Better-than-best-effort: QoS, Int-serv, Diff-serv, RSVP, RTP

Shivkumar Kalyanaraman
Rensselaer Polytechnic Institute
shivkuma@ecse.rpi.edu
http://www.ecse.rpi.edu/Homepages/shivkuma
Based in part on slides of Jim Kurose, Srinivasan Seshan, S. Keshav

Why better-than-best-effort (QoS-enabled) Internet?
Quality of Service (QoS) building blocks
End-to-end protocols: RTP, H.323,
Network protocols:
- Integrated Services(int-serv), RSVP.
- Scalable differentiated services for ISPs: diff-serv
- Control plane: QoS routing, traffic engineering, policy management, pricing models

Overview

Why Better-than-Best-Effort (QoS)?
- To support a wider range of applications
  - Real-time, Multimedia etc
- To develop sustainable economic models and new private networking services
  - Current flat priced models, and best-effort services do not cut it for businesses

Quality of Service: What is it?
Multimedia applications: network audio and video
network provides application with level of performance needed for application to function.

What is QoS?
- “Better performance” as described by a set of parameters or measured by a set of metrics.
- Generic parameters:
  - Bandwidth
  - Delay, Delay-jitter
  - Packet loss rate (or probability)
- Transport/Application-specific parameters:
  - Timeouts
  - Percentage of “important” packets lost

What is QoS (contd)?
- These parameters can be measured at several granularities:
  - “micro” flow, aggregate flow, population.
- QoS considered “better” if
  - a) more parameters can be specified
  - b) QoS can be specified at a fine-granularity.
- QoS spectrum:

<table>
<thead>
<tr>
<th>Best Effort</th>
<th>Leased Line</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
In a FIFO service discipline, the performance assigned to one flow is convoluted with the arrivals of packets from all other flows!

Cant get QoS with a “free-for-all”

Need to use new scheduling disciplines which provide “isolation” of performance from arrival rates of background traffic

Fundamental Problems

Scheduling Discipline

FIFO

Irrespective of scheduling discipline chosen:
- Average backlog (delay) is constant
- Average bandwidth is constant
- Zero-sum game => need to “set-aside” resources for premium services

QoS Components

- QoS => set aside resources for premium services
- QoS components:
  - a) Specification of premium services: (Service/SLA design)
  - b) How much resources to set aside? (admission control/provisioning)
  - c) How to ensure network resource utilization, do load balancing, flexibly manage traffic aggregates and paths? (QoS routing, traffic engineering)

How to upgrade the Internet for QoS?

- Approach: de-couple end-system evolution from network evolution
- End-to-end protocols: RTP, H.323 etc to spur the growth of adaptive multimedia applications
  - Assume best-effort or better-than-best-effort clouds
- Network protocols: Intserv, Diffserv, RSVP, MPLS, COPS ...
  - To support better-than-best-effort capabilities at the network (IP) level
Mechanisms: Queuing/Scheduling

- Use a few bits in header to indicate which queue (class) a packet goes into (also branded as CoS).
- High $$$ users classified into high priority queues, which also may be less populated.
  - lower delay and low likelihood of packet drop
- Ideas: priority, round-robin, classification, aggregation.

Mechanisms: Buffer Mgmt/Priority Drop

- Drop RED and BLUE packets
- Drop only BLUE packets
  - Ideas: packet marking, queue thresholds, differential dropping, buffer assignments.

Mechanisms: Traffic Shaping/Policing

- Token bucket: limits input to specified Burst Size (b) and Average Rate (r).
- Traffic sent over any time $T < rT + b$
  - a.k.a Linear bounded arrival process (LBAP)
- Excess traffic may be queued, marked BLUE, or simply dropped.

Focus: Scheduling Policies

- Priority Queuing: classes have different priorities; class may depend on explicit marking or other header info, eg IP source or destination, TCP Port numbers, etc.
  - Transmit a packet from the highest priority class with a non-empty queue.
  - Preemptive and non-preemptive versions.

Generalized Processor Sharing (GPS)

- Assume a fluid model of traffic
  - Visit each non-empty queue in turn (RR)
  - Serve infinitesimal from each
  - Leads to “max-min” fairness
  - GPS is un-implementable!
  - We cannot serve infinitesimals, only packets.
**Bit-by-bit Round Robin**
- Single flow: clock ticks when a bit is transmitted.
  - For packet $i$:
    - $P_i$ = length, $A_i$ = arrival time, $S_i$ = begin transmit time, $F_i$ = finish transmit time
    - $F_i = S_i + P_i = \max\{F_{i-1}, A_i\} + P_i$
- Multiple flows: clock ticks when a bit from all active flows is transmitted
- Can calculate $F_i$ for each packet if number of flows is known at all times
- This can be complicated

**Fair Queuing (FQ)**
- Mapping bit-by-bit schedule onto packet transmission schedule
- Transmit packet with the lowest $F_i$ at any given time
- Variation: Weighted Fair Queuing (WFQ)

**FQ Example**

**Putting it together: Parekh-Gallager theorem**
- Let a connection be allocated weights at each WFQ scheduler along its path, so that the least bandwidth it is allocated is $g$
- Let it be leaky-bucket regulated such that # bits sent in time $[t_1, t_2] \leq g(t_2 - t_1) + \sigma$
- Let the connection pass through $K$ schedulers, where the $k$th scheduler has a rate $r(k)$
- Let the largest packet size in the network be $P$

\[ \text{end-to-end delay} \leq \sigma / g + P / g + \sum_i P / r(k) \]

**Significance**
- P-G Theorem shows that WFQ scheduling can provide end-to-end delay bounds in a network of multiplexed bottlenecks
- WFQ provides both bandwidth and delay guarantees
- Bound holds regardless of cross traffic behavior (isolation)
- Needs shapers at the entrance of the network
- Can be generalized for networks where schedulers are variants of WFQ, and the link service rate changes over time

**Integrated Services (intserv)**
- An architecture for providing QOS guarantees in IP networks for individual application sessions
- Relies on resource reservation, and routers need to maintain state information of allocated resources (eg: $g$) and respond to new Call setup requests
**Signaling semantics**
- Classic scheme: sender initiated
- **SETUP, SETUP_ACK, SETUP_RESPONSE**
- Admission control
- Tentative resource reservation and confirmation
- Simplex and duplex setup: no multicast support

**RSVP: Internet Signaling**
- Creates and maintains distributed reservation state
- De-coupled from routing:
  - Multicast trees setup by routing protocols, not RSVP (unlike ATM or telephony signaling)
  - Receiver-initiated: scales for multicast
  - Soft-state: reservation times out unless refreshed
- Latest paths discovered through “PATH” messages (forward direction) and used by RESV mesgs (reverse direction).

---

**Call Admission**
- Session must first declare its QoS requirement and characterize the traffic it will send through the network
- **R-spec**: defines the QoS being requested
- **T-spec**: defines the traffic characteristics
- A signaling protocol is needed to carry the R-spec and T-spec to the routers where reservation is required; RSVP is a leading candidate for such signaling protocol

**Differentiated Services (diffserv)**
- Intended to address the following difficulties with Intserv and RSVP:
  - **Scalability**: maintaining states by routers in high speed networks is difficult due to the very large number of flows
  - **Flexible Service Models**: Intserv has only two classes, want to provide more qualitative service classes; want to provide ‘relative’ service distinction (Platinum, Gold, Silver, …)
  - **Simpler signaling**: (than RSVP) many applications and users may only want to specify a more qualitative notion of service.

---

**Differentiated Services Model**
- **Edge routers**: traffic conditioning (policing, marking, dropping), SLA negotiation
  - Set values in DS-byte in IP header based upon negotiated service and observed traffic.
- **Interior routers**: traffic classification and forwarding (near stateless core!)
  - Use DS-byte as index into forwarding table
**Diffserv Architecture**

**Edge router:**
- per-flow traffic management
- marks packets as in-profile and out-profile

**Core router:**
- per class TM
- buffering and scheduling based on marking at edge
- preference given to in-profile packets
- Assured Forwarding

**Packet format support**

- Packet is marked in the Type of Service (TOS) in IPv4, and Traffic Class in IPv6: renamed as “DS”
- 6 bits used for Differentiated Service Code Point (DSCP) and determine PHB that the packet will receive
- 2 bits are currently unused

**Traffic Conditioning**

- It may be desirable to limit traffic injection rate of some class; user declares traffic profile (e.g., rate and burst size); traffic is metered and shaped if non-conforming

**Per-hop Behavior (PHB)**

- PHB: name for interior router data-plane functions
- Includes scheduling, buffer management, shaping etc.
- Logical spec: PHB does not specify mechanisms to use to ensure performance behavior
- Examples:
  - Class A gets x% of outgoing link bandwidth over time intervals of a specified length
  - Class A packets leave first before packets from class B

**PHB (contd)**

- PHBs under consideration:
  - **Expedited Forwarding**: departure rate of packets from a class equals or exceeds a specified rate (logical link with a minimum guaranteed rate)
  - Emulates leased-line behavior
  - **Assured Forwarding**: 4 classes, each guaranteed a minimum amount of bandwidth and buffering; each with three drop preference partitions
  - Emulates frame-relay behavior

**End-to-end: Real-Time Protocol (RTP)**

- Provides standard packet format for real-time applications
- Typically runs over UDP
- Specifies header fields below
- **Payload Type**: 7 bits, providing 128 possible different types of encoding; e.g., PCM, MPEG2 video, etc.
- **Sequence Number**: 16 bits; used to detect packet loss
Real-Time Protocol (RTP)

- **Timestamp**: 32 bytes; gives the sampling instant of the first audio/video byte in the packet; used to remove jitter introduced by the network
- **Synchronization Source identifier (SSRC)**: 32 bits; an id for the source of a stream; assigned randomly by the source

RTP Control Protocol (RTCP)

- Protocol specifies report packets exchanged between sources and destinations of multimedia information
- Three reports are defined: Receiver reception, Sender, and Source description
- Reports contain statistics such as the number of packets sent, number of packets lost, inter-arrival jitter
- Used to modify sender transmission rates and for diagnostics purposes

End-to-end Adaptive Applications

- Video Coding, Error Concealment, Unequal Error Protection (UEP)
- Packetization, Marking, Source Buffer Management
- Congestion control

Video Coding, Error Concealment, Unequal Error Protection (UEP)
Packetization, Marking, playout Buffer Management Congestion control

Internet
End-to-end Closed-loop control

Eg: Streaming & RTSP

- User interactive control is provided, e.g. the public protocol Real Time Streaming Protocol (RTSP)
- Helper Application: displays content, which is typically requested via a Web browser; e.g. RealPlayer; typical functions:
  - Decompression
  - Jitter removal
  - Error correction: use redundant packets to be used for reconstruction of original stream
  - GUI for user control

Using a Streaming Server

- Web browser requests and receives a Meta File (a file describing the object)
- Browser launches the appropriate Player and passes it the Meta File;
- Player contacts a streaming server, may use a choice of UDP vs. TCP to get the stream

Web Server
(1) HTTP request-response for presentation description file
Meta Player
(3) audio/video file requested and sent
Web Browser
(1) HTTP request-response for presentation description file
(2) presentation description file
Streaming Server
receiver

Receiver Adaptation Options

- If UDP: Server sends at a rate appropriate for client; to reduce jitter, Player buffers initially for 2-5 seconds, then starts display
- If TCP: sender sends at maximum possible rate; retransmit when error is encountered; Player uses a much large buffer to smooth delivery rate of TCP

Web Server
user buffer
fill rate = n/10i
sink rate = n/10i
no decompression employed
prohibited
video

Web Browser
stream network
fill rate = n/10i
sink rate = n/10i
no decompression employed
prohibited
video

Web Server
user buffer
fill rate = n/10i
sink rate = n/10i
no decompression employed
prohibited
video

Web Browser
stream network
fill rate = n/10i
sink rate = n/10i
no decompression employed
prohibited
video
**H.323**
- H.323 is an ITU standard for multimedia communications over best-effort LANs.
- Part of a larger set of standards (H.32X) for videoconferencing over data networks.
- H.323 includes both stand-alone devices and embedded personal computer technology as well as point-to-point and multipoint conferences.
- H.323 addresses call control, multimedia management, and bandwidth management as well as interfaces between LANs and other networks.

**Network Core: Traffic Engineering**
- Performance optimization of operational networks.
- Traffic-oriented: meet QoS of flows.
- Resource-oriented: optimization of network resource utilization.
  - Minimize overall congestion.
  - Maximize overall utilization.
  - Control over routing.

**Control Plane: MPLS**
- Provides a framework for routing evolution.
- De-couples forwarding from routing control.
- Explicit routing.
- Constraint-based (QoS) routing, load-balancing.
- Traffic engineering: aggregating traffic flows into trunks, and mapping them onto pre-defined paths.
- Provides a framework for integrating IP, ATM, and frame-relay cores.
- Allows re-engineering of the ATM control plane, and the IP forwarding plane.

**MPLS: Building Blocks**
- Label: short, fixed length field.
- Carrying label in header:
  - Use VCI/VPI or DLCI in ATM or FR.
  - New “shim” header for other link layers.

**MPLS: Building Blocks (Continued)**
- Forwarding table structure:
  - Incoming label + subentry = outgoing label, outgoing interface, next-hop address (will include PHBs for diff-serv).
- Forwarding algorithm: Label swapping.
  - Use label as an index (exact match).
MPLS: Building Blocks (Continued)

- **Control component:**
  - Responsible for distributing routing & label-binding information: extensions to routing protocols, RSVP, LDP

MPLS Traffic Engineering

- Load balancing, explicit (constraint-based) routing
- Avoids limitations of destination-based forwarding
- Allows mapping of traffic into hierarchically aggregatable trunks (LSPs)

Virtual Private Networks with MPLS

- MPLS encapsulation provides opaque tunneling support for VPNs
- Security and performance (QoS) attributes can then be assigned to such tunnels (LSPs)

COPS

- Common Open Policy Service
- Initially designed for adding policy control to RSVP
- Now being extended to support provisioning
- Uses TCP; stateful exchange; common object model

Open problems: Multi-Provider Internetwork QoS

New approach: Edge-based building blocks
Closed-loop QoS Building Blocks

- **Scheduler**: Differentiates service on a packet-by-packet basis
- **Loops**: Differentiates service on an RTT-by-RTT basis using purely edge-based policy configuration.

QoS: an application-level approach

- Sophisticated services in application
- Architecturally "above" network core
- Open services: Let 1000 flowers bloom

QoS: an application-level approach

Application-level infrastructure
- Accommodate network-level service
- Additional tailoring of user services

Content Delivery: motivation

- Reduces load on server
- Avoids network congestion

Content Delivery: idea

- Reduces load on server
- Avoids network congestion

Content Delivery: congestion
CDN: Architectural Layout

- Publisher informs RR of Content Availability.
- Content Pushed to Distribution System.
- Client Requests Content, Requested redirected to RR.
- RR finds the most suitable Surrogate
- Surrogate services client request.

Summary

- QoS big picture, building blocks
- Integrated services: RSVP, 2 services, scheduling, admission control etc
- Diff-serv: edge-routers, core routers; DS byte marking and PHBs
- Real-time transport/middleware: RTP, H.323
- Traffic Engineering, MPLS, COPS
- Open problems: deployment of inter-domain QoS, Application-level QoS, Content delivery/web caching