Networking and protocols for real-time signal transmissions

by Hans-Peter Schwefel

- Mm1 Introduction & simple performance models (HPS)
- Mm2 Real-time Support in Wireless Technologies (HPS)
- Mm3 Transport Layer Aspects and Header Compression (HPS)
- Mm4 IP Quality of Service: Advanced Concepts (HPS)
- Mm5 Session Signalling and Application Layer/Codecs (SVA)

hps@kom.auc.dk
http://www.kom.auc.dk/~hps

Background: IP Protocol Stack

Network Layer (Layer 3):
- Internet Protocol (IPv4, IPv6)
- Packet (IP datagram) transmission between end-systems [hosts] (packet size up to 65535 bytes, often restricted by Layer 2 protocols)
- Routing using 32 bit addresses (v4)
- Unreliable, connectionless transmission: loss, duplication, reordering can occur

Transport Layer (Layer 4):
- RTP provides mechanisms for synchronization/merging
- RTCP provides basic QoS monitoring
- But missing mechanisms for provisioning of QoS

→ Network Layer Task
Motivation: Quality of Service

• Advantages of Packet-Based Transport (as opposed to circuit switched)
  – Flexibility
  – Optimal Use of Link Capacities, Multiplex-Gain for bursty traffic

• Drawbacks
  – Buffering/Queueing at routers can be necessary
  – Delay / Jitter / Packet Loss can occur
  – Overhead from Headers (20 Byte IPv4, 20 Byte TCP)

→ Protocol Improvements for Real-Time Applications necessary
  – Packet Prioritization (DiffServ, TypeOfService field in IPv4)
  – Resource Reservation (IntServ)
  – Traffic Engineering (QoS Routing, static/dynamic MPLS paths)
  – Connection Admission / Traffic Policing / Shaping
  – Header Compression

Quality of Service: Scope / Parameters

• User Plane QoS
  – End-2-End Packet Delay (in particular interactive applications)
  – Delay Jitter
  – Packet Loss
  – Throughput/Goodput

• Application Level QoS
  – e.g. Video/Voice Quality (depending on codecs)

• Signalling Plane
  – Call Setup Delays
  – Fraction of blocked Calls

• Behavior at Handover
  – Dropped Calls
  – Delayed / Lost packets

• Reliability Aspects
  – Failure probabilities of entities
  – Downtime distribution

Main Focus here
Realtime Applications: VoIP, Video, etc.
See MM5 of WNII
Over-Provisioning

- Design network to be able to deal with worst-case traffic scenario
- Advantage:
  - no impact on architecture, protocols and user equipment
  - simplicity
- Problems:
  - Traffic depends on number of active users, user mobility, type of application, daily utilization profile \(\rightarrow\) difficult forecasting
  - Data traffic tends to be very bursty (even `self-similar´)
    \(\rightarrow\) waste of resources if planned for worst-case scenario
    \(\rightarrow\) can be very expensive
  - Unforeseeable events can occur (new applications; changes in user behavior, e.g. always-on)
Content

1. Introduction/Motivation
   • Network QoS, QoS Parameters
2. Overprovisioning
3. Integrated Services
   • Architecture & functionalities
   • QoS Signalling
4. Differentiated Services
   • Architecture, Functionalities
   • Buffer Management: RED
5. Traffic Engineering
   • QoS Routing
   • Multi-Protocol Label Switching (MPLS)
6. Summary
7. Use Case: QoS for Gaming Applications

Integrated Services (IntServ) / RSVP

• Fundamental Idea: Reserve necessary resources for each traffic flow along its transmission path, which requires:
  – Connection Admission Control (CAC): traffic specification + info about available resources at router → admission decision (if no, then re-routing)
  – Packet Classification: which flow does it belong to?
  – Packet Scheduling: make sure, flow obtains resources as specified
**IntServ: functionalities**

- Connection/Call Admission Control (CAC)
  - Easy for constant bit rate (CBR) flows
  - Difficult tasks for bursty traffic
  
  Alternatives:
  - Peak-Rate Allocation → no multiplex gain
  - Mean-rate allocation → large delays and losses possible
  - Intermediate solution: effective bandwidths
    Frequently based on limit theorems
    - Large deviations theory
    - High multiplex degrees

- Packet Classifier
  - Flow specifications by so-called 'filters'
  - Specifies ranges of value for L3/L4 header fields

- Packet Scheduler
  - Multiple queues
  - Scheduling principles: WFQ, strict priority, EDF, ...
  - In addition: buffer management, e.g. RED (see later)

**IntServ: Signalling, RSVP**

- Signalling by Resource Reservation Protocol (RSVP)
  - Path Message: sender initiated, description of traffic parameters (Tspec) and path
  - Resv Message: receiver initiated, causes connection admission/reservation along path; specifies QoS parameters (Rspec)
  - Other messages for reservation teardown and error treatment

  Principles
  - In-path signalling
  - Multi-cast support
  - Soft-State concept: periodic refresh of reservation required

  **Advantages:**
  - Fine Granularity: per flow treatment, flexible set of QoS parameters
  - Able to provide QoS guarantees (if admission, classification, scheduling is performed correctly)

  **Disadvantages**
  - Scalability problem: management of state for each single flow
  - Complexity (already connection admission can be complex, e.g. effective bandwidths, etc.)
**IntServ: RSVP Messages I**

- **Path Message**
  - **Tspec:** Traffic specification
    - Token Bucket Rate
    - Token Bucket Size
    - Peak Data Rate
    - Minimum Policed Unit
    - Packet Size
  - **Adspec:** Network Resources on Path
    - Non QoS Hop-count
    - Available Path Bandwidth
    - Minimum Path Latency
    - Path MTU
  - **Sender Template:** Filter Specification
    - IP source address, protocol type, port number, etc.
  - Previous router on path

**IntServ: RSVP Messages II**

- **Resv Message**
  - Next hop in path (receiver → sender)
  - Flow-spec
    - Tspec
    - Rspec
  - List of Filter Specs (description of sender for which the reservation is intended)
  - Reservation style
    - Wildcard filter: shared, one reservation for all senders
    - Fixed filter: distinct, one per sender
    - Shared explicit: one reservation for specified list of senders
RSVP Messages: FlowSpec (Controlled Load)

<table>
<thead>
<tr>
<th>31</th>
<th>24 23</th>
<th>16 15</th>
<th>8 7</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0 (a)</td>
<td>reserved</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>5 (c)</td>
<td>0</td>
<td>reserved</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>127 (e)</td>
<td>0 (f)</td>
<td>5 (g)</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Token Bucket Rate [r] (32-bit IEEE floating point number)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Token Bucket Size [b] (32-bit IEEE floating point number)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Peak Data Rate [p] (32-bit IEEE floating point number)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Minimum Policed Unit [m] (32-bit integer)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Maximum Packet Size [M] (32-bit integer)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(a) - Message format version number (3)
(b) - Overall length (7 words not including header)
(c) - Service header, service number 5 (Controlled-Load)
(d) - Length of controlled-load data, 6 words not including per-service header
(e) - Parameter 127, parameter 127 (Token Bucket TSpec)
(f) - Parameter 127 flags (none set)
(g) - Parameter 127 length, 5 words not including per-service header

Reservation Types:

• Guaranteed Service: Bandwidth and Delay Guarantees, No Loss
• Controlled Load: Only Bandwidth Guarantee

Additional Issues in QoS signalling

- Inter-domain signaling
- Off-path signaling
- Arbitrary placement of initiator and receiver
- Bi-directional signaling/ sender-initiated signaling
- Mobility support
- Implementation size & complexity (~own transport protocol on top of IP, multicast support, etc.)
- How to secure RSVP in a real-world environment
QoS signalling scenarios

- State kept at more than two entities.
- Protocol requires interaction with other protocols (routing, security, AAA, mobility, etc.)

Content

1. Introduction/Motivation
   - Network QoS, QoS Parameters
2. Overprovisioning
3. Integrated Services
   - Architecture & functionalities
   - QoS Signalling
4. Differentiated Services
   - Architecture, Functionalities
   - Buffer Management: RED
5. Traffic Engineering
   - QoS Routing
   - Multi-Protocol Label Switching (MPLS)
6. Summary
7. Use Case: QoS for Gaming Applications
**Differentiated Services (DiffServ)**

- Basic idea: reduce queueing delay/loss for critical traffic by preferential treatment at routers (multiple queues)
  → improve per-hop transmission behavior
- Packets marked by DiffServ Code Points (DSCPs, 6bit)
- Various scheduling disciplines at routers possible (e.g. static priority, weighted fair queueing)
- Advantage: Simple and scalable
- Problem: No performance guarantees unless used in conjunction with connection admission and traffic shaping/policing at ingress routers

**DiffServ Code Points (DSCP)**

- **Differentiated Services (DS) Byte**
  - **Per Hop Behaviour**
  - **Currently Unused**
- **IPv4**
- **IPv6**

---

**SLA**: Service Level Agreement
**DiffServ: Influencing QoS**

- At Border Routers
  - Traffic classification and Marking → DiffServ Class (e.g. EF, AF, BE)
  - Traffic Policing/Shaping/Conditioning, e.g.
    - (Token) Leaky Bucket
    - Time-sliding window 3 color marker
      - Two thresholds: Committed Information Rate, Peak Information Rate
      - Time-sliding window for measurement of average rate
        \[ \Lambda_t = \frac{\Lambda_{t-1} \Delta + \sigma}{\Delta + \left( t_t - t_{t-1} \right)} \]
  - At Interior DiffServ Router
    - Scheduling: Strict Priority, Weighted Fair Queueing, etc.
    - Buffer Management, e.g. Random Early Drop RED, RIO
      - Possibly different drop precedence

---

**Buffer Management: RED**

- Packet loss due to congestion (full buffers)
  - Bursty loss
  - Abrupt occurrence (no direct 'warning' previously, except for increasing RTTs) → difficult for adaptivity (TCP rate throttling, adaptive applications)
- Modified Buffer Strategy: Random Early Drop (RED)
  - Based on weighted average queue-length
    - Weighting factor 0≤w≤1 (e.g. w=0.002)
  - Two thresholds, Qₗ<Qₜ, for average queue-length
  - Linearly increasing (0,...,pₘ) packet discarding probability between thresholds
    \[ p(q) = \begin{cases} 
    0 & \text{for } q ≤ Q_l \\
    \frac{p_m}{Q_h - Q_l} \cdot (q_l - q) & \text{for } Q_l < q ≤ Q_h \\
    1 & \text{for } q > Q_h 
  \end{cases} \]
  - Early indication of congestion via single packet drops
Model of RED queues

- Bottleneck RED queue model
  - CBR arrivals, deterministic with rate $\lambda$
  - Deterministic service rate $\mu$

- Queue-length behavior at packet arrival instances using fluid-flow service approximation:
  - $A_i()$ is random variable describing number of arriving packets, i.e. $A_i()=0$ with discarding probability $p()$.
  - $\lambda > \mu$
  - $\lambda \neq \mu$
  - $\lambda = \mu$
  - $\lambda < \mu$

\[
\begin{align*}
\text{Stochastic process:} & \\
q_{i+1}^+ & = \left( q_i^+ - \frac{\mu}{\lambda} \right) + A_i(\tilde{q}_i^+) \\
q_{i+1}^- & = (1-w)q_i^- + w\left[q_{i+1}^-\right].
\end{align*}
\]

- Deterministic approximation, replacing $A_i()$ by its expected value

\[
\begin{align*}
\text{Deterministic RED:} & \\
q_{i+1}^+ & = \left( q_i^+ - \frac{\mu}{\lambda} \right) + 1 - p\left(q_i^-\right) \\
q_{i+1}^- & = (1-w)q_i^- + w\left[q_{i+1}^-\right].
\end{align*}
\]

\[\hat{q} = \lim_{i\to\infty} q_i^+ = \lim_{i\to\infty} q_i^- = \frac{\lambda - \mu}{\mu} Q_h - Q_l + Q_0.\]

RED Queue: Behavior in Overload ($\lambda > \mu$)

- Two different behavior patterns
  - Oscillations, for $w < 4p_{\text{low}}/(Q_h - Q_l)$.
  - Exponential convergence towards steady-state limit

- No convergence of probabilistic models in oscillating case

\[
\text{Convergence time [seconds]} = \begin{cases} 
\frac{1}{2\pi} & \text{for oscillations} \\
\log_{10}(0.5) & \text{for exponential convergence}
\end{cases}
\]

- No convergence of probabilistic models in oscillating case
  - Consequence of probabilistic discarding decision on packet level
1. Introduction/Motivation
   • Network QoS, QoS Parameters
2. Overprovisioning
3. Integrated Services
   • Architecture & functionalities
   • QoS Signalling
4. Differentiated Services
   • Architecture, Functionalities
   • Buffer Management: RED
5. Traffic Engineering
   • QoS Routing
   • Multi-Protocol Label Switching (MPLS)
6. Summary
7. Use Case: QoS for Gaming Applications

Traffic Engineering (TE)

• TE = distribute traffic over network links in order to avoid congestion
• IP routing (OSPF, IS-IS, RIP, etc.)
  – Based on destination IP address
  – No possibility for distinguishing traffic classes
  – Link costs normally statically assigned (sometimes even hop-count used)
  – Modification of link costs possible, but implications on link utilizations not straightforward
• Alternatives
  – QoS Routing: Use QoS parameters for path selection
  – Establishment of explicit paths
    • Automatically
    • Via network management
    • Approaches:
      • Tunneling: e.g. L2TP, PPP
      • Multi-Protocol Label Switching (MPLS)
**Traffic Engineering: Time-scales**

- In traditional use: traffic engineering and configuration of link costs for routing done via network management
  - time-scales of minutes to hours (in best case)
- Shorter time-scales (flow-level, even packet level) via extensions, e.g. QoS routing

**QoS routing: Steps**

- Add QoS relevant information to link state advertisements (in addition to static link costs and connectivity relation)
- Two approaches
  - Source Routing: Compute full routes
  - Distributed Routing: determine next hop
- Modify path metrics using QoS parameters
- Constraint-based routing: eliminate certain paths not meeting constraints (e.g. Minimal bandwidth req.)

BUT: increased complexity, path selection in some cases np-complete!
QoS routing: example

- OSPF-TE: LSAs advertise
  - Cost
  - Residual bandwidth
  - Delay of links
- Widest-shortest path algorithm:
  - Among all paths with sufficient bandwidth
  - choose among those with the lowest hop count
  - If there are several feasible paths with identical hop count, choose the one with the highest residual bandwidth.
- Computation of routing tables at each node using a modified Bellman-Ford algorithm

Multi-Protocol Label Switching: Principles

- Border Router (ingress) assigns label to packet
  - e.g. based on protocol type, DSCP, QoS demands, etc.
- Forwarding within MPLS domain based on assigned label
- Necessary: establishment of Label Switched Paths (LSPs), using signalling protocols
  - CR-LDP
  - RSVP-TE
- Egress router removes label
### MPLS Encapsulation

**Label Stack Entry Format**

- **Label**: Label Value, 20 bits (0-16 reserved)
- **Exp.**: Experimental, 3 bits (was Class of Service)
- **S**: Bottom of Stack, 1 bit (1 = last entry in label stack)
- **TTL**: Time to Live, 8 bits

- Network layer must be inferable from value of bottom label of the stack
- TTL must be set to the value of the IP TTL field when packet is first labeled
- When last label is popped off stack, MPLS TTL to be copied to IP TTL field
- Pushing multiple labels may cause length of frame to exceed layer-2 MTU

**MPLS on LAN links uses ‘Shim’ Header Inserted Between Layer 2 and Layer 3 Headers**

### Dynamic MPLS paths for aggregated traffic

- Dynamic MPLS path setup using source routing
- Question:
  - Trigger events?
  - Requirements/QoS parameters for new path?
Summary

1. Introduction/Motivation
   • Network QoS, QoS Parameters
2. Overprovisioning
3. Integrated Services
   • Architecture & functionalities
   • QoS Signalling
4. Differentiated Services
   • Architecture, Functionalities
   • Buffer Management: RED
5. Traffic Engineering
   • QoS Routing
   • Multi-Protocol Label Switching (MPLS)

6. Summary

7. Use Case: QoS for Gaming Applications

Topics not treated here

• Mapping of QoS Parameters (Applic➔L3➔L2➔PHY)
• Coupling of Mobility support and QoS
• Service Level Agreements (SLAs)
• End-to-end QoS signalling in UMTS (SIP/IMS) (MOB 9)
7. Use-Case: Quality of Service for Gaming Applications --
an Experimental Evaluation of DiffServ

Based on
Semester Project by Erling V. Matthiesen, Jakob K. Larsen, and
Flemming Olufsen
Supervisors: H. Schiøler, HP Schwefel
Aalborg University, Spring 2004

Gaming
Types of Games
– Real Time Strategy
– Massively Multiplayer Online Role Playing Game
– First Person Shooter (FPS)
  • Considered here:
    Counter-Strike
Content (Sect. 7)

7.1 QoS Requirements
- QoS Parameters
- User Satisfaction

7.2 Traffic Analysis/Modeling
- Inter-packet times, packet sizes
- Markov models

7.3 QoS Solution
- Candidates
- Experimental Setup

7.4 Experimental Results
- Measurement Methodology, Results, Discussion

7.5 Summary

Counter-Strike: QoS Requirements

- High requirements on
  - Delay (RTT<60ms)
  - Jitter

- Medium Requirements on
  - Packet Loss (<3%)
  - Bandwidth consumption rather low (22 players<1Mb/s)
Content (Sect. 7)

7.1 QoS Requirements
- QoS Parameters
- User Satisfaction

7.2 Traffic Analysis/Modeling
- Inter-packet times, packet sizes
- Markov models

7.3 QoS Solution
- Candidates
- Experimental Setup

7.4 Experimental Results
- Measurement Methodology, Results, Discussion

7.5 Summary

Game Traffic

- Traffic Measurements of Counter-Strike Session
  - Architecture:
    - 2 clients, 1 server
    - Within switched 100Mb/s Ethernet
  - Measurement of UDP traffic
    - Inter-packet time distribution (inter-departure time)
    - Packet Size distribution
    - Correlation properties (auto-, cross-)
  - ... for four different flows
    - Flow 1: Client 1 → Server
    - Flow 2: Client 2 → Server
    - Flow 3: Server → Client 1
    - Flow 4: Server → Client 2
Inter-departure time distribution

- Upstream: two modes, around 33 ms and 50 ms
- Downstream: single mode, around 60 ms

Packet size distribution

<table>
<thead>
<tr>
<th>Flow</th>
<th>Mean packet size</th>
<th>Standard deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flow 1</td>
<td>43.47 bytes</td>
<td>8.81 byte</td>
</tr>
<tr>
<td>Flow 2</td>
<td>46.32 bytes</td>
<td>8.56 byte</td>
</tr>
<tr>
<td>Flow 3</td>
<td>54.50 bytes</td>
<td>76.29 byte</td>
</tr>
<tr>
<td>Flow 4</td>
<td>56.10 bytes</td>
<td>81.15 byte</td>
</tr>
</tbody>
</table>
Modeling of Gaming Traffic

- Downstream (server → client)
  - i.i.d. normally distributed
- Upstream (client → server)
  - Mix of two normal distributions
  - Normal distributions fitted with the mean and standard deviation of the traffic sample
  - Selection of ‘short’ or ‘long’ inter-departure time determined by discrete time Markov chain
    - Transition matrix derived from traffic samples
      \[
      P_{\text{low}} = \begin{bmatrix}
      \text{slow} & \text{slow} & \text{slow} \rightarrow \text{fast} \\
      \text{fast} & \text{slow} & \text{fast} \rightarrow \text{fast}
      \end{bmatrix} = \begin{bmatrix}
      0.712406 & 0.28557 \\
      0.350855 & 0.649145
      \end{bmatrix}
      \]
      \[
      P_{\text{low2}} = \begin{bmatrix}
      \text{slow} & \text{slow} & \text{slow} \rightarrow \text{fast} \\
      \text{fast} & \text{slow} & \text{fast} \rightarrow \text{fast}
      \end{bmatrix} = \begin{bmatrix}
      0.136926 & 0.863074 \\
      0.209484 & 0.788296
      \end{bmatrix}
      \]

Auto-correlation of inter-packet times

Coefficient of auto-correlation
  → Seemingly some periodic properties in game traffic
  • Not achieved (of course) by two-state Markov model!
Content (Sect. 7)

7.1 QoS Requirements
- QoS Parameters
- User Satisfaction

7.2 Traffic Analysis/Modeling
- Inter-packet times, packet sizes
- Markov models

7.3 QoS Solution
- Candidates
- Experimental Setup

7.4 Experimental Results
- Measurement Methodology, Results, Discussion

7.5 Summary

Candidates for QoS support

- Traffic Engineering / Multi-Protocol Label Switching (MPLS)
  - Not supported by SW version of our Cisco Routers
- CISCO IP Multi-layer Switching (IP-MLS)
  - Reduced processing times, but otherwise no QoS support
- Resource Reservation Protocol (RSVP)
  - Available in routers but rather complex
- Differentiated Services (DiffServ)
  - Gaming Traffic in EF class
  - Considered in the project: mapping to two scheduling mechanisms
    - Low latency Queueing
    - Weighted fair queueing (see table)

<table>
<thead>
<tr>
<th>Precedence name</th>
<th>Value</th>
<th>Relative share of bandwidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best effort</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Class 1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Class 2</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Class 3</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>Class 4</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Expedited forwarding</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>Inter-protocol control</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>Network control</td>
<td>7</td>
<td>8</td>
</tr>
</tbody>
</table>
**Weighted Fair Queuing**

- Strict Priority for some classes, WFQ for others
- Configured for each interface
- Enabled by: `fair-queue 64 256 0`
  - Defines 256 queues
  - 64 packets long
- Disabled by: `no fair-queue`
- Router config.: see table

<table>
<thead>
<tr>
<th>Router</th>
<th>Type</th>
<th>Interface</th>
<th>QoS</th>
</tr>
</thead>
<tbody>
<tr>
<td>R_1</td>
<td>Classifier/router</td>
<td>FastEthernet0/0</td>
<td>WFQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Serial1/2</td>
<td>WFQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Serial1/3</td>
<td>WFQ</td>
</tr>
<tr>
<td>R_2</td>
<td>router</td>
<td>Serial1/2</td>
<td>WFQ</td>
</tr>
<tr>
<td>R_3</td>
<td>router</td>
<td>FastEthernet0/0</td>
<td>WFQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Serial1/2</td>
<td>WFQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Serial1/1</td>
<td>WFQ</td>
</tr>
<tr>
<td>R_4</td>
<td>Classifier/router</td>
<td>FastEthernet0/0</td>
<td>WFQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Serial0/0</td>
<td>WFQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Serial0/1</td>
<td>WFQ</td>
</tr>
</tbody>
</table>

**Low Latency Queuing**

- Mixture of strict priority and WFQ scheduling
- Requires CEF (Express Forwarding) to be enabled
- Enabled by: `priority 300 1000`
  - Gives a maximum throughput of 300kbit/s
  - Enables a burst of 1000 bytes exceeding the guaranteed bandwidth
  - Bandwidth is only restricted if congestion occurs
- Disabled by: `no priority`

<table>
<thead>
<tr>
<th>Router</th>
<th>Type</th>
<th>Interface</th>
<th>Class</th>
<th>BW [Kbit/s]</th>
<th>Max Burst [bytes]</th>
</tr>
</thead>
<tbody>
<tr>
<td>R_1</td>
<td>Classifier/</td>
<td>FastEthernet0/0</td>
<td>CBWFQ</td>
<td>300</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>router</td>
<td>Serial1/2</td>
<td>CBWFQ</td>
<td>300</td>
<td>1000</td>
</tr>
<tr>
<td>R_2</td>
<td>router</td>
<td>Serial1/2</td>
<td>CBWFQ</td>
<td>300</td>
<td>1000</td>
</tr>
<tr>
<td>R_3</td>
<td>router</td>
<td>FastEthernet0/0</td>
<td>CBWFQ</td>
<td>300</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>router</td>
<td>Serial1/2</td>
<td>CBWFQ</td>
<td>300</td>
<td>1000</td>
</tr>
<tr>
<td>R_4</td>
<td>classifier/</td>
<td>FastEthernet0/0</td>
<td>CBWFQ</td>
<td>300</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>router</td>
<td>Serial0/0</td>
<td>CBWFQ</td>
<td>300</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>router</td>
<td>Serial0/1</td>
<td>CBWFQ</td>
<td>300</td>
<td>1000</td>
</tr>
</tbody>
</table>
Content (Sect. 7)

7.1 QoS Requirements
- QoS Parameters
- User Satisfaction

7.2 Traffic Analysis/Modeling
- Inter-packet times, packet sizes
- Markov models

7.3 QoS Solution
- Candidates
- Experimental Setup

7.4 Experimental Results
- Measurement Methodology, Results, Discussion

7.5 Summary

Approach

- Routers configured via telnet
- Cisco configuration CLI used
- Scripts to enable and disable features developed

Traffic Generation
- Own implementation of server and client traffic generator
- cross traffic (CBR) generated with Iperf

Data Collection
- Traffic dumps taken with tcpdump
- Delay extracted with awk-scripts
- Statistics done with matlab
**Experiment Execution**

- Conducted in the IP Laboratory
- Equipment
  - 3 Cisco Routers 3620
  - 1 Cisco Router 3631
  - 2 Cisco Catalyst Switches 2950

---

**Experiment Execution: Small Setup**

[Diagram showing network setup with labels: S_1, S_2, R_1, Proxy, Game server, Game client1, Game client2, Cross traffic generator1, Cross traffic generator2.]
Experiment Execution: Large Setup

Experiment Results

- Experiments
  - Small setup, no crosstraffic
  - Small setup, 6M crosstraffic
  - Large setup, no crosstraffic
  - Large setup, 2M crosstraffic
  - Large setup, 6M crosstraffic
  - Large setup, 8M crosstraffic, CEF enabled
### Experiment Results: small setup, no cross-traffic

#### Downstream one-way packet delays

<table>
<thead>
<tr>
<th>QoS</th>
<th>Mean [ms]</th>
<th>Dev [ms]</th>
<th>Loss [%]</th>
<th>CrossLoss [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIFO</td>
<td>1,409</td>
<td>0,752</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>WFQ</td>
<td>1,562</td>
<td>0,610</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>LLQ</td>
<td>1,611</td>
<td>0,599</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

### Experiment Results: Large setup

6Mb/s cross-traffic

<table>
<thead>
<tr>
<th>QoS</th>
<th>Mean [ms]</th>
<th>Dev [ms]</th>
<th>Loss [%]</th>
<th>CrossLoss [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIFO</td>
<td>36,80</td>
<td>30,00</td>
<td>0,08</td>
<td>0,08</td>
</tr>
<tr>
<td>WFQ</td>
<td>174,80</td>
<td>36,10</td>
<td>17,00</td>
<td>8,28</td>
</tr>
<tr>
<td>LLQ</td>
<td>120,40</td>
<td>61,90</td>
<td>7,62</td>
<td>1,35</td>
</tr>
</tbody>
</table>

8Mb/s cross-traffic, CEF enabled,

<table>
<thead>
<tr>
<th>QoS</th>
<th>Mean [ms]</th>
<th>Dev [ms]</th>
<th>Loss [%]</th>
<th>CrossLoss [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIFO</td>
<td>0,61</td>
<td>0,46</td>
<td>0,00</td>
<td>2,50</td>
</tr>
<tr>
<td>WFQ</td>
<td>1,10</td>
<td>0,24</td>
<td>0,00</td>
<td>2,50</td>
</tr>
<tr>
<td>LLQ</td>
<td>1,10</td>
<td>0,21</td>
<td>0,00</td>
<td>2,50</td>
</tr>
</tbody>
</table>
**Experiment Results: large setup (II)**

![Graph showing experiment results](attachment:image.png)

**Conclusion**

- Delay (<30ms) and loss sensitive (<3%) game traffic
- Traffic model for game traffic by 2-state discrete Markov chain (determining parameters of a normally distributed inter-packet times)
- Experimental evaluation of DiffServ QoS concept
  - Game traffic in EF class, cross-traffic in BE class
  - WFQ and LLQ scheduling strategy
  - W/o cross-traffic: WFQ and LLQ increase processing delays → larger delays but still below required threshold
  - With cross-traffic: large delays and loss rates for game traffic in particular for WFQ, LLQ
    - → careful QoS configuration necessary (CEF enabling needed)
- Additional evaluation scenarios would need to be investigated
Acknowledgements

• Student work (all TU Munich)
  – Raimund Brandt (Seminar)
  – Sunhwei Shen (Seminar)
  – Stefan Rank (Master Thesis)

  www.control.auc.dk/~04gr832b/

• Presentations of H. Schulzrinne (Columbia University) and H. Tschofenig (Siemens Corporate Technologies)

• InfotechLecture notes: IP Based Networks and Applications, Chapter 3 (J. Charzinski), www.jcho.de/jc/IPNA

• Tutorial: IP Technology in 3rd Generation mobile networks, Siemens AG (J. Kross, L. Smith, H. Schwefel)