**Die ISO/OSI-Schichtenarchitektur**

**ISO/OSI Modell:**

|  |  |
| --- | --- |
| Bitübertragungsschicht | |
| Schicht 1 Physical | Maßnahmen und Verfahren zur Übertragung von Bitfolgen |
| Die Bitübertragungsschicht definiert die elektrische, mechanische und funktionale Schnittstelle zum Übertragungsmedium. Die Protokolle dieser Schicht unterscheiden sich nur nach dem eingesetzten Übertragungsmedium und -verfahren. Das Übertragungsmedium ist jedoch kein Bestandteil der Schicht 1. |
| Sicherungsschicht | |
| Schicht 2 Data Link | Logische Verbindungen mit Datenpaketen und elementare Fehlererkennungsmechanismen |
| Die Sicherungsschicht sorgt für eine zuverlässige und funktionierende Verbindung zwischen Endgerät und Übertragungsmedium. Zur Vermeidung von Übertragungsfehlern und Datenverlust enthält diese Schicht Funktionen zur Fehlererkennung, Fehlerbehebung und Datenflusskontrolle. Auf dieser Schicht findet auch die physikalische Adressierung von Datenpaketen statt. |
| Vermittlungsschicht | |
| Schicht 3 Network | Routing und Datenflusskontrolle |
| Die Vermittlungsschicht steuert die zeitliche und logische getrennte Kommunikation zwischen den Endgeräten, unabhängig vom Übertragungsmedium und der Topologie. Auf dieser Schicht erfolgt erstmals die logische Adressierung der Endgeräte. Die Adressierung ist eng mit dem Routing (Wegfindung vom Sender zum Empfänger) verbunden. |
| Transportschicht | |
| Schicht 4 Transport | Logische Ende-zu-Ende-Verbindungen |
| Die Transportschicht ist das Bindeglied zwischen den transportorientierten und anwendungsorientierten Schichten. Hier werden die Datenpakete einer Anwendung zugeordnet. |
| Kommunikationsschicht | |
| Schicht 5 Session | Prozeß-zu-Prozeß-Verbindungen |
| Die Kommunikationsschicht organisiert die Verbindungen zwischen den Endsystemen. Dazu sind Steuerungs- und Kontrollmechanismen für die Verbindung und dem Datenaustausch implementiert. |
| Darstellungsschicht | |
| Schicht 6 Presentation | Ausgabe von Daten in Standardformate |
| Die Darstellungsschicht wandelt die Daten in verschiedene Codecs und Formate. Hier werden die Daten zu oder von der Anwendungsschicht in ein geeignetes Format umgewandelt. |
| Anwendungsschicht | |
| Schicht 7 Application | Dienste, Anwendungen und Netzmanagement |
| Die Anwendungsschicht stellt Funktionen für die Anwendungen zur Verfügung. Diese Schicht stellt die Verbindung zu den unteren Schichten her. Auf dieser Ebene findet auch die Dateneingabe und -ausgabe statt. |

**OSI comparision with TCP/IP Protocol Stack**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| OSI# | OSI Layer Name | TCP/IP# | TCP/IP Layer Name |  |
| ⁷ | Application | 4 | Application |
| 6 | Presentation |
| 5 | Session |
| 4 | Transport | 3 | Transport |
| 3 | Network | 2 | Internet |
| 2 | Data Link | 1 | Network Access |
| 1 | Physical |

Es hat den Nachteil, dass für jedes Gerät, dass ein anderes Format benötigt wird, 3

Schichten betrachtet werden müssen anstatt nur eine.

* 1. **Adressierung von Geräten im lokalen Netzwerk**

1/ das ist ein Gerät von Apple . (Herstellerkennung (OUI), 24 Bit)

2/

* Es wird so viel einfacher zu spionieren ( As a result of users being trackable by their devices' MAC addresses, [Apple Inc.](https://en.wikipedia.org/wiki/Apple_Inc.) has started using random MAC addresses in their [iOS](https://en.wikipedia.org/wiki/IOS) line of devices while scanning for networks.)

-Weil die MAC-Adressen unabhängig von der Topologie oder dem Standort des

Rechners vergeben wird. Man kann nicht sagen an welchem Ort der Rechner ist, der

sendet oder an welchem Ort der Empfänger steht. Es müssten endlos lange Tabellen

darüber geführt werden, wo was steht. Ein anderer Grund wäre z. B. hätte ein Server

diese Adresse und er fällt aus, dann müssten viele Änderungen vorgenommen

werden.

* 1. Aufbau und Struktur eines Computernetzwerks

2/Ein Hub arbeitet (genauso wie ein Repeater) ausschließlich auf Ebene 1 des ISO/OSI-Referenzmodells (Bitübertragungsschicht) . Das Signal eines Netzteilnehmers wird nicht analysiert, sondern nur die übertragene Bit- bzw. Symbolebene wird regeneriert.

Grundsätzlich sorgt ein Switch innerhalb eines Segments (Broadcast-Domain) dafür, dass die Datenpakete, sogenannte „Frames“, an ihr Ziel kommen. Es arbeitet in 2te OSI Schicht (Datalink Layer ). aber heute die hoch-complex und intelligent switchs , mit ihren neuen service ( IP routing services, Quality of Service ( QoS), firewall functions, deep packet inspection...) , konnen bis 7the Schicht von OSI Model arbeiten.

3/

Bei Einsatz eines Hubs im Netz wird durch die Verkabelung im physikalischen Sinne eine Stern-Topologie realisiert. Der logische Aufbau ähnelt dem einer Bus-Topologie, weil jede gesendete Information alle Teilnehmer erreicht. Alle Teilnehmer in einem Netzwerk, die an einen Hub angeschlossen sind, befinden sich in derselben Kollisionsdomän.

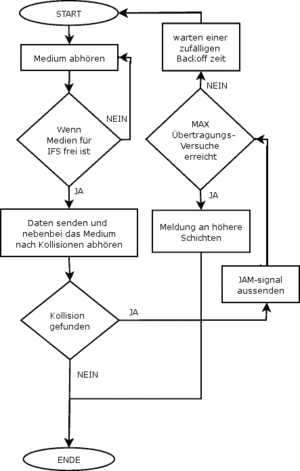
Die RTDT (Round-Trip-Delay-Time )ist die Zeit, die ein Datenframe benötigt, um vom einen Ende des Netzes zum weitestentfernten anderen Ende des Netzes zu gelangen - und wieder zurück. Wird das Netz zu groß, also die RTDT zu hoch, werden Kollisionen häufiger, unerkannte Kollisionen möglich und der gesamte Netzverkehr beeinträchtigt.

Ein Zugriffsverfahren wäre sehr nötig, da man damit

eventuell Kollisionen verhindern könnte.

4/ ein switch trennt Kollisionsdomäne , jedes Port repräsentiert eine eigene KollisionsDomän. So sind Kollisionen unmoglich.

* 1. **Der Zugriffsalgorithmus des Ethernet-Protokolls**

****

# Carrier Sense Multiple Access/Collision Detection

Jeder Rechner, der ein Paket versenden will, „hört“ das Übertragungsmedium ab, ob

es frei ist. Sollte das Übertragungsmedium frei sein, wird mit dem senden begonnen.

Während des Sendens wird auch mitgehört, ob es zu einer Kollision kommt. Sollte es

zu einer Kollision kommen, wird die Übertragung für eine bestimmte Zeit

abgebrochen und es wird dann wieder versucht zu senden.

# 4. Das Internet Protokoll

4.1 Richten Sie das Testbed entsprechend der Abbildung 7 ein. Starten Sie alle drei PCs neu, damit das Testnetz initial konfiguriert wird. Ändern Sie die IP-Adressen, konfigurieren Sie die Routen und Default-Gateways, wie sie in Abbildung 7 beschriftet sind.

**Netem**

Sudo bash –c ́echo 1 > /proc/sys/net/ipv4/ip\_forward`

Sudo ifconfig eth0 192.168.1.1 netmask 255.255.255.0

Sudo ifconfig eth1 192.168.2.1 netmask 255.255.255.0

 Sudo route add –net 192.168.1.0 netmask 255.255.255.0 gw 192.168.1.1

Sudo route add –net 192.168.2.0 netmask 255.255.255.0 gw 192.168.2.1

**hwp@hwp3**

sudo ifconfig eth0 192.168.2.2 netmask 255.255.255.0

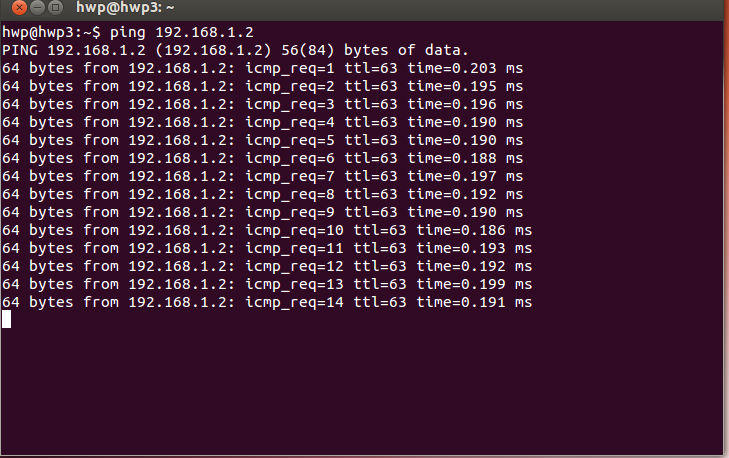
sudo route add default gw 192.168.2.1

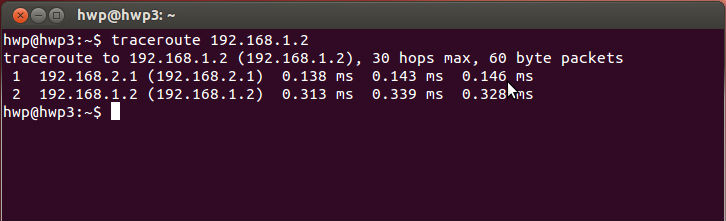
**hwp@hwp1**

sudo ifconfig eth0 192.168.1.2 netmask 255.255.255.0

sudo route add default gw 192.168.1.1

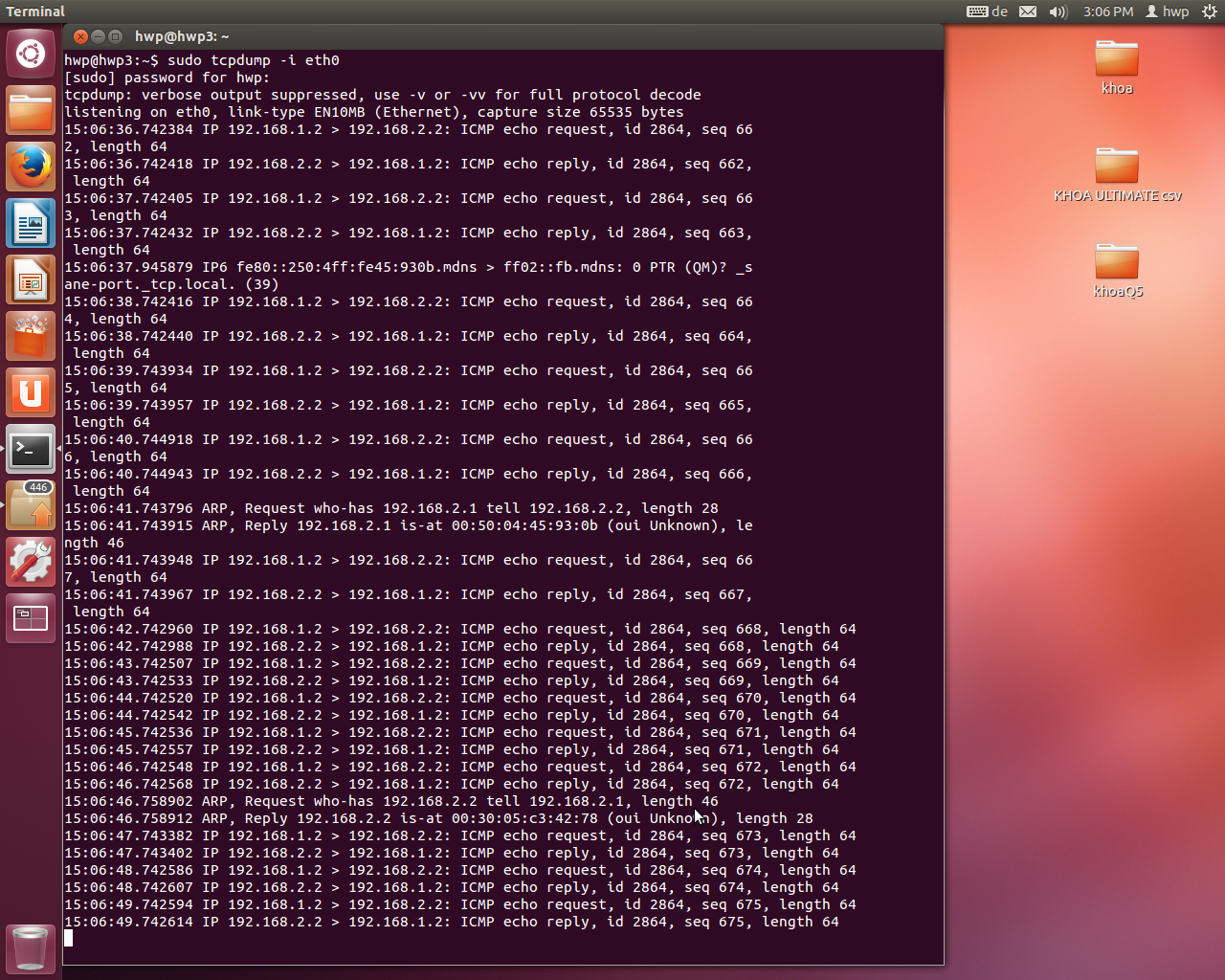
4.2 Überprüfen Sie ihre Konfiguration mit Hilfe der Tools ‘ping’ und ‘traceroute’. Was beobachten Sie?

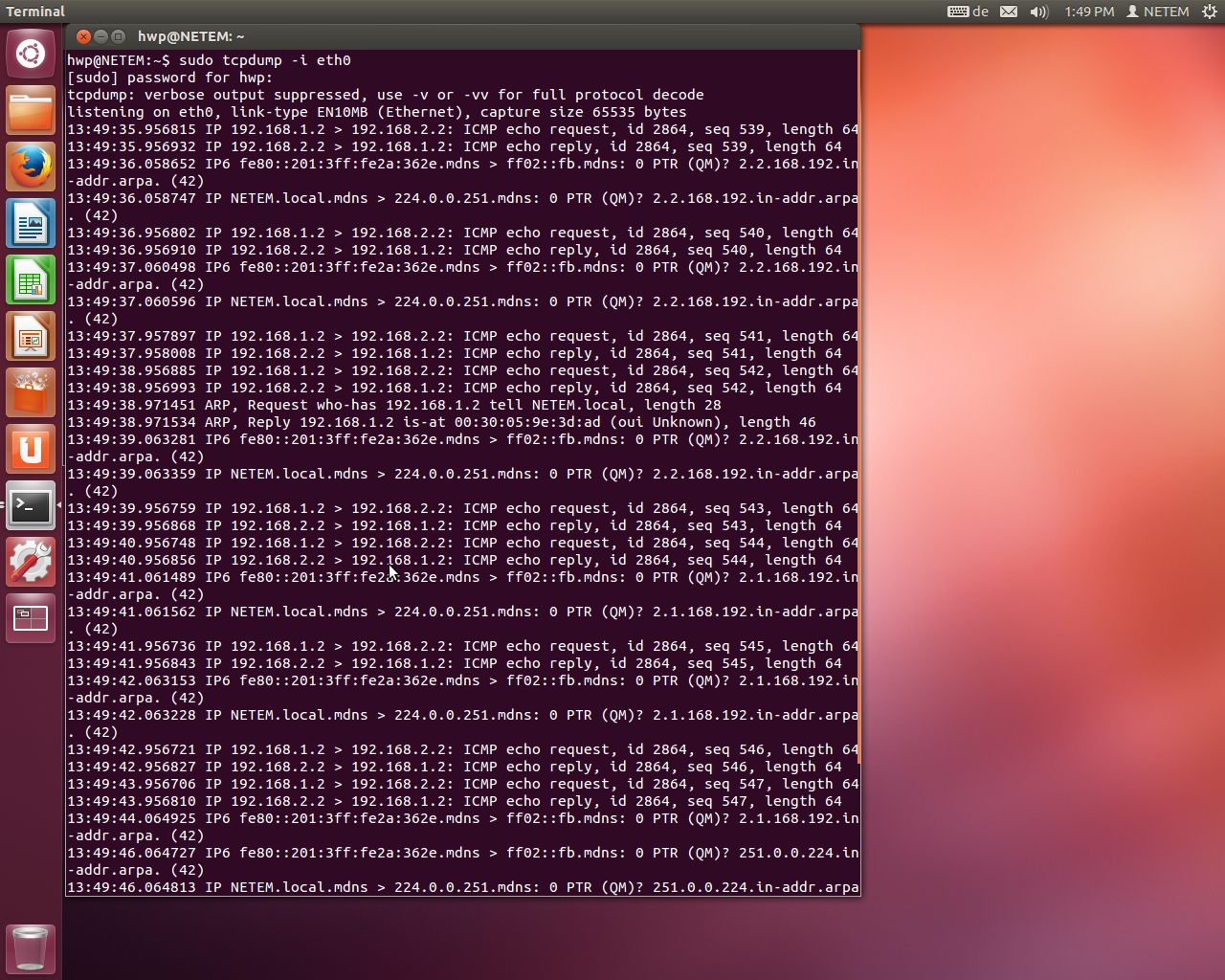




Der „traceroute“-Befehl zeigt, dass sich die beiden Rechner in verschiedenen Netzen befinden.

4.3 Überprüfen Sie den Fluss der ICMP Pakete mit Hilfe des Tools ‘tcpdump’, wenn Sie von Client 1 nach Client 2 ‘pingen’.





# 5. Transportprotokolle im Internet

5.2. Starten sie die Java Client Applikation auf einem der beiden Clients aus dem Testbed von 7 und die Server Applikation dementsprechend auf dem anderen Client.

Das Erstellen und Ausführung von Server-Anwendung an Client2 (für unsere Anwendung ist das Server):

hwp@hwp3:~$ javac Server\_TCP.java

hwp@hwp3:~$ java Server\_TCP

Das Erstellen und Ausführung von Client-Anwendung an Client1:

hwp@hwp1:~$ javac Server\_TCP.java

hwp@hwp1:~$ java Server\_TCP

Analog erfolgt das Vorgehen von UDP-Anwendung.

Berechnung von TCP-Anwendung (Client):

hwp@hwp1:~$ 4

hwp@hwp1:~$ 5

hwp@hwp1:~$ 9

Berechnung von UDP-Anwendung (Client):

hwp@hwp1:~$ 4

hwp@hwp1:~$ 5

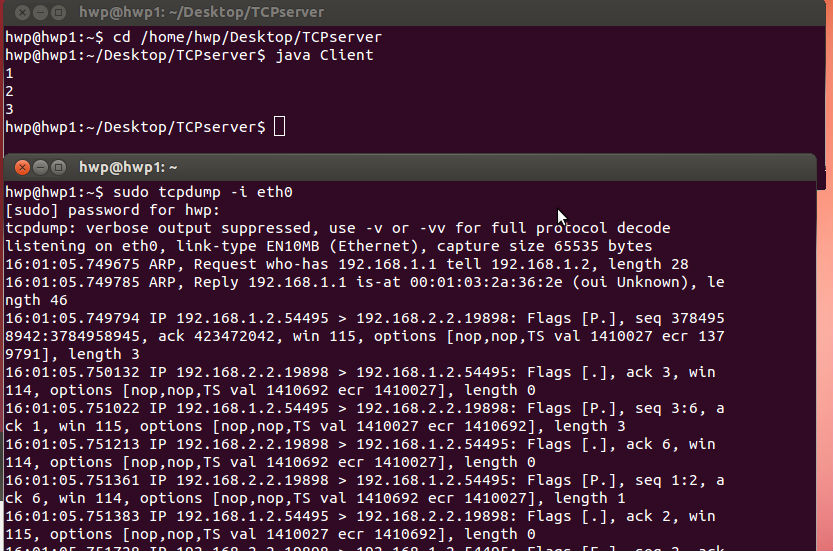
hwp@hwp1:~$ Ergebnis: 9

Berechnung von UDP-Anwendung (Server):

hwp@hwp3:~$ Empfangen: 4 5

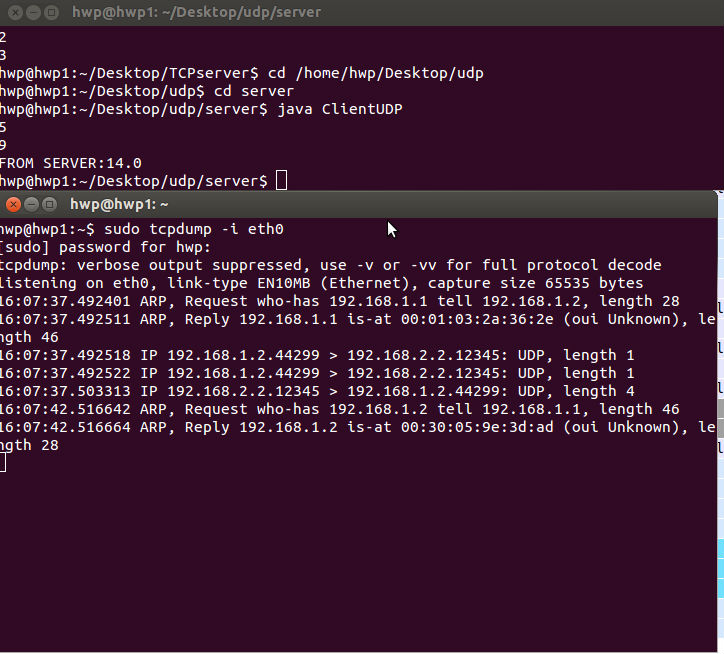
hwp@hwp3:~$ Ergebnis: 9

5.3. Überprüfen Sie die Funktionalität ihrer Anwendung durch Beobachtung des Flusses mit dem Tool ‘tcpdump’.



5.4. Wiederholen Sie das Experiment für eine UDP Server/Client Applikation.

tcpdump für UDP-Anwendung :



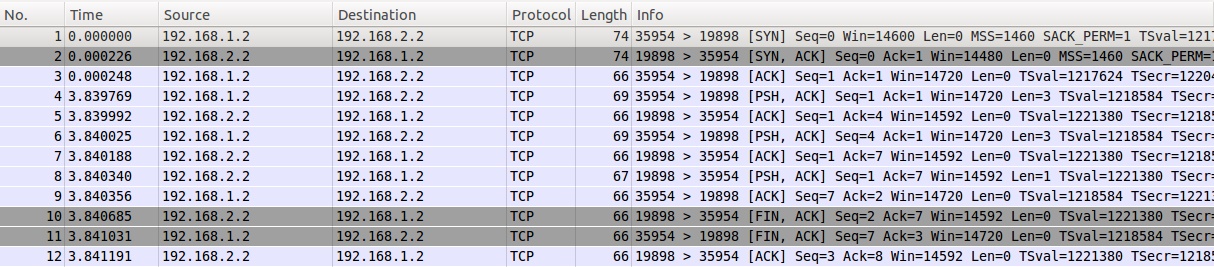
5.5. Vergleichen Sie die Startphasen zu Beginn einer Übertragung bei den beiden Transportprotokollen TCP und UDP. Erstellen Sie hierzu zwei aussagekräftige Screenshots mit dem Tool ‘Wireshark’ und erläutern sie diese in ihrem Bericht. Benutzen Sie hierfür den Modus “Flow Graph”.

Beide Protokolle starten mit der Anfrage, ob der Gateway existiert.

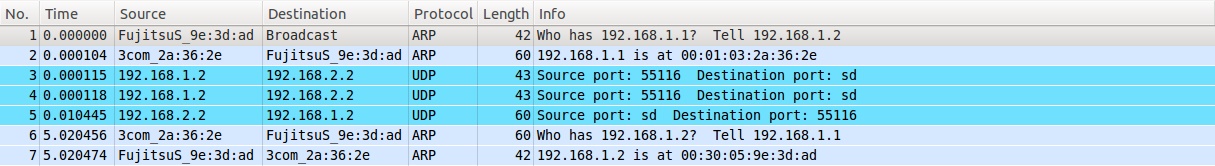
Wireshark berechtigen an Client1-Rechner:

hwp@hwp1:~$ gksudo wireshark

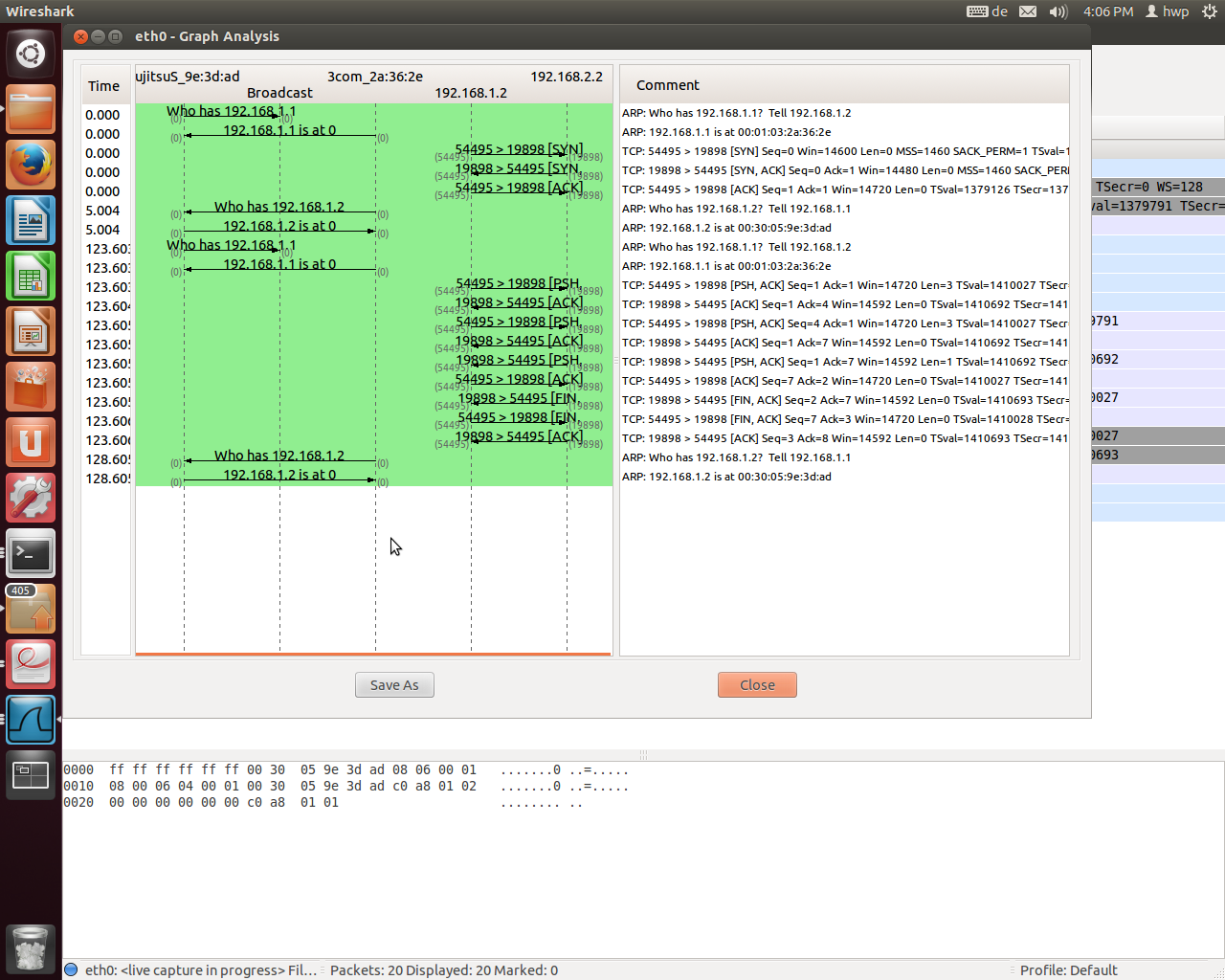
Wireshark-Bericht für TCP-Anwendung:



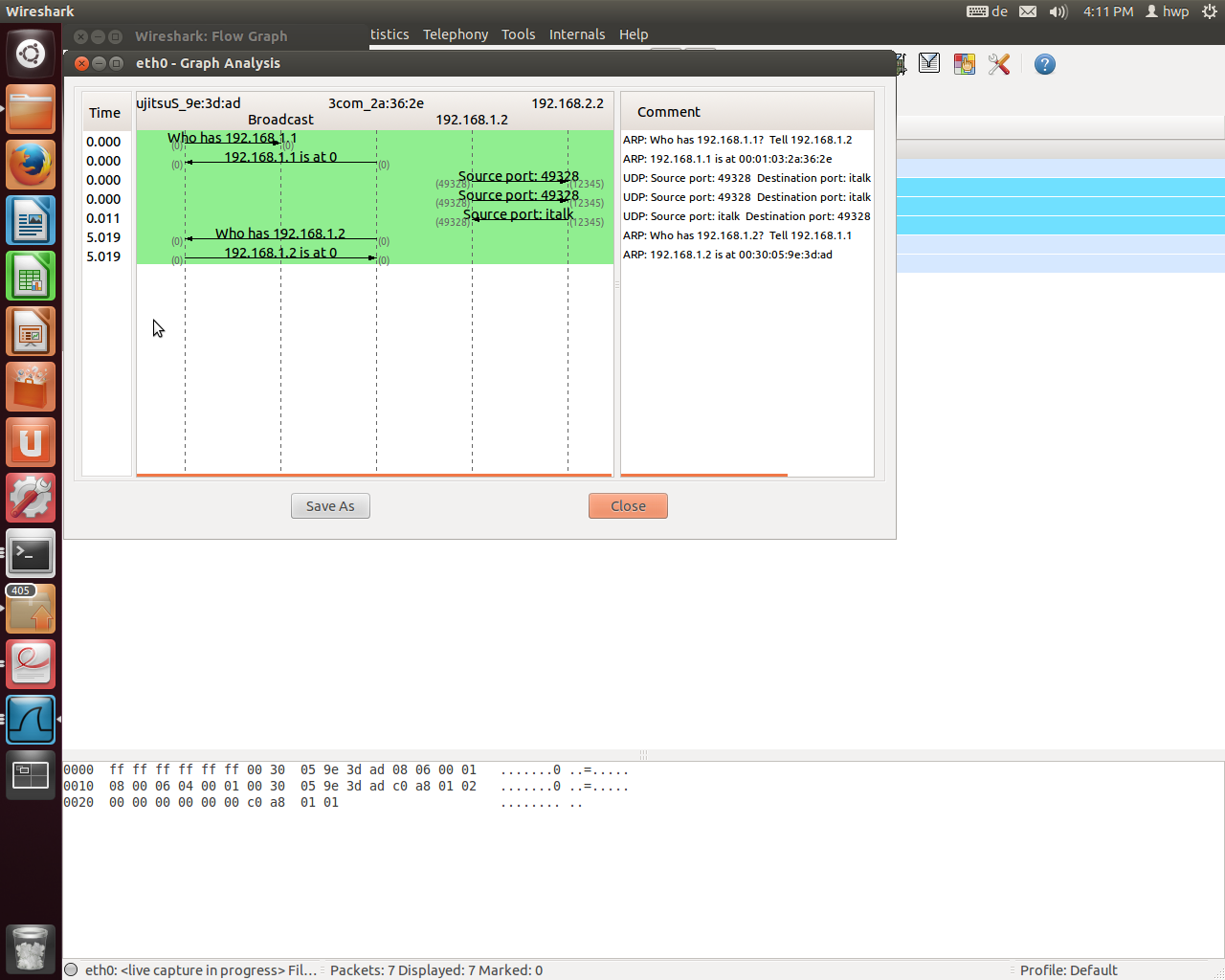
Wireshark-Bericht für UDP-Anwendung:



Flow Graph von TCP-Anwendung:



Flow Graph von UDP-Anwendung:



5.6. Einfluss von Paketverlust: Fügen Sie nun mittels Netem Paketverlust von 10%, 20%, 30% und 40% sowohl für den TCP als auch den UDP Fall hinzu und lassen Sie ihre Applikation mehrfach laufen.Was können sie beobachten? Warum?

Paketverlust-Simulation mithilfe „qdisc“ an NETEM-Rechner:

hwp@NETEM:~$ sudo tc qdisc add dev eth0 root netem loss 10%

hwp@NETEM:~$ sudo tc qdisc change dev eth0 root netem loss 20%

hwp@NETEM:~$ sudo tc qdisc change dev eth0 root netem loss 30%

hwp@NETEM:~$ sudo tc qdisc change dev eth0 root netem loss 40%

Paketverlust:

TCP: Was fällt auf: Verzögerung bei höherer Verlustzahl. Client braucht also bei höherer Verlustrate länger um Paket zu erhalten.

UDP: Bis 40% keine nennenswerte Zeitverzögerung.

Wenn Paket die Endestation nicht erreicht, benehmen sich TCP und UDP Protokolle ganz unterschiedlich. TCP hat eingebaute Funktion für Überprüfen des Versandserfolgs. D.h., dass beim Paketverlust, Sendestation den Paket wieder schickt und versucht die Kommunikation erfolgreich abzuschließen. Das ist beim UDP-Protokoll nicht der Fall. Da muss der Empfänger selber wieder nachfragen, da in UDP kein Mechanismus existiert um Versandserfolg zu überprüfen.

5.7. Einfluss von Paketverzögerung: Konfigurieren Sie in NetEm eine Verzögerung in eine Übertragungsrichtung von jeweils 1000ms und 5000ms zwischen Client 1 und Client 2. Führen Sie das Experiment sowohl für TCP als auch für UDP Verkehr durch. Was lässt sich beobachten? Beschreiben Sie ihre Beobachtungen bezüglich der geringen Verzögerung im Vergleich zu der Verzögerung bei größer Distanz.Welches Transportprotokoll würden Sie für große Distanzen bevorzugt einsetzen?

Paketverzögerung-Simulation an NETEM-Rechenr:

sudo tc qdisc change dev eth0 root netem delay 1000ms

sudo tc qdisc change dev eth0 root netem delay 5000ms

TCP: Client wartet länger auf Bearbeitung, Berechnung braucht also länger bei höherer delay-time.

UDP: Leichtere Verzögerung als bei TCP.

Mit Paketverzögerung kann man leicht zeigen warum UDP Protokoll besser für Fernkommunikation ist. Genau der Mechanismus für Wiederaufforderung beim TCP verursacht Probleme, weil es nicht klar ist ob es um Verzögerung oder um Paketverlust geht. Bei der längeren Verzögerung schickt TCP einen Paket mehrmals, was zur Verdopplung von empfangenen Dateien führt und Verursacht unnötiger Aufwand bei der Anfrage.

UDP vermeidet solche Probleme, da erfolgreicher Empfang nicht erfordert ist.