Trafficshaping

Bandbreitenmanagement und Latenzzeiten

Andreas Hoffmann

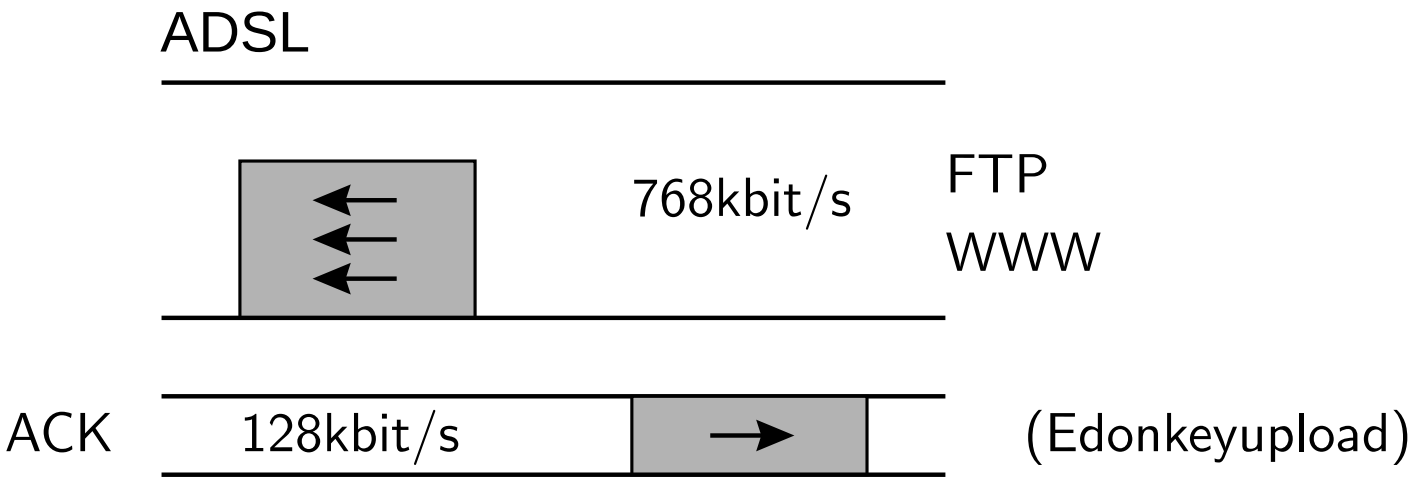
Praktische Anwendungen

- Priorisierung spezieller Verkehrsarten (z. B. http gegenüber ftp)
- Latenzzeiten bei interaktivem Verkehr verbessern (ssh)
- ADSL-typische Beienträchtigung des Downstreams durch den Upstream verhindern
- Beschränkung von Verbindungen
  - Simulation verschiedener Internetanbindungen (Webseiten testen)
  - Internetzugänge verschiedener Geschwindigkeit anbieten
- Lastverteilung über mehrere Anschlüsse
- ...

Privates Interesse

- Einbruch des Downstreams bei ausgelastetem Upstream

  Problem: Beim TCP-Protokoll muß jedes Paket mit einem ACK-Paket bestätigt werden.

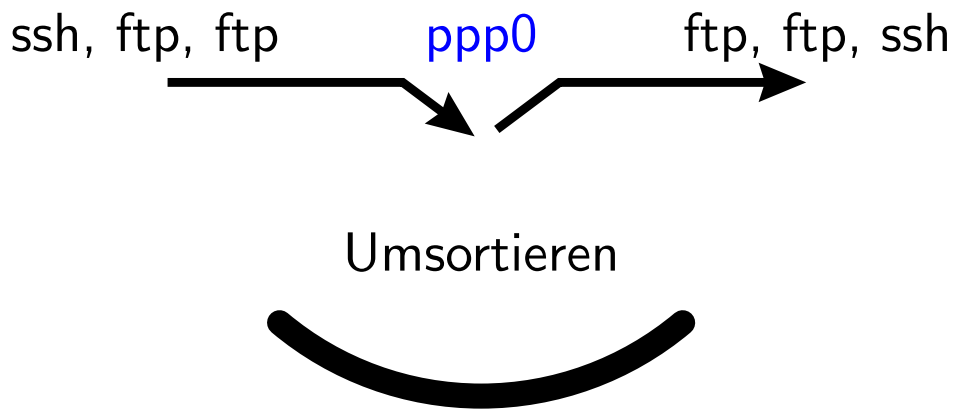


- Latenzzeiten von SSH-Verbindungen unerträglich lange bei gleichzeitigem Upload/Download

  Problem: Pakete müssen durch Puffer im ADSL/ISDN-Modem und beim Provider

Grundlegendes

Geregelt wird wie Pakete gesendet werden.



Pakete sollen z.B. über ein ppp0 Interface ins Internet.

Sie werden jetzt **erst** in eine Queue eingespeist und umsortiert.

Erst dann werden sie dem ppp0-Device zum aussenden überlassen.

Grundlegendes

Es läßt sich also nur Verkehr über ein Device regeln, bei dem man selbst den Verkehr kontrollieren kann.

- Nur **ausgehende** Pakete sind in vollem Umfang regelbar.
- **Eingehende** Pakete lassen sich aber in ihrer Rate drosseln.
- Es gibt ein sog. IMQ-Device, das man hinter den Eingang klemmen kann.

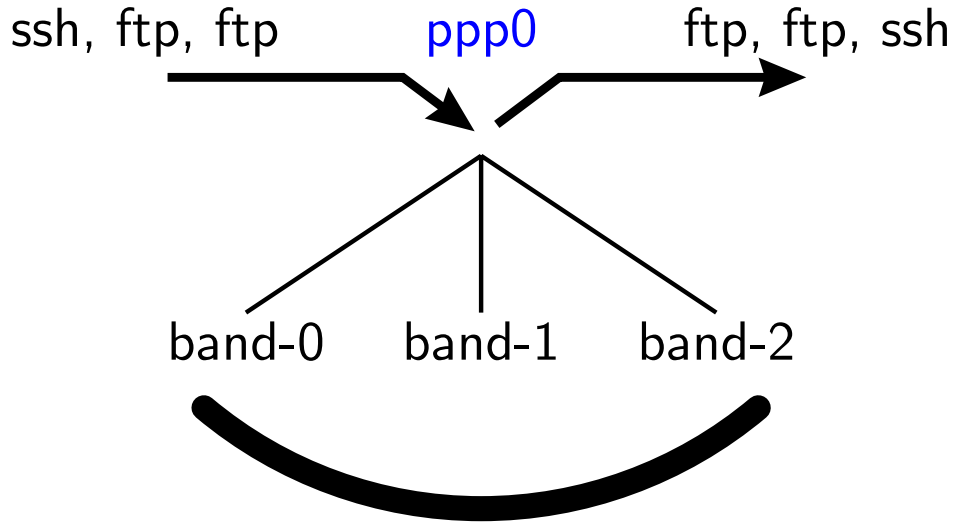
  Achtung: Erfordert Patches für den Kern, "iptables"-Ziele im Kern und iptables selbst.

Queueing Disciplines(Qdiscs)

PFIFO_FAST, nicht konfigurierbarer Default, 3 Bänder.

Mit einem Type of Service (TOS) Bit kann eine Anwendung ein Paket z.B. als eilig markieren.

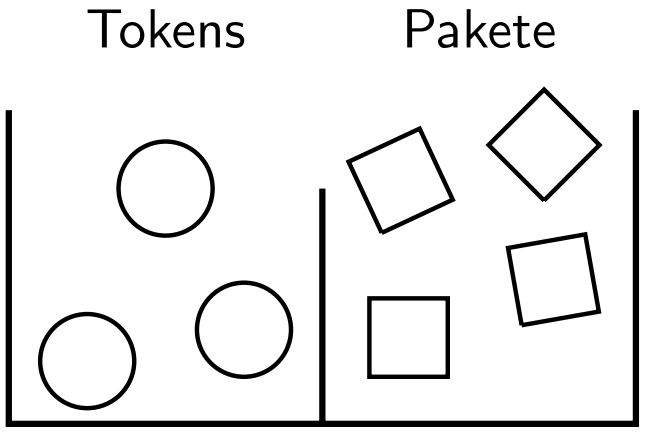
Einsortiert werden die Pakete nach der sog. Priomap basierend auf TOS-Bits.



Niedrigere Bänder haben immer Vorrang, in jedem Band gilt FIFO.

Weitere Beispiele

TBF (Token Bucket Filter)

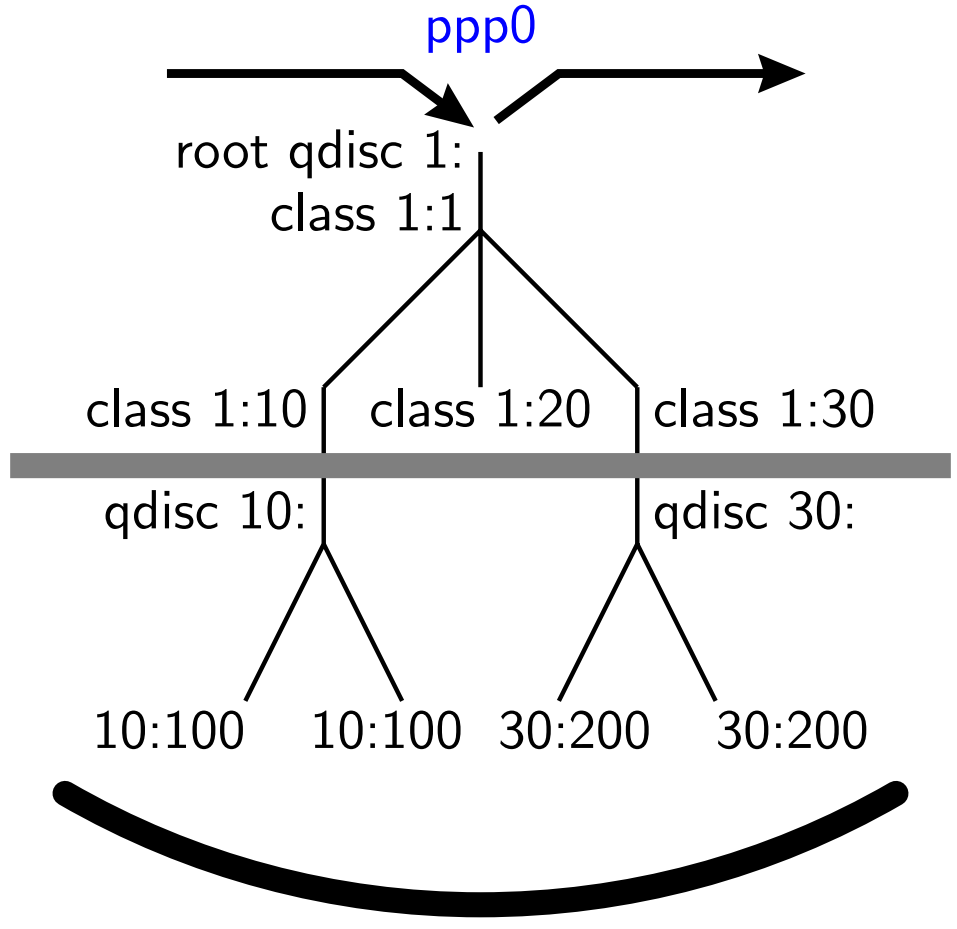


lartc: "A simple but *very* useful configuration is this:"

```
# tc qdisc add dev ppp0 root tbf rate 118kbit \
          latency 50ms burst 1540
```

`rate`: Rate, mit der Tokens eingeworfen werden
`burst`: Größe des Tokeneimers - Instantan verfügbare Tokens
`latency`: Verweildauer eines Pakets in der Schlange

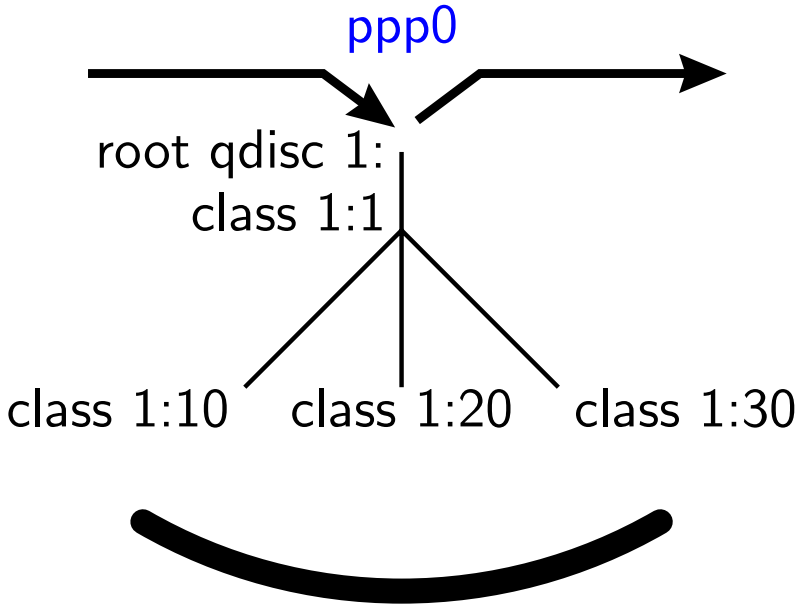
Klassenbehaftete Qdiscs



Erweiterung der Bänder von **PFIFO_FAST** zu selbst konfigurierbaren Klassen, zusammen mit Ratenregelung.

Qdiscs können Klassen enthalten und Klassen können Klassen oder Qdiscs enthalten.

Klassenbehaftete Qdiscs



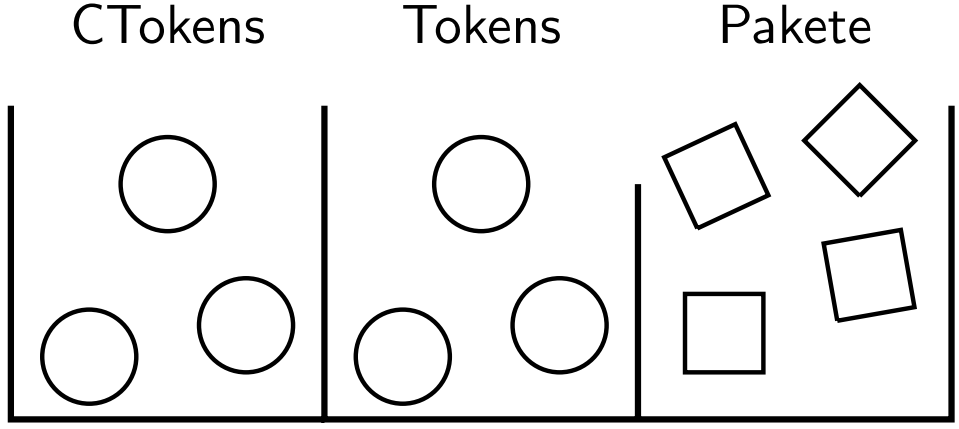
Ausgesendet werden Pakete von Qdiscs. Schneidet man, wie oft, den Baum an der grauen Linie ab, so erhalten 1:10, 1:20, 1:30 jeweils eine **FIFO**-Qdisc, wenn man nichts besseres (**SFQ**) angibt.

HTB Hierachical Token Buckets

- Klassenbehaftete Qdisc, Erweiterung von **TBF**: Einfacher und angebl. performanter als **CBQ**.
- Klassen teilen sich freie Kapazitäten

Neu bei **HTB**:

`ceil`: Zweiter Tokeneimer mit CTokens
`prio`: Wie bei **PFIFO_FAST** wird eine Klasse mit kleinerem `prio`-Wert zuerst bedient.



Wie bekommt man Pakete in Klassen?

Beispiel: SSH-Pakete in Klasse 1:10 einsortieren

1. Möglichkeit: `fw`-Filter

(a) Markieren mit iptables (`handle 11`)

```
iptables -A POSTROUTING -t mangle -o ppp0 \
         -p tcp -dport 22 -j MARK -set-mark 11
```

(b) Mittels `handle 11` und `fw`-Filter einsortieren

```
tc filter add dev ppp0 parent 1:0 protocol ip \
         prio 0 handle 11 fw flowid 1:10
```

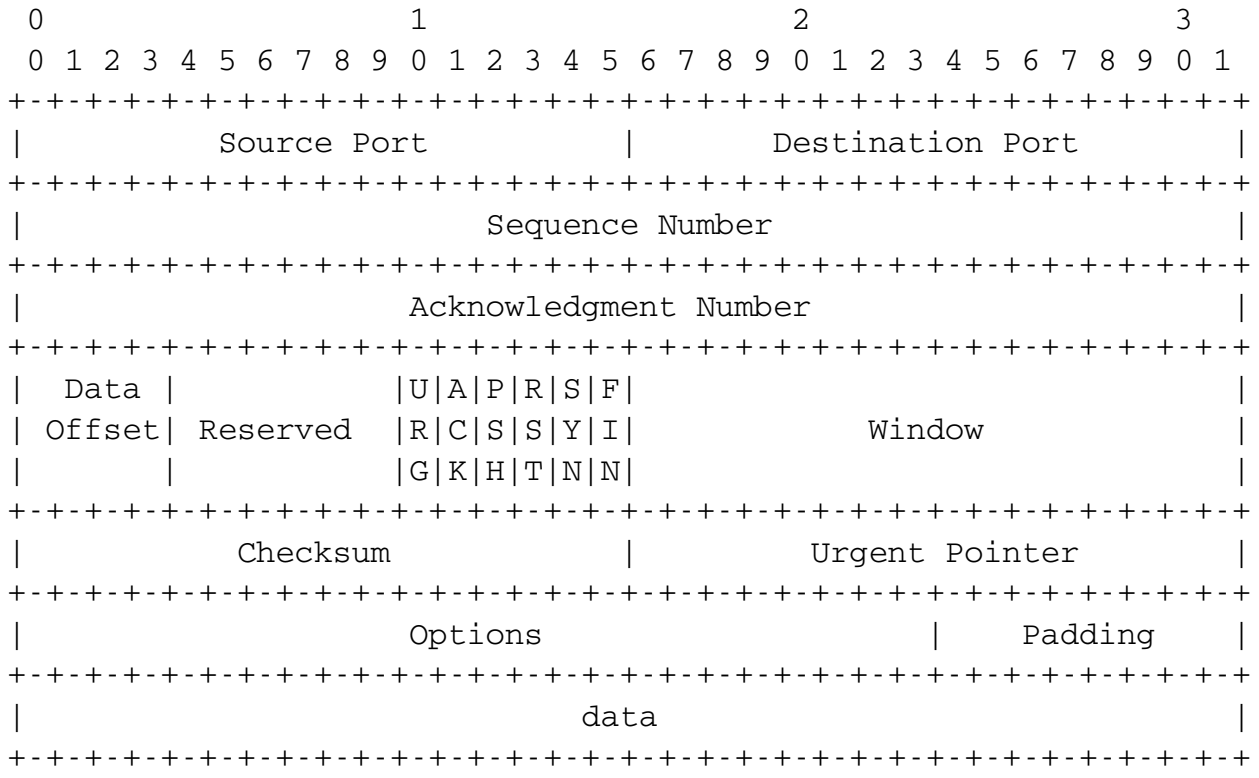
zusätzlich wird diese Sortierregel mit höchster Priorität 0 getestet.

Es lassen sich hiermit sogar die Pakete eines Nutzers markieren

```
iptables -A POSTROUTING -t mangle -o ppp0 \
         -m owner -uid-owner andi -j MARK \
         -set-mark 11
```

Wie bekommt man Pakete in Klassen?

2. Möglichkeit: u32-Filter



32 Bit
2 * 8 Bit Src-Port
2 * 8 Bit Dst-Port

Beispiel SSH-Paket: Port=22 entspricht 00 00 00 16 in Hex

```
# tc filter add dev ppp0 protocol ip parent 1:0 prio 0 \
u32 match u32 00000016 0000ffff at nexthdr+0 flowid 1:10
```

DSL-Rezepte

Geht auf allen halbwegs aktuellen Kernen:

- `tc qdisc add dev ppp0 root tbf rate 118kbit \ latency 50ms burst 1540`

  Zur Verbesserung der Latenzzeiten

Ab Kern 2.4.20 oder mit gepatchten Kernen:

- Wondershaper von `http://www.lartc.org`

```
DOWNLINK=750
UPLINK=118
DEV=ppp0
# low priority source ports
NOPRIOPORTSRC="4662 1214"
# low priority destination ports
NOPRIOPORTDST="4662 1214"
```

Verbessert auch Surfen bei gleichzeitigem Download. Nicht auf DSL beschränkt.

Quellen

- Dokumentation:
  - c’t Artikel 24/2002, S. 224
  - `http://www.lartc.org`
  - `http://www.docum.org`
  - Dort erwähnte Links zu "HTB, CBQ, IMQ, ..."
  - `http://tcng.sourceforge.net`
- Mailinglisten:
  - `http://www.lartc.org/#mailinglist`
- Voraussetzungen:
  - Implementiert seit Kernel 2.2.x
  - HTB3 erst ab Kernel 2.4.20
  - `tc` aus iproute2, für HTB3 Version 2.4.7

Persönliches Fazit

- Sehr wirksame Vorlagen im Netz erhältlich einfach in z.B. `/etc/ppp/ip-up.d/` kopieren, anpassen, klappt!
- Sehr mächtig
- IMQ-Device noch nicht im Standardkern
- Handhabung teils trickreich