



Data Communications & Networks

Session 10 – Main Theme Multimedia Networking

Dr. Jean-Claude Franchitti

*New York University
Computer Science Department
Courant Institute of Mathematical Sciences*

*Adapted from course textbook resources
Computer Networking: A Top-Down Approach, 5/E
Copyright 1996-2013
J.F. Kurose and K.W. Ross, All Rights Reserved*

Agenda



- 1 Session Overview

- 2 Multimedia Networking

- 3 Summary and Conclusion



- Course description and syllabus:

- » <http://www.nyu.edu/classes/jcf/csci-ga.2262-001/>

- » <http://cs.nyu.edu/courses/spring16/CSCI-GA.2262-001/index.html>

- Textbooks:

- » ***Computer Networking: A Top-Down Approach (6th Edition)***

James F. Kurose, Keith W. Ross

Addison Wesley

ISBN-10: 0132856204, ISBN-13: 978-0132856201, 6th Edition (02/24/12)





- Computer Networks and the Internet
- Application Layer
- Fundamental Data Structures: queues, ring buffers, finite state machines
- Data Encoding and Transmission
- Local Area Networks and Data Link Control
- Wireless Communications
- Packet Switching
- OSI and Internet Protocol Architecture
- Congestion Control and Flow Control Methods
- Internet Protocols (IP, ARP, UDP, TCP)
- Network (packet) Routing Algorithms (OSPF, Distance Vector)
- IP Multicast
- Sockets



- Introduction to Basic Networking Concepts (Network Stack)
- Origins of Naming, Addressing, and Routing (TCP, IP, DNS)
- Physical Communication Layer
- MAC Layer (Ethernet, Bridging)
- Routing Protocols (Link State, Distance Vector)
- Internet Routing (BGP, OSPF, Programmable Routers)
- TCP Basics (Reliable/Unreliable)
- Congestion Control
- QoS, Fair Queuing, and Queuing Theory
- Network Services – Multicast and Unicast
- Extensions to Internet Architecture (NATs, IPv6, Proxies)
- Network Hardware and Software (How to Build Networks, Routers)
- Overlay Networks and Services (How to Implement Network Services)
- Network Firewalls, Network Security, and Enterprise Networks



Information



Common Realization



Knowledge/Competency Pattern



Governance



Alignment



Solution Approach

Agenda



- 1 Session Overview

- 2 Multimedia Networking

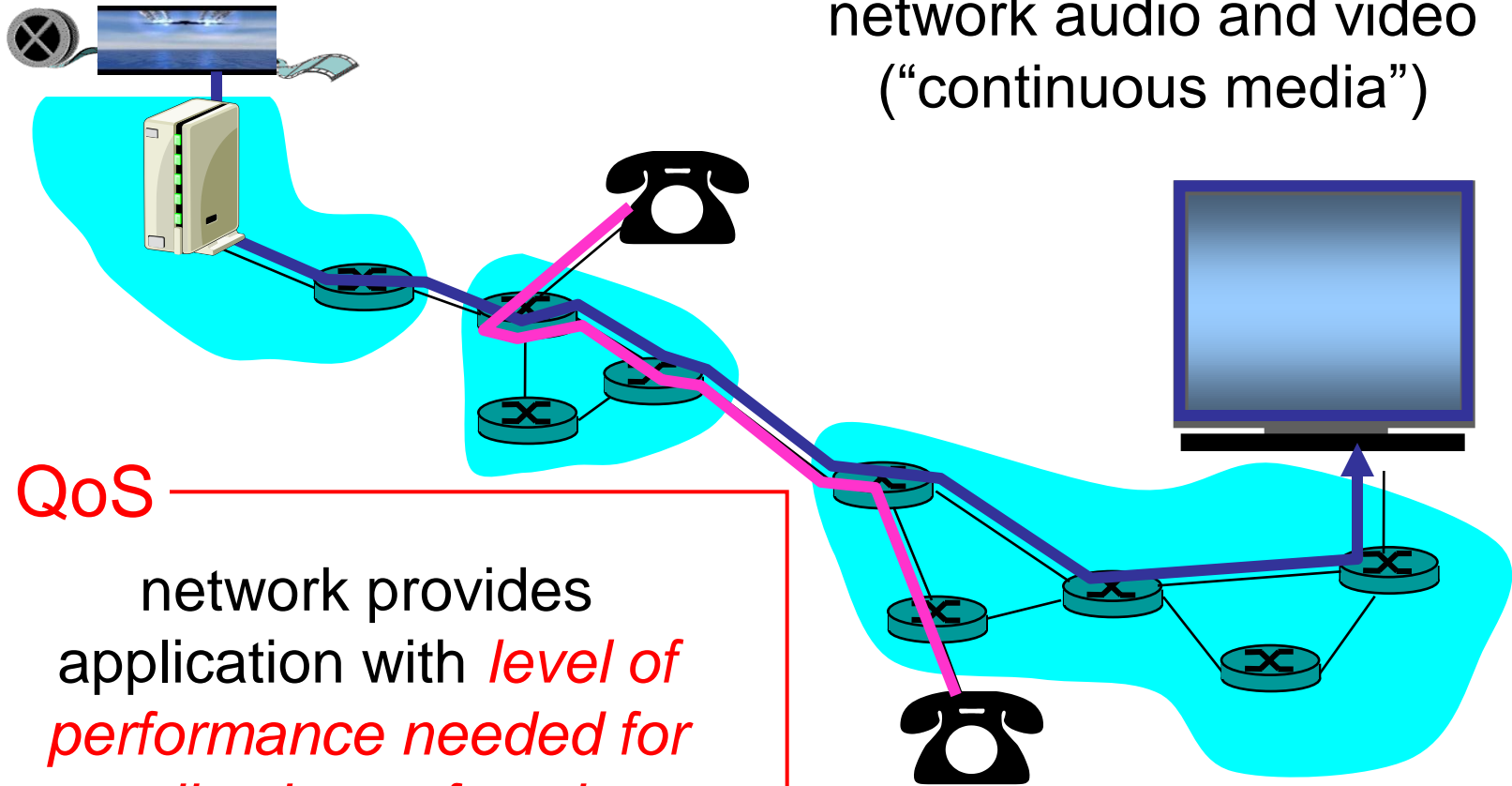
- 3 Summary and Conclusion



- Session Overview
 - Multimedia networking applications
 - Streaming stored audio and video
 - Making the best out of best effort service
 - Protocols for real-time interactive applications
RTP, RTCP, SIP
 - Providing multiple classes of service
 - Providing QoS guarantees
- Summary

Multimedia and Quality of Service: What is it?

multimedia applications:
network audio and video
("continuous media")



QoS

network provides
application with *level of
performance needed for
application to function.*

Principles

- classify multimedia applications
- identify network services applications need
- making the best of best effort service

Protocols and Architectures

- specific protocols for best-effort
- mechanisms for providing QoS
- architectures for QoS

Classes of MM applications:

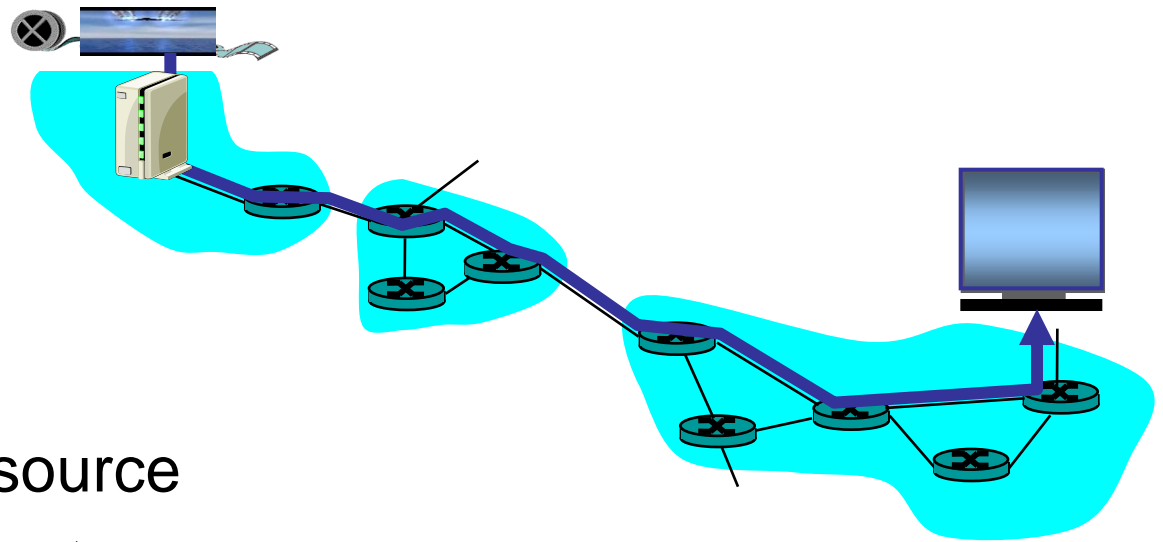
- 1) stored streaming
- 2) live streaming
- 3) interactive, real-time

Jitter is the variability of packet delays within the same packet stream

Fundamental characteristics:

- typically **delay sensitive**
 - » end-to-end delay
 - » delay jitter
- **loss tolerant**: infrequent losses cause minor glitches
- antithesis of data, which are loss *intolerant* but delay *tolerant*.

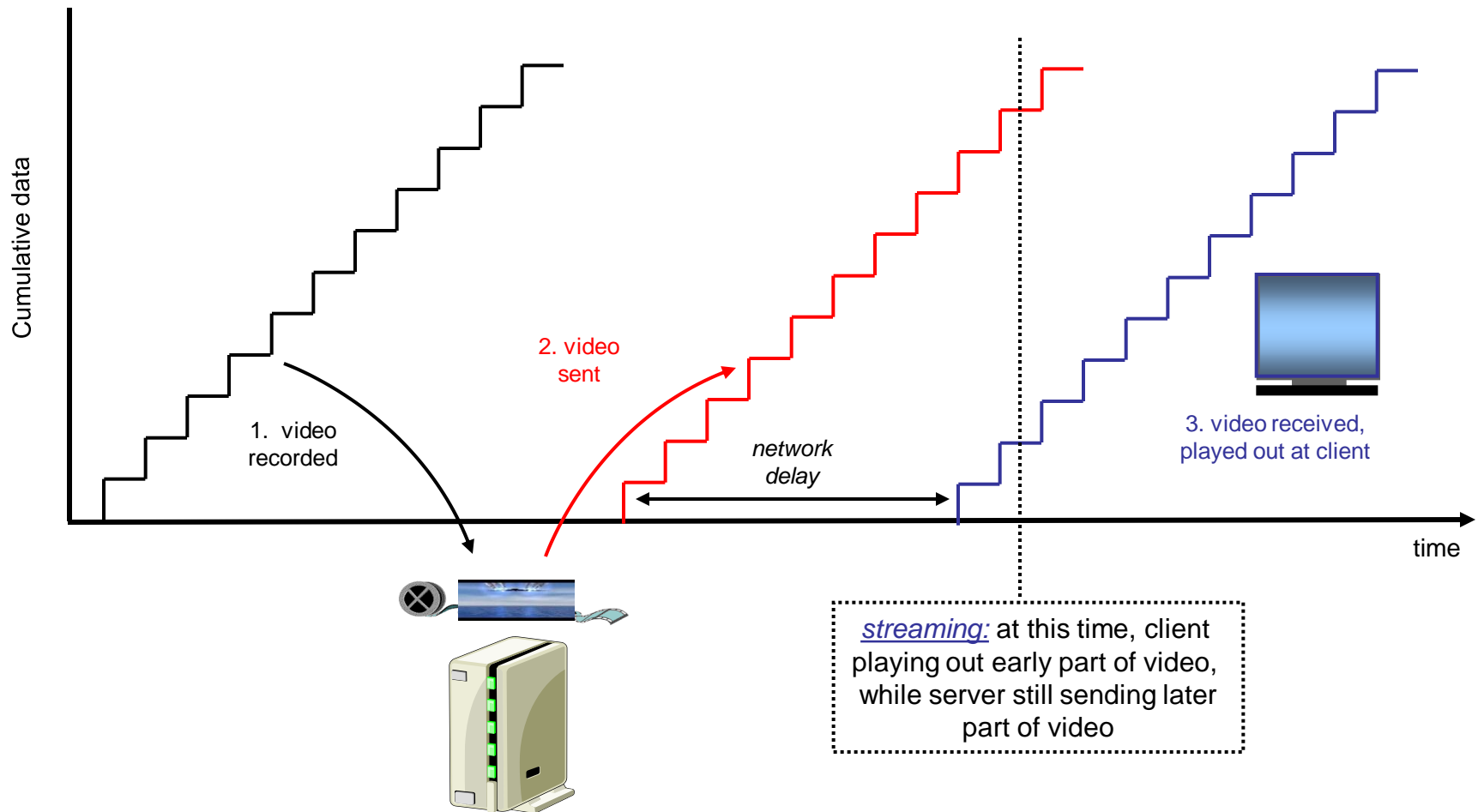
Streaming Stored Multimedia



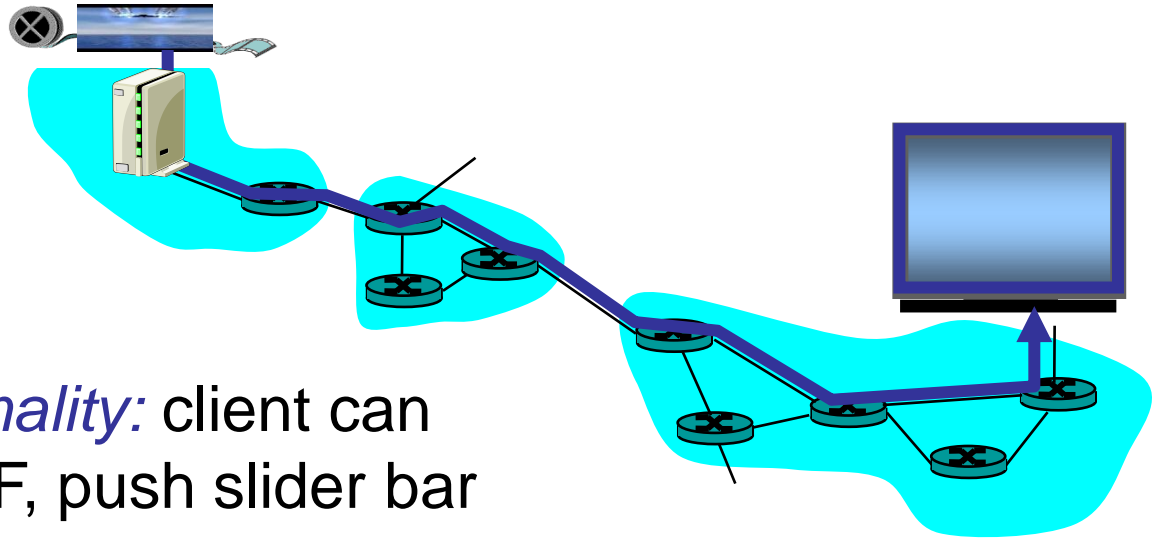
Stored streaming:

- ❑ media stored at source
- ❑ transmitted to client
- ❑ streaming: client playout begins *before* all data has arrived
 - ❑ timing constraint for still-to-be transmitted data: in time for playout

Streaming Stored Multimedia: What is it?



Streaming *Stored* Multimedia: Interactivity



- ❑ *VCR-like functionality*: client can pause, rewind, FF, push slider bar
 - 10 sec initial delay OK
 - 1-2 sec until command effect OK

- ❑ timing constraint for still-to-be transmitted data: in time for playout

Examples:

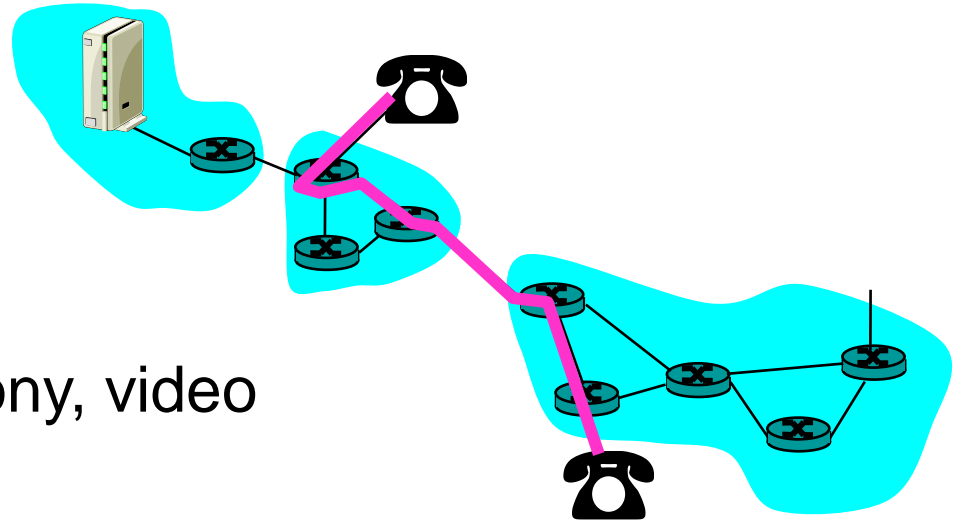
- Internet radio talk show
- live sporting event

Streaming (as with streaming *stored* multimedia)

- playback buffer
- playback can lag tens of seconds after transmission
- still have timing constraint

Interactivity

- fast forward impossible
- rewind, pause possible!



□ **applications:** IP telephony, video conference, distributed interactive worlds

■ **end-end delay requirements:**

- » audio: < 150 msec good, < 400 msec OK
 - includes application-level (packetization) and network delays
 - higher delays noticeable, impair interactivity

■ **session initialization**

- » how does callee advertise its IP address, port number, encoding algorithms?

TCP/UDP/IP: “best-effort service”

- **no** guarantees on delay, loss



? ? ? ? ? ?
But you said multimedia apps requires ?
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

How should the Internet evolve to better support multimedia?

Integrated services philosophy:

- fundamental changes in Internet so that apps can reserve end-to-end bandwidth
- requires new, complex software in hosts & routers

Laissez-faire

- no major changes
- more bandwidth when needed
- content distribution, application-layer multicast
 - » application layer

Differentiated services philosophy:

- fewer changes to Internet infrastructure, yet provide 1st and 2nd class service



What's your opinion?

A few words about audio compression

- analog signal sampled at constant rate
 - » telephone: 8,000 samples/sec
 - » CD music: 44,100 samples/sec
- each sample quantized, i.e., rounded
 - » e.g., $2^8=256$ possible quantized values
- each quantized value represented by bits
 - » 8 bits for 256 values
- example: 8,000 samples/sec, 256 quantized values --> 64,000 bps
- receiver converts bits back to analog signal:
 - » some quality reduction

Example rates

- CD: 1.411 Mbps
- MP3: 96, 128, 160 kbps
- Internet telephony: 5.3 kbps and up

A few words about video compression

- video: sequence of images displayed at constant rate
 - » e.g. 24 images/sec
- digital image: array of pixels
 - » each pixel represented by bits
- redundancy
 - » spatial (within image)
 - » temporal (from one image to next)


Examples:

- MPEG 1 (CD-ROM) 1.5 Mbps
- MPEG2 (DVD) 3-6 Mbps
- MPEG4 (often used in Internet, < 1 Mbps)

Research:

- layered (scalable) video
 - » adapt layers to available bandwidth



- Session Overview
- Multimedia networking applications
-  ■ Streaming stored audio and video
- Making the best out of best effort service
- Protocols for real-time interactive applications
RTP, RTCP, SIP
- Providing multiple classes of service
- Providing QoS guarantees
- Summary

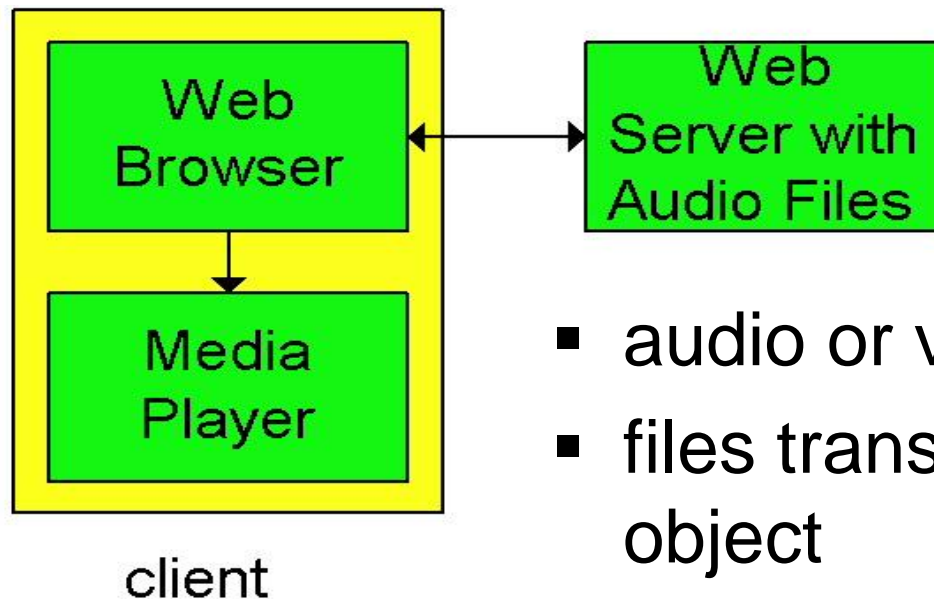
application-level streaming techniques for making the best out of best effort service:

- » client-side buffering
- » use of UDP versus TCP
- » multiple encodings of multimedia

Media Player

- jitter removal
- decompression
- error concealment
- graphical user interface w/ controls for interactivity

Internet multimedia: simplest approach

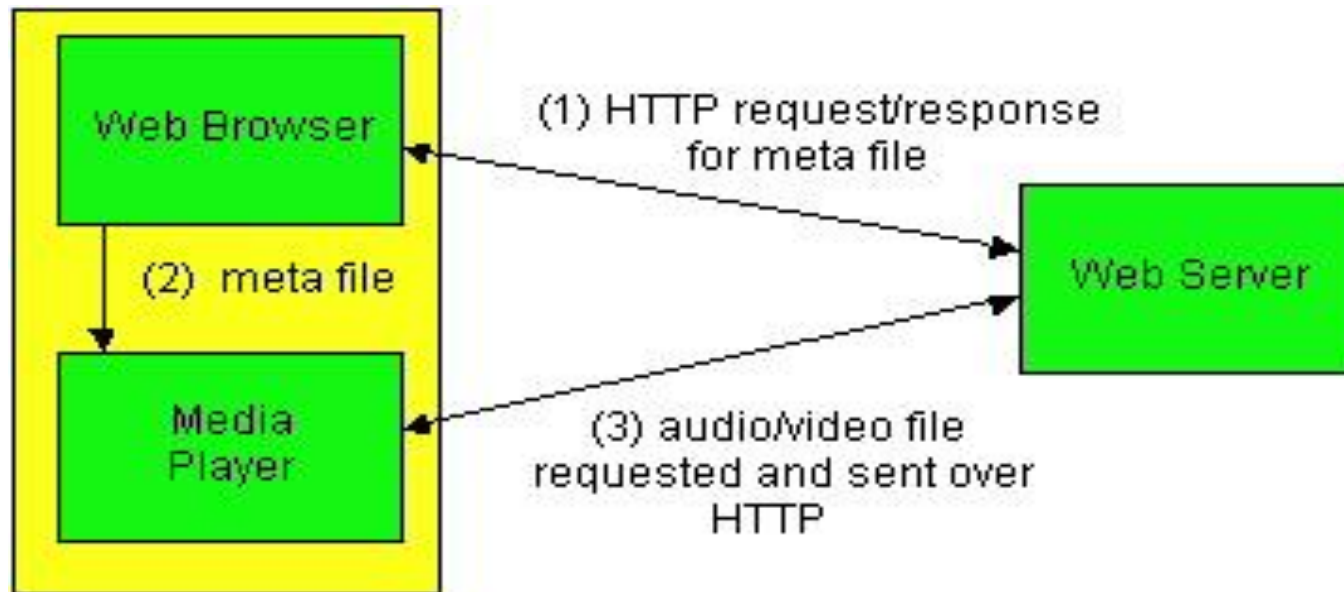


- audio or video stored in file
- files transferred as HTTP object
 - » received in entirety at client
 - » then passed to player

audio, video not streamed:

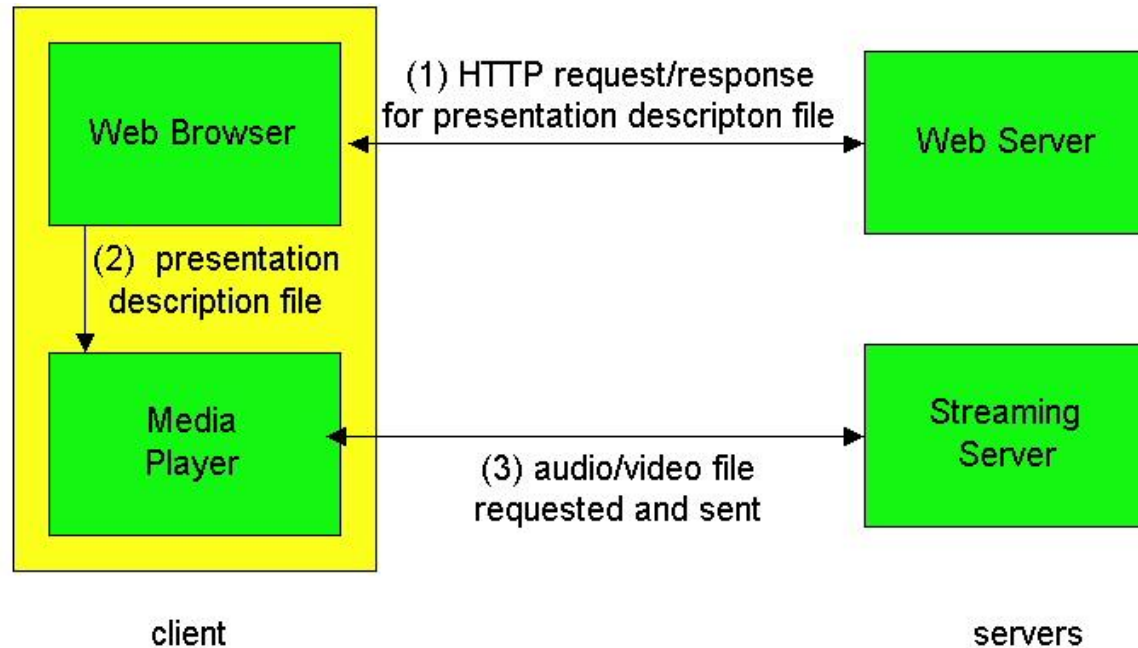
- no, “pipelining,” long delays until playout!

Internet multimedia: streaming approach



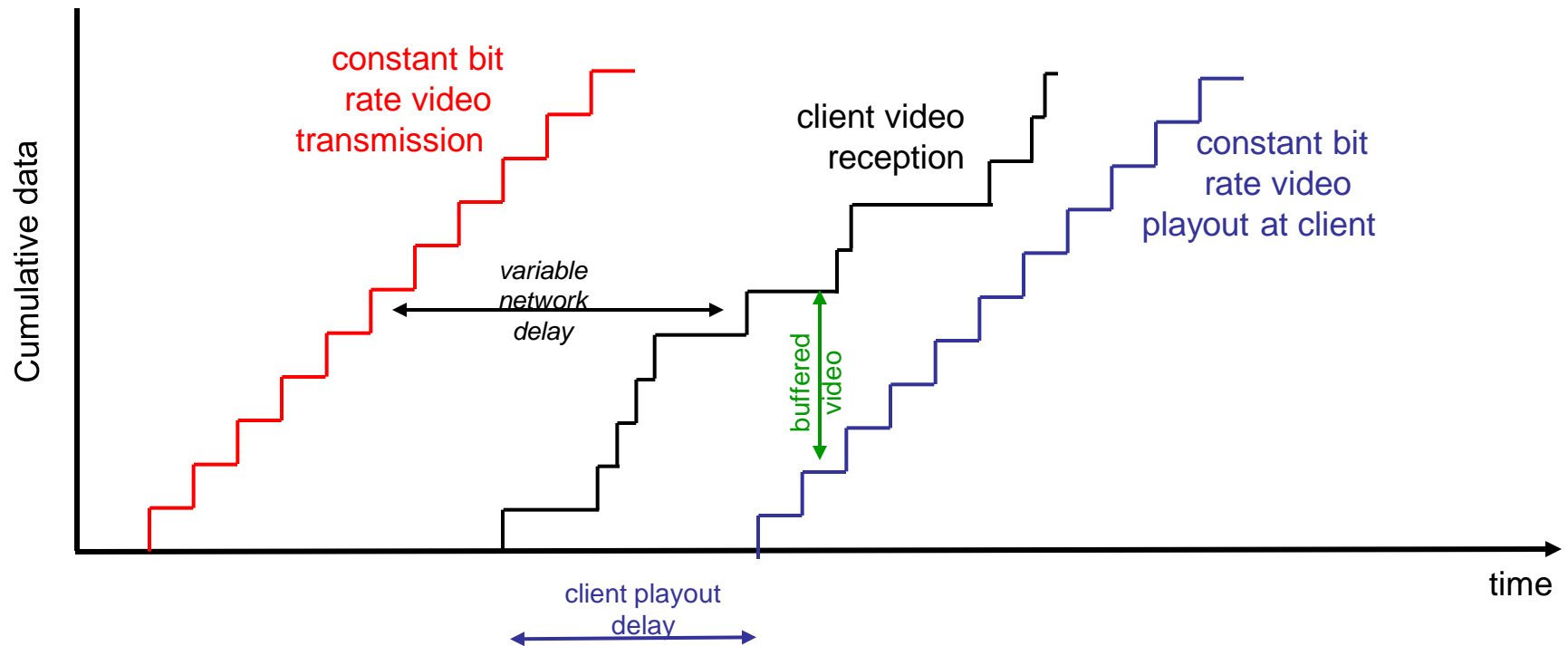
- ❑ browser GETs **metafile**
- ❑ browser launches player, passing metafile
- ❑ player contacts server
- ❑ server **streams** audio/video to player

Streaming from a streaming server



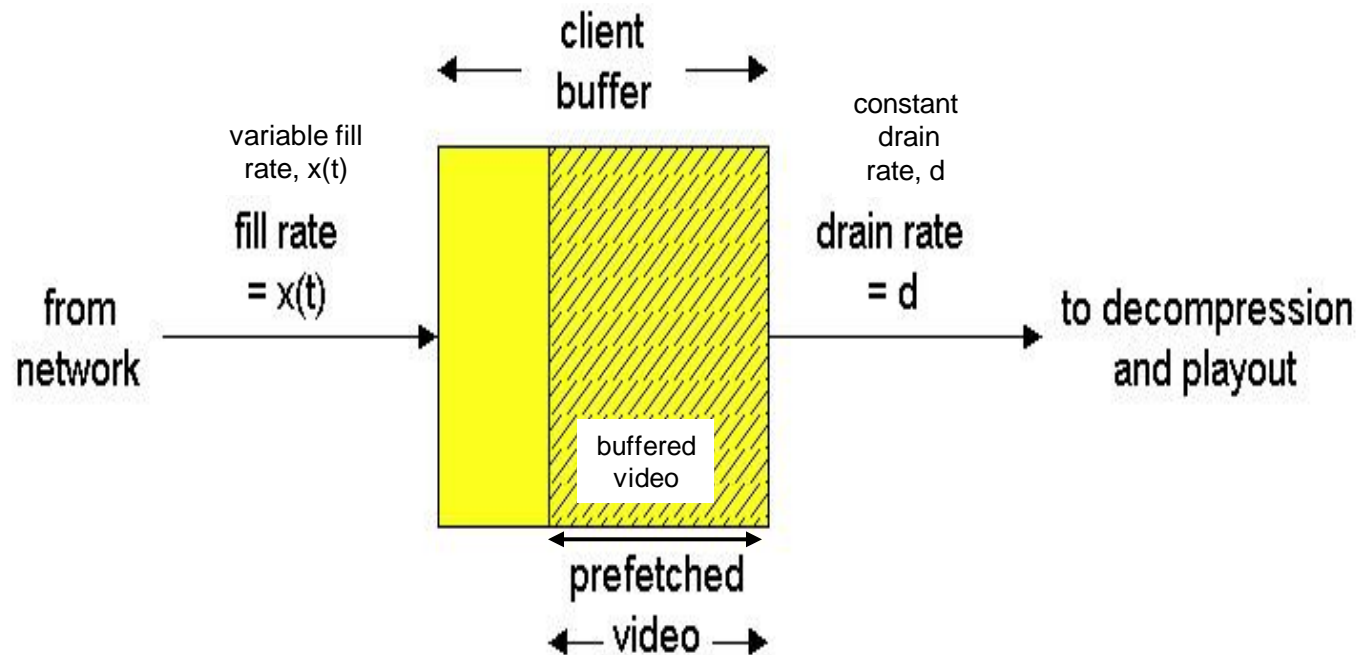
- allows for non-HTTP protocol between server, media player
- UDP or TCP for step (3), more shortly

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

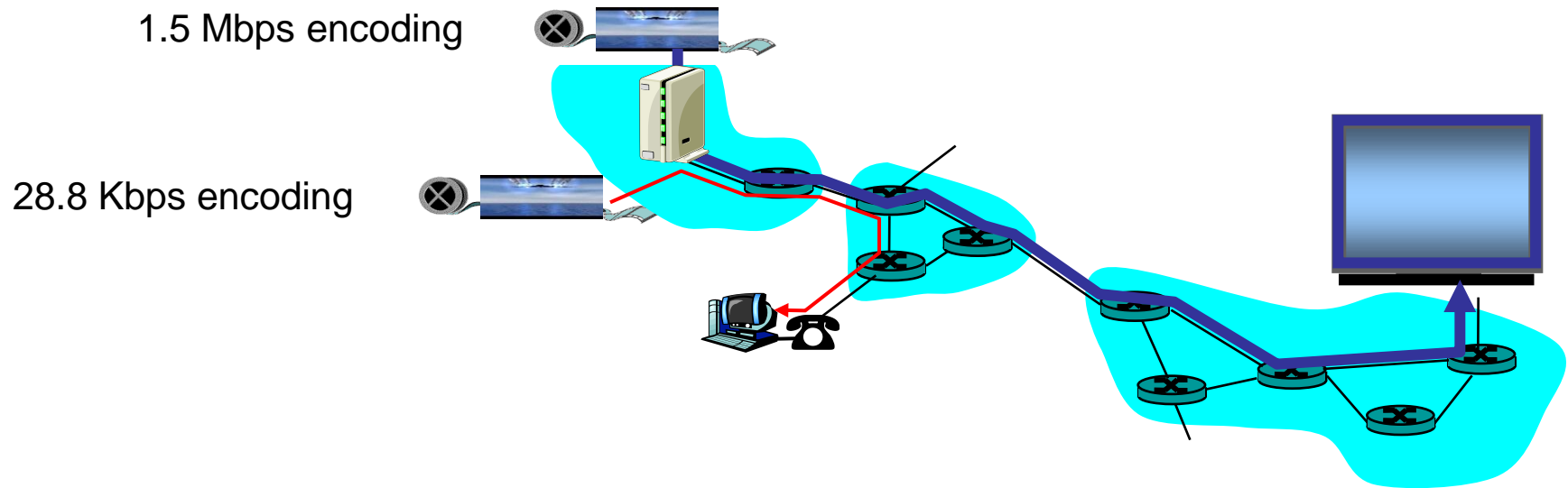
UDP

- 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 remove network 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
- 100 Mbps Ethernet

A: server stores, transmits multiple copies of video, encoded at different rates

HTTP

- does not target multimedia content
- no commands for fast forward, etc.

RTSP: RFC 2326

- client-server application layer protocol
- user control: rewind, fast forward, pause, resume, repositioning, etc...

What it doesn't do:

- doesn't define how audio/video is encapsulated for streaming over network
- doesn't restrict how streamed media is transported (UDP or TCP possible)
- doesn't specify how media player buffers audio/video

FTP uses an “out-of-band” control channel:

- file transferred over one TCP connection.
- control info (directory changes, file deletion, rename) sent over separate TCP connection
- “out-of-band”, “in-band” channels use different port numbers

RTSP messages also sent out-of-band:

- RTSP control messages use different port numbers than media stream: out-of-band.
 - » port 554
- media stream is considered “in-band”.

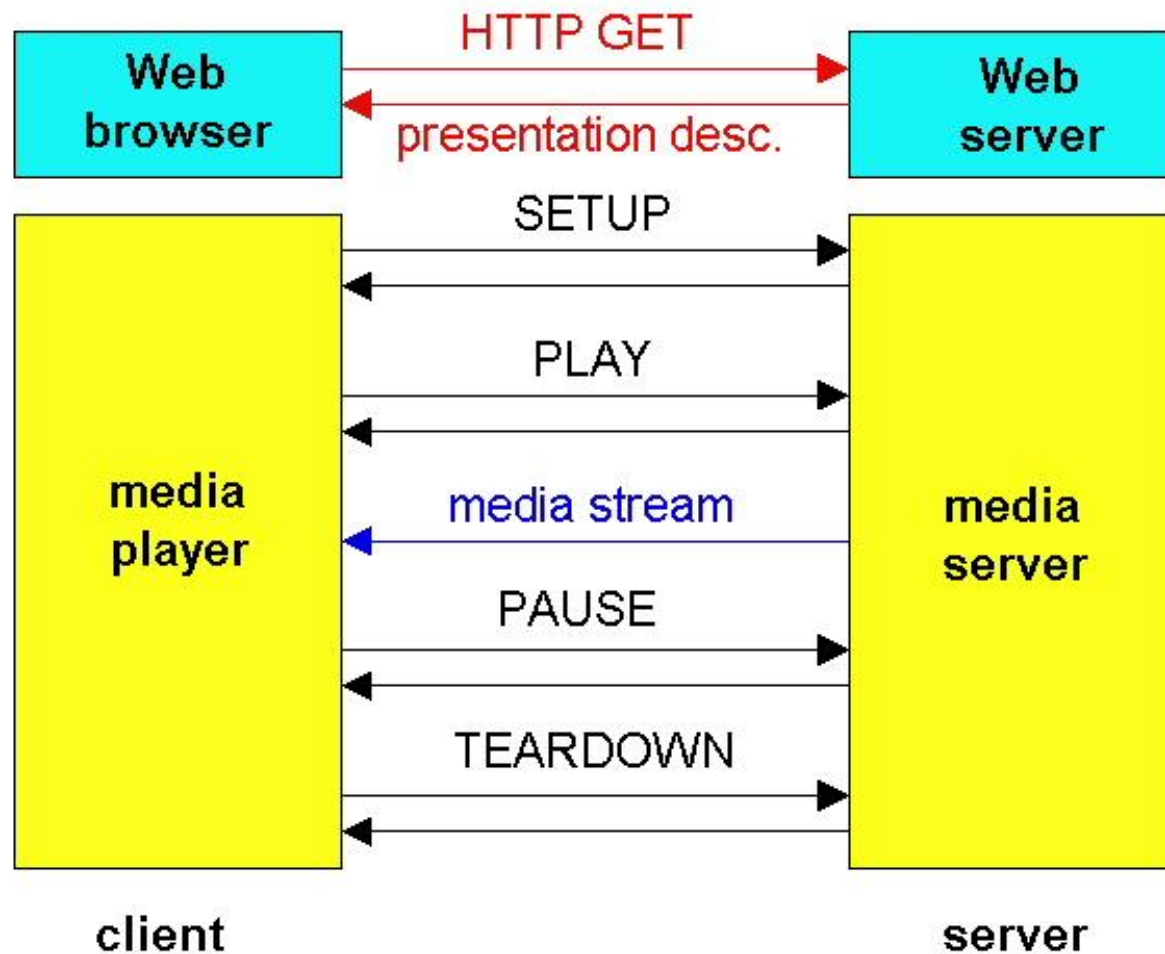
Scenario:

- metafile communicated to web browser
- browser launches player
- player sets up an RTSP control connection, data connection to streaming server

Metafile Example

```
<title>Twister</title>
<session>
  <group language=en lipsync>
    <switch>
      <track type=audio
        e="PCMU/8000/1"
        src = "rtsp://audio.example.com/twister/audio.en/lofi">
      <track type=audio
        e="DVI4/16000/2" pt="90 DVI4/8000/1"
        src="rtsp://audio.example.com/twister/audio.en/hifi">
    </switch>
  <track type="video/jpeg"
    src="rtsp://video.example.com/twister/video">
  </group>
</session>
```

RTSP Operation



RTSP Exchange Example

C: SETUP rtsp://audio.example.com/twister/audio RTSP/1.0
Transport: rtp/udp; compression; port=3056; mode=PLAY

S: RTSP/1.0 200 1 OK
Session 4231


C: PLAY rtsp://audio.example.com/twister/audio.en/lofi RTSP/1.0
Session: 4231
Range: npt=0-

C: PAUSE rtsp://audio.example.com/twister/audio.en/lofi RTSP/1.0
Session: 4231
Range: npt=37

C: TEARDOWN rtsp://audio.example.com/twister/audio.en/lofi RTSP/1.0
Session: 4231

S: 200 3 OK



- Session Overview
- Multimedia networking applications
- Streaming stored audio and video
-  ■ Making the best out of best effort service
- Protocols for real-time interactive applications
RTP, RTCP, SIP
- Providing multiple classes of service
- Providing QoS guarantees
- Summary

- PC-2-PC phone
 - » Skype
- PC-2-phone
 - » Dialpad
 - » Net2phone
 - » Skype
- videoconference with webcams
 - » Skype
 - » Polycom

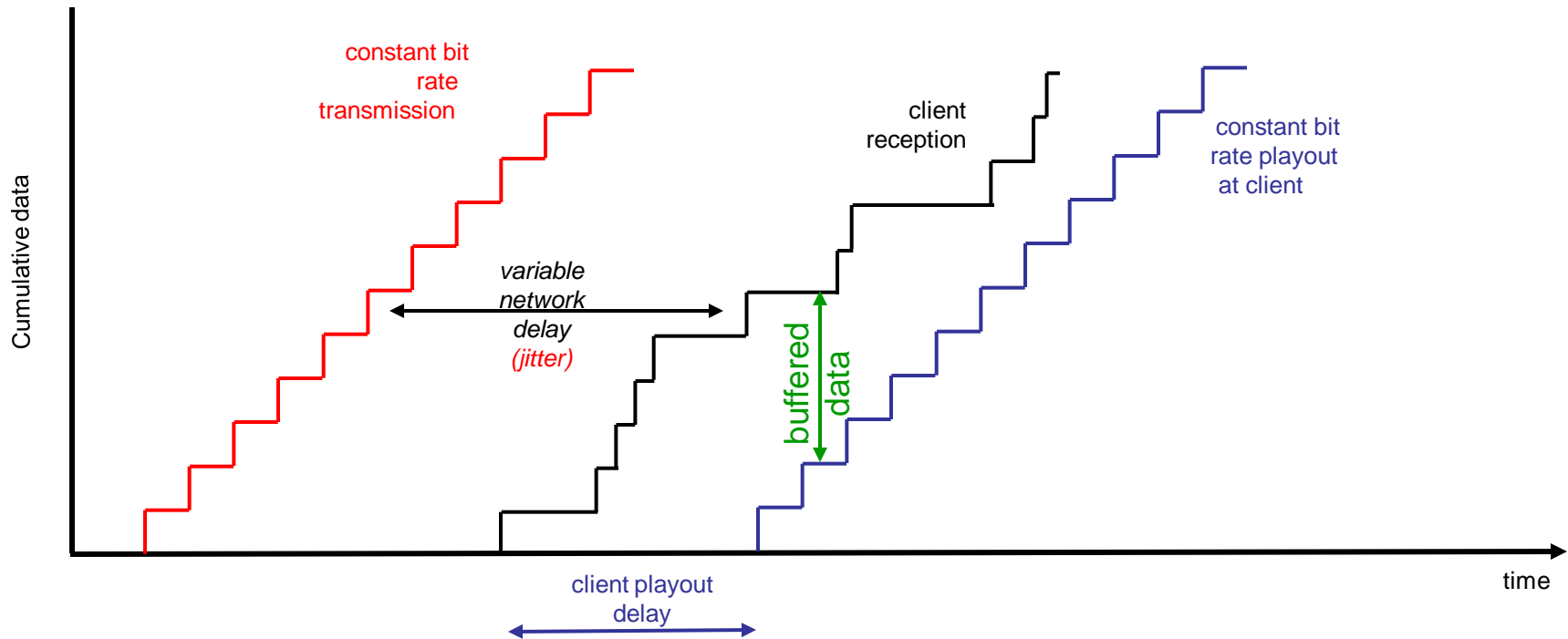
Going to now look at
a PC-2-PC Internet
phone example in
detail

Introduce Internet Phone by way of an example

- speaker's audio: alternating talk spurts, silent periods.
 - » 64 kbps during talk spurt
 - » pkts 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

- **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.

Delay Jitter

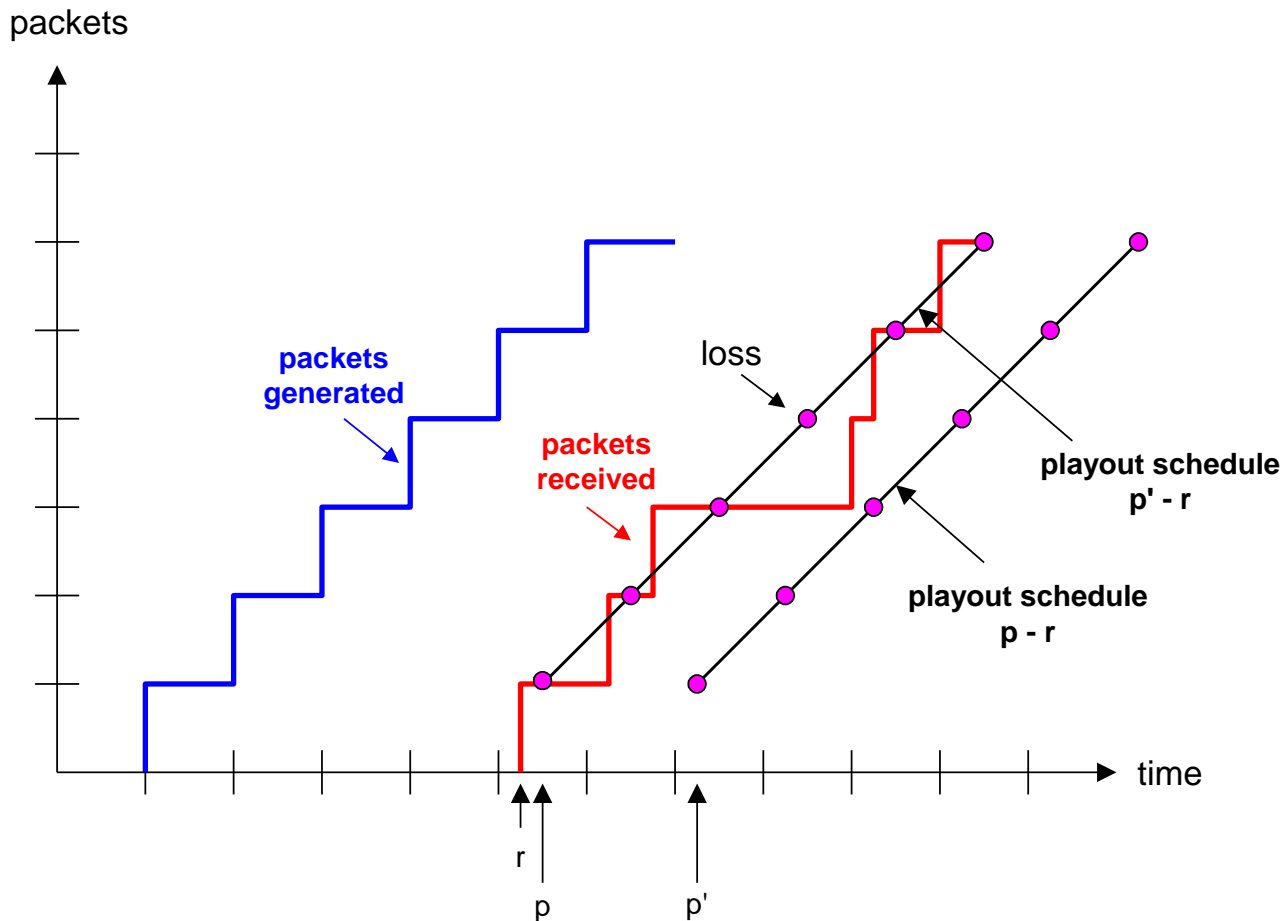


- consider end-to-end delays of two consecutive packets: difference can be more or less than 20 msec (transmission time difference)

- 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 in choosing 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 (1)

- 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.

t_i = timestamp of the i th packet

r_i = the time packet i is received by receiver

p_i = the time packet i is played at receiver

$r_i - t_i$ = network delay for i th packet

d_i = estimate of average network delay after receiving i th packet

dynamic estimate of average delay at receiver:

$$d_i = (1 - u)d_{i-1} + u(r_i - t_i)$$

where u is a fixed constant (e.g., $u = .01$).

Adaptive playout delay (2)

- also useful to estimate average deviation of delay, v_i :

$$v_i = (1 - u)v_{i-1} + u |r_i - t_i - d_i|$$

- estimates d_i , v_i calculated for every received packet (but used only at start of talk spurt)
- for first packet in talk spurt, playout time is:

$$p_i = t_i + d_i + Kv_i$$

where K is positive constant

- remaining packets in talkspurt are played out periodically

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 --> 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 --> talk spurt begins.

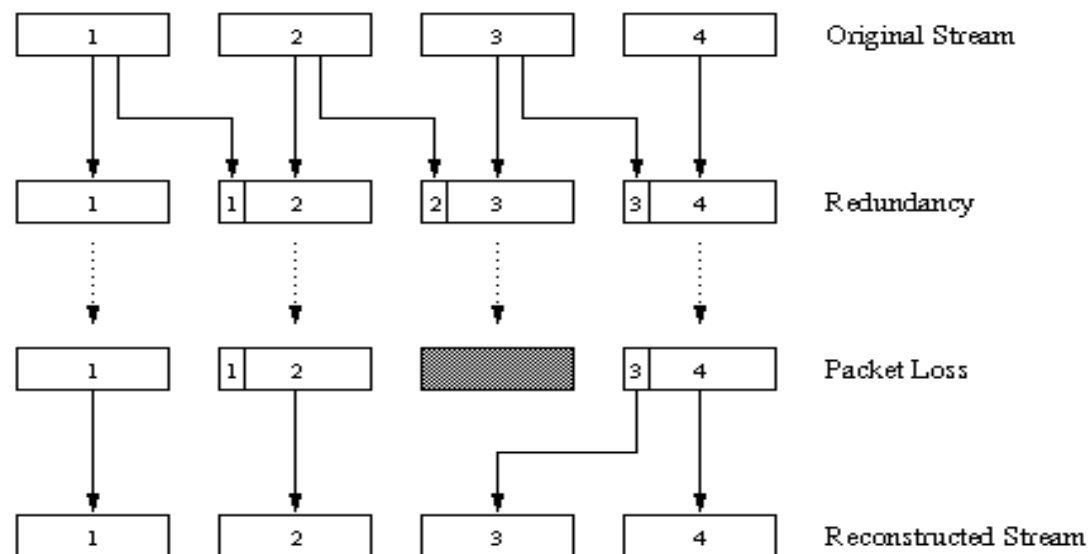
Forward Error Correction (FEC): simple scheme

- for every group of n chunks create redundant chunk by exclusive OR-ing n original chunks
- send out $n+1$ chunks, increasing bandwidth by factor $1/n$.
- can reconstruct original n chunks if at most one lost chunk from $n+1$ chunks
- playout delay: enough time to receive all $n+1$ packets
- tradeoff:
 - » increase n , less bandwidth waste
 - » increase n , longer playout delay
 - » increase n , higher probability that 2 or more chunks will be lost

Recovery from packet loss (2)

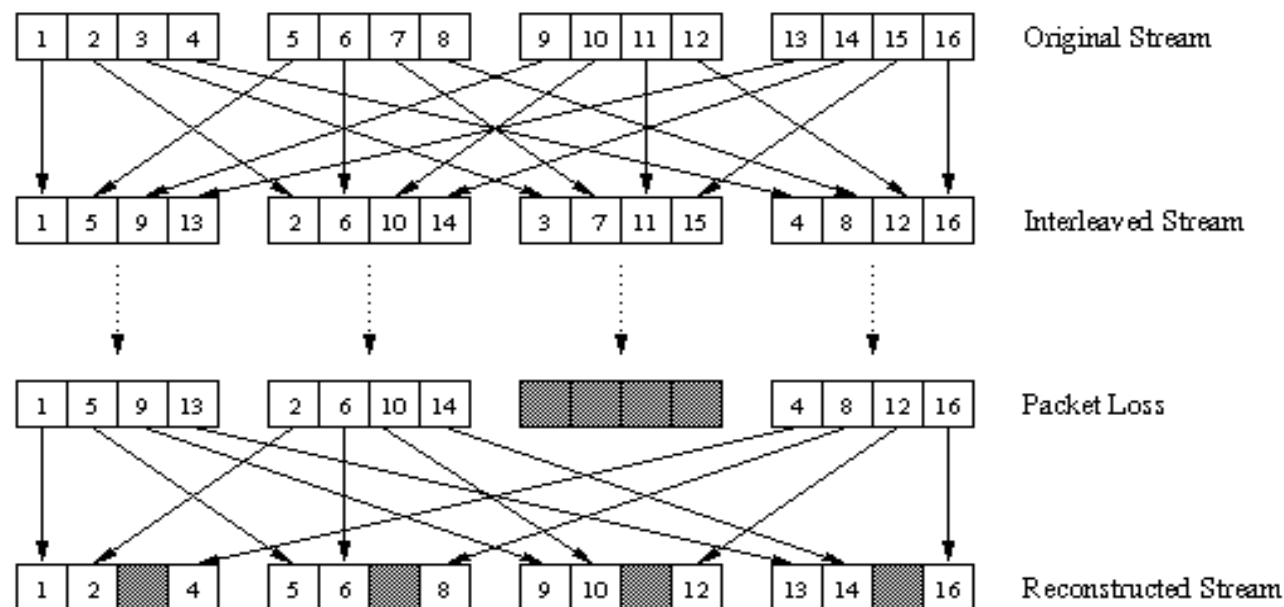
2nd FEC scheme

- ❑ “piggyback lower quality stream”
- ❑ send lower resolution audio stream as redundant information
- ❑ e.g., nominal stream PCM at 64 kbps and redundant stream GSM at 13 kbps.



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

Recovery from packet loss (3)



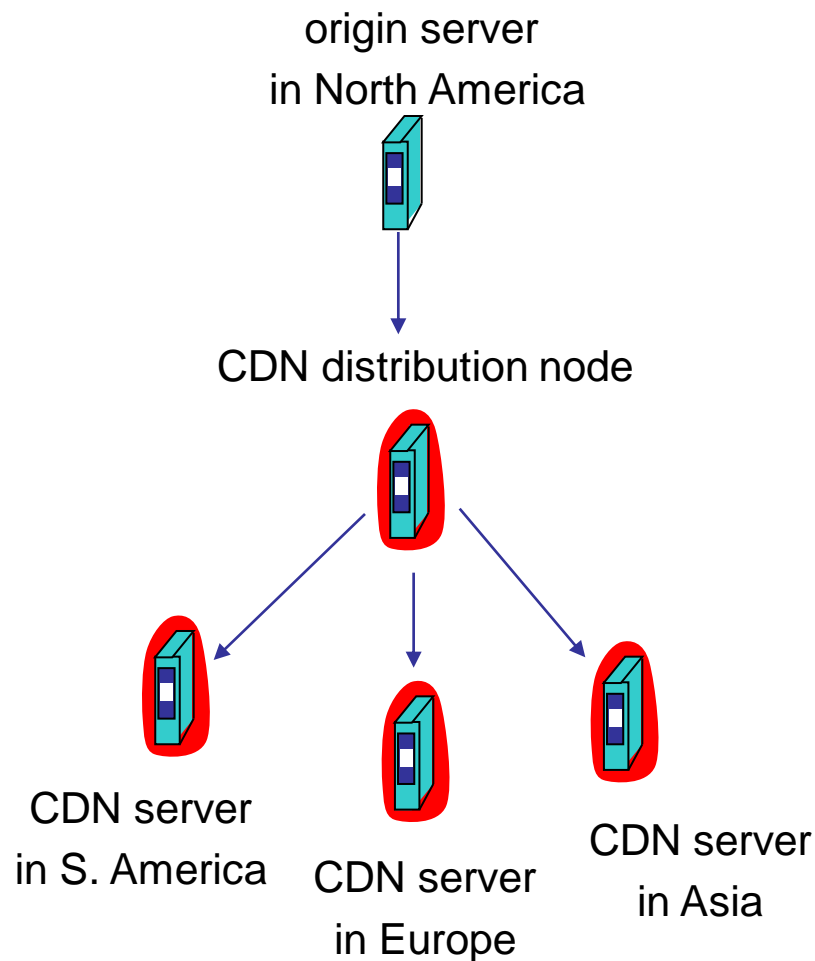
Interleaving

- chunks divided into smaller units
- for example, four 5 msec units per chunk
- packet contains small units from different chunks
- if packet lost, still have most of every chunk
- no redundancy overhead, but increases playout delay

Content distribution networks (CDNs)

Content replication

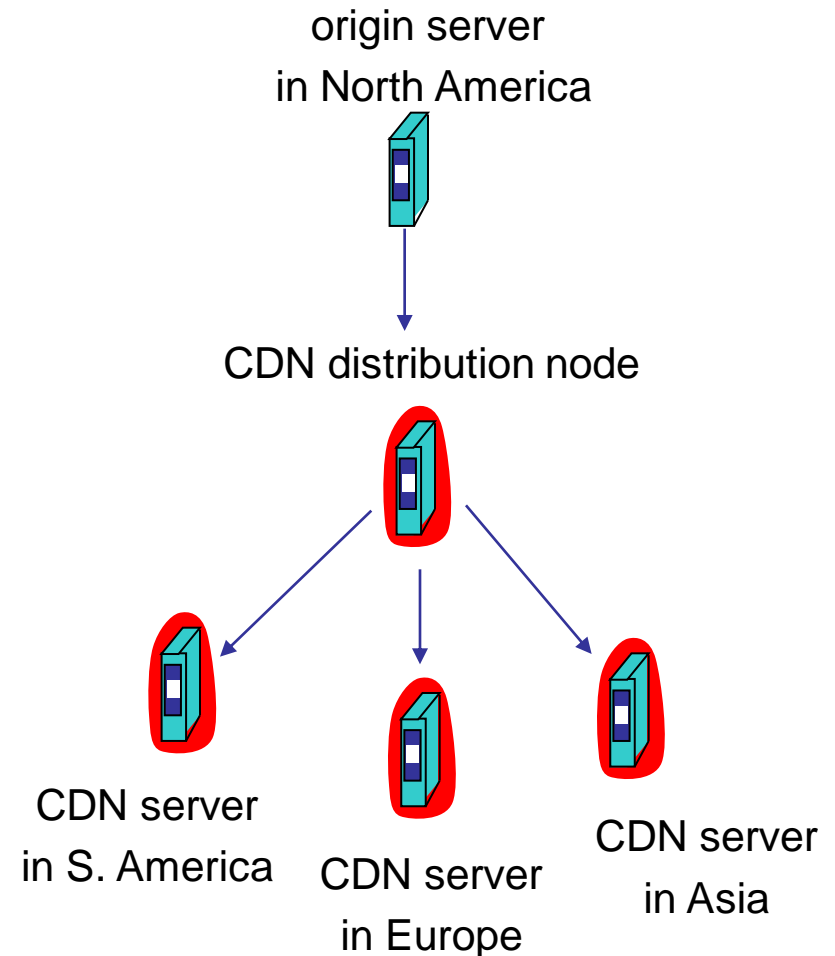
- challenging to stream large files (e.g., video) from single origin server in real time
- *solution*: replicate content at hundreds of servers throughout Internet
 - » content downloaded to CDN servers ahead of time
 - » *placing content “close” to user avoids impairments (loss, delay) of sending content over long paths*
 - » CDN server typically in edge/access network



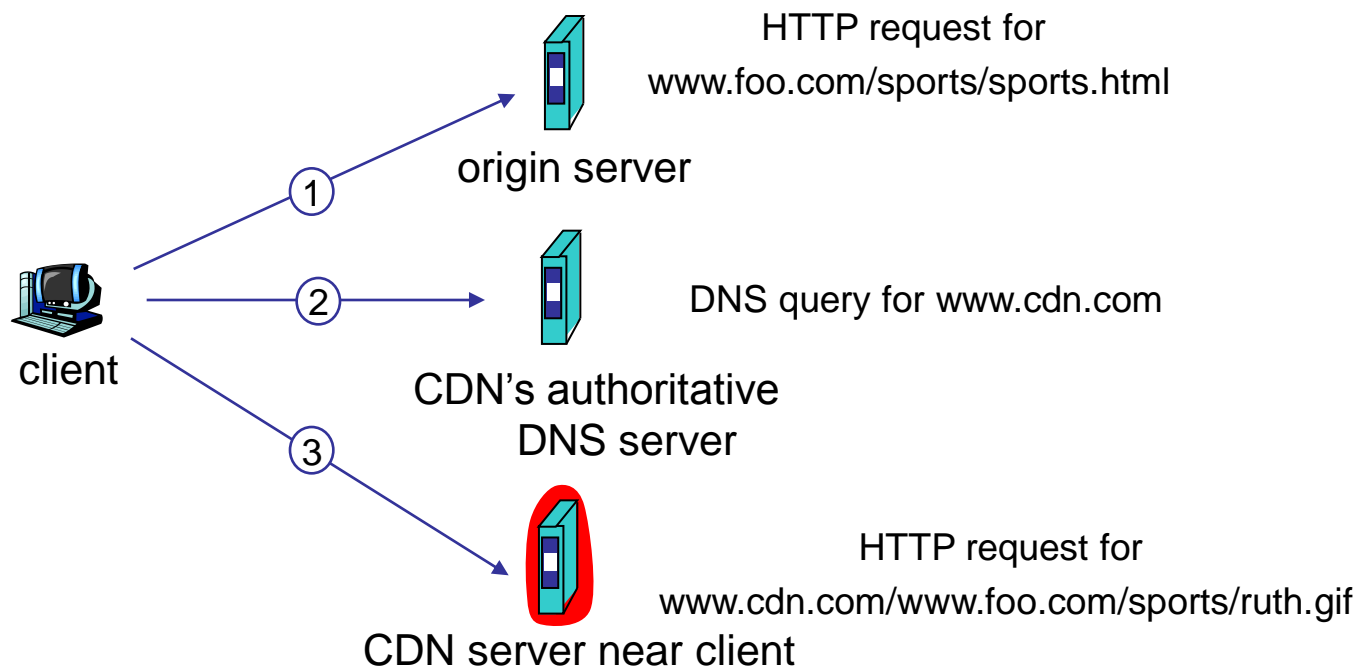
Content distribution networks (CDNs)

Content replication

- CDN (e.g., Akamai) customer is the content provider (e.g., CNN)
- CDN replicates customers' content in CDN servers.
- when provider updates content, CDN updates servers



CDN example



origin server (www.foo.com)

- distributes HTML
- replaces:
`http://www.foo.com/sports.ruth.gif`
with
`http://www.cdn.com/www.foo.com/sports/ruth.gif`

CDN company (cdn.com)

- ❑ distributes gif files
- ❑ uses its authoritative DNS server to route redirect requests


routing requests

- CDN creates a “map”, indicating distances from leaf ISPs and CDN nodes
- when query arrives at authoritative DNS server:
 - » server determines ISP from which query originates
 - » uses “map” to determine best CDN server
- CDN nodes create application-layer overlay network

Summary: 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, error concealment
 - » retransmissions, time permitting
- CDN: bring content closer to clients

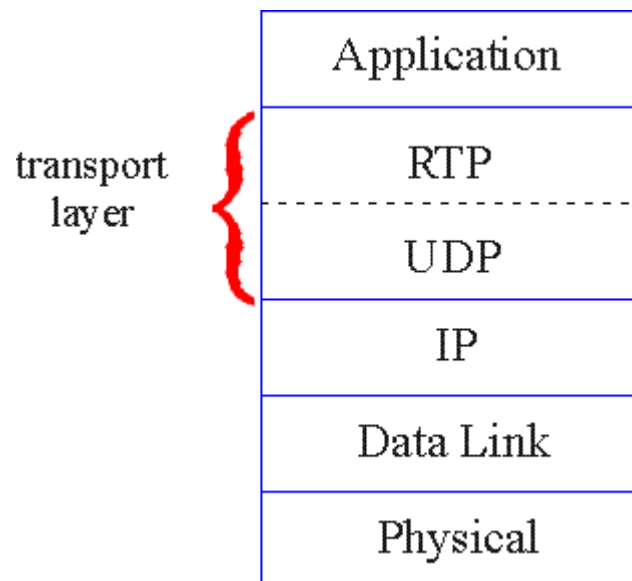


- Session Overview
- Multimedia networking applications
- Streaming stored audio and video
- Making the best out of best effort service
-  ■ Protocols for real-time interactive applications
RTP, RTCP, SIP
- Providing multiple classes of service
- Providing QoS guarantees
- Summary

- RTP specifies packet structure for packets carrying audio, video data
- RFC 3550
- RTP packet provides
 - » payload type identification
 - » packet sequence numbering
 - » time stamping
- RTP runs in end systems
- RTP packets encapsulated in UDP segments
- interoperability: if two Internet phone applications run RTP, then they may be able to work together

RTP libraries provide transport-layer interface that extends UDP:

- port numbers, IP addresses
- payload type identification
- packet sequence numbering
- time-stamping



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

- RTP does **not** provide any mechanism to ensure timely data delivery or other QoS guarantees.
- RTP encapsulation is only seen at end systems (not) by intermediate routers.
 - » routers providing best-effort service, making no special effort to ensure that RTP packets arrive at destination in timely matter.



RTP Header

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

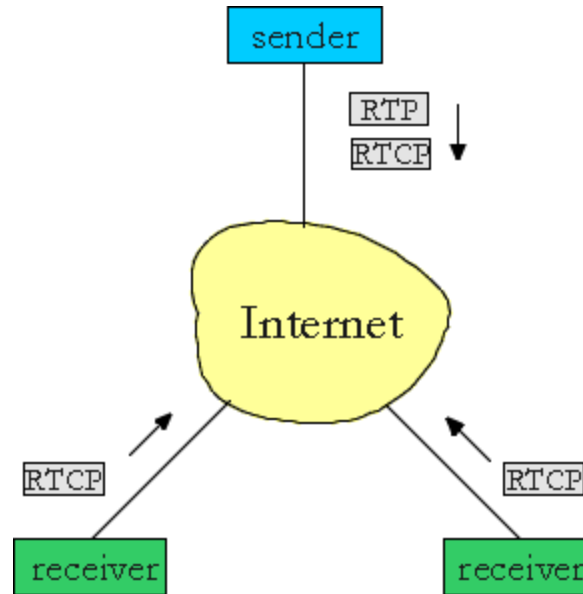
- Timestamp field (32 bytes long): sampling instant of first byte in this RTP data packet
 - » for audio, timestamp clock typically increments by one for each sampling period (for example, each 125 usecs for 8 KHz sampling clock)
 - » if application generates chunks of 160 encoded samples, then timestamp 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 source of t RTP stream. Each stream in RTP session should have distinct SSRC.

- build a server that encapsulates stored video frames into RTP packets
 - » grab video frame, add RTP headers, create UDP segments, send segments to UDP socket
 - » include seq numbers and time stamps
 - » client RTP provided for you
- also write client side of RTSP
 - » issue play/pause commands
 - » server RTSP provided for you

Real-Time Control Protocol (RTCP)

- works in conjunction with RTP.
- each participant in RTP session periodically transmits RTCP control packets to all other participants.
- each RTCP packet contains sender and/or receiver reports
 - » report statistics useful to application: # packets sent, # packets lost, interarrival jitter, etc.
- feedback can be used to control performance
 - » sender may modify its transmissions based on feedback

RTCP - Continued



- ❑ each RTP session: typically a single multicast address; all RTP /RTCP packets belonging to session use multicast address.
- ❑ RTP, RTCP packets distinguished from each other via distinct port numbers.
- ❑ to limit traffic, each participant reduces RTCP traffic as number of conference participants increases

Receiver report packets:

- fraction of packets lost, last sequence number, average interarrival jitter

Sender report packets:

- SSRC of RTP stream, current time, number of packets sent, number of bytes sent

Source description packets:

- e-mail address of sender, sender's name, SSRC of associated RTP stream
- provide mapping between the SSRC and the user/host name

Synchronization of Streams

- RTCP can synchronize different media streams within a RTP session
- consider videoconferencing app for which each sender generates one RTP stream for video, one for audio.
- timestamps in RTP packets tied to the video, audio sampling clocks
 - » *not* tied to wall-clock time
- each RTCP sender-report packet contains (for most recently generated packet in associated RTP stream):
 - » timestamp of RTP packet
 - » wall-clock time for when packet was created.
- receivers uses association to synchronize playout of audio, video

- RTCP attempts to limit its traffic to 5% of session bandwidth.

Example

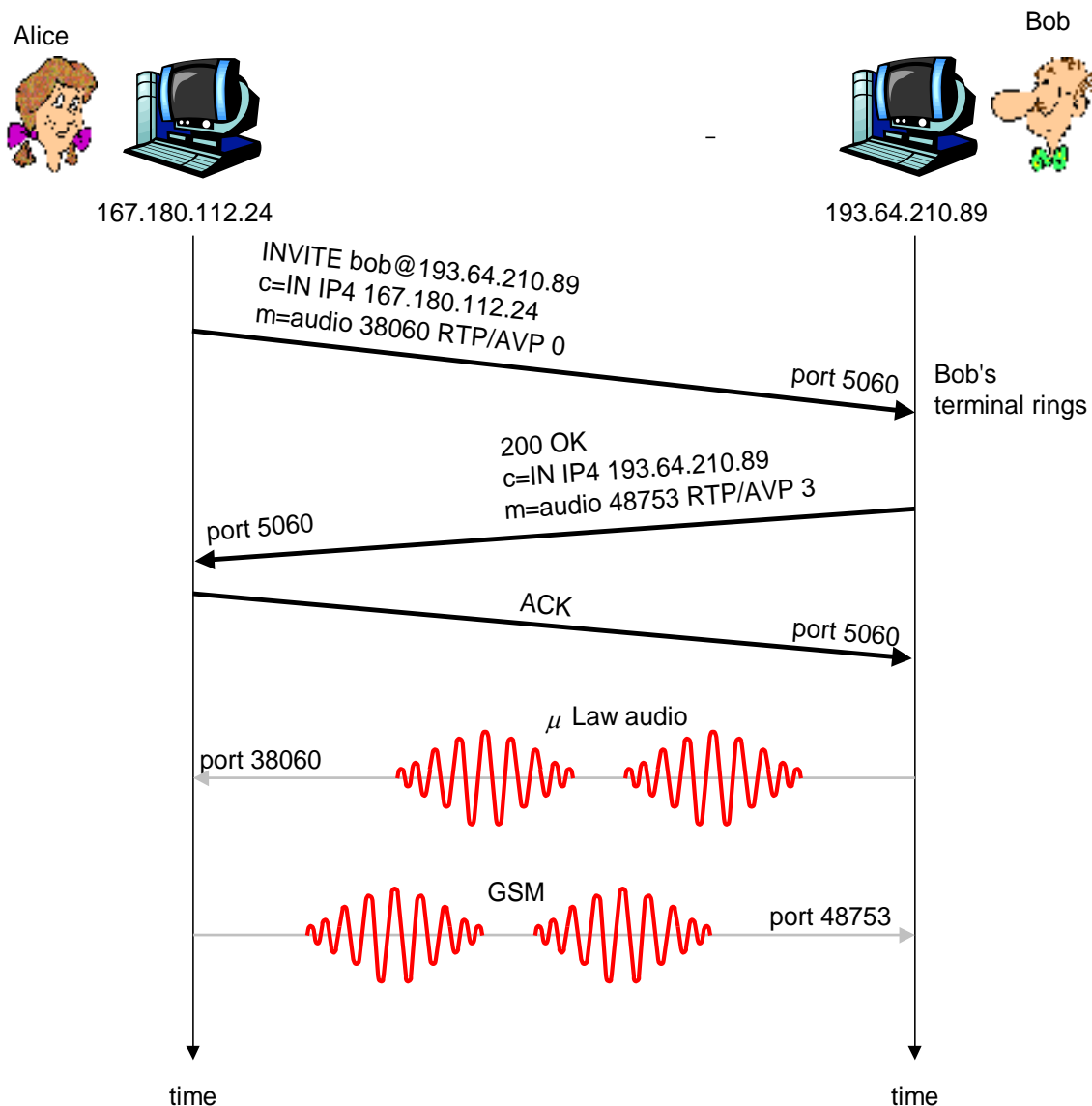
- Suppose one sender, sending video at 2 Mbps. Then RTCP attempts to limit its traffic to 100 Kbps.
- RTCP gives 75% of rate to receivers; remaining 25% to sender
- 75 kbps is equally shared among receivers:
 - » with R receivers, each receiver gets to send RTCP traffic at $75/R$ kbps.
- sender gets to send RTCP traffic at 25 kbps.
- participant determines RTCP packet transmission period by calculating avg RTCP packet size (across entire session) and dividing by allocated rate

SIP long-term vision:

- all telephone calls, video conference calls take place over Internet
- people are identified by names or e-mail addresses, rather than by phone numbers
- you can reach callee, no matter where callee roams, no matter what IP device callee is currently using

- Setting up a call, SIP provides mechanisms ..
 - » for caller to let callee know she wants to establish a call
 - » so caller, callee can agree on media type, encoding
 - » 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, hold calls

Setting up a call to known IP address



- Alice's SIP invite message indicates her port number, IP address, encoding she 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, listing his encoders Alice can then send new INVITE message, advertising different encoder
- rejecting a call
 - » Bob can reject with replies "busy," "gone," "payment required," "forbidden"
- media can be sent over RTP or some other protocol

Example of SIP message

```
INVITE sip:bob@domain.com SIP/2.0
Via: SIP/2.0/UDP 167.180.112.24
From: sip:alice@hereway.com
To: sip:bob@domain.com
Call-ID: a2e3a@pigeon.hereway.com
Content-Type: application/sdp
Content-Length: 885

c=IN IP4 167.180.112.24
m=audio 38060 RTP/AVP 0
```

Notes:

- HTTP message syntax
- sdp = session description protocol
- Call-ID is unique for every call.

❑ Here we don't know Bob's IP address. Intermediate SIP servers needed.

❑ Alice sends, receives SIP messages using SIP default port 506

❑ Alice specifies in Via: header that SIP client sends, receives SIP messages over UDP

- caller wants to call callee, but only has callee's name or e-mail address.
- need to get IP address of callee's current host:
 - » user moves around
 - » DHCP protocol
 - » user has different IP devices (PC, PDA, car device)

- result can be based on:
 - » time of day (work, home)
 - » caller (don't want boss to call you at home)
 - » status of callee (calls sent to voicemail when callee is already talking to someone)

Service provided by SIP servers:

- SIP registrar server
- SIP proxy server

- when Bob starts SIP client, client sends SIP REGISTER message to Bob's registrar server
(similar function needed by Instant Messaging)

Register Message:

```
REGISTER sip:domain.com SIP/2.0
Via: SIP/2.0/UDP 193.64.210.89
From: sip:bob@domain.com
To: sip:bob@domain.com
Expires: 3600
```

- Alice sends invite message to her proxy server
 - » contains address sip:bob@domain.com
- proxy responsible for routing SIP messages to callee
 - » possibly through multiple proxies.
- callee sends response back through the same set of proxies.
- proxy returns SIP response message to Alice
 - » contains Bob's IP address
- proxy analogous to local DNS server

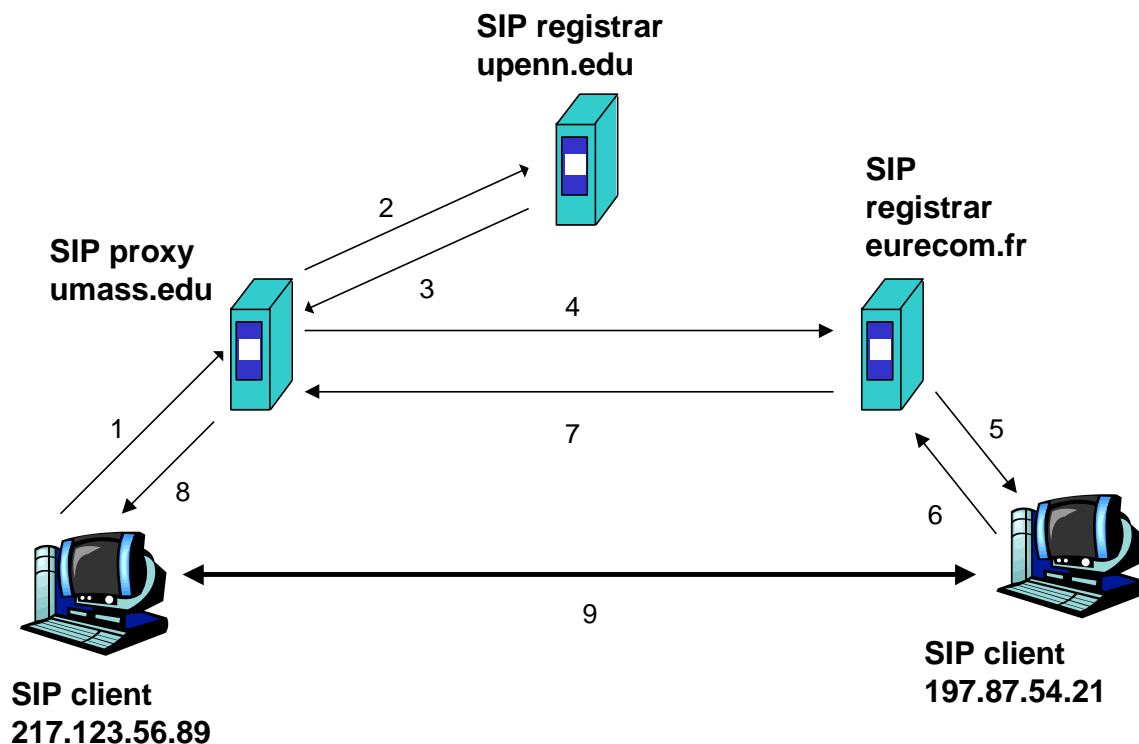
Example

Caller jim@umass.edu
with places a
call to keith@upenn.edu

(1) Jim sends INVITE message to umass SIP proxy.
(2) Proxy forwards request to upenn registrar server.
(3) upenn server returns redirect response, indicating that it should try keith@eurecom.fr

(4) umass proxy sends INVITE to eurecom registrar.
(5) eurecom registrar forwards INVITE to 197.87.54.21, which is running keith's SIP client.
(6-8) SIP response sent back
(9) media sent directly between clients.


Note: also a SIP ack message, which is not shown.



Comparison with H.323

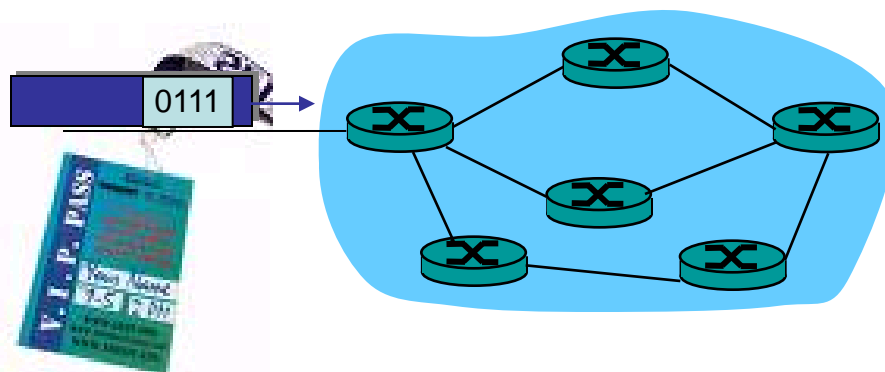
- H.323 is another signaling protocol for real-time, interactive
- H.323 is a complete, vertically integrated suite of protocols for multimedia conferencing: signaling, registration, admission control, transport, codecs
- SIP is a single component. Works with RTP, but does not mandate it. Can be combined with other protocols, services
- H.323 comes from the ITU (telephony).
- SIP comes from IETF: Borrows much of its concepts from HTTP
 - » SIP has Web flavor, whereas H.323 has telephony flavor.
- SIP uses the KISS principle: Keep it simple stupid.



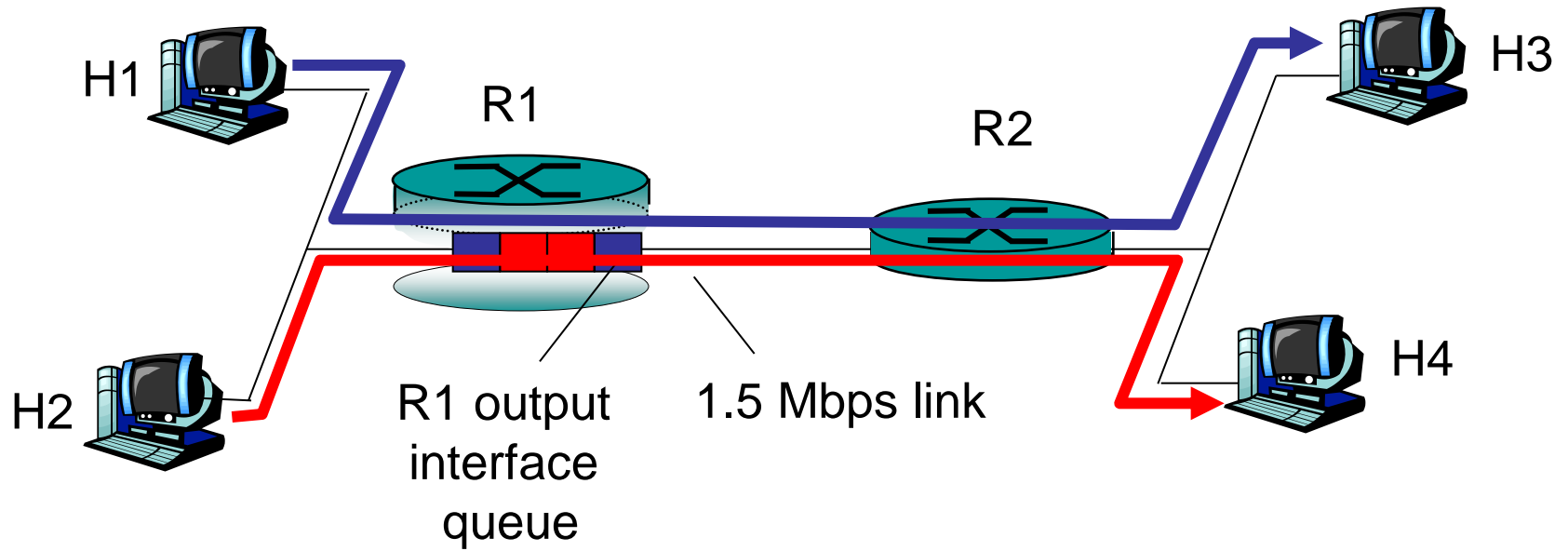
- Session Overview
- Multimedia networking applications
- Streaming stored audio and video
- Making the best out of best effort service
- Protocols for real-time interactive applications
RTP, RTCP, SIP
-  ■ Providing multiple classes of service
- Providing QoS guarantees
- Summary

Providing Multiple Classes of Service

- thus far: making the best of best effort service
 - » one-size fits all service model
 - alternative: multiple classes of service
 - » partition traffic into classes
 - » network treats different classes of traffic differently (analogy: VIP service vs regular service)
- granularity: differential service among multiple classes, not among individual connections
- history: ToS bits

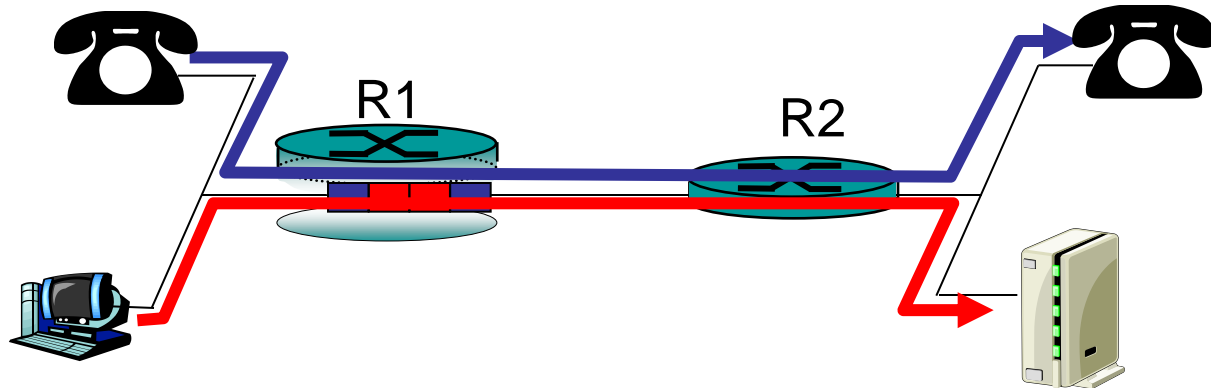


Multiple classes of service: scenario



Scenario 1: mixed FTP and audio

- Example: 1Mbps IP phone, FTP share 1.5 Mbps link.
 - » bursts of FTP can congest router, cause audio loss
 - » want to give priority to audio over FTP

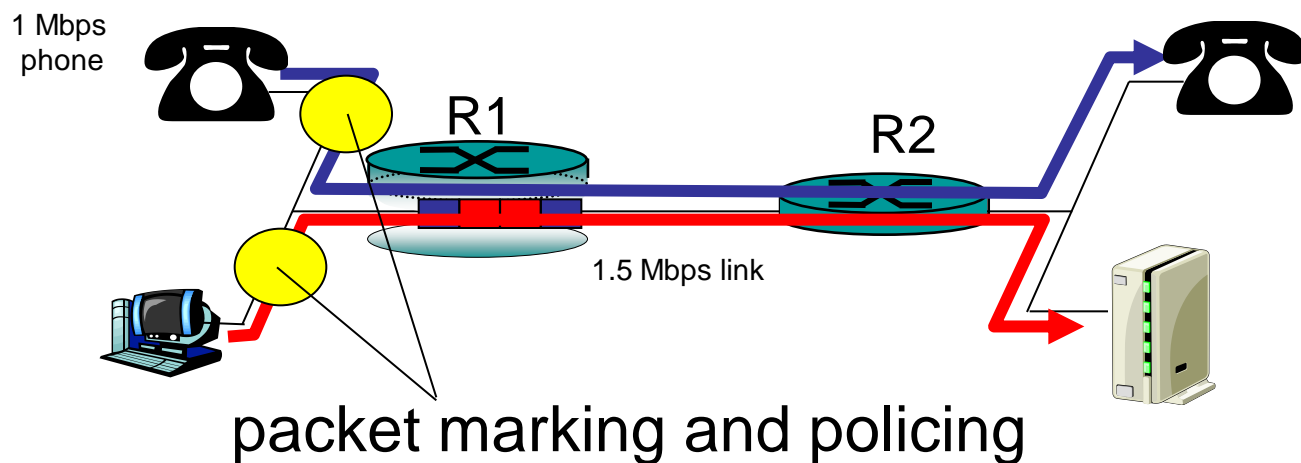


Principle 1

packet marking needed for router to distinguish between different classes; and new router policy to treat packets accordingly

Principles for QOS Guarantees (more)

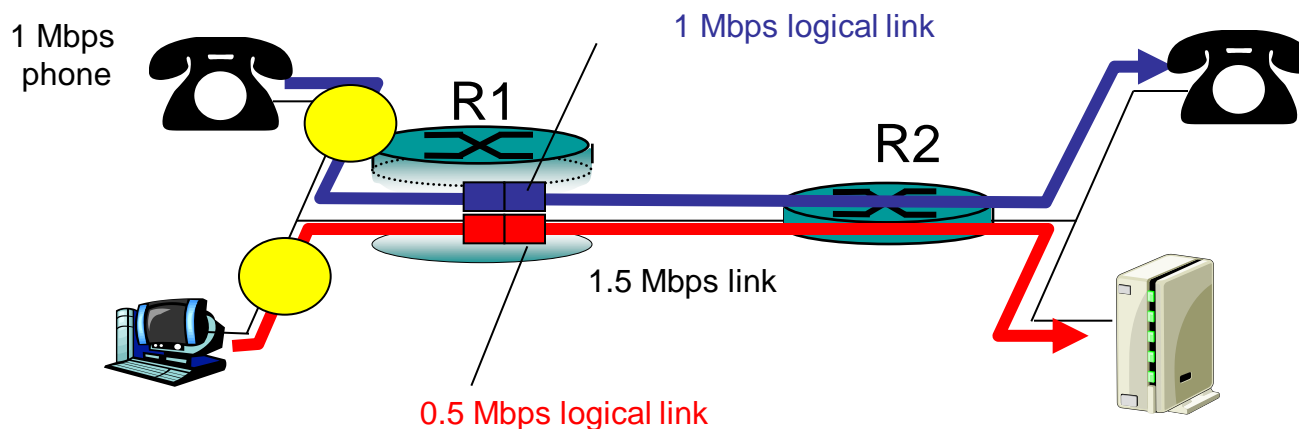
- what if applications misbehave (audio sends higher than declared rate)
 - » policing: force source adherence to bandwidth allocations
- marking and policing at network edge:
 - » similar to ATM UNI (User Network Interface)



Principle 2

provide protection (*isolation*) for one class from others

- Allocating *fixed* (non-sharable) bandwidth to flow: *inefficient* use of bandwidth if flows doesn't use its allocation

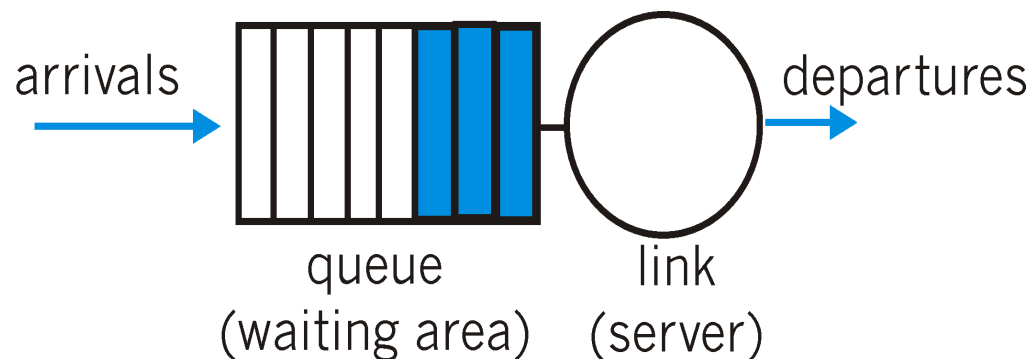


Principle 3

While providing isolation, it is desirable to use resources as efficiently as possible

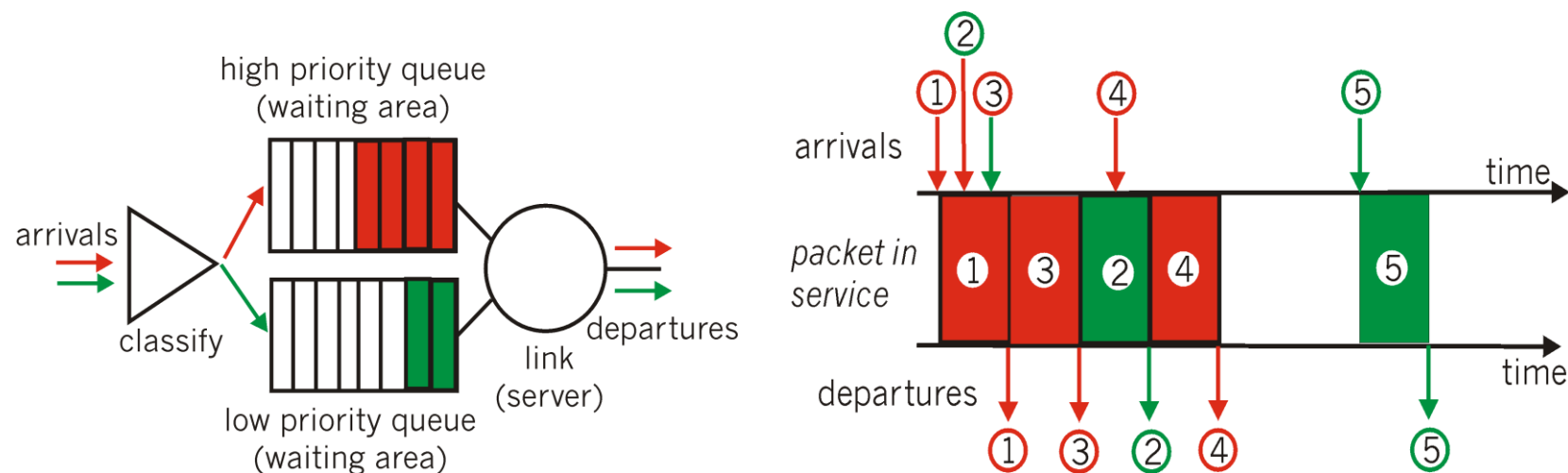
Scheduling And Policing Mechanisms

- **scheduling**: choose next packet to send on link
- **FIFO (first in first out) scheduling**: send in order of arrival to queue
 - » real-world example?
 - » **discard policy**: if packet arrives to full queue: who to discard?
 - Tail drop: drop arriving packet
 - priority: drop/remove on priority basis
 - random: drop/remove randomly



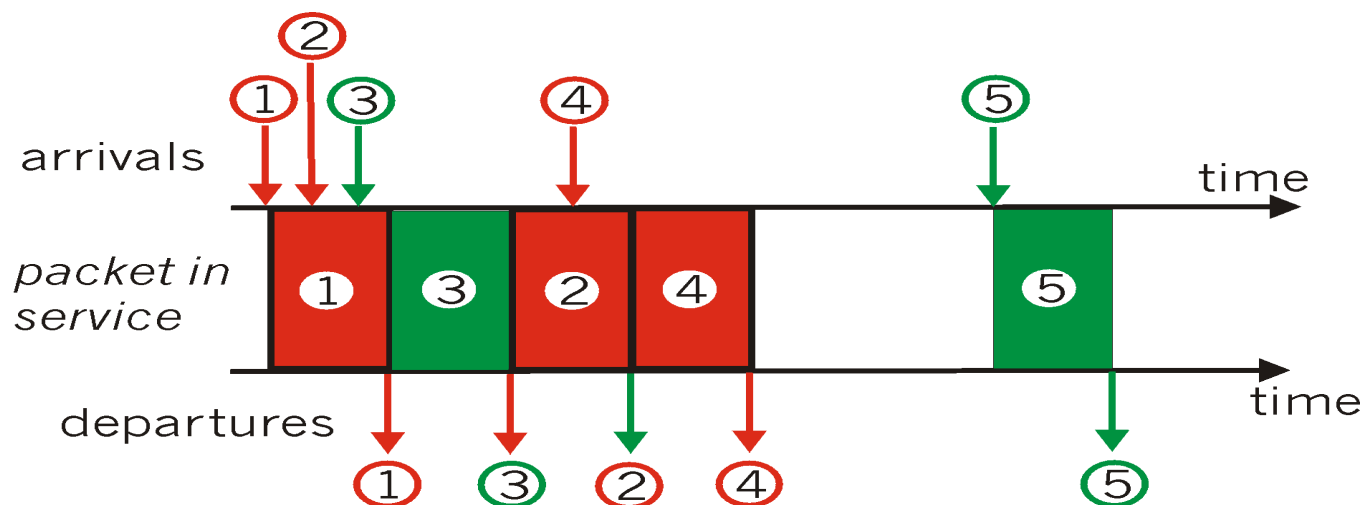
Priority scheduling: transmit highest priority queued packet

- multiple *classes*, with different priorities
 - » class may depend on marking or other header info, e.g. IP source/dest, port numbers, etc..
 - » Real world example?



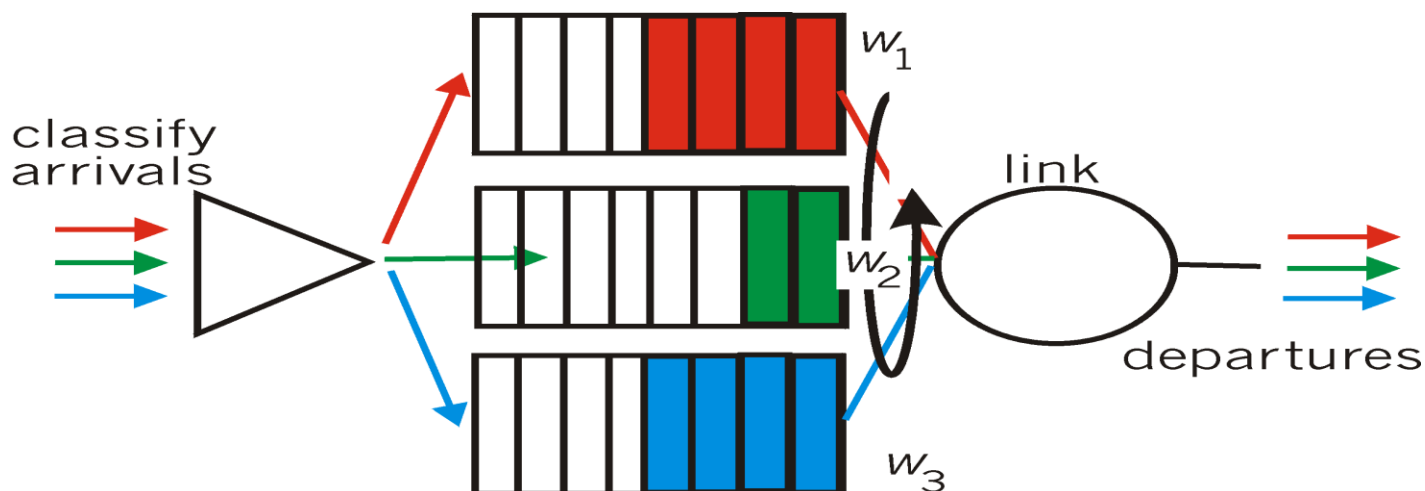
round robin scheduling:

- multiple classes
- cyclically scan class queues, serving one from each class (if available)
- real world example?



Weighted Fair Queuing:

- generalized Round Robin
- each class gets weighted amount of service in each cycle
- real-world example?

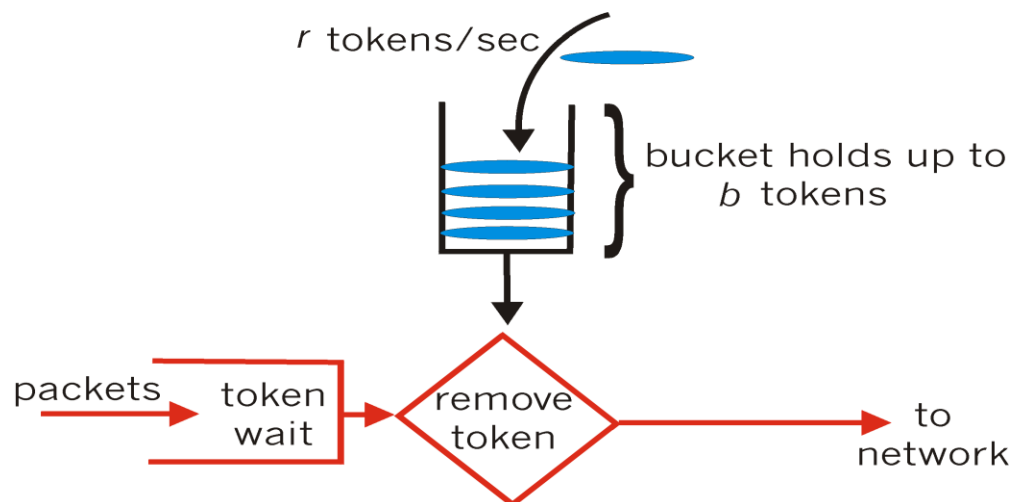


Goal: limit traffic to not exceed declared parameters

Three common-used criteria:

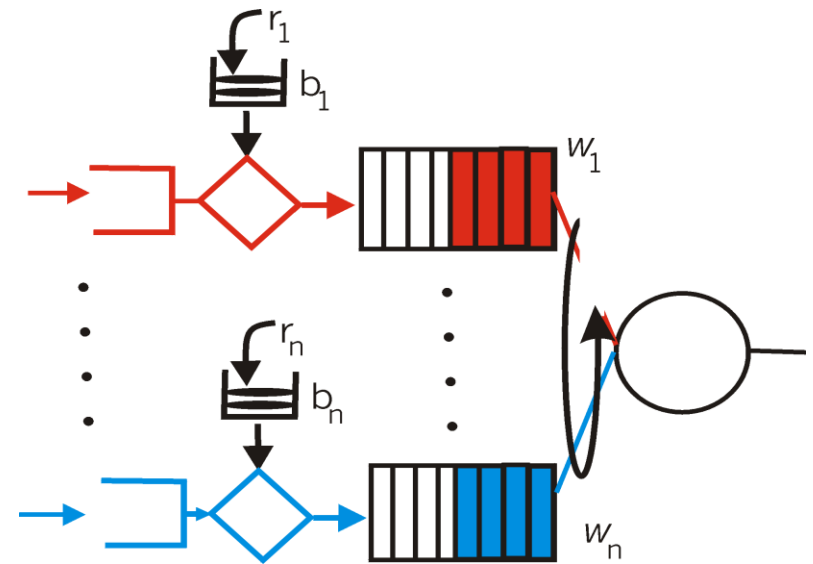
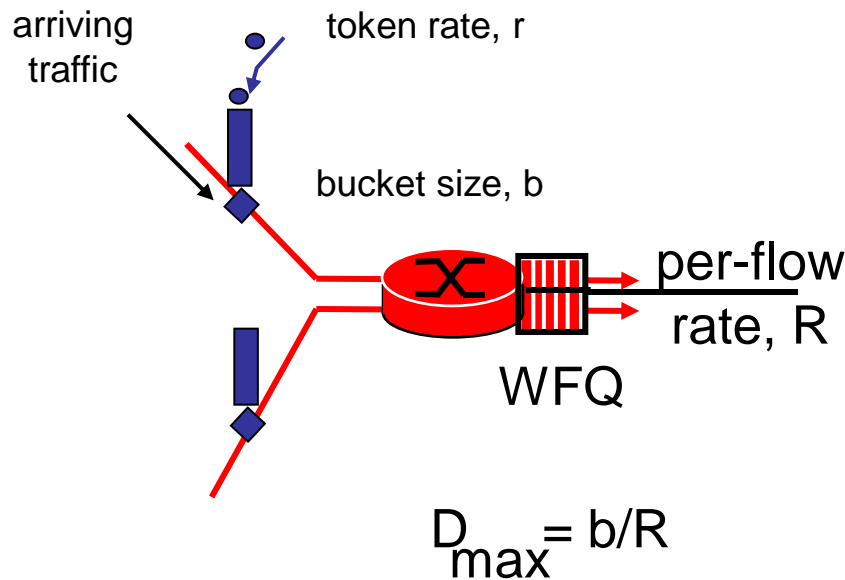
- *(Long term) Average Rate:* how many pkts can be sent per unit time (in the long run)
 - » crucial question: what is the interval length: 100 packets per sec or 6000 packets per min have same average!
- *Peak Rate:* e.g., 6000 pkts per min. (ppm) avg.; 1500 ppm peak rate
- *(Max.) Burst Size:* max. number of pkts sent consecutively (with no intervening idle)

Token Bucket: limit input to specified Burst Size and Average Rate.



- bucket can hold b tokens
- tokens generated at rate r token/sec unless bucket full
- *over interval of length t : number of packets admitted less than or equal to $(r t + b)$.*

- token bucket, WFQ combine to provide guaranteed upper bound on delay, i.e., *QoS guarantee!*



- want “qualitative” service classes
 - » “behaves like a wire”
 - » relative service distinction: Platinum, Gold, Silver
- *scalability*: simple functions in network core, relatively complex functions at edge routers (or hosts)
 - » signaling, maintaining per-flow router state difficult with large number of flows
- don't define service classes, provide functional components to build service classes

Diffserv Architecture



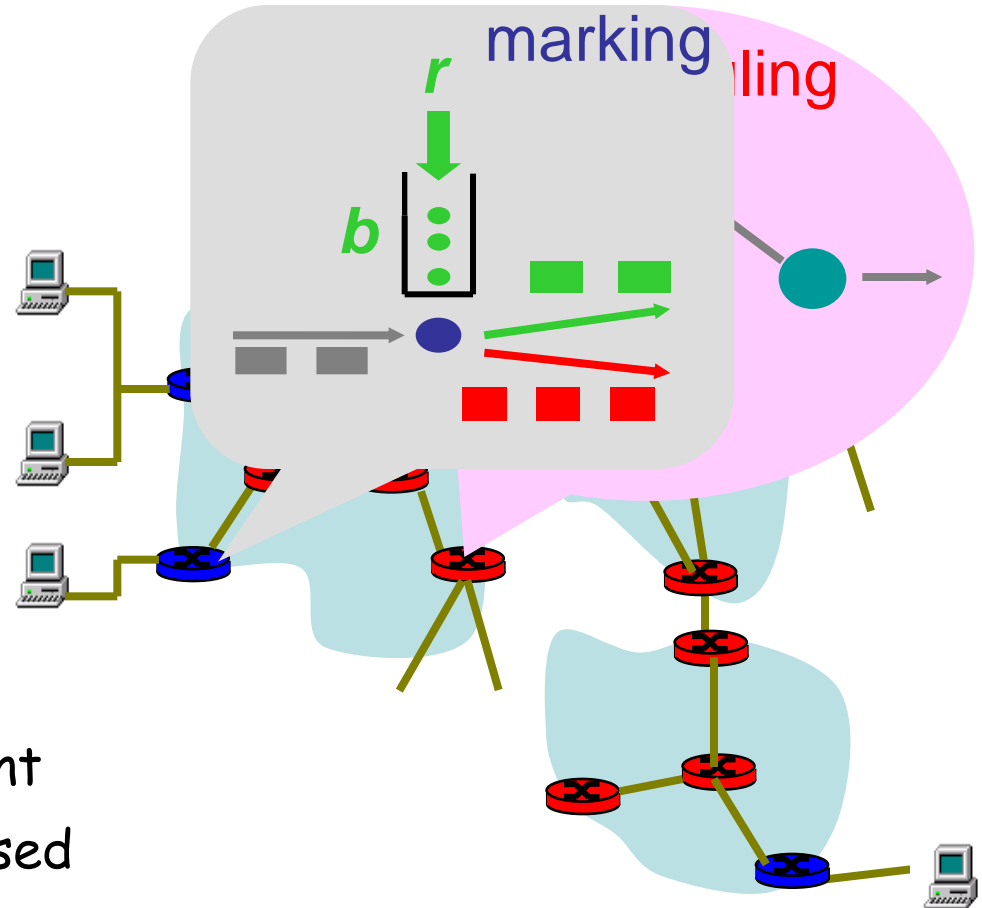
Edge router:

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



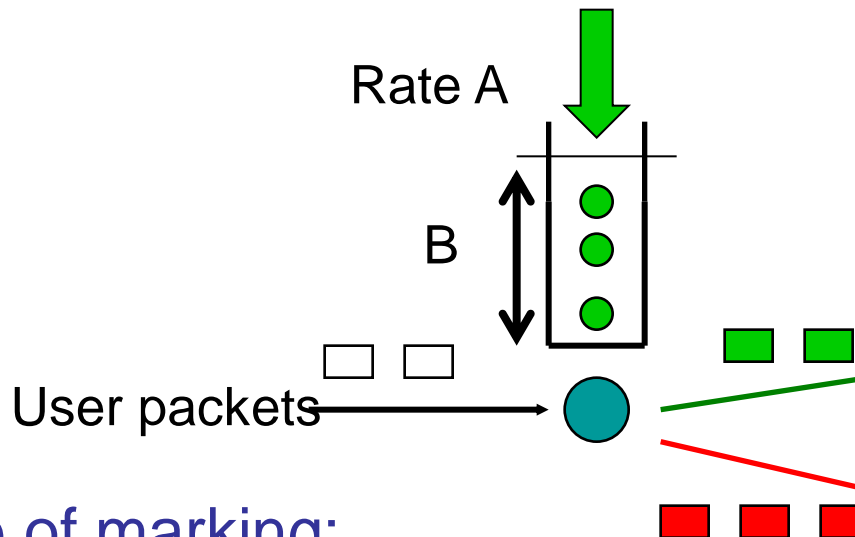
Core router:

- **per class** traffic management
- buffering and scheduling based on **marking** at edge
- preference given to **in-profile** packets



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:

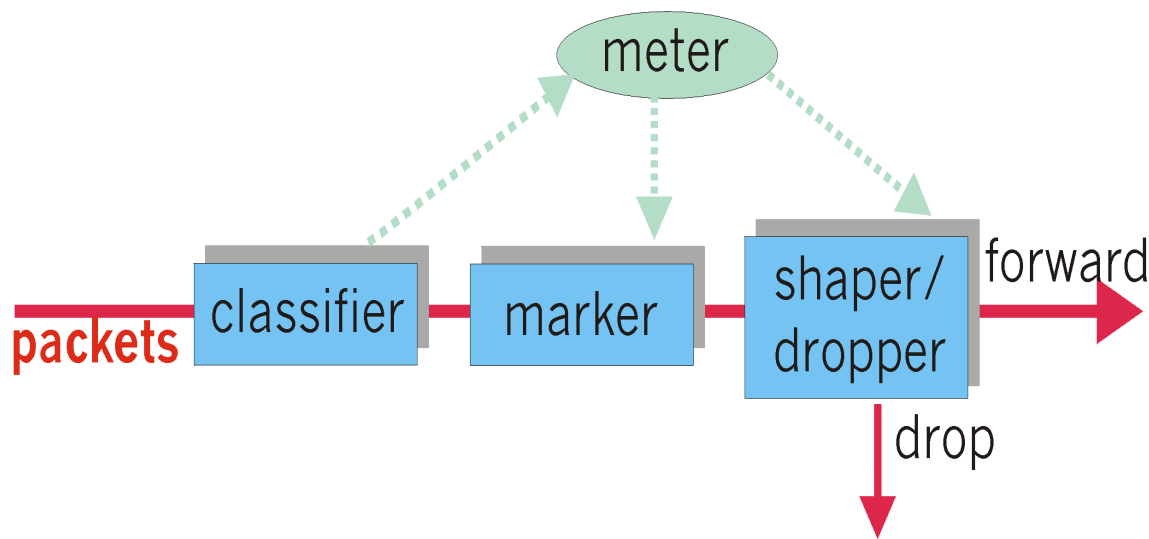
- ❑ class-based marking: packets of different classes marked differently
- ❑ intra-class marking: conforming portion of flow marked differently than non-conforming one

- 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



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




- PHB result in a different observable (measurable) forwarding performance behavior
- 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

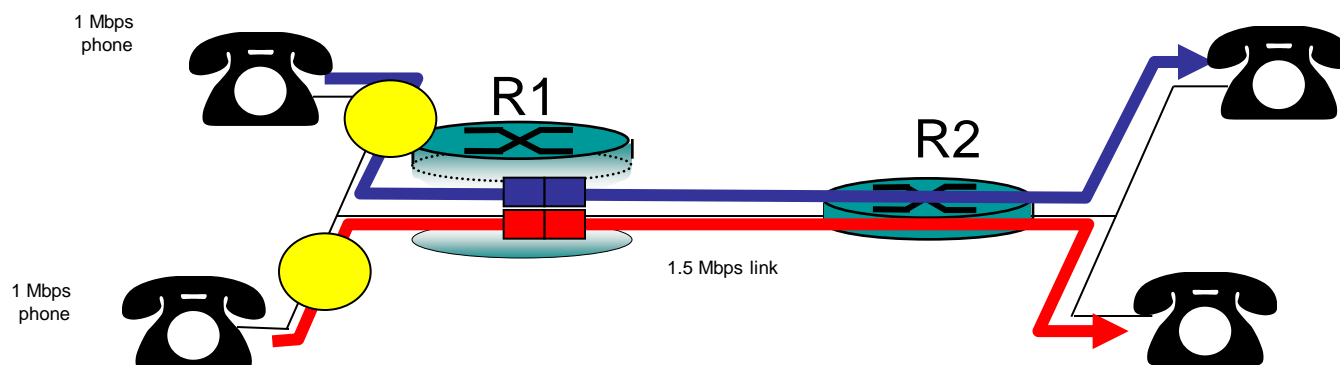
PHBs being developed:

- **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



- Session Overview
- Multimedia networking applications
- Streaming stored audio and video
- Making the best out of best effort service
- Protocols for real-time interactive applications
RTP, RTCP, SIP
- Providing multiple classes of service
-  ■ Providing QoS guarantees
- Summary

- *Basic fact of life:* can not support traffic demands beyond link capacity

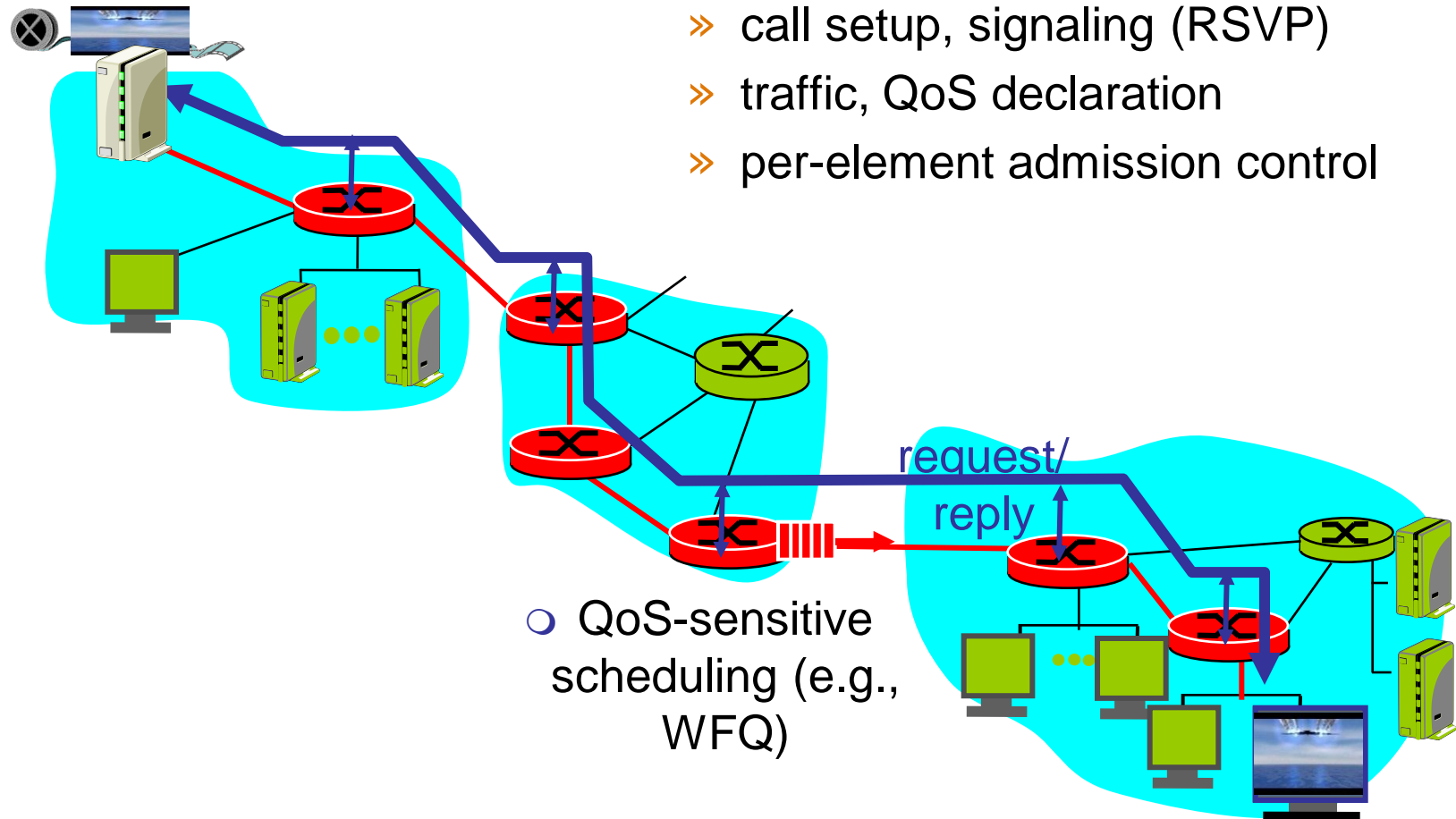


Principle 4

Call Admission: flow declares its needs, network may block call (e.g., busy signal) if it cannot meet needs

■ Resource reservation

- » call setup, signaling (RSVP)
- » traffic, QoS declaration
- » per-element admission control



- 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?

Arriving session must :

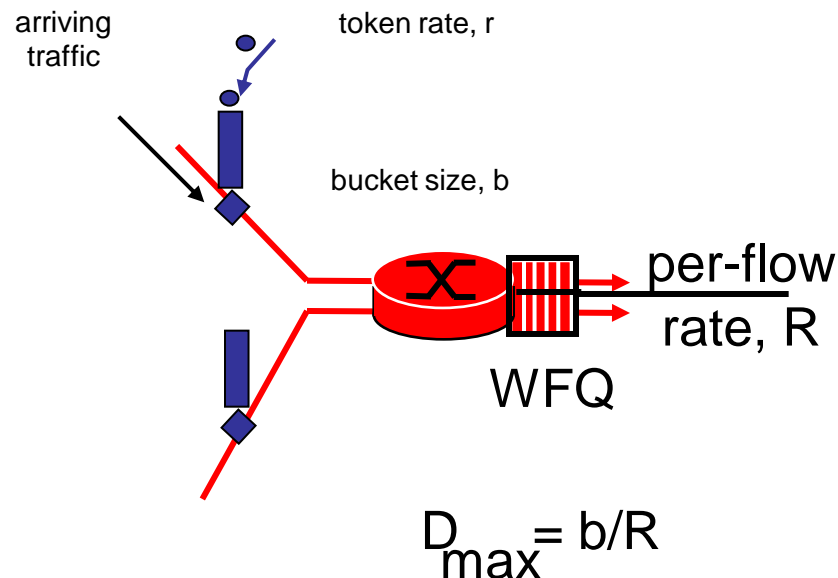
- declare its QOS requirement
 - » **R-spec**: defines the QOS being requested
- characterize traffic it will send into network
 - » **T-spec**: defines traffic characteristics
- signaling protocol: needed to carry R-spec and T-spec to routers (where reservation is required)
 - » **RSVP**

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."



connectionless (stateless)
forwarding by IP routers



best effort service



no network signaling protocols
in initial IP design

- **New requirement:** reserve resources along end-to-end path (end system, routers) for QoS for multimedia applications
- **RSVP:** Resource Reservation Protocol [RFC 2205]
 - » “ ... allow users to communicate requirements to network in robust and efficient way.” i.e., signaling !
- earlier Internet Signaling protocol: ST-II [RFC 1819]


RSVP Design Goals

1. accommodate **heterogeneous receivers** (different bandwidth along paths)
2. accommodate different applications **with different resource requirements**
3. make **multicast a first class service**, with adaptation to multicast group membership
4. **leverage existing multicast/unicast routing**, with adaptation to changes in underlying unicast, multicast routes
5. **control protocol overhead** to grow (at worst) linear in # receivers
6. **modular design** for heterogeneous underlying technologies

- ❑ specify how resources are to be reserved
 - ❑ rather: a mechanism for communicating needs
- ❑ determine routes packets will take
 - ❑ that's the job of routing protocols
 - ❑ signaling decoupled from routing
- ❑ interact with forwarding of packets
 - ❑ separation of control (signaling) and data (forwarding) planes

- senders, receiver join a multicast group
 - » done outside of RSVP
 - » senders need not join group
- sender-to-network signaling
 - » *path message*: make sender presence known to routers
 - » path teardown: delete sender's path state from routers
- receiver-to-network signaling
 - » *reservation message*: reserve resources from sender(s) to receiver
 - » reservation teardown: remove receiver reservations
- network-to-end-system signaling
 - » path error
 - » reservation error



- Session Overview
- Multimedia networking applications
- Streaming stored audio and video
- Making the best out of best effort service
- Protocols for real-time interactive applications
RTP, RTCP, SIP
- Providing multiple classes of service
- Providing QoS guarantees
-  ■ Summary

Principles

- classify multimedia applications
- identify network services applications need
- making the best of best effort service

Protocols and Architectures

- specific protocols for best-effort
- mechanisms for providing QoS
- architectures for QoS
 - » multiple classes of service
 - » QoS guarantees, admission control

Agenda

1 Session Overview

2 Multimedia Networking

3 Summary and Conclusion





- Multimedia networking applications
- Streaming stored audio and video
- Making the best out of best effort service
- Protocols for real-time interactive applications
RTP, RTCP, SIP
- Providing multiple classes of service
- Providing QoS guarantees
- Summary

Assignments & Readings

- Readings



» Chapter 7

- No Assignment

Next Session: Network Security