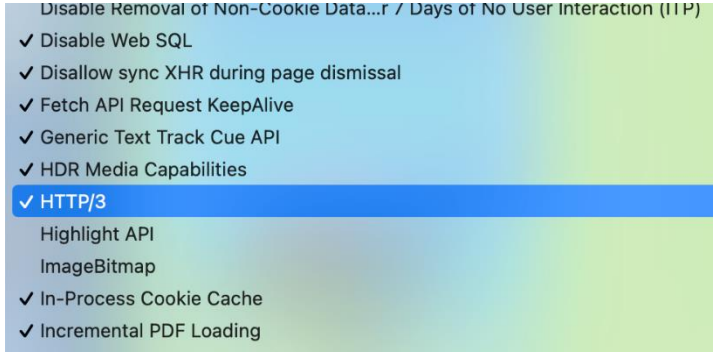
→ Safari → Erweitert → Experimental Features → **HTTP/3 → ON**

Ab **macOS Big Sur** Safari Browser öffnen

→ Einstellungen → Erweitert → Menü 'Entwickler' anzeigen → **ON**

Safari schliessen und neu starten, dann

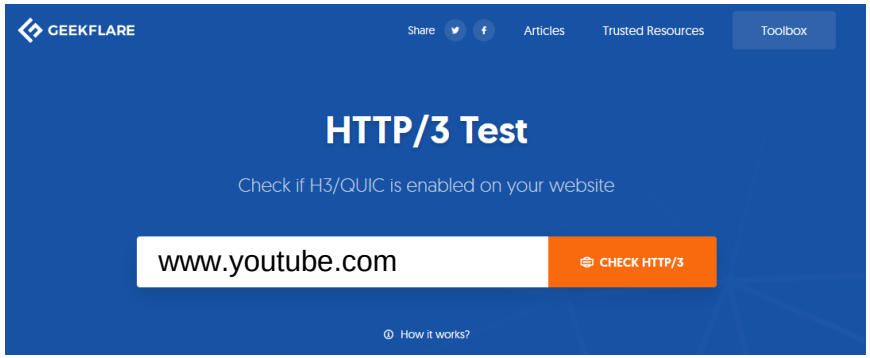
→ Entwickler → Experimentelle Funktionen → **HTTP/3 → ON**

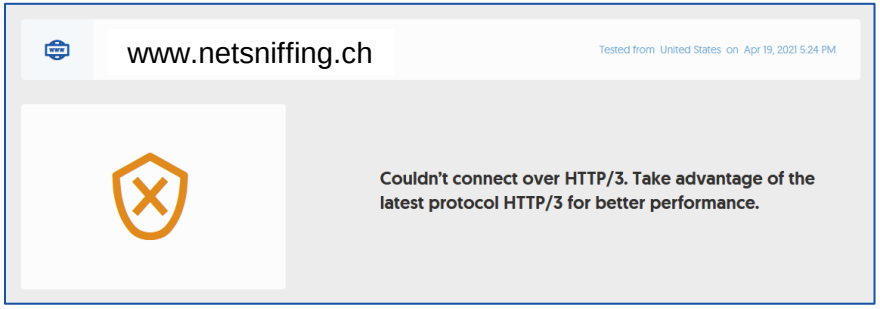


Quick UDP Internet Connections (QUIC)

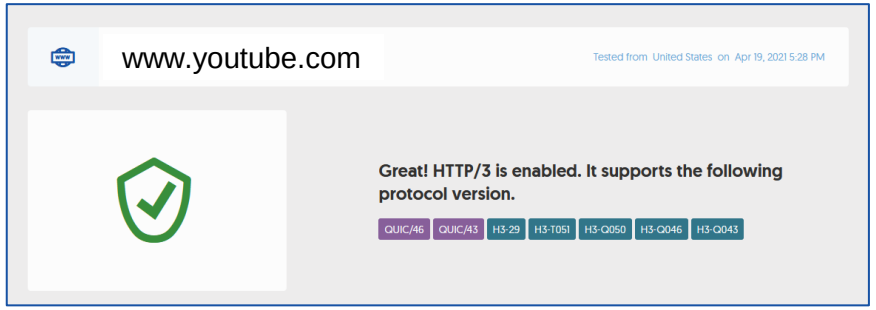
Überprüfen ob ein Webserver QUIC unterstützt

- Bei Verwendung eines anderen Browsers als CHROME muss HTTP/3 manuell aktiviert werden
- Aufruf der QUIC Testseite: https://gf.dev/http3-test
- URL Eingabe des zu testenden Webservers und Feld CHECK HTTP/3 betätigen





Leider **NEIN**



OK

Quick UDP Internet Connections (QUIC)

Technische Unterschiede von **TCP mit TLS** und **QUIC mit TLS**

- TCP verwendet den **3-Way Handshake** für den Verbindungsaufbau.
- QUIC über **UDP ohne den 3-Way Handshake**, dadurch schnellerer Verbindungsaufbau

- Beim TCP findet danach (optional) der TLS **4-Way Handshake** statt
- QUIC **verwendet immer TLS** (V1.3) und startet sofort mit dem **TLS Handshake**

- Beim **TCP** ist es möglich, mit Hilfe der **ACKs** die Qualität einer Verbindung zu beurteilen
- **UDP verwendet keine ACKs**, diese werden von QUIC auf der **Applikationsschicht** behandelt

- Auch bei **verschlüsseltem Payload** (HTTPS) werden Lost Frames, Retransmissions etc. erkannt
- Der QUIC-Empfänger **meldet verlorene Pakete,** und der Sender **wiederholt** die Übertragung

- TCP verwendet **TCP-Port und IP-Adresse (Socket)** als Session-Kennzeichnung
  Wechsel von Port oder IP-Adresse ist während einer TCP-Verbindung nicht möglich
- QUIC kennzeichnet eine Verbindung mit **Source- und Destination-Connection-ID (SCID, DCID)**
  QUIC ermöglicht dadurch den (fliegenden) **Wechsel** von **UDP-Port und IP-Adresse !!!**

- TCP verwendet **Sequenz-Nummern** (zählt Bytes) in Datenpaketen und Acknowledges.
- QUIC verwendet **Paketnummern** im Bereich 0 bis 4'611'686'018'427'387'903 (2^{62})

Quick UDP Internet Connections (QUIC)

Technische Unterschiede von **TCP mit TLS** und **QUIC mit TLS** (Fortsetzung)

- TCP verwendet **eine hochzählende Sequenznummer pro Frame, 1 Frame** enthält **1 Paket**
- QUIC kann **mehrere Pakete in einem UDP-Datagram** (Frame) übertragen !!!
- ***Obige Eigenschaft wird für Wireshark Filtering, Coloring usw. besonders anspruchsvoll.***

- TCP erhöht die Sequenznummer nur bei Paketen mit Applikationsdaten im Payload
- QUIC hat **separate Paket-Zähler** für folgende drei Gruppen:
  - Initial-Pakete
  - Handshake-Pakete
  - Application-Data-Pakete

- Der TCP-Header hat immer die Länge von **40 Bytes** (ohne Optionen)
- **QUIC** bietet **zwei Header-Längen:**
  - Langer Packet Header für den Verbindungsaufbau
  - Kurzer Packet Header wenn die Verbindung steht

- TCP steuert die Übertragungsgeschwindigkeit mit **Window Size** und **Congestion Control**
- Quick hat eine verbesserte **Congestion Control**, ohne die **Einschränkungen der Window Size**

- **TCP** unterstützt mehrere **parallele Sessions (genannt Streams)** zwischen zwei Hosts
- QUIC unterstützt ebenfalls **verschiedene Streams**, jedoch innerhalb **derselben UDP-Port-Nr.**

Quick UDP Internet Connections (QUIC)

Technische Unterschiede von **TCP mit TLS** und **QUIC mit TLS** (Fortsetzung)

- Im TCP werden die Streams durch **verschiedene TCP-Port-Nummern** unterschieden
- Im QUIC werden die Streams durch eine **STREAM ID im Header** unterschieden

- Ein TCP Frame enthält immer nur Daten für **ein und denselben Stream**
- Ein QUIC Frame kann Pakete für **verschiedene Streams** enthalten (limitiert durch Paketlänge)
- ***Obige Eigenschaft wird für Wireshark Filtering, Coloring usw. besonders anspruchsvoll.***
- QUIC unterstützt 4 verschiedene Stream-Typen:
  - 0x00 Client-Initiated, Bidirectional
  - 0x01 Server-Initiated, Bidirectional
  - 0x02 Client-Initiated, Unidirectional
  - 0x03 Server-Initiated, Unidirectional

Dies sind nur einige der **wichtigsten Unterschiede** von **QUIC.** Es gibt noch viele weitere Funktionen.

QUIC ist wesentlich komplexer und anspruchsvoller als **TCP**!

Da viele QUIC-Header-Informationen **verschlüsselt** sind, wird die **Fehlersuche** und die **Implementation** in Netzwerkkomponenten wie **Firewalls, Load Balancer** etc. wesentlich erschwert. Trotzdem ist zu erwarten, dass QUIC nach Verabschiedung des IETF RFCs stark an Verbreitung zunehmen wird. Der letzte Draft vom 15. Jan. 2021 ist Version 34

 **QUIC Analyse** ist ab sofort integriert in unserem **TCP/IP Wireshark Kurs** 

Quick UDP Internet Connections (QUIC)

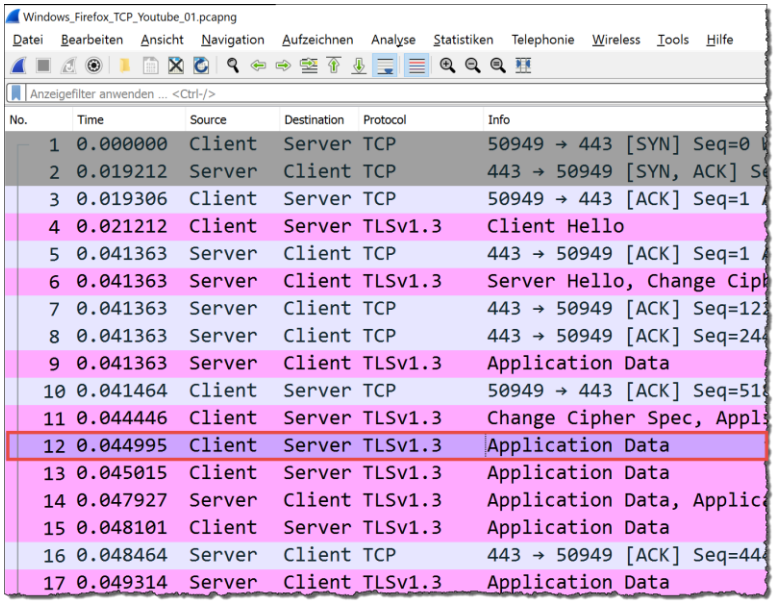
QUIC Analyse mit Wireshark

Da sich QUIC noch im Draft-Status befindet, sind bis zum RFC noch Änderungen an den Feldern zu erwarten. Trotzdem decodiert Wireshark bereits die meisten Header im Detail.

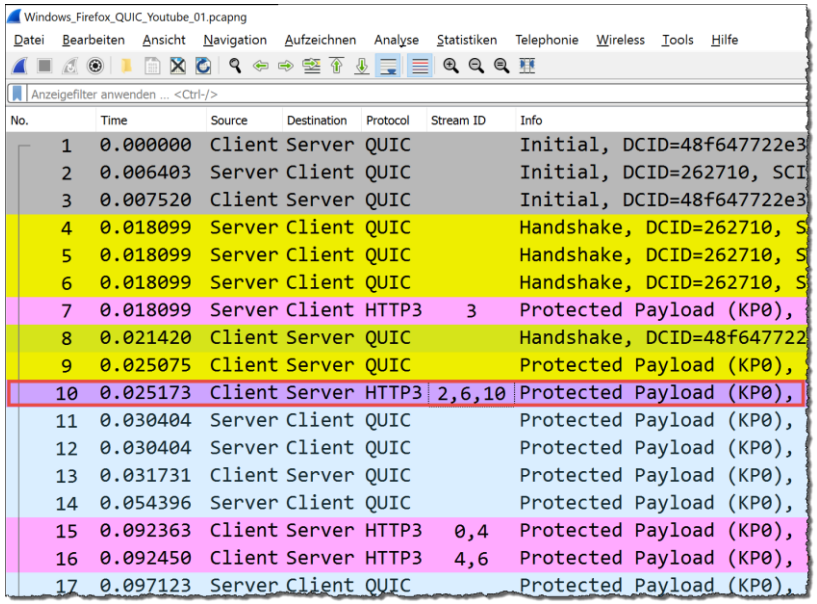
Natürlich fehlt für QUIC noch die komfortable Expert-Unterstützung wie z.B. beim TCP und wird wohl diesen hohen Level wegen der Verschlüsselung kaum je erreichen.

Die bei QUIC verwendete TLS1.3 Verschlüsselung gilt (mit normalen Mitteln) als unknackbar, es sei denn der Session key vom Client oder Server ist verfügbar (mehr im nächsten Newsletter).

Vergleich: Verbindungsaufbau mit Windows10, IPv6 und Firefox zu www.youtube.com



***TCP: 44.9ms** bis zum ersten Datenpaket vom Client*



***QUIC: 25.1ms** bis zum ersten Datenpaket vom Client*

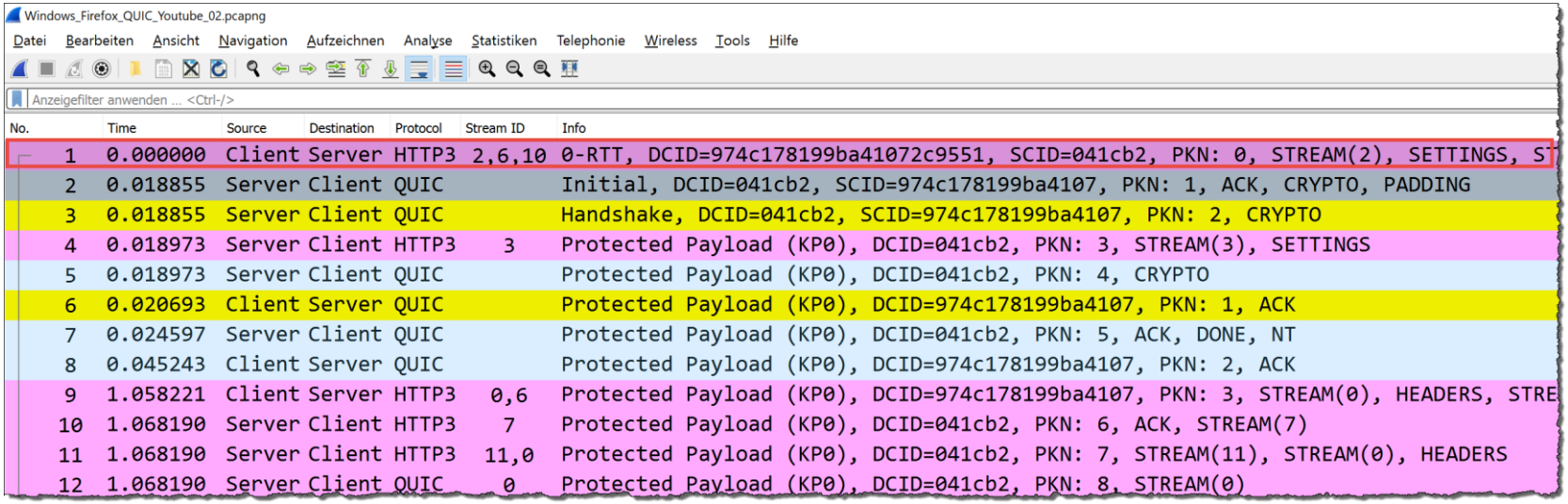
Quick UDP Internet Connections (QUIC)

QUIC Analyse mit Wireshark

TLS 1.3 (RFC 8446) braucht für den Verbindungsaufbau nur One Round Trip Time (1-RTT), im Vergleich zu TLS 1.2 mit Two RTTs (2-RTT).

Für den **Wiederaufbau** (Resumption) einer Verbindung zum selben Server braucht TLS 1.2 One RTT (1-RTT)

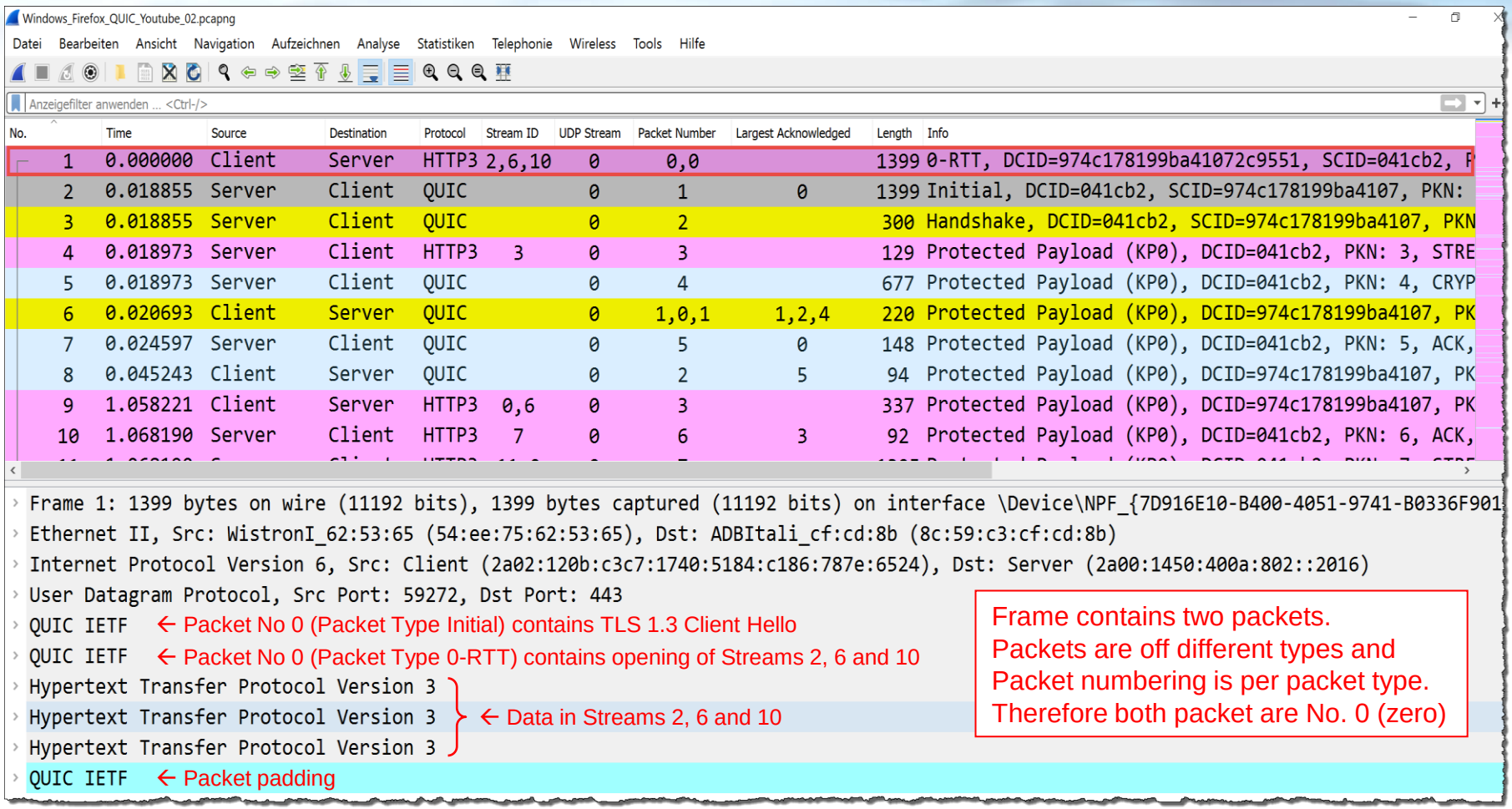
In **TLS 1.3** wurde die Funktion **Zero Round Trip Time** (0-RTT) implementiert. Da bei QUIC der TCP 3-way Handshake wegfällt, erfolgt Datentransfer und Verbindungswiederaufbau im ersten Frame.



Frame 1 QUIC-Resumption: *0-RTT Verbindung mit Stream-Daten vom Client bereits im ersten Datenpaket*

Quick UDP Internet Connections (QUIC)

QUIC Analyse mit Wireshark: Stream ID, Paket Number, Acks



Frame 1: *Client startet 0-RTT Verbindung, enthält TLS Client Hello, öffnet 3 Streams und sendet HTTP/3 Daten*

Quick UDP Internet Connections (QUIC)

QUIC Analyse mit Wireshark: Stream ID, Paket Number, Acks

No.	Time	Source	Destination	Protocol	Stream ID	UDP Stream	Packet Number	Largest Acknowledged	Length	Info
1	0.000000	Client	Server	HTTP3	2,6,10	0	0	0	1399	0-RTT, DCID=974c178199ba41072c9551, SCID=041cb2, P
2	0.018855	Server	Client	QUIC		0	1	0	1399	Initial, DCID=041cb2, SCID=974c178199ba4107, PKN:
3	0.018855	Server	Client	QUIC		0	2		300	Handshake, DCID=041cb2, SCID=974c178199ba4107, PKN
4	0.018973	Server	Client	HTTP3	3	0	3		129	Protected Payload (KP0), DCID=041cb2, PKN: 3, STRE

Frame 2: *Server antwortet mit TLS Server Hello und bestätigt Paket 0 (Packet Type Initial)*

No.	Time	Source	Destination	Protocol	Stream ID	UDP Stream	Packet Number	Largest Acknowledged	Length	Info
1	0.000000	Client	Server	HTTP3	2,6,10	0	0,0		1399	0-RTT, DCID=974c178199ba41072c9551, SCID=041cb2, PKN:
2	0.018855	Server	Client	QUIC		0	1	0	1399	Initial, DCID=041cb2, SCID=974c178199ba4107, PKN: 1, A
3	0.018855	Server	Client	QUIC		0	2		300	Handshake, DCID=041cb2, SCID=974c178199ba4107, PKN: 2,
4	0.018973	Server	Client	HTTP3	3	0	3		129	Protected Payload (KP0), DCID=041cb2, PKN: 3, STREAM(3
5	0.018973	Server	Client	QUIC		0	4		677	Protected Payload (KP0), DCID=041cb2, PKN: 4, CRYPTO

Frame 3: *Server schickt TLS Finished* ***Frame 4:*** *Server öffnet Stream 3* ***Frame 5:*** *Server schickt New Session Ticket*

No.	Time	Source	Destination	Protocol	Stream ID	UDP Stream	Packet Number	Largest Acknowledged	Length	Info
1	0.000000	Client	Server	HTTP3	2,6,10	0	0,0		1399	0-RTT, DCID=974c178199ba41072c9551, SCID=041cb2, PKN:
2	0.018855	Server	Client	QUIC		0	1	0	1399	Initial, DCID=041cb2, SCID=974c178199ba4107, PKN: 1, A
3	0.018855	Server	Client	QUIC		0	2		300	Handshake, DCID=041cb2, SCID=974c178199ba4107, PKN: 2,
4	0.018973	Server	Client	HTTP3	3	0	3		129	Protected Payload (KP0), DCID=041cb2, PKN: 3, STREAM(3
5	0.018973	Server	Client	QUIC		0	4		677	Protected Payload (KP0), DCID=041cb2, PKN: 4, CRYPTO
6	0.020693	Client	Server	QUIC		0	1,0,1	1,2,4	220	Protected Payload (KP0), DCID=974c178199ba4107, PKN: 1
7	0.024597	Server	Client	QUIC		0	5	0	148	Protected Payload (KP0), DCID=041cb2, PKN: 5, ACK, DON

Frame 6: *Client schickt TLS Finished und bestätigt Empfang Pakete 1 -4 vom Server*

No.	Time	Source	Destination	Protocol	Stream ID	UDP Stream	Packet Number	Largest Acknowledged	Length	Info
1	0.000000	Client	Server	HTTP3	2,6,10	0	0,0		1399	0-RTT, DCID=974c178199ba41072c9551, SCID=041cb2, PKN:
2	0.018855	Server	Client	QUIC		0	1	0	1399	Initial, DCID=041cb2, SCID=974c178199ba4107, PKN: 1, A
3	0.018855	Server	Client	QUIC		0	2		300	Handshake, DCID=041cb2, SCID=974c178199ba4107, PKN: 2,
4	0.018973	Server	Client	HTTP3	3	0	3		129	Protected Payload (KP0), DCID=041cb2, PKN: 3, STREAM(3
5	0.018973	Server	Client	QUIC		0	4		677	Protected Payload (KP0), DCID=041cb2, PKN: 4, CRYPTO
6	0.020693	Client	Server	QUIC		0	1,0,1	1,2,4	220	Protected Payload (KP0), DCID=974c178199ba4107, PKN: 1
7	0.024597	Server	Client	QUIC		0	5	0	148	Protected Payload (KP0), DCID=041cb2, PKN: 5, ACK, DON

Frame 7: *Server schickt TLS Handshake Done und bestätigt Paket 0 (Packet Type 0-RTT)*

Quick UDP Internet Connections (QUIC)

Zusammenfassung und Ausblick

- Diese Einführung vermittelt nur einen ersten Einblick in die Komplexität von QUIC
- Die Wireshark-Analyse wird durch folgende QUIC Eigenschaften wesentlich schwieriger:
  - Starke Verschlüsselung mit TLS 1.3
  - Zahlreiche unterschiedliche Header und Frame-Formate
  - Die meisten Felder im Header haben variable Grössen
  - Mehrere und unterschiedliche Pakete im gleichen Frame sind möglich (vergleichbar mit SCTP)
  - Drei verschiedene Paketgruppen mit eigenem Paketzähler und eigenen Bestätigungen
  - Mehrere, parallele Streams im selben Frame, begrenzt nur durch die max. Ethernet Framelänge
  - Filtern auf einen Stream wird dadurch verunmöglicht, da Wireshark nur ganze Frames filtern kann
  - Und viele Eigenschaften mehr…
- Der QUIC Rollout betrifft nicht nur Server und Clients, auch Firewalls, Loadbalancer etc.
- Implementierungsprobleme in vielen Komponenten basierend auf TCP sind zu erwarten
- Umso wichtiger werden QUIC-Kenntnisse und Analysemöglichkeiten mit Wireshark
- Die Wireshark-Decodierung ist noch nicht komplett, und ein Expert-System fehlt (noch)

- Aktuell standen mir noch keine Tracefiles mit Übertragungsfehlern zur Verfügung
- Ich werde QUIC-Updates und Fehlersituationen in weiteren Newslettern behandeln

⇨ **Feedback und QUIC-Tracefiles sind von euch Sniffern gerne willkommen** ⇦

Quick UDP Internet Connections (QUIC)

Persönlicher Kommentar

- Eines der häufigsten Argumente von QUIC vs. TCP ist das **Head-of-line (HOL) Blocking** von TCP. Dies ist die Eigenschaft beim TCP-Empfänger, die Auslieferung weiterer Daten zu stoppen, wenn ein TCP-Paket verloren ging (ein Feature, kein Bug). Das heisst jedoch nicht, dass die Übertragung gestoppt wird, da der Empfangs-Buffer (Windows-Size) weiterhin gefüllt werden kann. Wird das fehlende Paket nachgeliefert, bevor der Buffer voll ist (geschieht meistens, wie Wireshark-Analysen zeigen), entsteht keine Verzögerung der Übertragung.
- QUIC löst dieses Problem mit **verschiedenen Streams** über eine UDP-Verbindung, was vergleichbar ist mit verschiedenen parallelen TCP-Sessions. Da die meisten Browser sowieso per Default **mehrere TCP-Sessions** (auch Streams genannt) öffnen, erachte ich den Durchsatzgewinn von QUIC nicht als erheblich.

- Ein weiteres Argument für QUIC wird mit **weniger Handshake-Zyklen** beim Verbindungaufbau angepriesen. Wie mein Beispiel zeigt, verkürzt sich der Aufbau tatsächlich um ca. 20ms (kein TCP 3-way Handshake). Ob dieser einmalige Gewinn beim Aufbau für den Benutzer spürbar ist, wage ich zu bezweifeln (auch nicht bei mehreren parallelen TCP-Sessions, diese werden gleichzeitig und nicht sequenziell gestartet).
- Ein weiterer, oft gehörter Vorteil ist der **schnellere Aufbau der TLS 1.3** Verschlüsselung (1-RTT oder 0-RTT). Dieses Argument greift gar nicht, da **TLS 1.3 auch über TCP** verwendet werden kann und bereits wird.

- Für mich die beeindruckendste QUIC Neuerung ist die Möglichkeit, **während laufender Übertragung fliegend die IP-Adresse und den UDP-Port zu wechseln**. Sicher ein Argument bei der zunehmenden Mobilität, aber gleichzeitig graut mir davor, so ein Tracefile analysieren zu müssen, um einen Fehler einzugrenzen.
- Und was ist mit den unzähligen Netzgeräten wie **Firewalls, Load balancern, Traffic shapern** (auch viele Router mit Tuning- und Priorisierungsoptionen) die den **TCP-Header** für ihre Funktionen verwenden?

⇨ Über allem sehe ich noch nicht, wo und wie QUIC gegenüber TCP **markante** Durchsatzsteigerungen erreichen soll, und bleibe deshalb weiterhin **TCP-Fan.**

10 Gbit/s Aufzeichnung

Hardware Produkte

- Von Dualcomm Technology Inc ist neu auch ein **10 Gbit/s TAP** verfügbar.
- Der kostengünstige **ETAP-XG** ist ein universell einsetzbarer, mobiler Monitoring- TAP. Er kann mit SFPs (Small Form-Factor Pluggable) für Kupfer und Glas bestückt werden und unterstützt die Ethernet-Datenraten 100M/1G/2.5G/5G/10G.
- Der nur etwa zigarettenschachtelgrosse TAP funktioniert im sogenannten Non-Aggregation-Mode und bietet je einen Monitor Port für Transmit (TX) und Receive (RX) (Full-Duplex).

- Der ETAP-XG kann/braucht nicht konfiguriert zu werden, er wird einfach in eine bestehende Kupfer- oder Glas-Strecke eingefügt.
- Er ist für die Datenverbindung völlig transparent, hat keine MAC-Adresse und funktioniert deshalb auch bei Switch-Port-Security.
- Die Frames werden mit allen Headern und TAGs (VLAN, VXLAN, MPLS usw.) auf die beiden Monitor-Ports kopiert.
- Da der TAP zwei separate Datenströme (TX/RX) liefert, muss entweder gleichzeitig mit zwei Aufzeichnungsgeräten oder einem Aufzeichnungsgerät mit zwei Ethernet-Adaptern aufgezeichnet werden.

- Siehe unter Tipps, Tricks und Traces, wie mit Wireshark zwei Files zusammengefügt werden.

- Der **ETAP-XG** ist ab sofort auf unserer Webseite zum Preis von CHF 699.- bestellbar.

10 Gbit/s Aufzeichnung

Hardware Produkte

- Ein Notebook mit Wireshark ist für Aufzeichnungen über 1Gbit/s überfordert und verliert Frames. Mit entsprechenden Capture-Filtern kann diese Situation verhindert werden.
- Bei 10 Gbit/s ist die Aufzeichnung mit einem Notebook praktisch nicht mehr möglich, dazu wird zusätzliche Hardware benötigt.
- Marktführer in diesem Bereich ist die Firma ProfiTap mit ihren ProfiShark Network TAPs. Diese erlauben das Zusammenführen und Zwischenspeichern von RX und TX Streams bis 10 Gbit/s.

- Der ProfiShark liefert die Daten über die USB 3.0 Schnittstelle mit bis maximal 5 Gbit/s an einen Notebook mit Wireshark
- Für Datenraten über 5 Gbit/s kann Packet Slicing und Capture Filtering auf dem ProfiShark aktiviert werden.
- Schweizer Vertreter von ProfiTap ist die Firma ISATEL Electronic AG

ISATEL offeriert meinen Newsletter Abonnenten bis Ende 2021 einen exklusiven Rabatt von 5% auf alle ProfiShark Produkte.

Erwähnen sie bei der Bestellung als Referenz Leutert NetServices

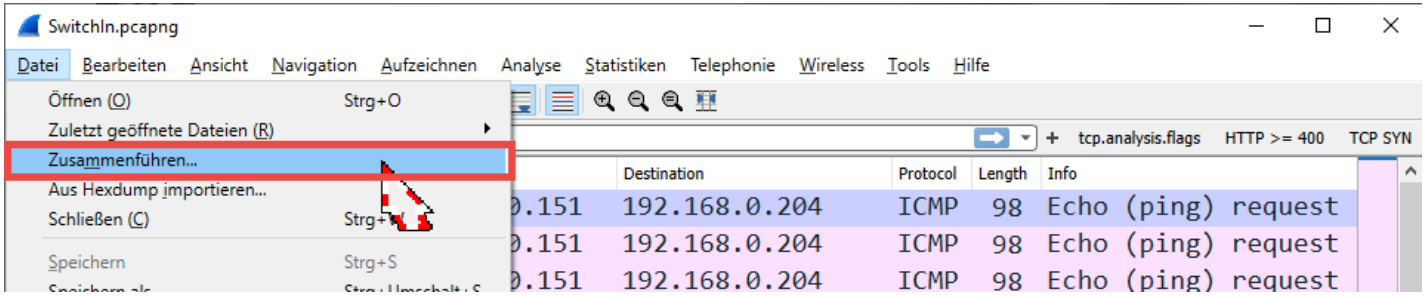
(Nein, ich verdiene da nicht mit, ich empfehle dieses Produkt aus Überzeugung)

Tipps, Tricks & Traces

Wireshark Tracefiles zusammenführen

- Bei der Fehlereingrenzung ist es oft sinnvoll, an zwei (oder mehr) Messpunkten Daten aufzuzeichnen, z.B. um die Quelle von Paketverlusten einzugrenzen.
- Sind die Messpunkte distanzmässig nahe beieinander, ist die einfachste Methode, das Aufzeichnungsgerät mit einem zweiten Ethernet Adapter zu bestücken und gleichzeitig auf beiden die Aufzeichnung starten.
- Damit ist gewährt, dass die Zeitstempel beider Quellen synchronisiert sind.
- Trotzdem ist eine Nachbearbeitung der Aufzeichnung notwendig, da die Frames durch die Buffer auf den Ethernet-Adaptern in der Reihenfolge vertauscht dargestellt werden.
- Das genaue Vorgehen ist beschrieben in unserem Newsletter https://www.netsniffing.ch/download/Wireshark_Newsletter_2012_09.pdf

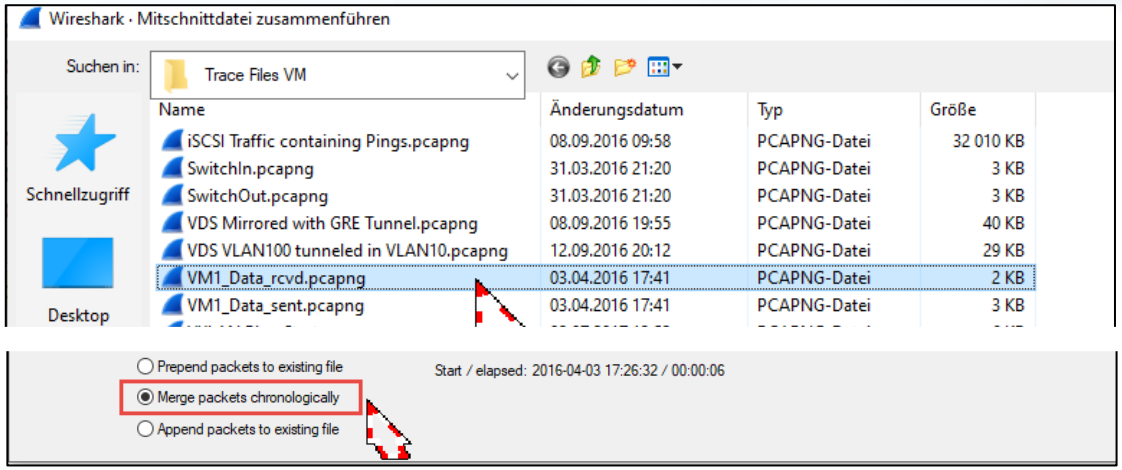
- Sind die Messpunkte distanziert, müssen zwei Aufzeichnungsgeräte eingesetzt werden. Die Analyse wird vereinfacht, wenn die Tracefiles zusammengefügt (merged) werden.
- Nach dem Öffnen eines Files mit Wireshark ist die Option Zusammenführen aktiviert



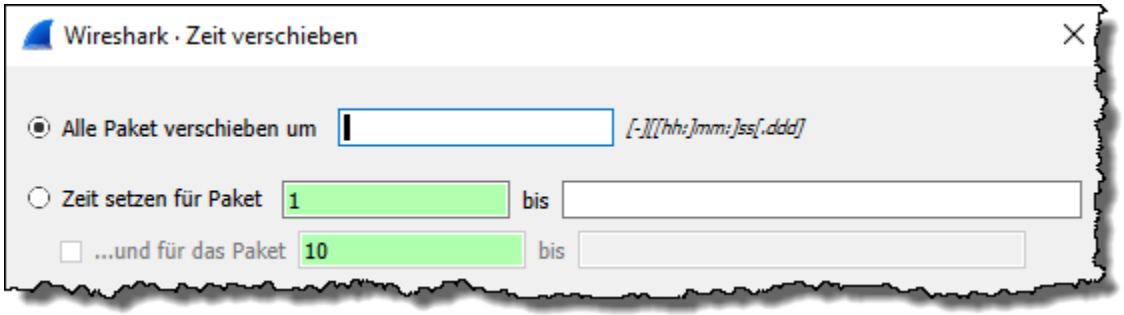
Tipps, Tricks & Traces

Wireshark Tracefiles zusammenführen

- Im nächsten Schritt das zweite File und die Option Merge packets chronologically wählen



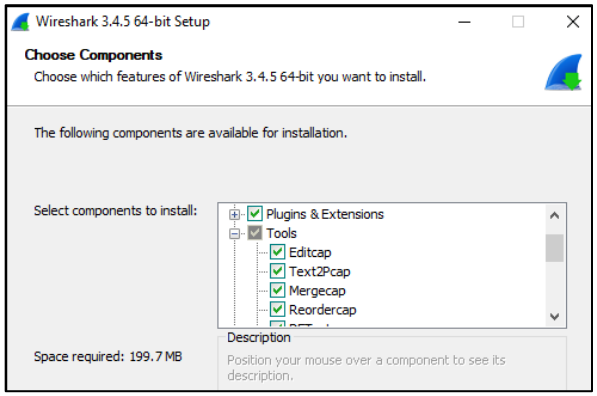
- In den meisten Fällen ist es unwahrscheinlich, dass die Zeitbasis der beiden Aufzeichnungsgeräte auf die Millisekunde genau synchronisiert ist. Das heisst die Reihenfolge der Pakete kann falsch sein. Leicht zu erkennen, wenn z.B. eine DNS-Antwort vor der Anfrage erscheint ☺
- Dies kann unter → Bearbeiten → Zeitverschieben **vor** dem Zusammenführen korrigiert werden

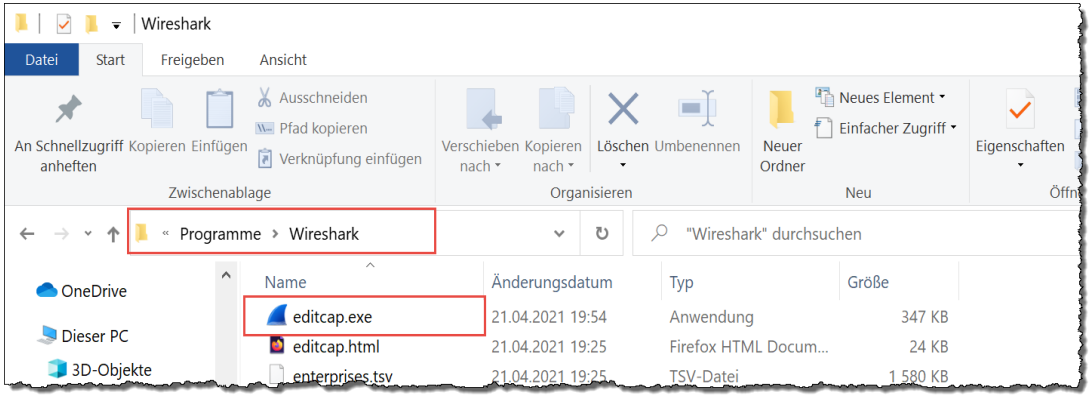


Tipps, Tricks & Traces

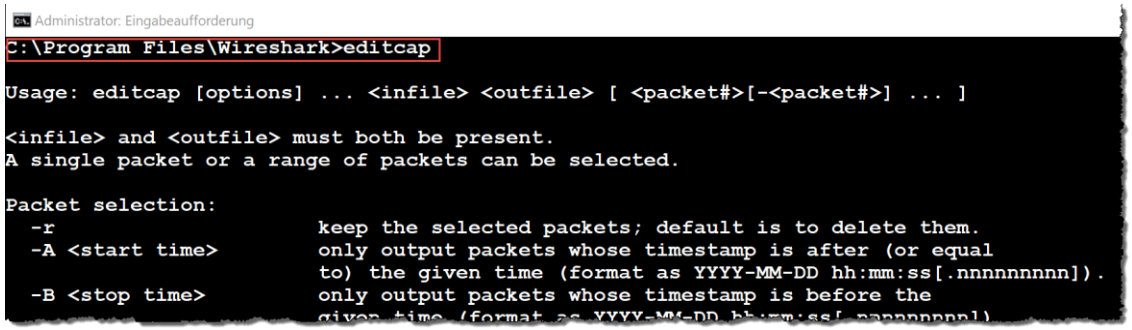
Pakete editieren mit Editcap

- Editcap.exe ist ein mächtiges Werkzeug zum Bearbeiten von aufgezeichneten Tracefiles. Es kann grosse Files in kleinere aufteilen, Pakete kürzen, Felder aus Paketen entfernen usw.
- Falls sie beim Installieren von Wireshark die Option Tools nicht geändert haben, wird Editcap per Default im selben Ordner wie Wireshark installiert sein.





- Eingabe von editcap im cmd Fenster zeigt die Syntax und die zahlreichen Optionen



Tipps, Tricks & Traces

Pakete editieren mit Editcap

- Editcap unterstützt .pcap und .pcapng Files. Wenn sich die Files nicht im selben Order wie editcap befinden, muss der Pfad zum File angegeben werden. Wichtig ist auch, dass die Filenamen keine Leerstellen enthalten (File allenfalls vorher umbenennen)
- Detaillierte Infos unter https://www.wireshark.org/docs/man-pages/editcap.html

```
editcap -c 100000 capture_in.pcapng capture_out.pcapng
```

-c xxx teilt ein bestehendes File auf in Zielfiles mit der angegeben Anzahl Pakete

```
editcap -s 200 capture_in.pcapng capture_out.pcapng
```

-s xxx kürzt alle Pakete auf die angegebe Länge in Bytes und speichert diese im Zielfile

```
editcap -C 38 -L capture_GRE.pcapng capture_No_GRE.pcapng
```

-C xxx entfernt die angegebene Anzahl Bytes am Anfang jedes Paketes & passt die Länge an

```
editcap -C 12:4 -L capture_vlan.pcap capture_no_vlan.pcap
```

-C 12:4 -L entfernt die Anzahl Bytes an einer bestimmten Position & passt die Paketlänge neu an

```
editcap -d capture_in.pcapng capture_out.pcapng
```

-d entfernt Duplikate von Paketen aufgrund übereinstimmender Länge und des MD5 Wertes

```
editcap -F snoop capture_in.pcap capture_out.snoop
```

-F xxx konvertiert das Format des Input Files in das gewählte Output Format

Unsere Wireshark-Protokoll-Kurse & andere Events

- **TCP/IP Analyse mit Wireshark** (neu auch mit QUIC)
  14. - 16. Juni 2021, HSR Hochschule für Technik Rapperswil → Zur Anmeldung bei HSR
- **WLAN Netzwerkanalyse mit Wireshark**, WaveXpert und WiSpy
  13. – 14. Dez. 2021, HSR Hochschule für Technik Rapperswil → Zur Anmeldung bei HSR
- **VoIP Analyse mit Wireshark,** SIP und RTP
  18. Oktober 2021, HSR Hochschule für Technik Rapperswil → Zur Anmeldung bei HSR

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

- **Einführung Netzwerkanalyse, Wireshark Tipps & Tricks, TCP/IP, WLAN, VoIP und IPv6**

Das **SharkFest`21** findet einmal mehr nur virtuell statt.
Dafür ist die Teilnahme kostenlos.
Ich werde wiederum einige Sessions präsentieren.
Anmeldung: https://sharkfesteurope.wireshark.org/index

Unser Newsletter Archiv finden sie unter: https://www.netsniffing.ch/de/wireshark-infos/newsletter

Es würde uns freuen, Sie in einem unserer Kurse begrüssen zu können.

Have fun and enjoy sniffing, Rolf Leutert