Quality of Service in IP Networks

by

Harald Welte <laforge@gnumonks.org>

Contents

- Definition of QoS
- Why QoS
- IP Networks are not designed for QoS
- How to do the impossible
- What can Linux based systems help
- Advanced Concepts (DiffServ, IntServ, RSVP, ...)
- References / Further Reading

Definiton of QoS

- Provide Service Differentiation
- Performance Assurance by
  - Bandwitdh guarantees
    - for streaming multimedia traffic
    - priorizing certain important applications
  - Latency guarantees
    - for voice over IP
    - for interactive character-oriented applications (ssh,telnet)
  - Packet-loss guarantees
    - for unreliable layer-4 protocols
    - to avoid retransmits

Why QoS

- Decide how and who available bandwidth is devided
- Limit available bandwidth for certain users / applications
- Guarantee bandwidth for certain users / applications
- Divide bandwidth more equally between users / applications

IP networks not designed for QoS

Properties of IP-based networks:

- offer a "best-effort" service
- make NO guarantees about
  - bandwidth
  - latency
  - packet loss
- provide a non-reliable packet transport

Conclusion: IP networks are not suitable for QoS

How to do the Impossible

As IP Networks including Hardware (Routers, ...) are widely deployed, all QoS efforts have to layer on top of the existing technology.

- There’s no real solution to control latency
  - latency widely dependent on routing, which may be dynamic
- There’s no real solution to control packet loss
  - packet loss may occurr on any intermediate router
- But we can control bandwidth usage!
  - The sender can limit bandwidth for outgoing streams
  - Intermediate routers BEFORE a bottleneck can control bandwidth usage

What can Linux systems do?

- Bandwidth limiting at the sender application
  - not many applications support it
  - server often out of control (on Internet, ...)
  - server doesn't know what's between him and the client
- Bandwidth control on intermediate router before bottleneck
  - Ideal case because this is where packet loss would occurr
  - Sophisticated queue scheduling on the outgoing queue
  - Variety of different queue scheduling algorithms
- Flow throttling at the Receiver
  - Worst case, because influence is limited
  - Theoretically possible for TCP, no implementation yet.
  - Ingress qdisc might help

Bandwidth limiting at server

- Some Internet Servers support bandwidth limiting
  - ProFTPd (builtin support)
  - Apache (using contributed mod_bandwidth)
- Using those features it is easy to limit
  - maximum bandwidth used per connection
  - maximum bandwidth used per client (IP/network)
  - maximum bandwidth used by one virtual host (webserver/ftpserver)

Router before bottleneck

The router receives more packets on his incoming interface(s) than it can send out on the outgoing interface. It has to build a queue of packets (usually a FIFO one) and starts dropping packets as soon as the queue is full