Quality of Service
EECS 122: Lecture 15

Policing at the edge of the network controls the amount of traffic the network layer has to allocate.

Scheduling in conjunction with packet dropping control performance within a router.

Scheduling mechanisms determine how the bandwidth of an output port is shared:
- Mainly used to manage delay
- Signaling allows for flows.

Model of router queues
Flow Set up

- Flow signals to the network its
  - Statistics (e.g. 24Mb/s constant bit rate)  
    - Traffic Spec Service Level requirement
    - Max, average, 95% level etc.
  - If the network cannot fulfill the requirement it rejects the flow
    - May suggest a change
  - If the network can fulfill the requirement it reserves resources to carry the flow traffic and accepts the flow

Token Buckets
Admission Control
Resource Reservation

Today

End to End QoS

- Network Layer: Multiple routers
  - Intserv
  - Diffserv

- Application Layer
  - Adaptive Playback Buffers
    - Streaming
    - Voice
IETF Integrated Services

- architecture for providing QoS guarantees in IP networks for individual application sessions
- resource reservation: routers maintain state info (a la VC) of allocated resources, QoS req’s
- admit/deny new call setup requests:

**Question:** can newly arriving flow be admitted with performance guarantees while not violated QoS guarantees made to already admitted flows?

Intserv: QoS guarantee scenario

- **Resource reservation**
  - call setup, signaling (RSVP)
  - traffic, QoS declaration
  - per-element admission control

- QoS-sensitive scheduling (e.g., WFQ)
Call Admission

Arriving session must:
- declare its QoS requirement
  - R-spec: defines the QoS being requested
- characterize traffic it will send into network
  - T-spec: defines traffic characteristics
- signaling protocol: needed to carry R-spec and T-spec to routers (where reservation is required)
  - RSVP (Resource Reservation Protocol): will cover this when we discuss multicast

Intserv QoS: Service models [rfc2211, rfc 2212]

Guaranteed service:
- worst case traffic arrival: leaky-bucket-policed source
- simple (mathematically provable) bound on delay [Parekh 1992, Cruz 1988]

Controlled load service:
- “a quality of service closely approximating the QoS that same flow would receive from an unloaded network element.”

\[ D_{\text{max}} = \frac{b}{R} \]
Intserv Example

- **Goal:** achieve per-flow bandwidth and delay guarantees

Step 1: Ask Permission...

- Example: achieve per-flow bandwidth and delay guarantees

Sender sends $T_{spec}$, $R_{spec}$
Step 2: Establish Path

- RSVP Signaling Protocol

Sender

Receiver

RSVP Signaling Protocol
Path established

Per-flow state on all routers in path

What about DATAGRAM routing?

Step 3: Reserve buffer resources

- Configure router queues

Sender

Per-flow state on all routers in path

What about DATAGRAM routing?

Receiver
Step 4: Traffic Flows

- We will discuss joins later...

Traffic Flows
Traffic Flows

Per-flow scheduling on each router

Token Bucket + WFQ = Delay Bound

- No packet has a queueing delay of more than $b/R$ seconds
IETF Differentiated Services

Concerns with Intserv:
- **Scalability**: signaling, maintaining per-flow router state difficult with large number of flows
- **Flexible Service Models**: Intserv has only two classes. Also want “qualitative” service classes
  - “behaves like a wire”
  - relative service distinction: Platinum, Gold, Silver

Diffserv approach:
- simple functions in network core, relatively complex functions at edge routers (or hosts)
- Don’t define service classes, provide functional components to build service classes

DiffServ Architecture

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

**Core router**:
- per class traffic management
- buffering and scheduling based on marking at edge
- preference given to in-profile packets
- Assured Forwarding
**Edge-router Packet Marking**

- **profile:** pre-negotiated rate A, bucket size B
- **packet marking at edge based on per-flow profile**

Possible usage of marking:
- **class-based marking:** packets of different classes marked differently
- **intra-class marking:** conforming portion of flow marked differently than non-conforming one

**Classification and Shaping**

may be desirable to limit traffic injection rate of some class:
- **user declares traffic profile** (e.g., rate, burst size)
- traffic metered, shaped if non-conforming
**Forwarding: Per Hop Behaviors (PHBs)**

- Try to control QoS per router hop rather than end to end.
- PHB specifies observable (measurable) forwarding performance behavior
  - E.g. Don’t any packets of class 11 by more than 20ms.
- PHB does not specify what mechanisms to use to ensure required PHB 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

---

**Classification and Conditioning**

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

PHBs being developed:

- **Expedited Forwarding**: pkt departure rate of a class equals or exceeds specified rate
  - logical link with a minimum guaranteed rate

- **Assured Forwarding**: 4 classes of traffic
  - each guaranteed minimum amount of bandwidth
  - each with three drop preference partitions

### Comparison

<table>
<thead>
<tr>
<th></th>
<th>Best-Effort</th>
<th>DiffServ</th>
<th>Intserv</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service scope</td>
<td>End-to-end</td>
<td>Domain</td>
<td>End-to-end</td>
</tr>
<tr>
<td>Complexity</td>
<td>No setup</td>
<td>Long term setup</td>
<td>Per flow setup</td>
</tr>
<tr>
<td>Scalability</td>
<td>Highly scalable (nodes maintain only routing state)</td>
<td>Scalable (edge routers maintain per aggregate state, core routers per class state)</td>
<td>Not scalable (each router maintains per flow state)</td>
</tr>
</tbody>
</table>
Multimedia Over Today’s Internet

TCP/UDP/IP: “best-effort service”
- **no** guarantees on delay, loss

But you said multimedia apps require **QoS and level of performance to be effective**!

Today’s Internet multimedia applications use application-level techniques to mitigate (as best possible) effects of delay, loss

Example: Video

- e.g. 10fps, 600x500 bytes per frame $\approx 24\text{Mb/s}$

Simplify by assuming that Camera sends at a fixed rate

Nick McKeown
Router Effect on Delay

Router Effects on Traffic
Network Effects on Traffic

Cumulative Bits

Source

Router 1

Router n

Delay's do not build up independently in each router

Svc function at router 1 is arrival function at router 2
Network Effects on Traffic

Cumulative Bits vs. Time

- Source
- Router n
- bits in the network
- delay

Network Effect on Delay

Prob vs. Delay/latency

- Delay variation or Jitter
- e.g. 200ms
- Min
- 99%

Nick McKeown
Smooth out non-interactive traffic with a playback buffer

Cumulative Bits vs. Time

Source -> Destination

Playout buffer

Playback Point

Nick McKeown
Playback point adapts

The Playout buffer can grow quickly!

1. Has a maximum instantaneous rate of $\leq 100\text{Mb/s}$
2. Has an average rate of $\leq 24\text{Mb/s}$
3. Has a short-term average rate of $\leq 45\text{Mb/s}$
Streaming Stored Multimedia

Application-level streaming techniques for making the best out of best effort service:
- client side buffering
- use of UDP versus TCP
- multiple encodings of multimedia

Streaming over HTTP

- browser GETs metafile
- browser launches player, passing metafile
- player contacts server
- server streams audio/video to player
Streaming from a streaming server

This architecture allows for non-HTTP protocol between server and media player.
- Can also use UDP instead of TCP.

Streaming Multimedia: Client Buffering

- Client-side buffering, playout delay compensate for network-added delay, delay jitter.
Streaming Multimedia: Client Buffering

- Client-side buffering, playout delay compensate for network-added delay, delay jitter

Streaming Multimedia: UDP or TCP?

**UDP**
- server sends at rate appropriate for client (oblivious to network congestion !)
  - often send rate = encoding rate = constant rate
  - then, fill rate = constant rate - packet loss
- short playout delay (2-5 seconds) to compensate for network delay jitter
- error recover: time permitting

**TCP**
- send at maximum possible rate under TCP
- fill rate fluctuates due to TCP congestion control
- larger playout delay: smooth TCP delivery rate
- HTTP/TCP passes more easily through firewalls
Streaming Multimedia: client rate(s)

Q: how to handle different client receive rate capabilities?
- 28.8 Kbps dialup
- 100Mbps Ethernet
A: server stores, transmits multiple copies of video, encoded at different rates

User Control of Streaming Media:
RTSP

HTTP
- Does not target multimedia content
- No commands for fast forward, etc.

RTSP: RFC 2326
- Client-server application layer protocol.
- For user to control display: rewind, fast forward, pause, resume, repositioning, etc...

What it doesn’t do:
- does not define how audio/video is encapsulated for streaming over network
- does not restrict how streamed media is transported; it can be transported over UDP or TCP
- does not specify how the media player buffers audio/video
RTSP (Real Time Streaming Protocol): Out of band control

FTP uses an “out-of-band” control channel:
- A file is transferred over one TCP connection.
- Control information (directory changes, file deletion, file renaming, etc.) is sent over a separate TCP connection.
- The “out-of-band” and “in-band” channels use different port numbers.

RTSP messages are also sent out-of-band:
- RTSP control messages use different port numbers than the media stream: out-of-band.
  - Port 554
- The media stream is considered “in-band”.

Real-time interactive applications
Example: Internet Telephony

- Examples
  - PC2PC Phone
  - instant messaging services are providing this
  - videoconference with Webcams
- Late packets are as good as lost!
- Call Set up: Session Initiation Protocol (SIP)
  - No network layer connection assumed
- Packets transported using Real Time Protocol (RTP)
- Most of the work done in the receiver…
SIP

- Session Initiation Protocol
- Comes from IETF

**SIP long-term vision**

- All telephone calls and video conference calls take place over the Internet
- People are identified by names or e-mail addresses, rather than by phone numbers.
- You can reach the callee, no matter where the callee roams, no matter what IP device the callee is currently using.

---

SIP Services

- **Setting up a call**
  - Provides mechanisms for caller to let callee know she wants to establish a call
  - Provides mechanisms so that caller and callee can agree on media type and encoding.
  - Provides mechanisms to end call.

- **Determine current IP address of callee.**
  - Maps mnemonic identifier to current IP address

- **Call management**
  - Add new media streams during call
  - Change encoding during call
  - Invite others
  - Transfer and hold calls
Setting up a call to a known IP address

- Alice's SIP invite message indicates her port number & IP address. Indicates encoding that Alice prefers to receive (PCM ulaw)

- Bob's 200 OK message indicates his port number, IP address & preferred encoding (GSM)

- SIP messages can be sent over TCP or UDP; here sent over RTP/UDP.

- Default SIP port number is 5060.

---

Setting up a call (more)

- Codec negotiation:
  - Suppose Bob doesn’t have PCM ulaw encoder.
  - Bob will instead reply with 606 Not Acceptable Reply and list encoders he can use.
  - Alice can then send a new INVITE message, advertising an appropriate encoder.

- Rejecting the call
  - Bob can reject with replies “busy,” “gone,” “payment required,” “forbidden”.
  - Media can be sent over RTP or some other protocol.
Internet Phone Traffic Characteristics

- speaker’s audio: alternating talk spurts, silent periods.
  - 64 kbps during talk spurt
- packets generated only during talk spurts
  - 20 msec chunks at 8 Kbytes/sec: 160 bytes data
- application-layer header added to each chunk.
- Chunk+header encapsulated into UDP segment.
- application sends UDP segment into socket every 20 msec during talkspurt.

Internet Phone: Packet Loss and Delay

- network loss: IP datagram lost due to network congestion (router buffer overflow)
- delay loss: IP datagram arrives too late for playout at receiver
  - delays: processing, queueing in network; end-system (sender, receiver) delays
  - typical maximum tolerable delay: 400 ms
- loss tolerance: depending on voice encoding, losses concealed, packet loss rates between 1% and 10% can be tolerated.
The Problem with Delay and Jitter

- Consider the end-to-end delays of two consecutive packets: difference can be more or less than 20 msec

Internet Phone: Fixed Playout Delay

- Receiver attempts to playout each chunk exactly q msecs after chunk was generated.
  - chunk has time stamp t: play out chunk at t+q.
  - chunk arrives after t+q: data arrives too late for playout, data “lost”
- Tradeoff for q:
  - large q: less packet loss
  - small q: better interactive experience
Fixed Playout Delay

- Sender generates packets every 20 msec during talk spurt.
- First packet received at time $r$.
- First playout schedule begins at $p$.
- Second playout schedule begins at $p'$.

Adaptive Playout Delay, I

- **Goal**: minimize playout delay, keeping late loss rate low.
- **Approach**: adaptive playout delay adjustment:
  - Estimate network delay, adjust playout delay at beginning of each talk spurt.
  - Silent periods compressed and elongated.
  - Chunks still played out every 20 msec during talk spurt.
  - Maintain average delay estimate at the receiver:
    - Moving Average
  - Also, estimate jitter.
Marking Talkspurts

**Q:** How does receiver determine whether packet is first in a talkspurt?

- If no loss, receiver looks at successive timestamps.
  - difference of successive stamps > 20 msec \(\rightarrow\) talk spurt begins.
- With loss possible, receiver must look at both time stamps and sequence numbers.
  - difference of successive stamps > 20 msec and sequence numbers without gaps \(\rightarrow\) talk spurt begins.

---

Estimating Jitter

Suppose the network delay estimate when packet \(i\) arrives is \(D_{i}\), but packet \(i\) itself is delayed by \(d_{i}\) time units.

The deviation from the estimate is \(|D_{i} - d_{i}|\).

The receiver maintains an estimate of this variation for each packet. This is \(J_{i}\), its estimate of the jitter. Again, this is a moving average estimate.

When the first packet of a talkspurt is received, hold it in the buffer for \(K J_{i}\) time units to allow for time for more of the packets from the talkspurt to arrive.

Play the rest of the packets from that talkspurt out periodically...
Recovering from loss

**FEC scheme**
- "piggyback lower quality stream"
- send lower resolution audio stream as the redundant information
- for example, nominal stream PCM at 64 kbps and redundant stream GSM at 13 kbps.

Whenever there is non-consecutive loss, the receiver can conceal the loss. Can also append (n-1)st and (n-2)nd low-bit rate chunk.

Another approach to recover from loss

**Interleaving**
- chunks are broken up into smaller units
- for example, 4 5 msec units per chunk
- Packet contains small units from different chunks
- if packet is lost, still have most of every chunk
- has no redundancy overhead
- but adds to playout delay
Real-Time Protocol (RTP)

- RTP specifies a packet structure for packets carrying audio and video data
- RFC 1889.
- RTP packet provides:
  - payload type identification
  - packet sequence numbering
  - timestamping
- RTP runs in the end systems.
- RTP packets are encapsulated in UDP segments
- Interoperability: If two Internet phone applications run RTP, then they may be able to work together

RTP Header

Payload Type (7 bits): Indicates type of encoding currently being used. If sender changes encoding in middle of conference, sender informs the receiver through this payload type field.

- Payload type 0: PCM mu-law, 64 kbps
- Payload type 3, GSM, 13 kbps
- Payload type 7, LPC, 2.4 kbps
- Payload type 26, Motion JPEG
- Payload type 31, H.261
- Payload type 33, MPEG2 video

Sequence Number (16 bits): Increments by one for each RTP packet sent, and may be used to detect packet loss and to restore packet sequence.
RTP Header (2)

- **Timestamp field (32 bytes long).** Reflects the sampling instant of the first byte in the RTP data packet.
  - For audio, timestamp clock typically increments by one for each sampling period (for example, each 125 usecs for a 8 KHz sampling clock)
  - if application generates chunks of 160 encoded samples, then timestamp increases by 160 for each RTP packet when source is active. Timestamp clock continues to increase at constant rate when source is inactive.

- **SSRC field (32 bits long).** Identifies the source of the RTP stream. Each stream in a RTP session should have a distinct SSRC.

RTP runs on top of UDP

RTP libraries provide a transport-layer interface that extend UDP:
- port numbers, IP addresses
- payload type identification
- packet sequence numbering
- time-stamping
RTP Example

- Consider sending 64 kbps PCM-encoded voice over RTP.
- Application collects the encoded data in chunks, e.g., every 20 msec = 160 bytes in a chunk.
- The audio chunk along with the RTP header form the RTP packet, which is encapsulated into a UDP segment.
- RTP header indicates type of audio encoding in each packet
  - sender can change encoding during a conference.
- RTP header also contains sequence numbers and timestamps.

Internet Multimedia: bag of tricks

- use UDP to avoid TCP congestion control (delays) for time-sensitive traffic
- client-side adaptive playout delay: to compensate for delay
- server side matches stream bandwidth to available client-to-server path bandwidth
  - chose among pre-encoded stream rates
  - dynamic server encoding rate
- error recovery (on top of UDP)
  - FEC, interleaving
  - retransmissions, time permitting
  - conceal errors: repeat nearby data
- Has to be VERY application specific…
Conclusions

- QoS at the network layer has been studied extensively but implementation has been slow.
- Non-Interactive Multimedia applications can be built effectively on today’s infrastructure.
- Interactive Applications can be deployed at small scale but may ultimately be the reason QoS is more widely deployed.