FORSCHUNGSZENTRUM JÜLICH GmbH
Zentralinstitut für Angewandte Mathematik
D-52425 Jülich, Tel. (02461) 61-6402

Interner Bericht

Alternative Transportprotokolle und der Einfluss des Netzes: Low-Level-Benchmarks im X-WIN und VIOLA

Franz Petri, Thomas Eickermann

FZJ-ZAM-IB-2007-10

August 2007

(letzte Änderung: 14.09.2007)

Bericht erstellt für das D-Grid Integrationsprojekt (DGI)

D-Grid Integrationsprojekt (DGI)

Fachgebiet 3-3 – Alternative Transportprotokolle

Alternative Transportprotokolle und der Einfluss des Netzes: Low-Level-Benchmarks im X-WIN und VIOLA

Version 1.0, August 2007

Autoren

Franz Petri (ZAM, Forschungszentrum Jülich)

Thomas Eickermann (ZAM, Forschungszentrum Jülich)

Das diesem Bericht zugrundeliegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung und Forschung unter dem Förderkennzeichen 01AK800B gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt bei den Autoren.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Um einen reibungslosen Betrieb im Internet zu garantieren, werden eine Großzahl von Komponenten eingesetzt. Diese werden oft von verschiedenen Providern betrieben und von Datenströmen vieler Benutzern gleichzeitig durchlaufen. Die zu Verfügung stehende Bandbreite wird geteilt. Pakete passieren auf dem Weg vom Sender zum Empfänger die Queues zahlreicher aktiver Komponenten wie Switches, Router und Firewalls, wobei es je nach Auslastung der jeweiligen Komponente zu starken Schwankungen der Paketlaufzeiten oder auch zu Paketverlust kommen kann, was wiederum Auswirkung auf das Verhalten der Transportschicht hat.Im Gegensatz zu den heute aktiven Produktionsnetzen wie dem X-WIN bietet ein Weitverkehrsnetz wie das VIOLA-Testbed latenzkritischen Anwendungen quasi exklusive Bandbreitenbereitstellung. Durch dedizierte optische Verbindungstechnik und Authentifizierungs- und Zugangsmechanismen wird die Anzahl der aktiven Komponenten wie Router und Firewalls – und somit die Round Trip Time (RTT) innerhalb des Netzwerkes – auf ein Minimum reduziert.

Aus Sicht der Netzwerkapplikation müssen zwei Arten von Latenz unterschieden werden:

- Round Trip Time: Tauschen Anwendungen kleine, nur wenige Byte große Nachrichten aus, die in ein einzelnes Ethernet-Paket passen, so sind sie der reinen Paketlaufzeit bzw. Round Trip Time unterworfen. Diese Art von Latenz ist maßgeblich durch die physikalische Länge der Leitung sowie die Anzahl der aktiven Komponenten zwischen Sender und Empfänger bestimmt.
- Nachrichtenlaufzeit: Werden – beispielsweise mehrere Mbyte - große Nachrichten übertragen, so wird die Laufzeit einer solchen Nachricht durch die Durchsatzrate bestimmt, die über das Netzwerk möglich ist. Große Nachrichten benötigen bei hoher Durchsatzrate weniger Zeit vom Sender zum Empfänger als bei niedriger. Insbesondere bei Datenübertragung mit TCP hängt diese Art von Latenz entscheidend von der Auslastung der Netz-Infrastruktur ab. Aufgrund hoher Last entstehende Paketverluste vermindern den Durchsatz von TCP-basierten Anwendungen erheblich. Eine hohe RTT wirkt hierbei zusätzlich wie ein Hebel, weil sie verhindert, dass TCP eine Reduktion der Durchsatzrate schnell wieder kompensieren kann.

Im Rahmen des DGI FG3.3 wurden zahlreiche Messung innerhalb und außerhalb von Netzwerk-Laborumgebungen durchgeführt[1]. Hierbei wurde deutlich, dass sich insbesondere die nicht-deterministischen Eigenschaften eines realen Produktionsnetzes in einer Laborumgebung nur schwer abbilden lassen. Diese „Real-Life“-Effekte variieren mit der Anzahl der aktiven Komponenten in einer Verbindung und der Anzahl der Benutzer und ihres Querverkehrs auf diesen Komponenten. Dem vorliegenden Bericht liegen Messungen zugrunde, die sowohl in Produktionsnetzen als auch im dedizierten VIOLA-Testbed mit verschiedenen Transportprotokollvarianten wie z.B. TCP BIC oder UDT durchgeführt wurden.

Neben quantitativen Messungen mit reinen Low-Level-TCP-Benchmarks wie iperf, tcppp oder udtpp wurde untersucht, wie sich die o.g. Arten von Latenzen, beispielsweise in KoDaVis oder IMB, darstellen.

1 Siehe hierzu:

http://dgi.d-grid.de/fileadmin/user_upload/documents/DGI-FG3-3/FG3-3_Analyse-TCP.pdf

http://dgi.d-grid.de/fileadmin/user_upload/documents/DGI-FG3-3/FG3-3_Laufzeittests_VISIT_v1_0.pdf

http://dgi.d-grid.de/fileadmin/user_upload/documents/DGI-FG3-3/FG3-3_Messungen_VIOLA_v1_0.pdf

2 Eingesetzte Software

2.1 iperf

Iperf, das im Rahmen des National Laboratory for Applied Network Research (NLANR)-Projekts entwickelte und unter http://dast.nlanr.net/Projects/Iperf2.0/iperf-2.0.2.tar.gz bereitgestellte Werkzeug zur Messung von TCP- und UDP-Durchsatz, wurde in der Version 2.0.2 benutzt.

2.2 tcppp

Um Nachrichtenlaufzeiten in Abhängikeit von verschiedenen Nachrichtengrößen präzise messen zu können, wurde das Ping-Pong-Programm tcppp benutzt, das Teil der vom Forschungszentrum Jülich entwickelten Visit-Suite (http://www.fz-juelich.de/zam/visit/) ist. Dieses Benchmark-Tool wird sowohl als Server und als auch als Client gestartet; der Client überträgt eine Nachricht frei konfigurierbarer Größe an den Server („Ping“); nachdem dieser die Nachricht komplett empfangen hat, sendet er sie an den Client zurück („Pong“). Der Client misst die Zeit zwischen dem Senden des ersten Bytes und dem vollständigen Empfang des Server-“Pongs“.

2.3 udtpp

Im Rahmen der Arbeiten für DGI FG3.3 wurde eine um das UDP-basierte Transportprotokoll UDT (http://udt.sourceforge.net/) erweiterte Version der tcppp-Ping-Pong-Software geschaffen, die analog zu tcppp funktioniert, jedoch UDT anstatt TCP als Transportprotokoll verwendet. UDT ist eine vom National Center for Data Mining der University of Illinois Chicago entwickelte Transportprotokollvariante, die TCP-Mechanismen wie Staukontrolle und Zuverlässigkeit im Userspace implementiert.

2.4 MPI

2.4.1 metaMPICH

Ein mögliche Netzwerksituation ist aus Sicht einer Hochleistungsumgebung der Austausch von Nachrichten durch MPI. Im vorliegenden Falle wurde die metaMPICH-Bibliothek benutzt (http://www.lfbs.rwth-aachen.de/~martin/MetaMPICH/); metaMPICH ist Teil der mp-mpich-Distribution, eine Erweiterung der MPI-Implentierung MPICH um netzwerkübergreifende Kommunikations-, insbesondere TCP-Devices. Die metaMPICH-Bibliothek wurde als Teil der mp-mpich-Distribution Version 1.3.0 installiert.

2.4.2 IMB

Um die tatsächliche MPI-Performanz der metaMPICH-Bibliothek zu messen, kam der Intel MPI Benchmark (IMB) in Version 2.3 zum Einsatz (http://www.intel.com/cd/software/products/asmona/eng/219848.htm).

2.5 KoDaVis

Mit Hilfe der Software KoDaVis (Kolaborative Daten-Visualisierung) wurde exemplarisch der Datendurchsatz gemessen, der sich für eine Visualisierungssession über das Netzwerk ergibt (http://www.viola-testbed.de/content/index.php?id=kodavis0). KoDaVis wurde auf dem Testsys-

tem installiert, auf einer Maschine als Datenserver und auf der anderen als -Client. Um Daten lokal mit Hilfe von AVS-Express rendern zu können, fordert KoDaVis von dem Datenserver die darzustellenden Daten an.

2.5.1 KoDaVis mit TCP

In der Standard-Implementierung benutzt KoDaVis TCP sowohl zur Kommunikation mit dem Datenserver als auch zum Transport der Visualisierungsdaten vom Datenserver zur Render-Engine.

2.5.2 KoDaVis mit UDT

Aufbauend auf der um UDT erweiterten VISIT-Bibliothek lässt sich auch UDT als Transportprotokoll in einer entsprechend angepassten KoDaVis-Version nutzen. KoDaVis greift hierbei nicht direkt auf die UDT-Sockets zu, sondern implementiert den Zugriff auf die tieferliegenden Transport- und Netzwerkschichten über die UDT-fähige VTS-Schnittstelle der VISIT-Bibliothek.

3 Aufbau der Messumgebung

3.1 Hardware

Es wurden Messungen zwischen zwei Testservern durchgeführt. Auf diesen Endsystemen kamen pro Maschine folgende Komponenten zum Einsatz:

- Tyan Thunder K8WE (S2895), 4x 512 MB DDR400
- Chipsatz: AMD 8131, Nvidia Professional 2200&2050
- CPU: 2x AMD Opteron Dual Core 265 (1800 Mhz)
- NIC: nVidia Corporation CK804 1 Gbit/s Ethernet Controller (Onboard)

3.2 Betriebssystem

Die Messungen wurden unter SLES 10 (Kernelversion 2.6.16.27-0.6-smp) und Opensuse 10.2 (Kernelversion 2.6.18.8-0.1-default) durchgeführt.

Die Treiber der Netzwerkinterfaces wurden geladen mit:

```
options forcedeth optimization_mode=1,1
```

Die TCP-Parameter waren folgendermaßen konfiguriert:

```
net.ipv4.tcp_congestion_control = reno
net.ipv4.tcp_moderate_rcvbuf = 1
net.ipv4.tcp_rmem = 4096        87380   4194304
net.ipv4.tcp_wmem = 4096        16384   4194304
net.ipv4.tcp_sack = 1
net.ipv4.tcp_timestamps = 1
net.core.netdev_max_backlog = 1000
net.core.rmem_default = 126976
net.core.wmem_default = 126976
```

```
net.core.rmem_max = 131071
net.core.wmem_max = 131071
```

3.3 Netzwerkumgebung

Einer der Testrechner war in der Maschinenhalle des ZAMs im Forschungszentrum Jülich installiert, ein weiterer wurde temporär im Rechenzentrum der RWTH Aachen aufgebaut. Folgende drei Szenarien wurden untersucht:

3.3.1 VIOLA

Für die Messungen zwischen Jülich und Aachen wurden die beiden Testsysteme über 1 Gigabit/s-Ethernet-Switches miteinander verbunden. Auf Layer 3 lagen beide Rechner ohne weitere dazwischen geschaltete Komponenten in einem gemeinsamen Subnetz. Die RTT betrug < 1 ms.

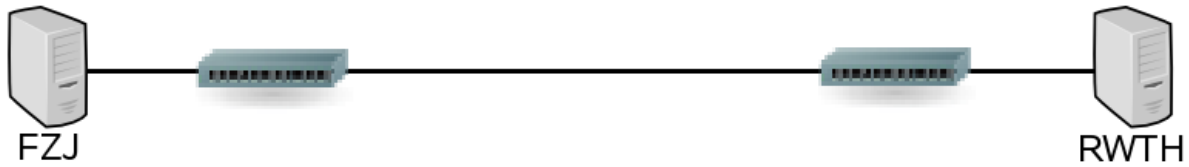

Abbildung 1: FZJ<->RWTH über VIOLA, geswitched, selbes Subnetz

3.3.2 Campuslink, mit RWTH-Firewall

Desweiteren wurden die Testsysteme sowohl in Jülich als auch in Aachen ins jeweilige Produktionsnetz integriert, um die Messungen zwischen den Maschinen über das Internet, also unter nicht-dedizierten Bedinungen, durchzuführen. Pakete zwischen den Testmaschinen waren so Störungen durch Querverkehr anderer Benutzer ausgesetzt und durchliefen sowohl auf jülicher wie auch auf aachener Seite aktive Komponenten wie Router und Firewalls. Die Route zwischen Jülich und Aachen lief hierbei über den sogenannten Campuslink. Die RTT schwankte im Durchschnitt zwischen 3 und 12 ms (Jitter).

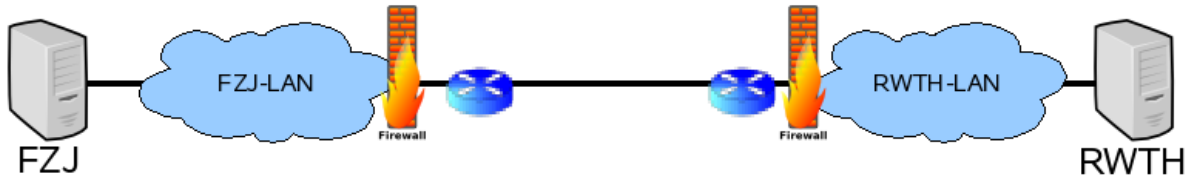

Abbildung 2: FZJ<->RWTH über Campuslink, mit RWTH-Firewall

3.3.3 X-WIN, ohne RWTH-Firewall

Um Effekte des lokalen Netzwerkes inklusive Firewall zu minimieren, wurde schließlich die Testmaschine in Aaechen AUßERHALB der RWTH-Firewall angeschlossen und auf Layer 3 so konfiguriert, dass die Pakete zwischen den Rechnern nicht über den direkten Campuslink, sondern über die X-WIN-default-Route in Frankfurt (FFM) lief. Die RTT betrug hierbei durchschnittlich < 5 ms.

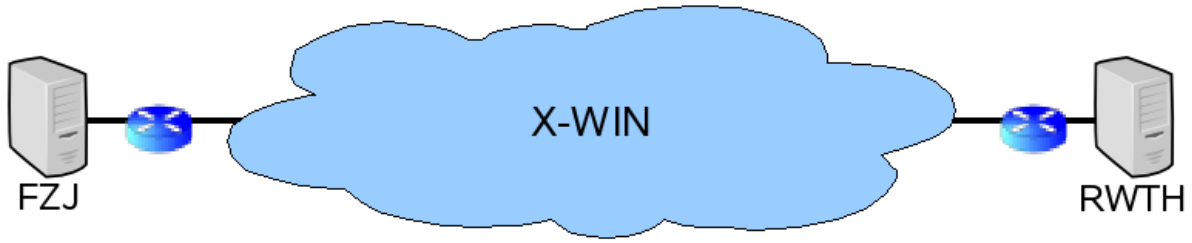

Abbildung 3: FZJ<->RWTH über X-WIN (FFM)

3.4 Getestete Transportprotokollvarianten

Es wurden Tests für folgende TCP-Varianten durchgeführt:

- TCP BIC
- TCP CUBIC
- TCP HIGHSPEED
- TCP HTCP
- TCP HYBLA
- TCP RENO
- TCP SCALABLE
- TCP VEGAS
- TCP WESTWOOD

Der modulare Aufbau des TCP-Stacks moderner Linux-Kernels erlaubt den Austausch der verwendeten Staukontrolle per

```
sysctl -w net.ipv4.tcp_congestion_control=<tcp-variante>
```

Außerdem wurden Messungen mit UDT mit Default Congestion-Control-Implementierung durchgeführt.

4 Durchführung und Ergebnisse der Messungen

4.1 iperf

Iperf-Server und -Client wurden mit Default-Parametern gestartet. Um die Messergebnisse mit

KoDaVis zu korrelieren, wurden zu den KoDaVis-Messungen identische Nachrichtengrößen (3384 Kbyte) und zu übertragende Datenmengen (10 GByte) gewählt.

```
iperf -s
iperf -c $srvaddress -n 10G -l 3384K -i 1
```

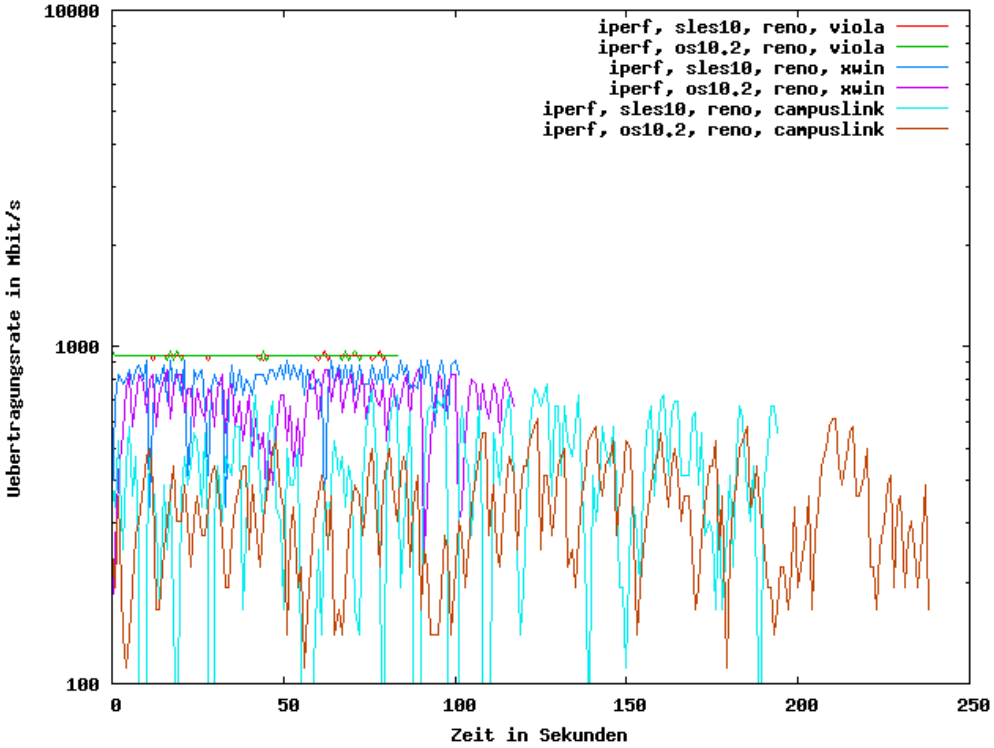

Abbildung 4: iperf, Übertragungsrate, Übersicht

In Abbildung 4 ist deutlich zu erkennen, wie in ungestörter, dedizierter Umgebung die Übertragungsrate beinahe konstant den theoretisch maximalen Wert von 1 Gbit/s erreicht. Erhöhen sich Störungen und Jitter, unter anderem hervorgerufen von der unter Last stehenden Firewall und anderer Komponenten, so verschlechtert sich das Bild. Um 10 Gbyte zu übertragen, benötigt man über VIOLA weniger als die Hälfte der Zeit als über das öffentlich Netz.

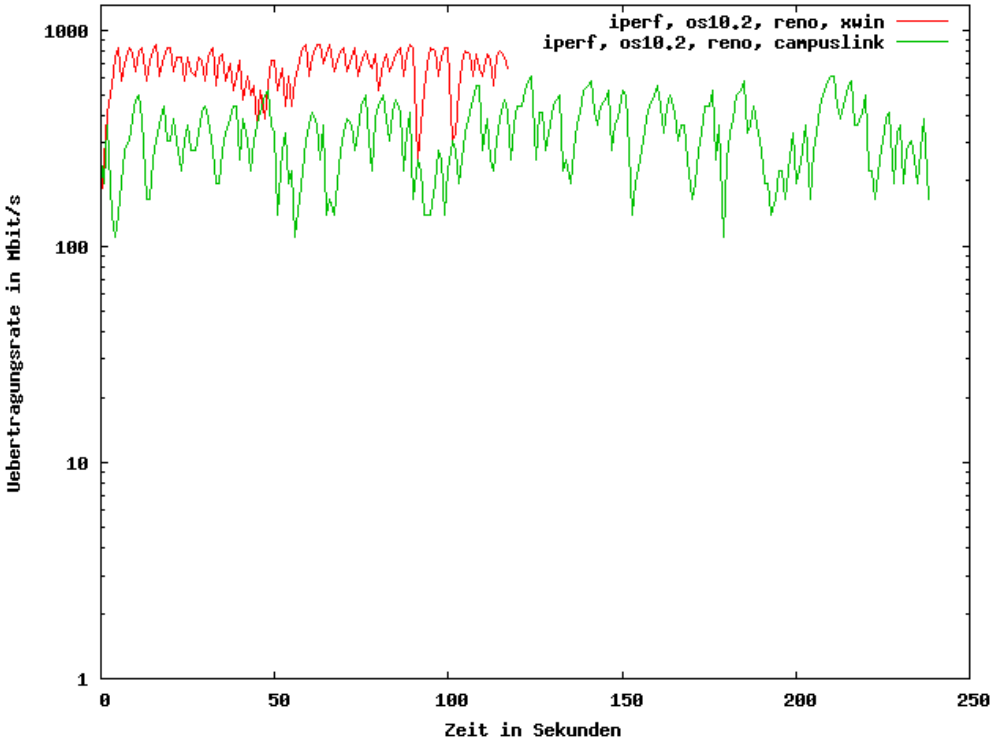

Abbildung 5: iperf, Vergleich X-Win und Campuslink

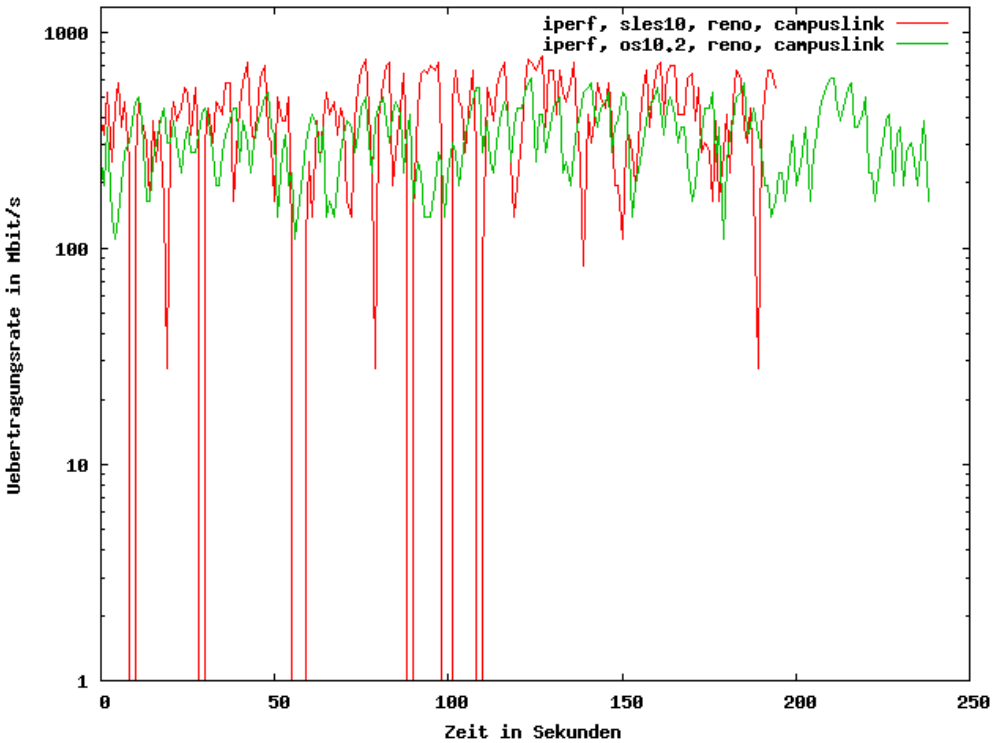

Abbildung 6: iperf, Vergleich SLES10 und OpenSuse 10.2

Die TCP-Reno-Version von SLES 10 benötigt weniger Zeit zur Übertragung der 10 Gbyte, zeigt aber auf den vermutlich von der Firewall induzierten Jitter sehr sprunghaftes Verhalten. Eine Analyse der Ausgabe von `netstat -s` zeigt ein deutlich häufigeres Einspringen des Fast-Retransmit-Mechanismus unter Open Suse 10.2 (Abbildung 6), welcher die Übertragung „glättet“.

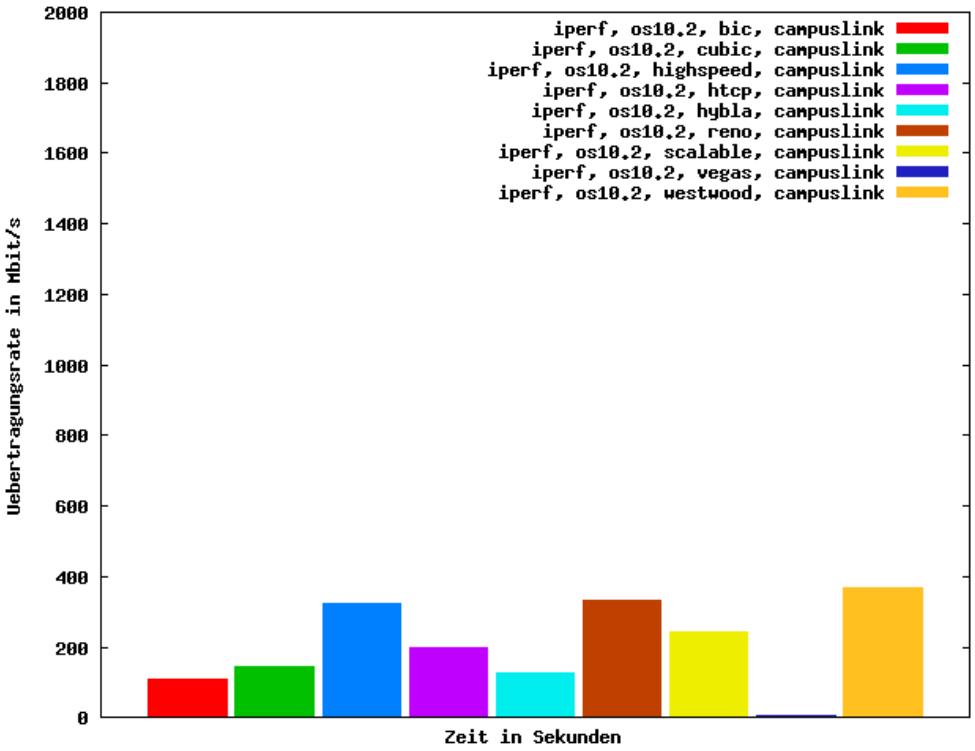

Abbildung 7: iperf, Vergleich TCP-Varianten, OpenSUSE 10.2, Campuslink

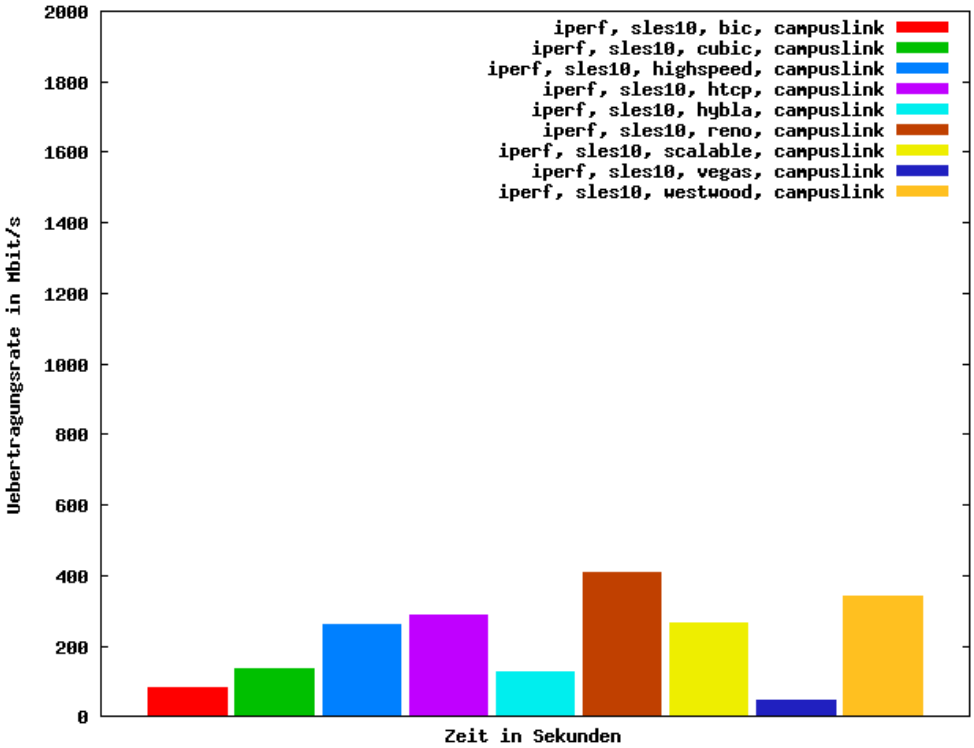

Abbildung 8: iperf, Vergleich TCP-Varianten, SLES10, Campuslink

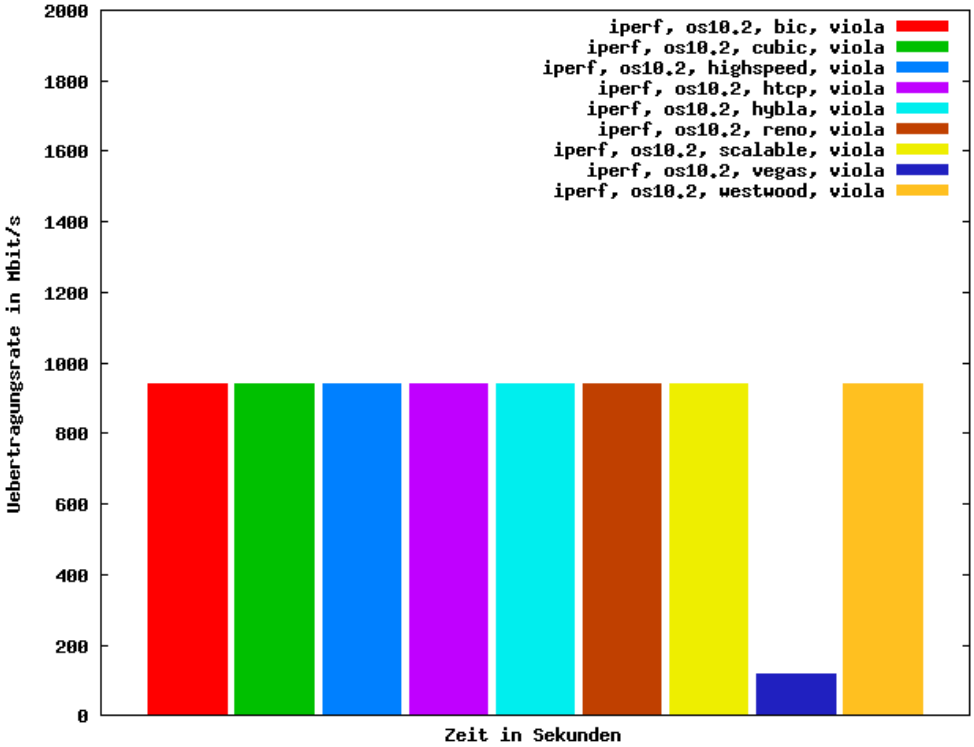

Abbildung 9: iperf, Vergleich TCP-Varianten, OpenSUSE 10.2, VIOLA

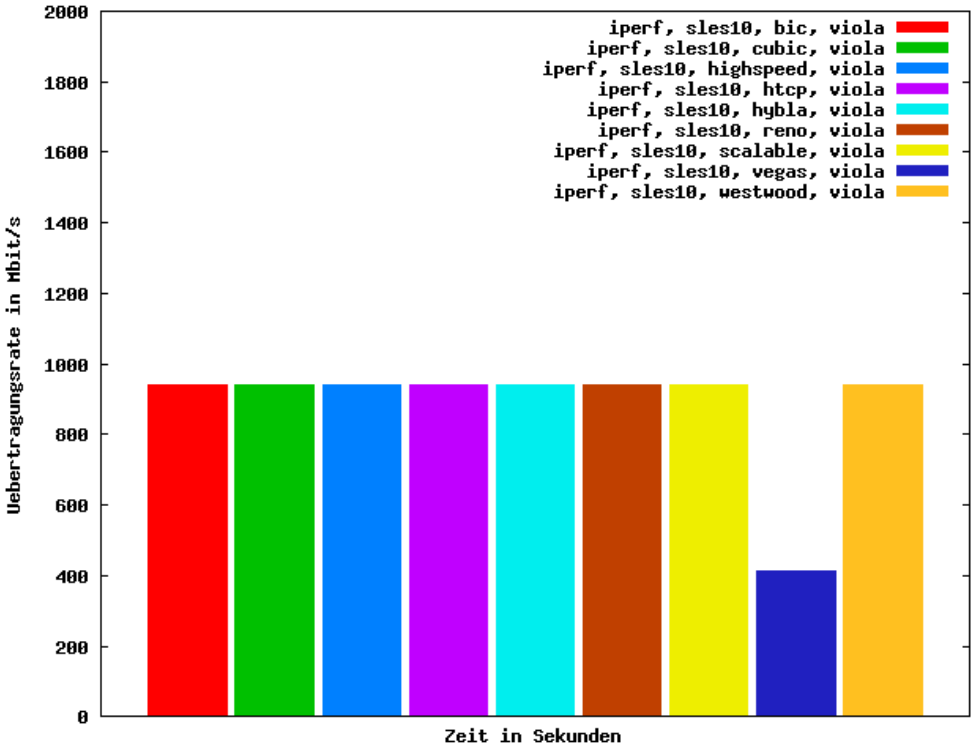

Abbildung 10: iperf, Vergleich TCP-Varianten, SLES10, VIOLA

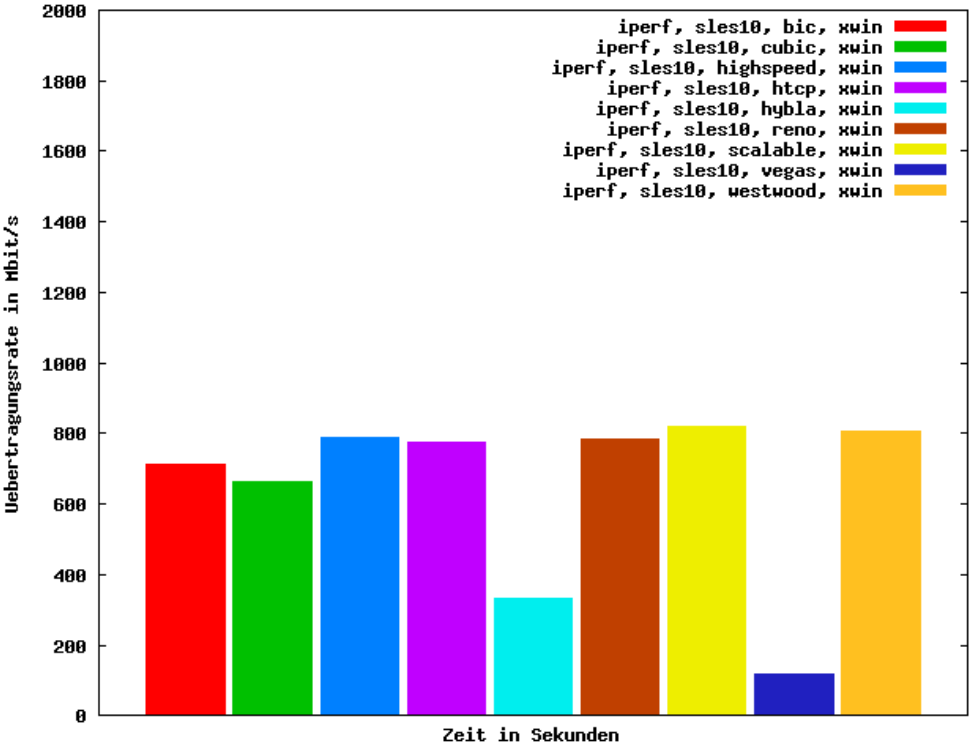

Abbildung 11: iperf, Vergleich TCP-Varianten, SLES10, X-WIN

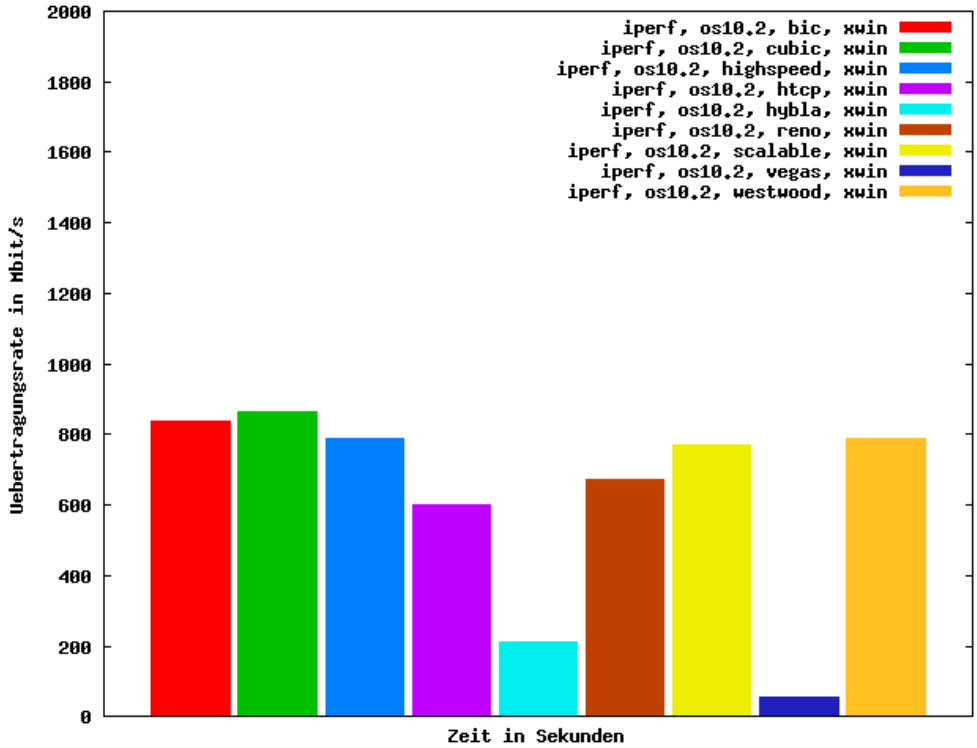

Abbildung 12: iperf, Vergleich TCP-Varianten, OpenSUSE 10.2, X-WIN

Abbildung 7 bis Abbildung 11 zeigen die durchschnittliche Übertragungsrate bei der Übertragung von 10 Gbyte Daten mit iperf. Auffällig ist, dass TCP VEGAS in beiden Kernelversionen (2.6.16.27-0.6-smp und 2.6.18.8-0.1-default) deutlich einbricht.

4.2 tcppp

tcppp-Server und -Client wurden folgendermaßen gestartet:

```
tcppp -b 0 -B 0 -S
tcppp -b 0 -B 0 -F pp.ini $srvaddress
```

„`-b 0 -B 0`“ bewirkt hierbei, dass die Applikation keinerlei Anpassungen an den Socket-Buffer-Größen vornimmt, wodurch die Autotuning-Mechanismen des Kernels zum Tragen kommen können.

Über die Konfigurationsdatei „`pp.ini`“ wurden die zu übertragenden Nachrichtengrößen an die Applikation übergeben. Es kamen Nachrichten zwischen 1 Byte und 10 000 000 Byte Größe zum Einsatz, mit einem Größen-Wachstumsfaktor von 1.2.

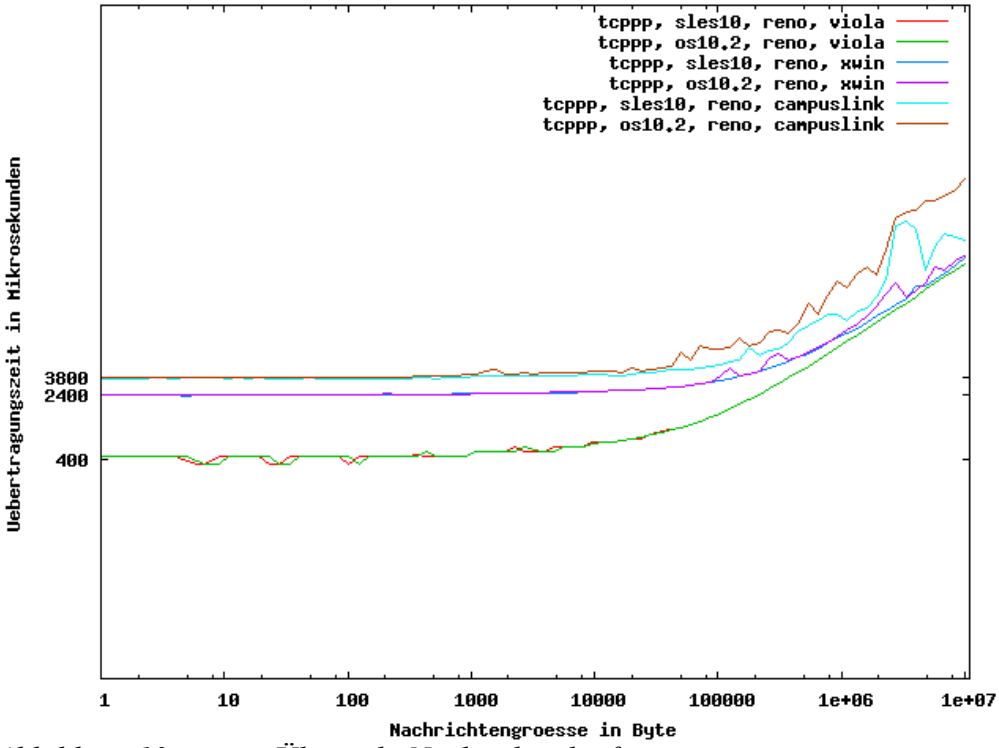

Abbildung 13: tcppp, Übersicht Nachrichtenlaufzeiten

In Abbildung 13 ist deutlich zu sehen, wie für kleine Nachrichtengrößen die durchschnittliche Nachrichtenlaufzeit (für die einfache Strecke gemessen) der (halben) RTT entspricht. Für größere Nachrichten verhält sich die Übertragung über das ungestörte Netz nahezu ideal: nach kurzer Übergangsphase ist die Nachrichtenlaufzeit direkt proportional zur Nachrichtengröße (ab ca 100 Kbyte). Unter gestörten Bedinungen wird deutlich, dass die Nachrichtenlaufzeit durch Schwankungen in der Übertragungsrate sprunghaft ansteigen kann.

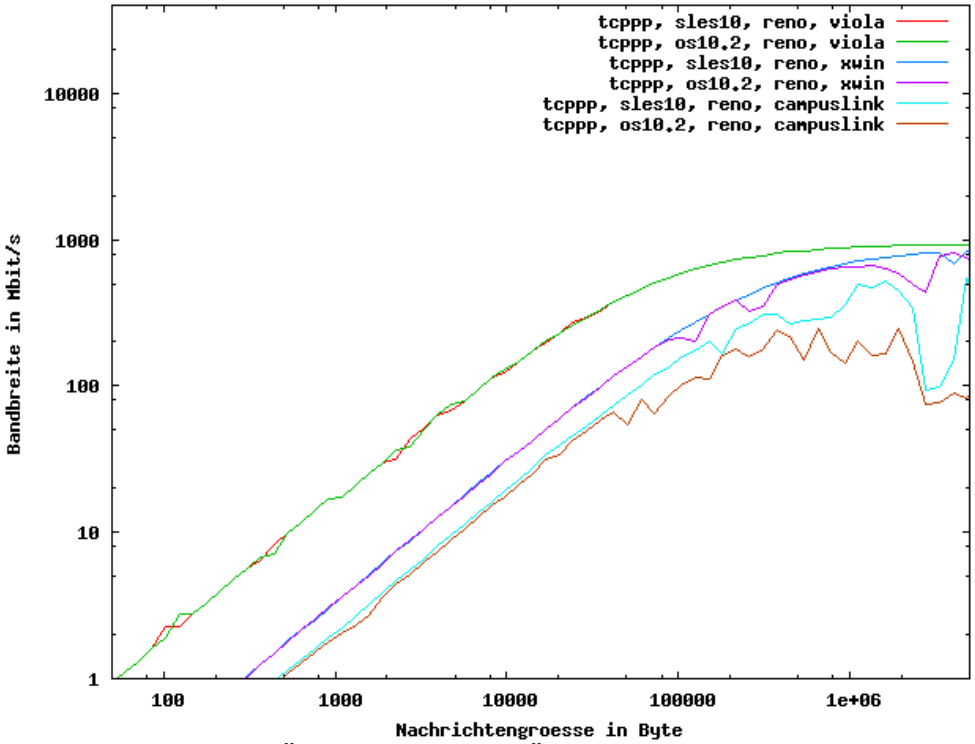

Abbildung 14: tcppp, Übertragungsrate, Übersicht

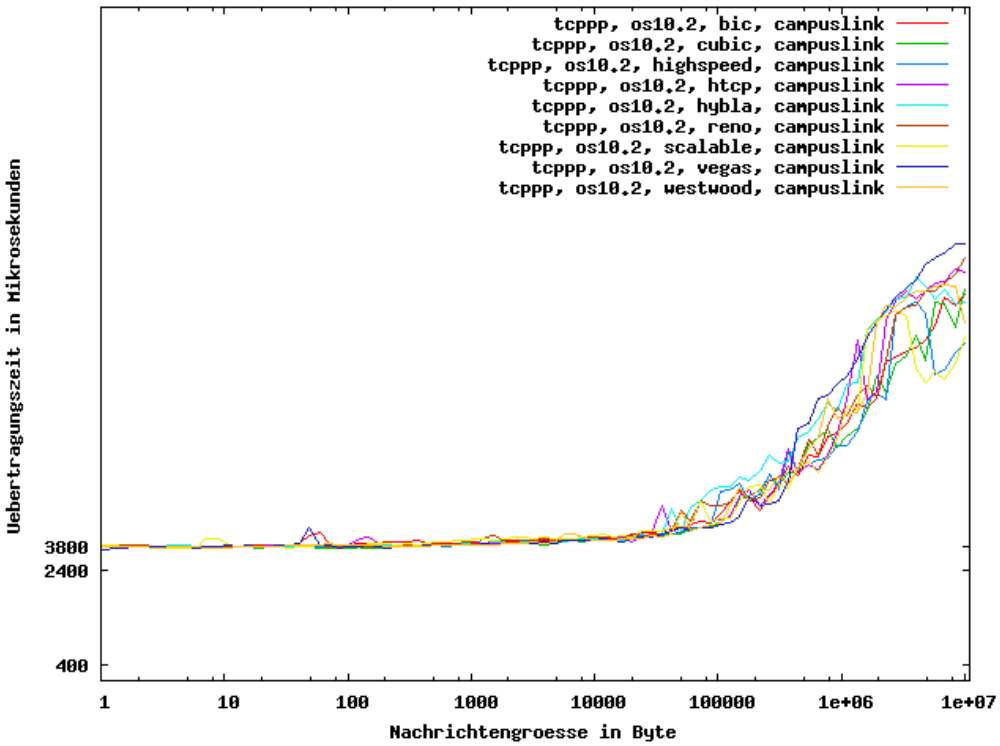

Abbildung 15: tcppp, Nachrichtenlaufzeiten, Vergleich verschiedener TCP-Varianten, OpenSUSE 10.2, Campuslink

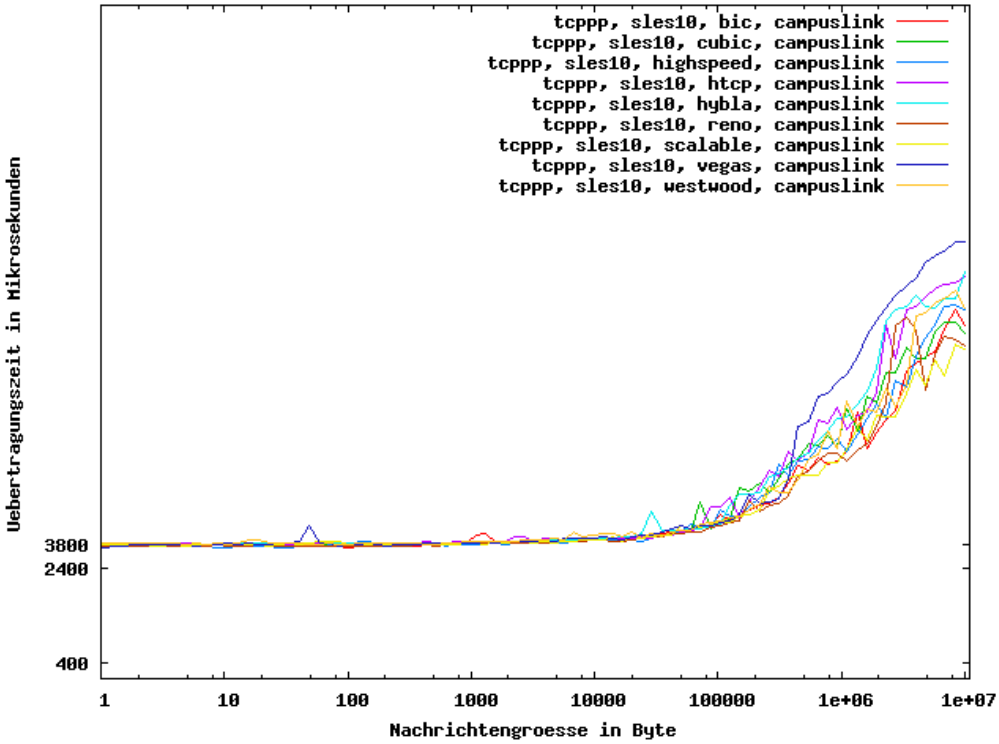

Abbildung 16: tcppp, Nachrichtenlaufzeiten, Vergleich verschiedener TCP-Varianten, SLES10, Campuslink

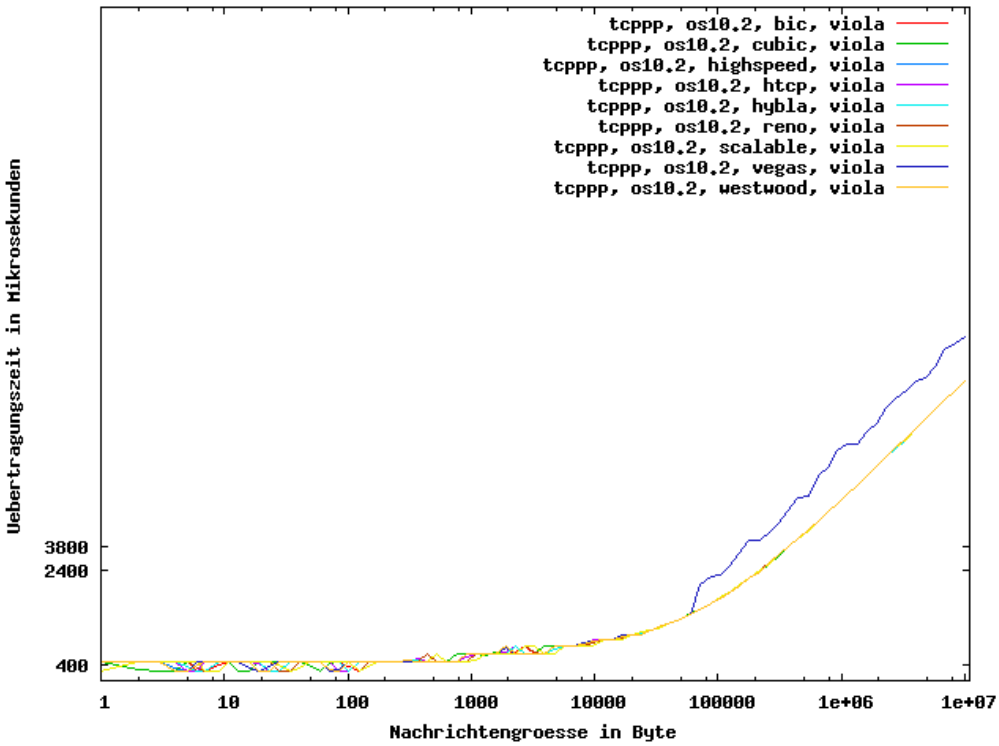

Abbildung 17: tcppp, Nachrichtenlaufzeiten, Vergleich verschiedener TCP-Varianten, OpenSUSE 10.2, VIOLA

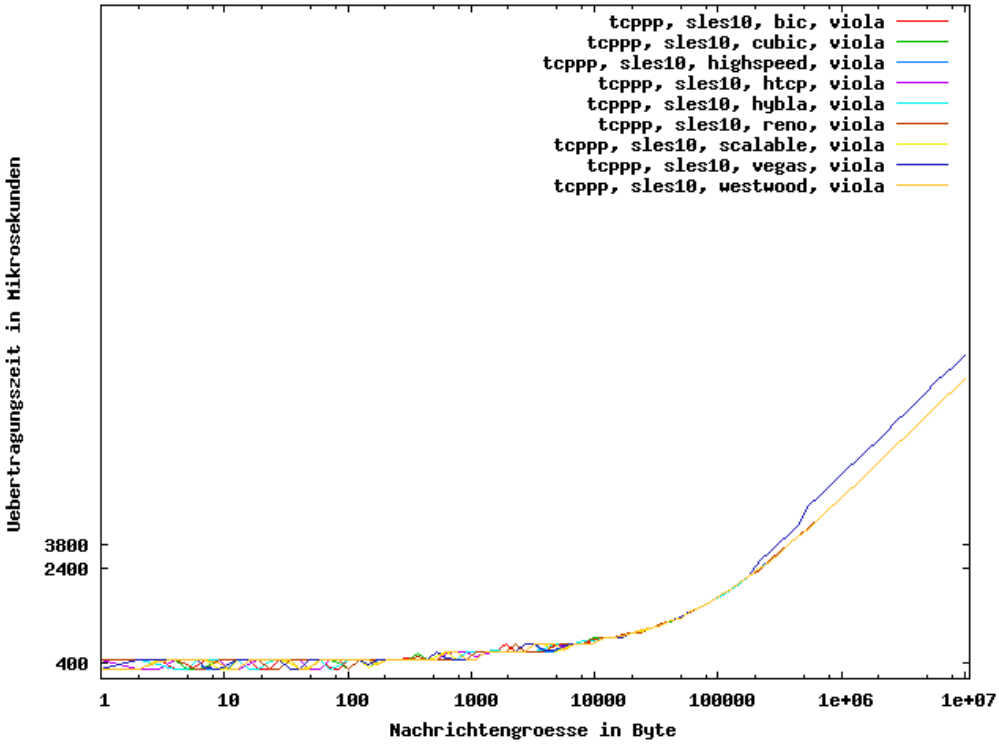

Abbildung 18: tcppp, Nachrichtenlaufzeiten, Vergleich verschiedener TCP-Varianten, SLES10, VIOLA

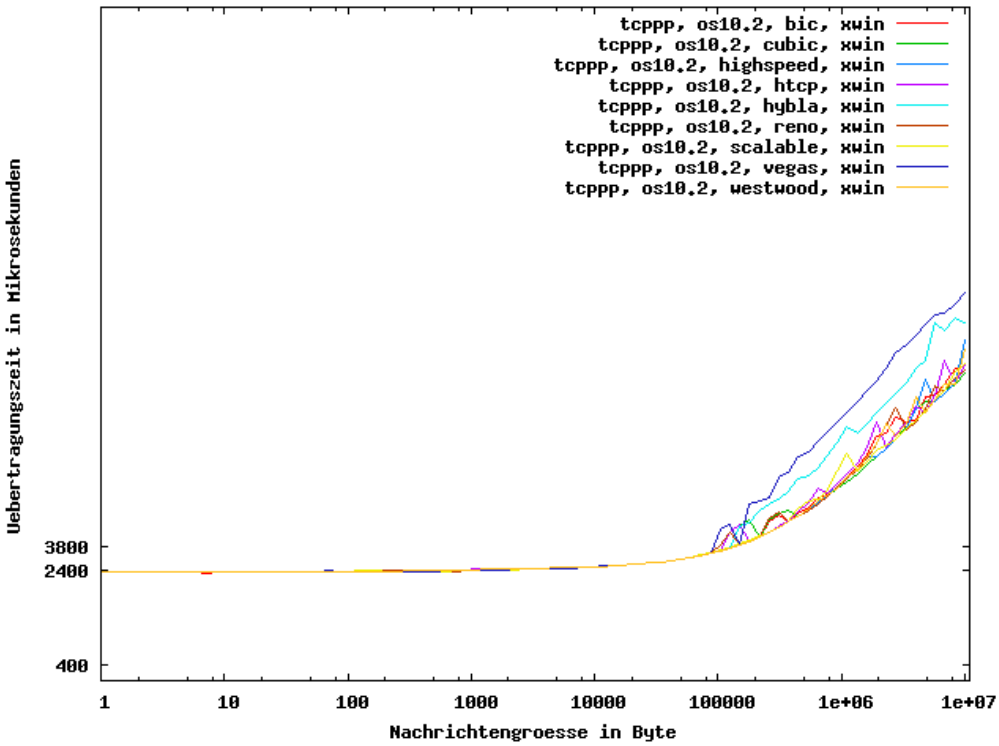

Abbildung 19: tcppp, Nachrichtenlaufzeiten, Vergleich verschiedener TCP-Varianten, OpenSUSE 10.2, X-WIN

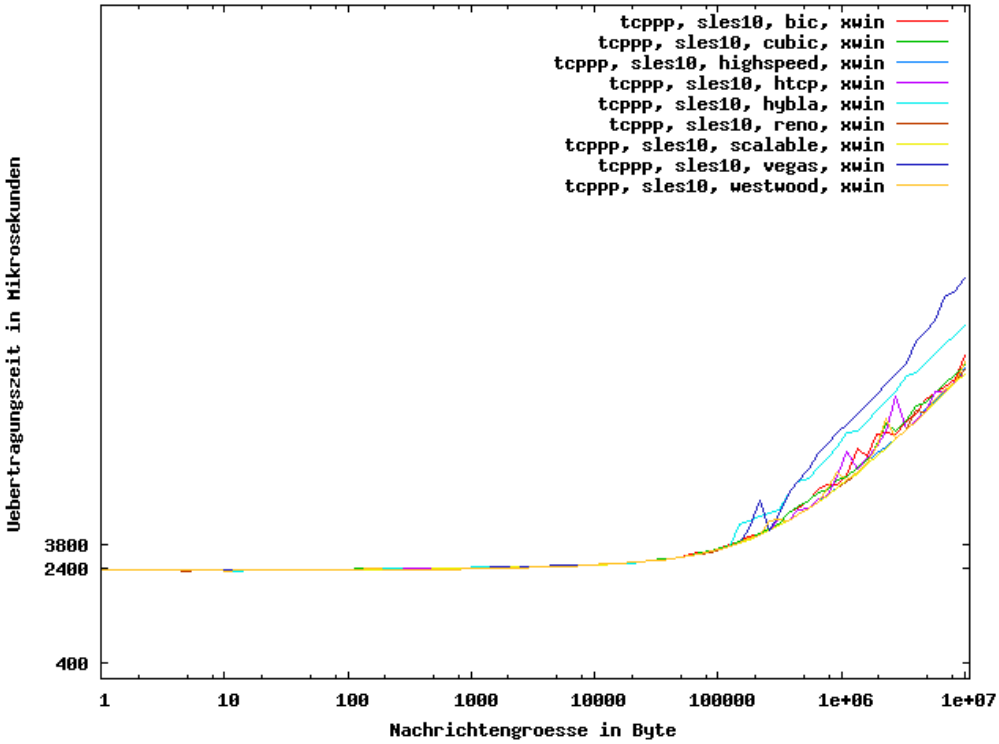

Abbildung 20: tcppp, Nachrichtenlaufzeiten, Vergleich verschiedener TCP-Varianten, SLES10, X-WIN

Abbildung 15 bis Abbildung 20 verdeutlichen nochmals die deutlich schlechtere Performanz von TCP VEGAS gegenüber den anderen TCP-Varianten.

4.3 udtpp

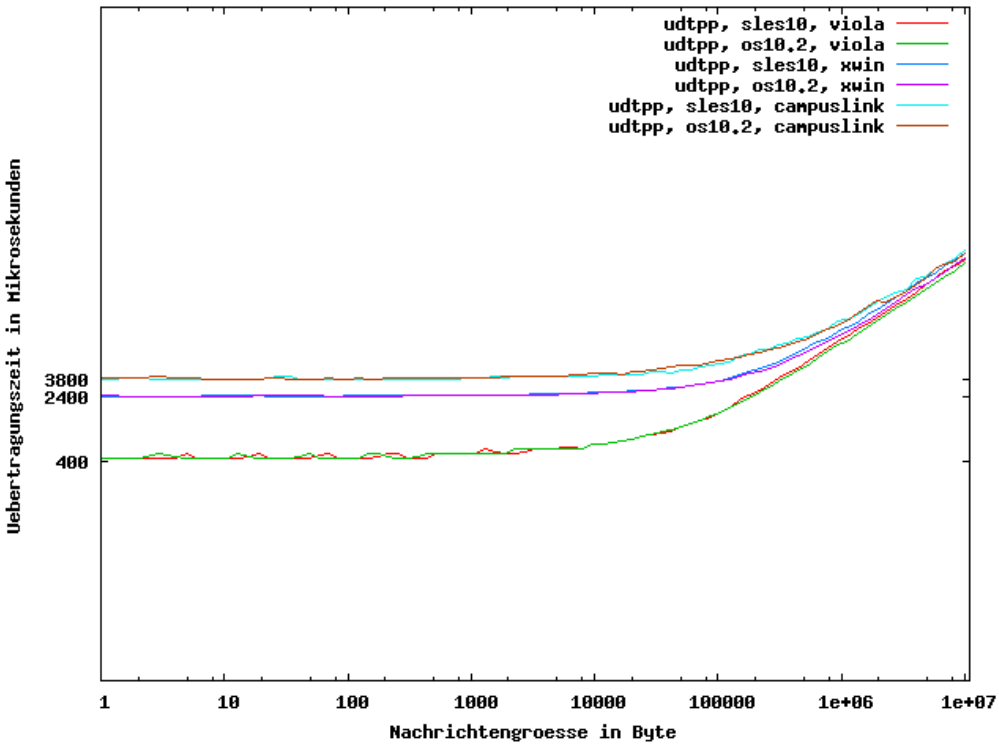

Abbildung 21: udtpp, Nachrichtenlaufzeiten in verschiedenen Netzwerkumgebungen

Das aggressive Verhalten der Default-Congestion-Control von UDT bewirkt, dass die Übertragung auch unter Jitter und Paketverlust eines Produktionsnetzes stabil bleibt, zu Ungunsten der Fairness gegenüber anderen TCP-Strömen allerdings.

4.4 IMB

Mit IMB wurde die „Pingpong“-Benchmark durchgeführt, wobei auf jeder Maschine jeweils ein Prozess, also insgesamt 2 Prozesse gestartet wurden, die mit Hilfe des TCP-Devices von metaMPICH MPI-Nachrichten zwischen 1 Byte und 4194304 Byte Größe austauschten.

Die Datei `metampich.cfg`, die auf beiden Maschinen identisch sein muss, enthielt folgende Eintragungen:

```
NUMHOSTS 2
ZAM_404 1
ZAM_406 1

OPTIONS
SECONDARY_DEVICE ch_usock

METAHOST ZAM_404 {
    TYPE=ch_usock;
    MPIROOT=/usr/local/viola/mp-mpich/mp-mpich-r4400;
```

```
    EXECPATH=/home/viola03/public/META-MULTIDEVICE/;
    NODES=zam404-mpich 1 (134.130.9.226);
}

METAHOST ZAM_406 {
    TYPE=ch_usock;
    MPIROOT=/usr/local/viola/mp-mpich/mp-mpich-r4400;
    EXECPATH=/home/viola03/public/META-MULTIDEVICE/;
    NODES=zam406-mpich 1 (194.95.187.8);
}

CONNECTIONS
PAIR ZAM_404 ZAM_406 0 -
PAIR ZAM_406 ZAM_404 0 -
```

Um diese Konfiguration lauffähig zu machen, mussten die geeigneten Namen und IP-Adressen der Testsysteme in `/etc/hosts` eingetragen werden:

```
zam404-mpich 134.130.9.226
zam406-mpich 194.95.187.8
```

Desweiteren musste der ssh-login zwischen den beiden Maschinen ohne weitere Benutzerinteraktion, also per Public-Key-Verfahren möglich sein.

IMB selber wurde folgendermaßen gestartet:

```
/usr/local/viola/mp-mpich/mp-mpich-r4400/bin/mpirun \
-meta /home/viola03/public/META-MULTIDEVICE/metampich.cfg \
/home/viola03/public/META-MULTIDEVICE/IMB-MPI1 PingPong
```

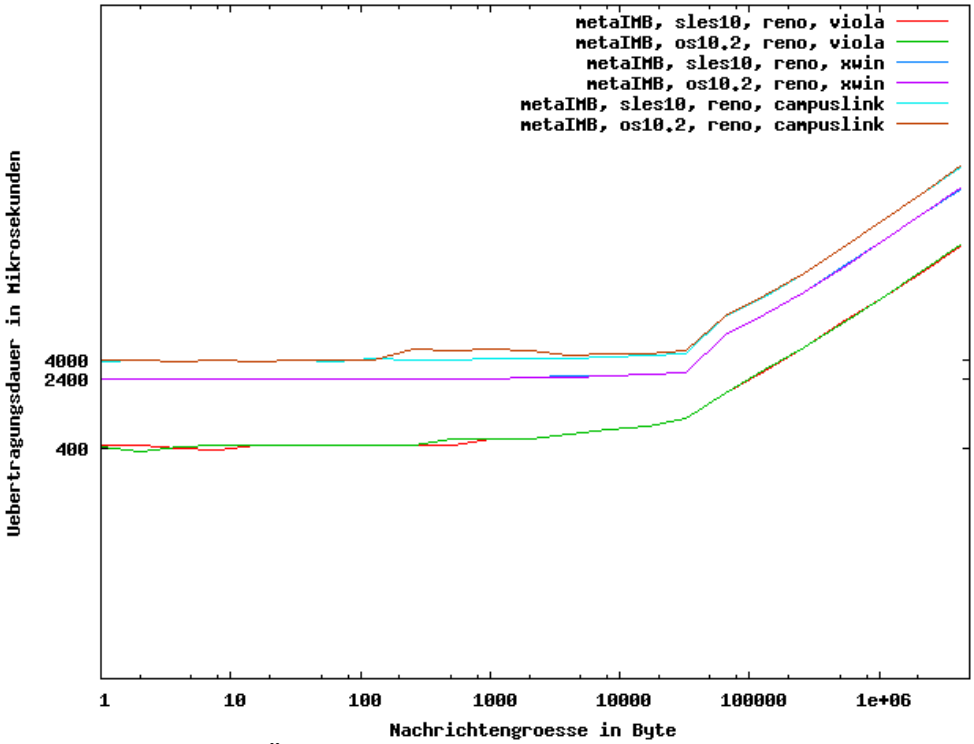

Abbildung 22: IMB, Übersicht Nachrichtenlaufzeit

Für MPI-Nachrichten gilt ebenfalls: kleine Nachrichten benötigen für die einfache Strecke vom Sender zum Empfänger die halbe RTT. Der im Vergleich zu den tcppp-Werten etwas geradlinigere Verlauf ist darauf zurückzuführen, dass die IMB-Benchmark pro Nachricht bis zu 1000 Pingpongs durchführt und somit Schwankungen nahezu komplett herausmittelt.

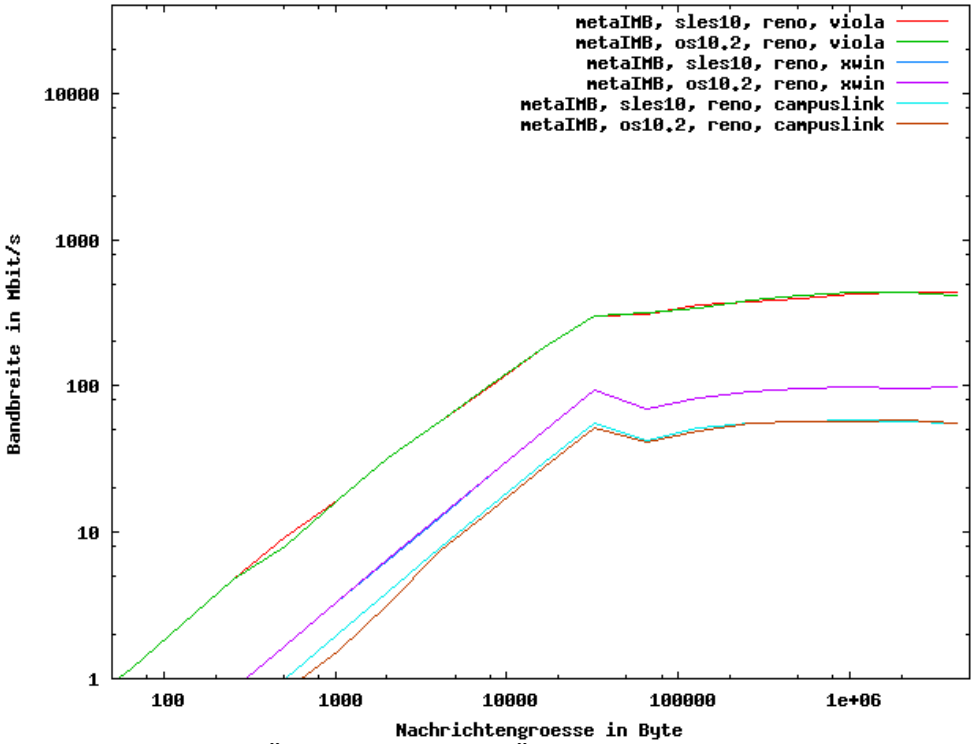

Abbildung 23: IMB, Übertragungsrate, Übersicht

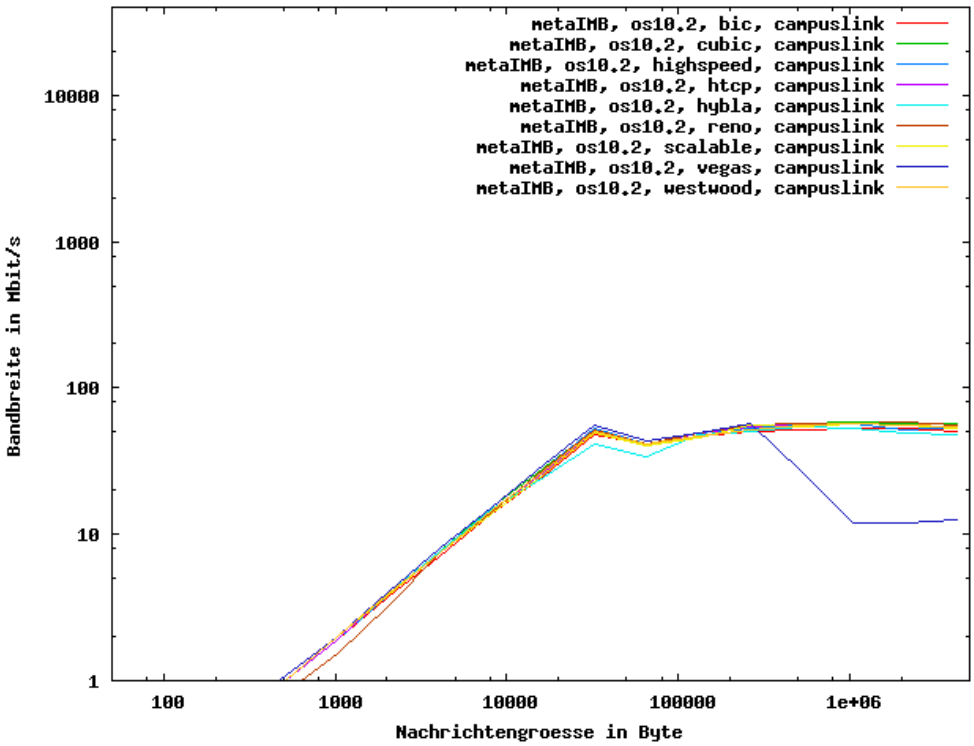

Abbildung 24: IMB, Vergleich verschiedener TCP-Varianten, OpenSUSE 10.2, Campuslink

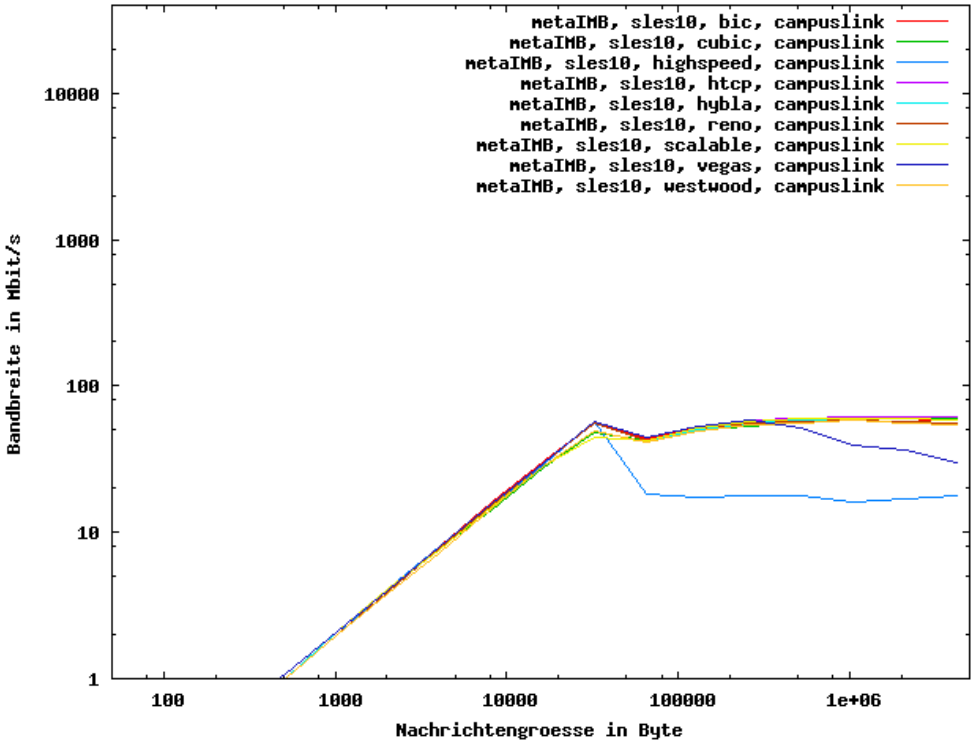

Abbildung 25: IMB, Vergleich verschiedener TCP-Varianten, SLES10, Campuslink

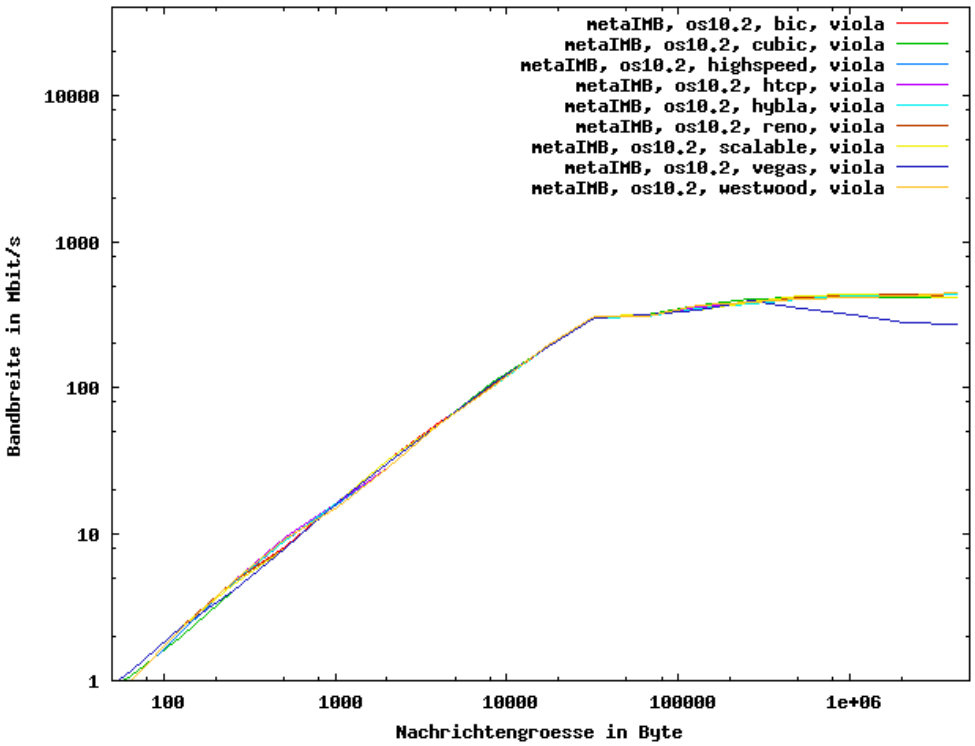

Abbildung 26: IMB, Vergleich verschiedener TCP-Varianten, OpenSUSE 10.2, VIOLA

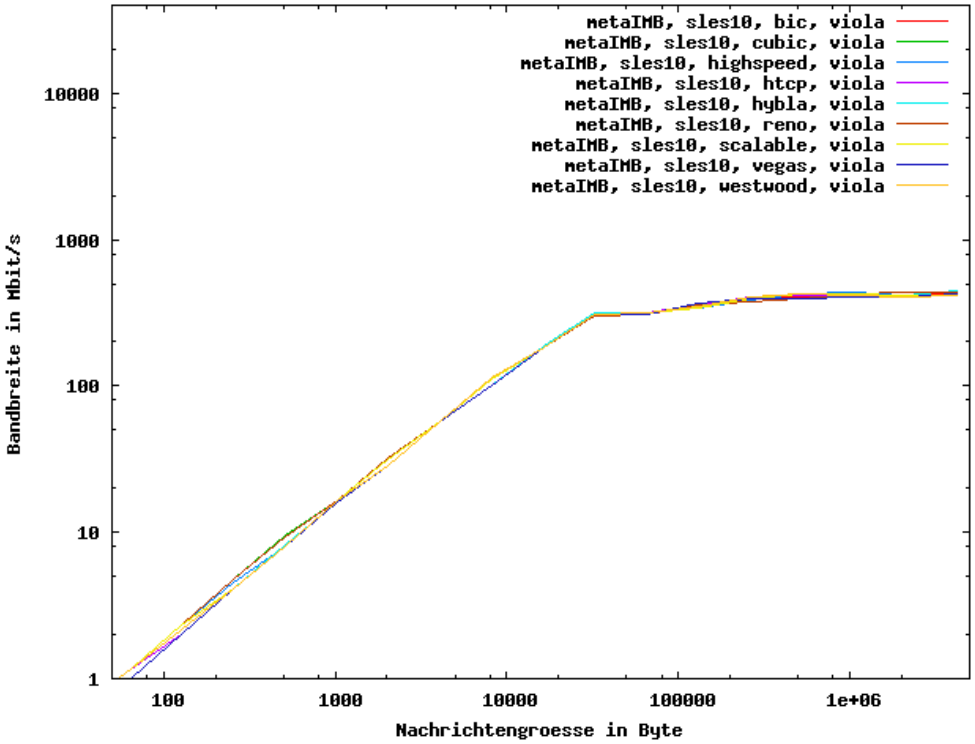

Abbildung 27: IMB, Vergleich verschiedener TCP-Varianten, SLES10, VIOLA

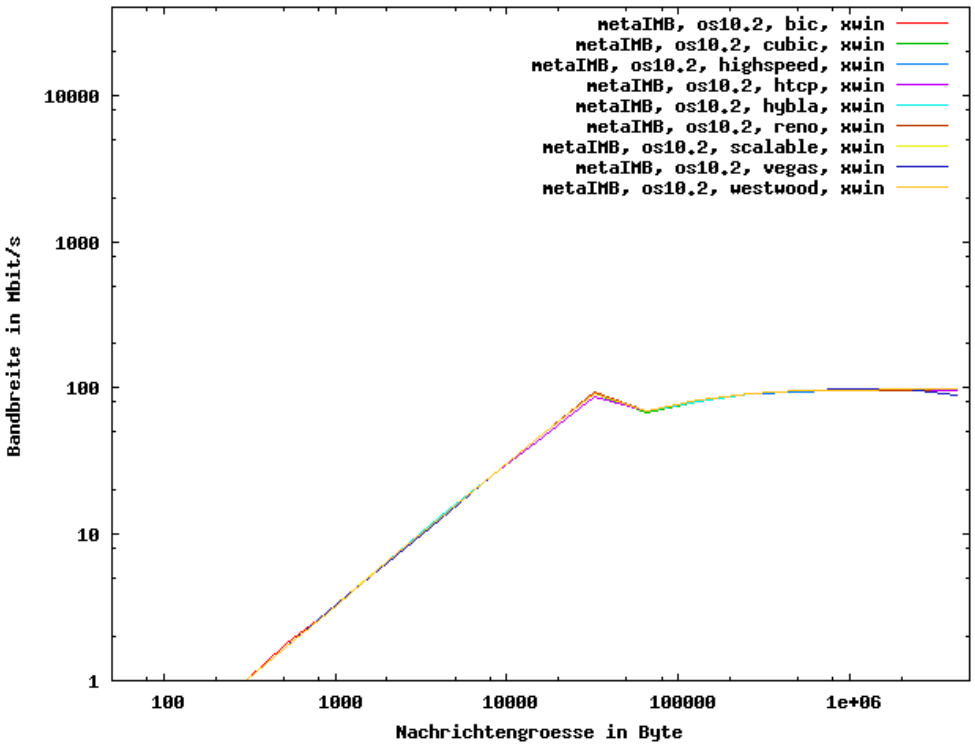

Abbildung 28: IMB, Vergleich verschiedener TCP-Varianten, OpenSUSE 10.2, X-WIN

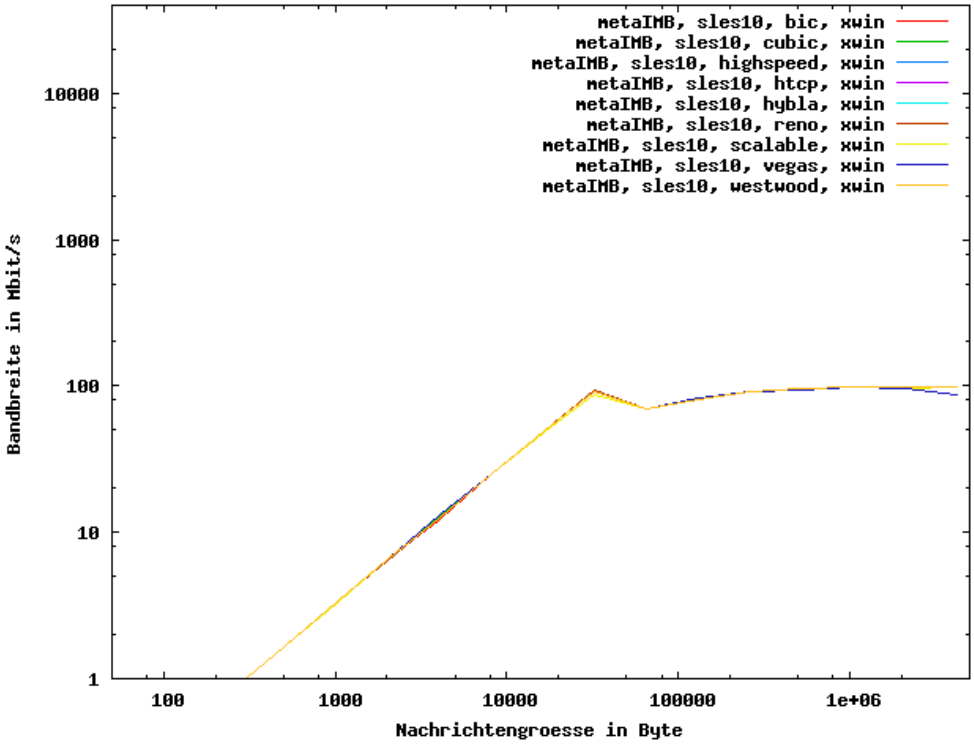

Abbildung 29: IMB, Vergleich verschiedener TCP-Varianten, SLES10, Campuslink

4.5 KoDaVis mit TCP

Bei den durchgeführten Messungen wurden Datenpakete der Größe 3384 Kbyte übertragen, die aufgrund der Caching-Mechanismen von KoDaVis direkt aus dem Arbeitsspeicher des Servers gelesen werden. Anschließend benötigt ein hinreichend modern ausgestattetes Renderingsystem durchschnittlich 80 ms, um diese Daten darzustellen. Die in den Diagrammen angegebenen Übertragungsraten sind bereits um diese Renderingzeit korrigiert.

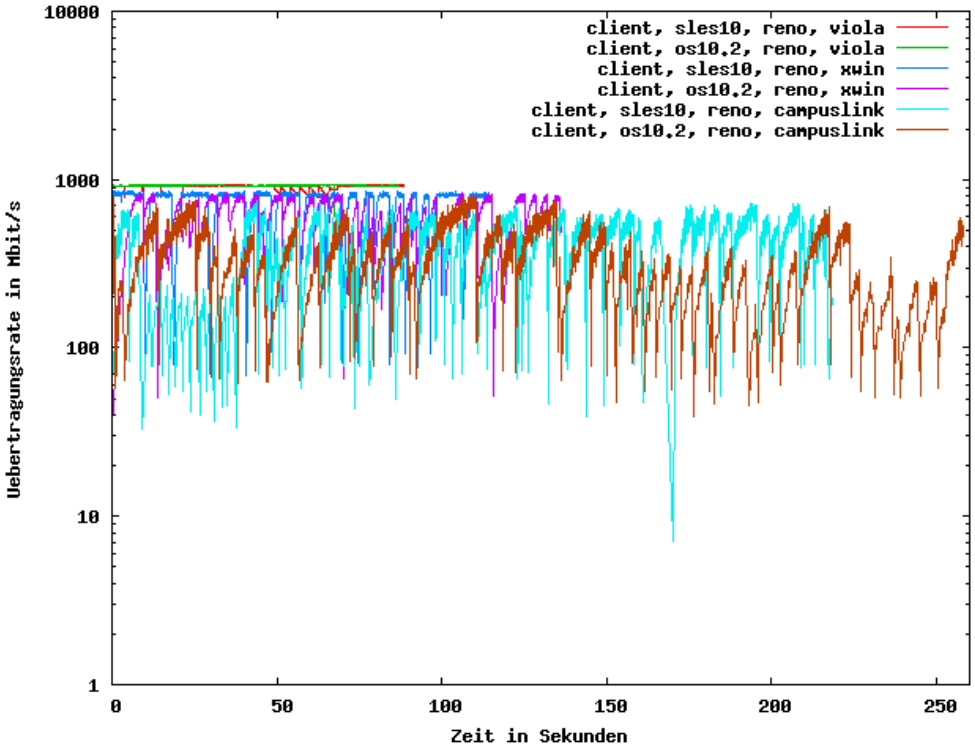

Abbildung 30: KoDaVis, Übertragungsrate, Übersicht

KoDaVis zeigt, ähnlich wie iperf, im ungestörten Netz nahezu ideales Verhalten. Die vom Datenserver angeforderten Datenpakete werden mit annähernd Wirespeed zur Visualisierungsapplikation übertragen. Je größer die Störungen durch Querverkehr, umso mehr verzögert sich die Übertragung.

	Campuslink	X-Win	VIOLA
Übertragungszeit	489 s	367 s	318 s

Tabelle 1: Visualisierung eines 10 GByte Datensatzes, Open Suse 10.2

Tabelle 1 zeigt den Einfluss der Netzwerkumgebung auf die Übertragungszeiten; die Zeit (80 ms pro Datenpaket), die das Rendering-System zur Darstellung der Daten benötigt, ist hier mit einbezogen. Die Übertragung über das VIOLA-Testbed ist hierbei ca 35 % schneller als über den regulären Campuslink und immer noch 15 % schneller, als die Übertragung unter Umgehung der lokalen Netzwerkinfrastruktur und Firewall.

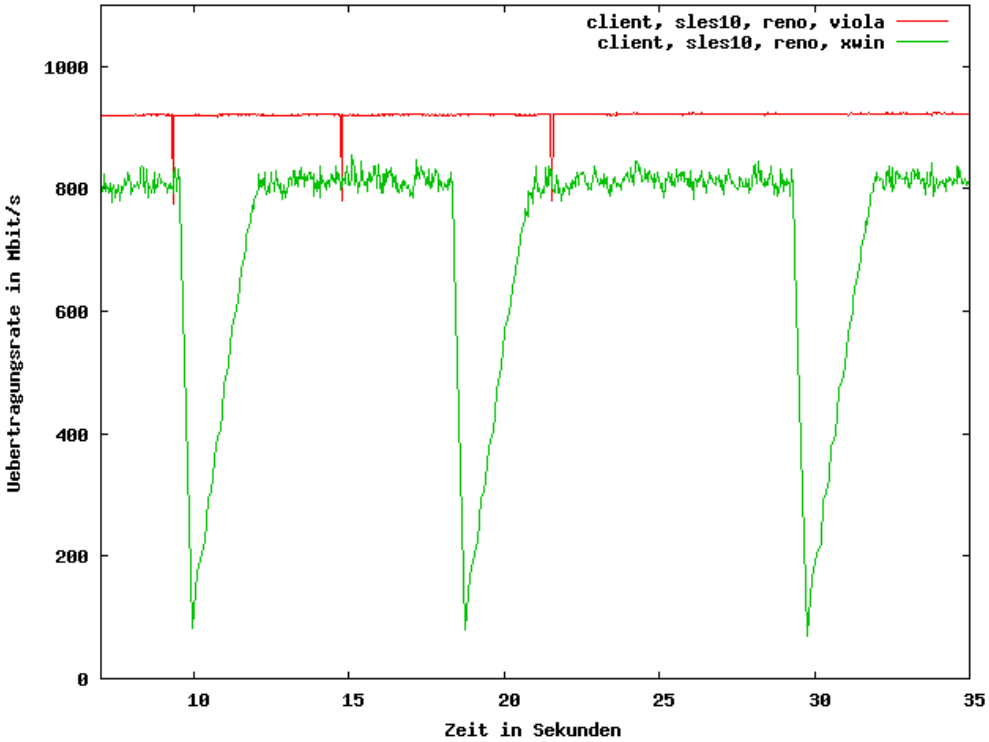

Abbildung 31: KoDaVis, Einfluss der RTT

Im Ausschnitt in Abbildung 31 ist beispielhaft die TCP-Reaktion auf Störung zu sehen. Die zeitliche „Dehnung“ der deutlich sichtbaren Congestion Avoidance (linearen Anstieg der Übertragungsrate nach Paketverlust) hängt direkt von der RTT ab.

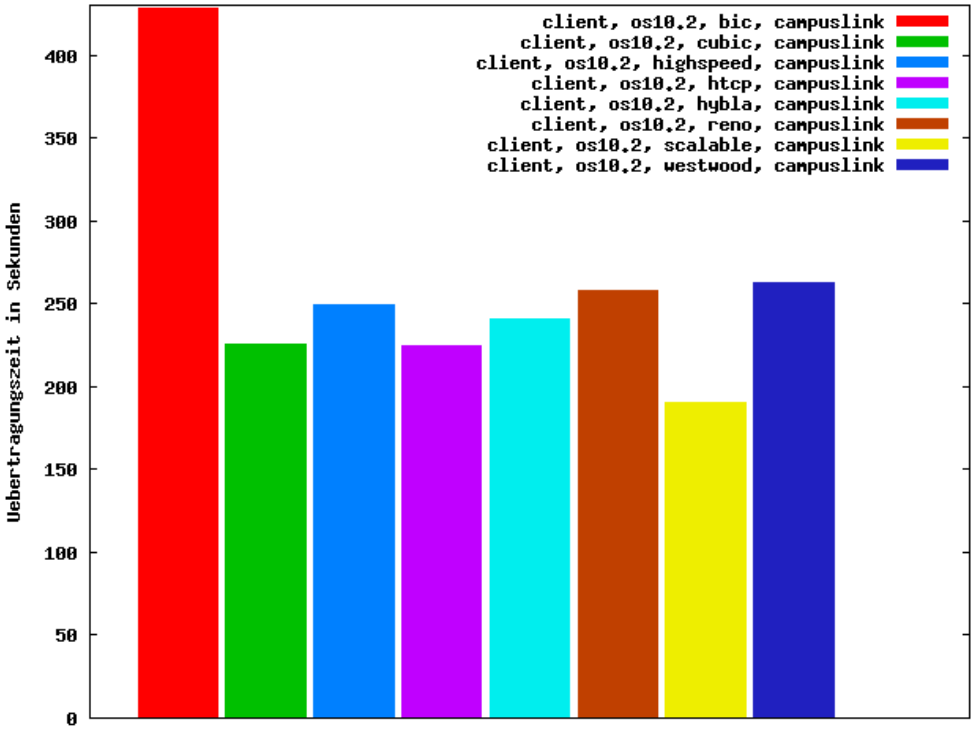

Abbildung 32: KoDaVis, Vergleich der Übertragungszeit mit verschiedenen TCP-Varianten, OpenSUSE 10.2, Campuslink

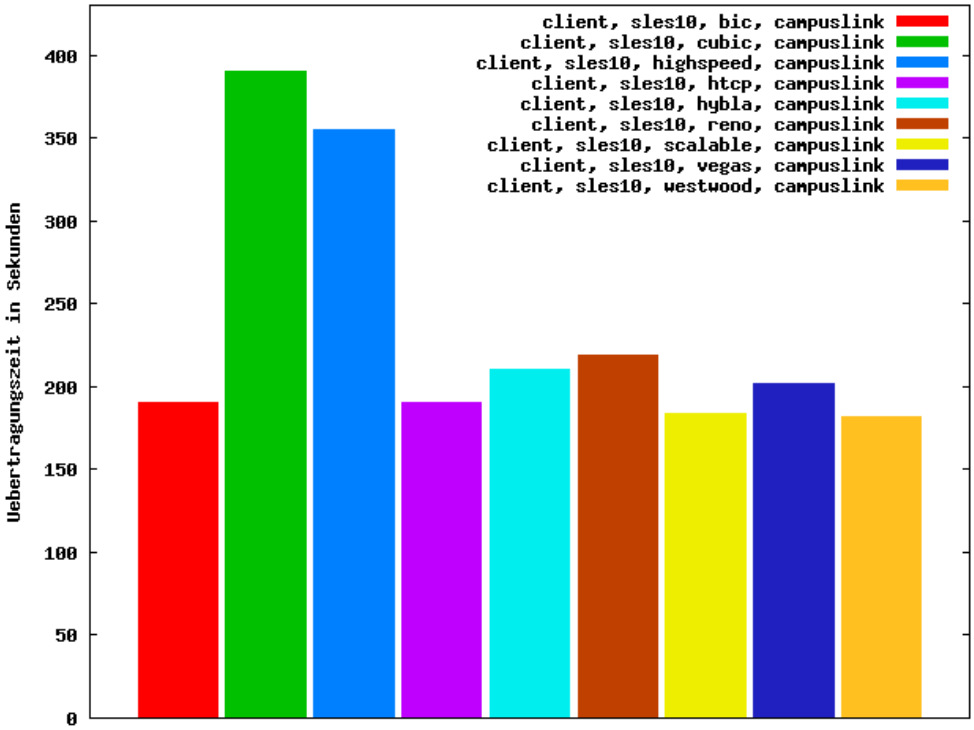

Abbildung 33: KoDaVis, Vergleich der Übertragungszeit mit verschiedenen TCP-Varianten, SLES10, Campuslink

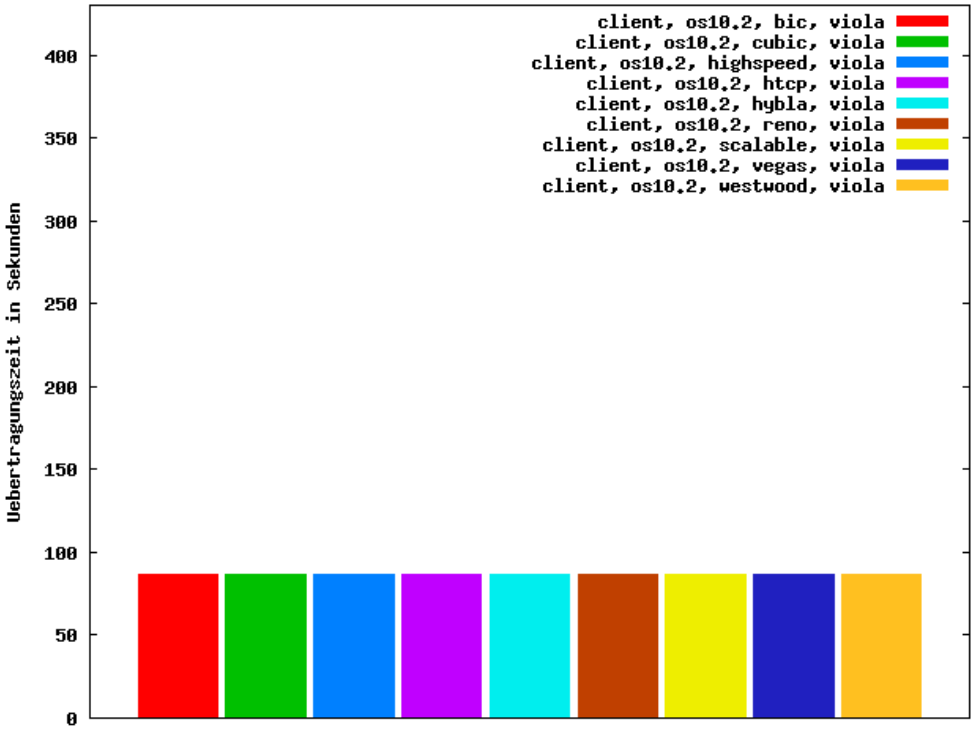

Abbildung 34: KoDaVis, Vergleich der Übertragungszeit mit verschiedenen TCP-Varianten, OpenSUSE 10.2, VIOLA

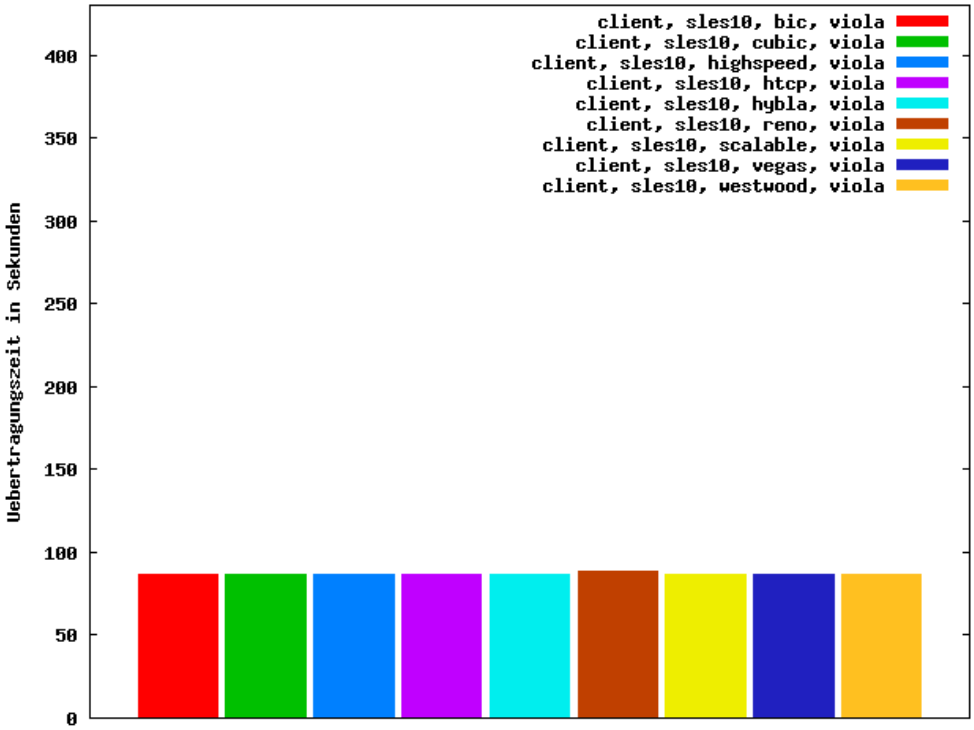

Abbildung 35: KoDaVis, Vergleich der Übertragungszeit mit verschiedenen TCP-Varianten, SLES10, VIOLA

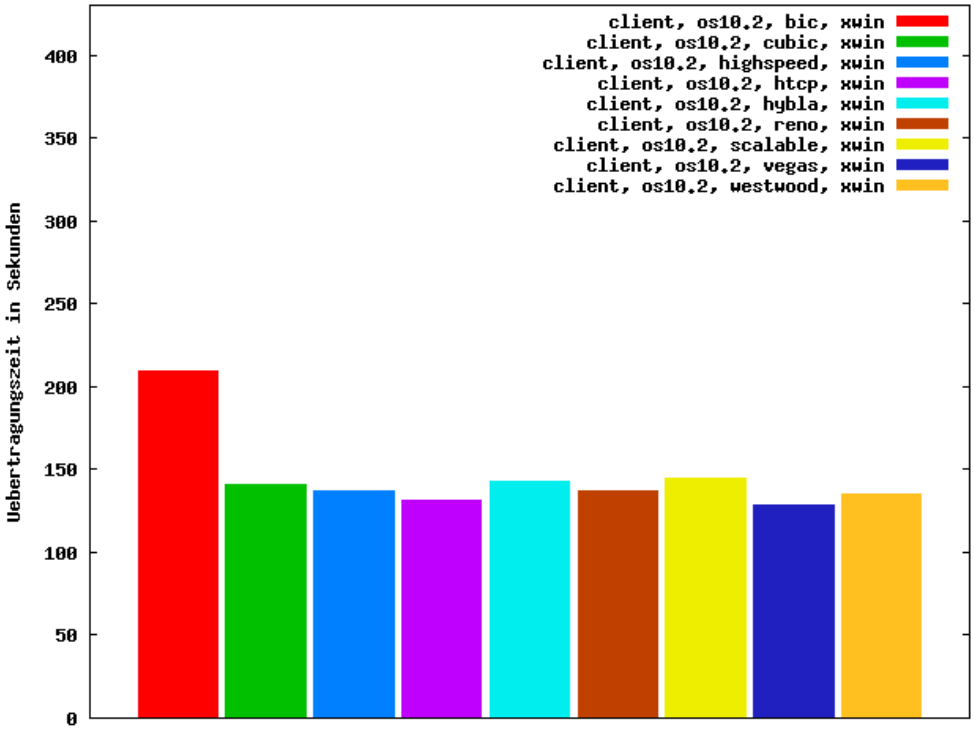

Abbildung 36: KoDaVis, Vergleich der Übertragungszeit mit verschiedenen TCP-Varianten, OpenSUSE 10.2, X-WIN

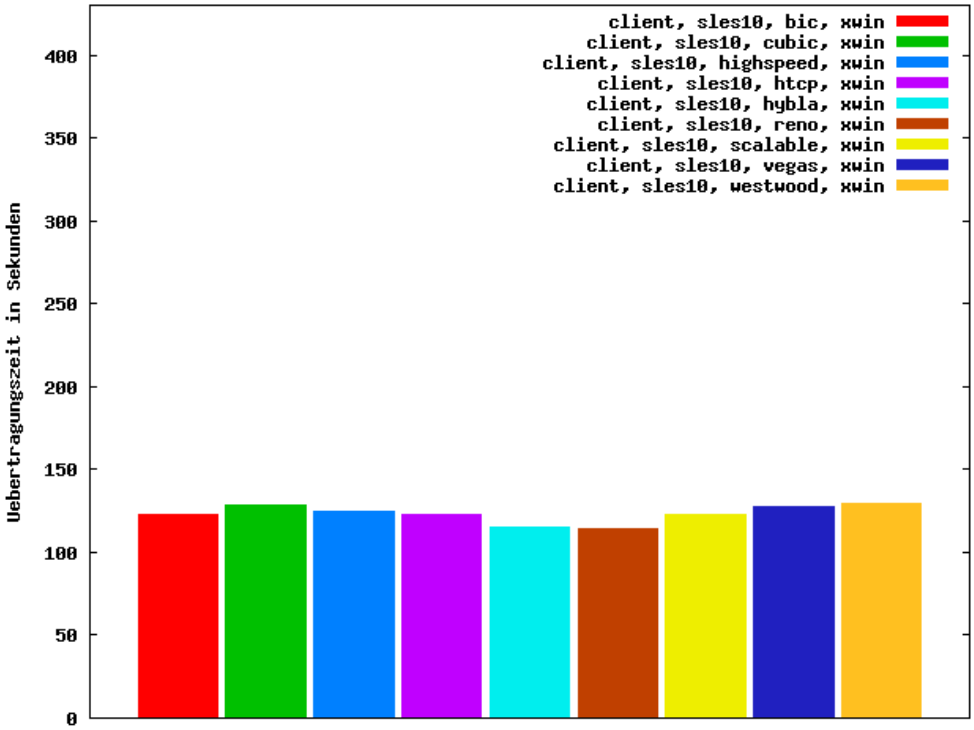

Abbildung 37: KoDaVis, Vergleich der Übertragungszeit mit verschiedenen TCP-Varianten, SLES10, X-WIN

4.6 KoDaVis mit UDT

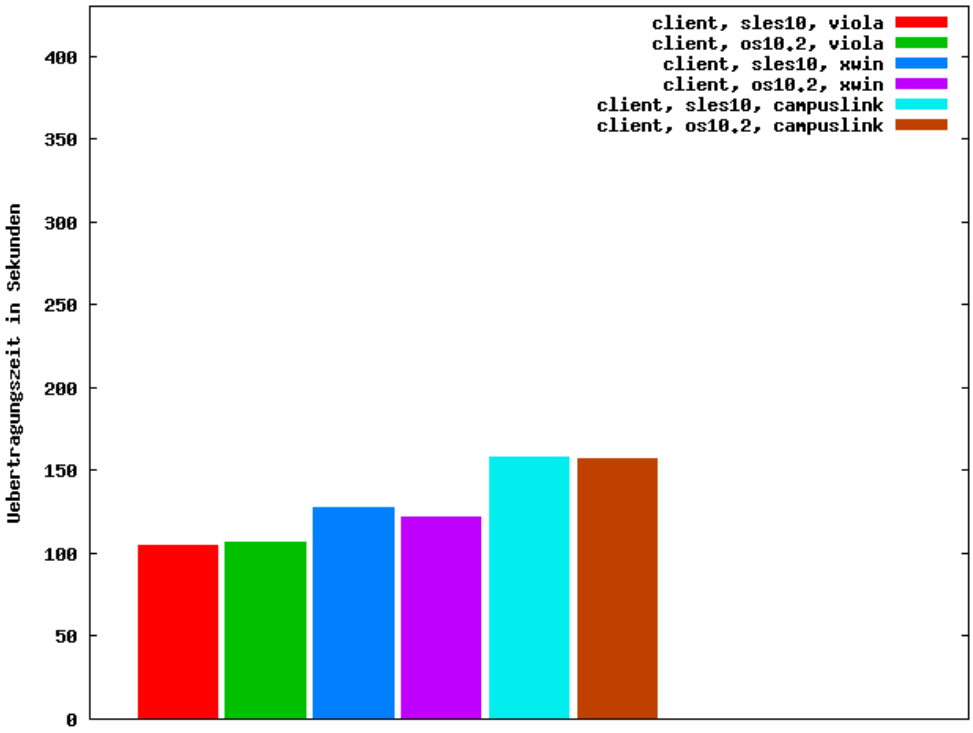

Abbildung 38: KoDaVis mit UDT, Vergleich verschiedener Netzwerkszenarien

5 Zusammenfassung und Schlussfolgerungen

Die vorliegenden Messungen zeigen einen deutlichen Unterschied zwischen einem öffentlichen, von Vielen gleichzeitig genutzten, und einem abgeschlossenen, dediziert geschalteten Netzwerk. Die diesem Bericht zugrunde liegenden Datenübertragungen erreichten bei den Messungen über das VIOLA-Testbed annähernd Wirespeed, mit Ausnahme der TCP-Variante TCP VEGAS. Die in einem öffentlichen Netz vorhandenen Strukturen, mit zahlreichen aktiven Komponenten wie Switches, Firewalls und Router beeinflussten die Messergebnisse in mehrerlei Hinsicht negativ:

- Selbst unbelastete Komponenten „verlängern“ die Leitung, was zu einer Erhöhung der RTT und damit einer Vergrößerung der Latenz kleiner Nachrichtengrößen führt (Faustregel: 1 ms pro Router/Switch).
- Komponenten unter Last induzieren Jitter, der oftmals die sehr fein auf die RTT abgestimmten verschiedenen TCP-Timeouts (z.B. RTO – Retransmit Time Out) triggert oder aus dem Takt bringt und somit die Durchsatzrate reduziert.
- Queues von Komponenten unter Last neigen zu Paketverlust, welcher dazu führt, dass TCP den Durchsatz reduziert. Dies bedeutet insbesondere für Übertragungen großer Nachrichten (> 100 Kbyte) eine Erhöhung der Übertragungsdauer, bzw. aus Applikationssicht: der Latenz.

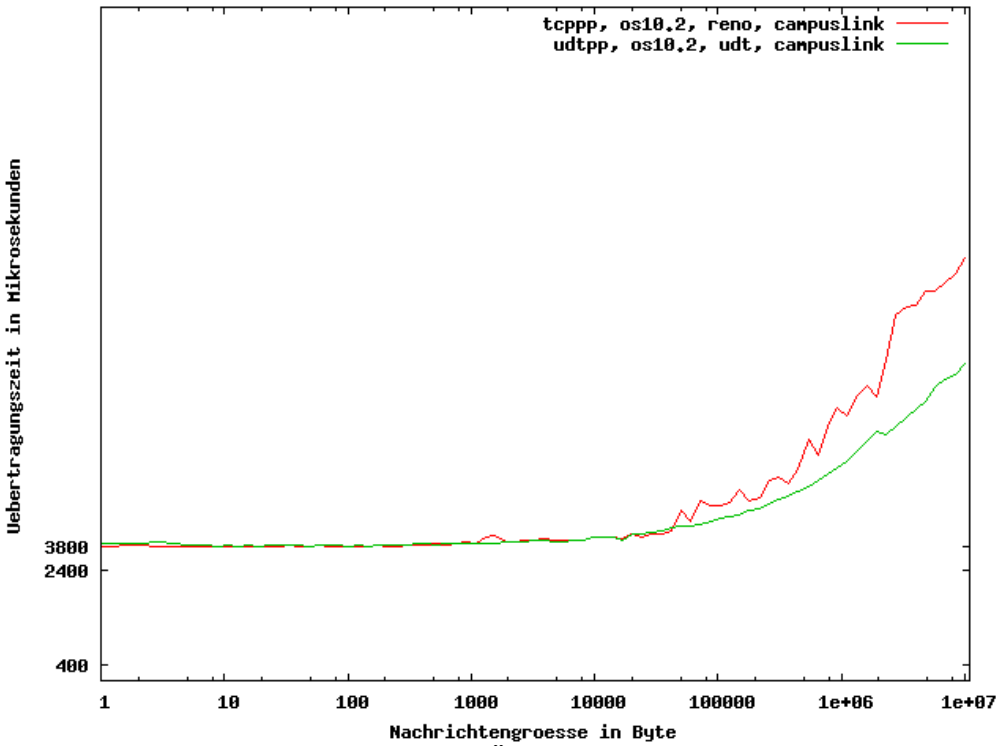

Abbildung 39: tcppp/udtpp, Vergleich Übertragungszeiten TCP-UDT

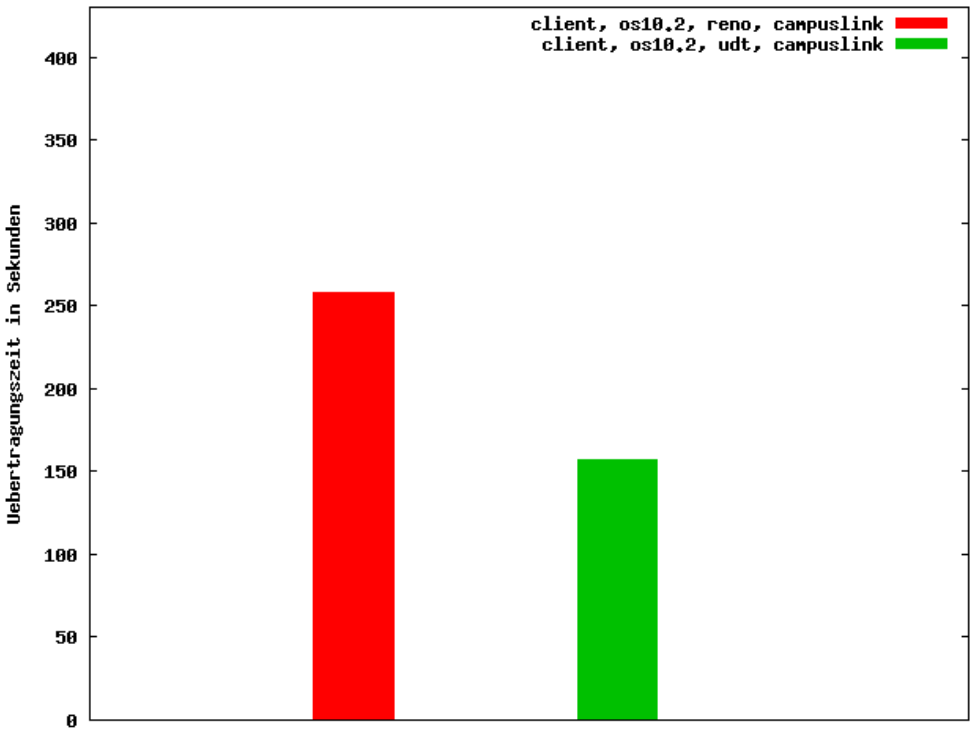

Abbildung 40: KoDaVis, Vergleich Übertragungszeit TCP-UDT

Abbildung 39 und Abbildung 40 zeigen jeweils deutlich, wie die sehr agressive, bzw. kaum vorhandene Default Congestion Control von UDT zu einer wesentlich besseren Durchsatzrate führt; dies allerdings klar zu Lasten der TCP-Fairness.

In den Untersuchungen zeigte sich, dass der Einfluss der Campusnetze mir ihrer lokalen Infrastruktur inklusive Firewall im Vergleich deutlich größer ist als der des Weitverkehrsnetz, sowohl im X-WIN-Szenario mit gemeinsam genutzer Bandbreite als auch in der annähernd idealen Situation im VIOLA-Testbed.