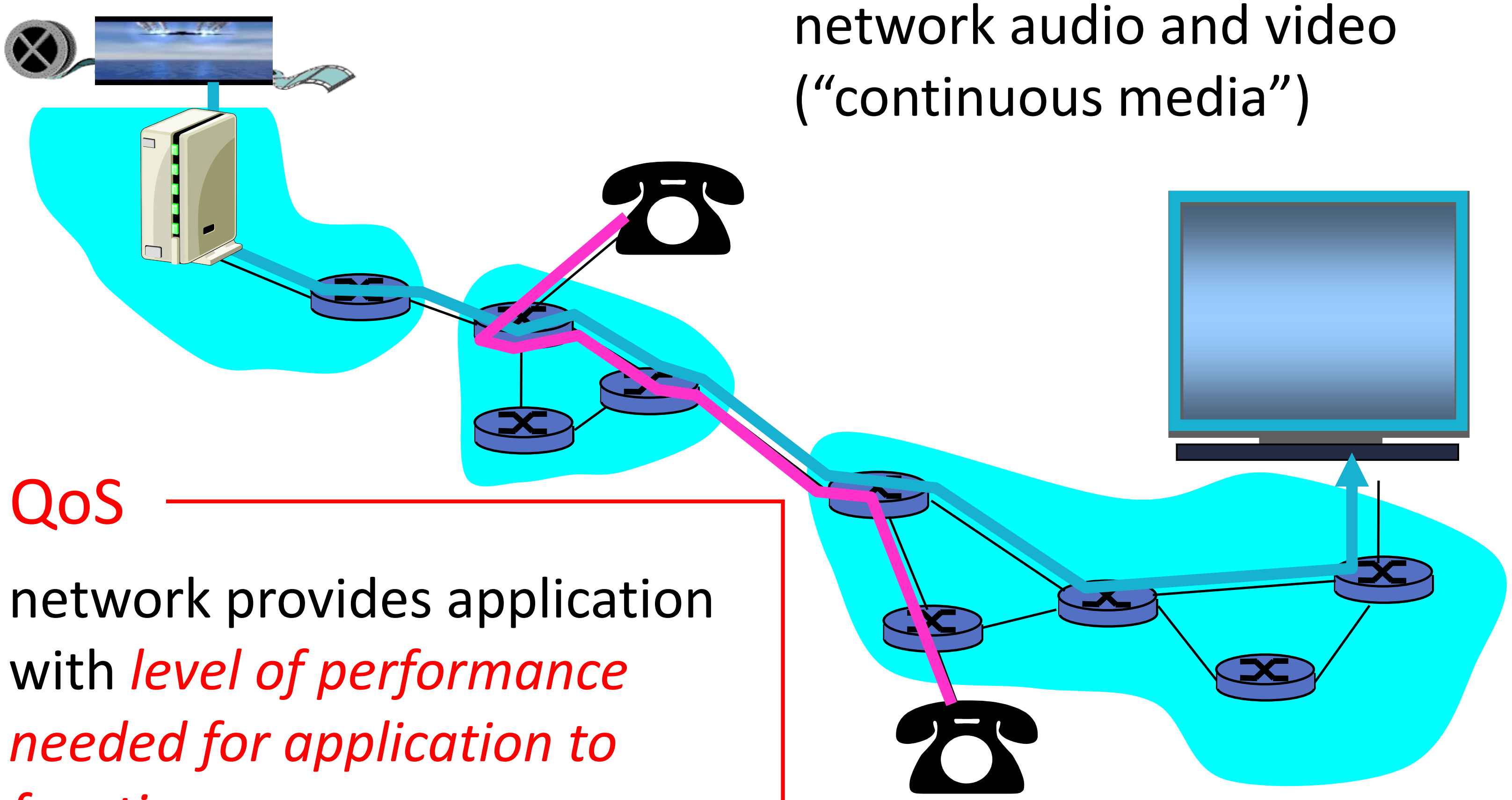
MULTI MEDIA NETWORKING

Double Tap to Add Subtitle

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.*

Chapter 7: goals

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

MM Networking Applications

Classes of MM applications:

1) stored streaming

2) live streaming

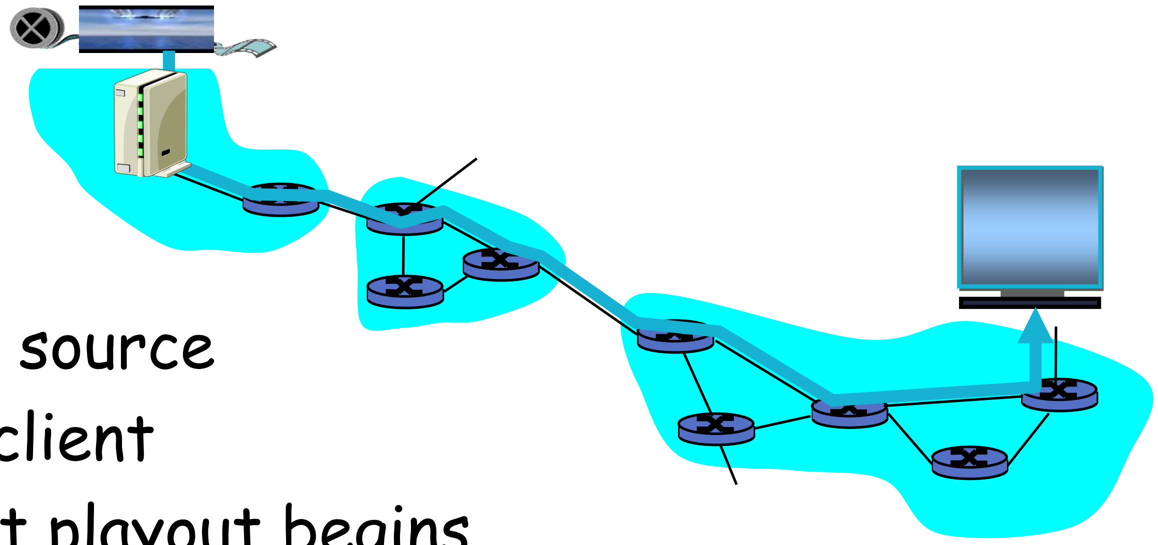
3) interactive, real-time

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*.

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

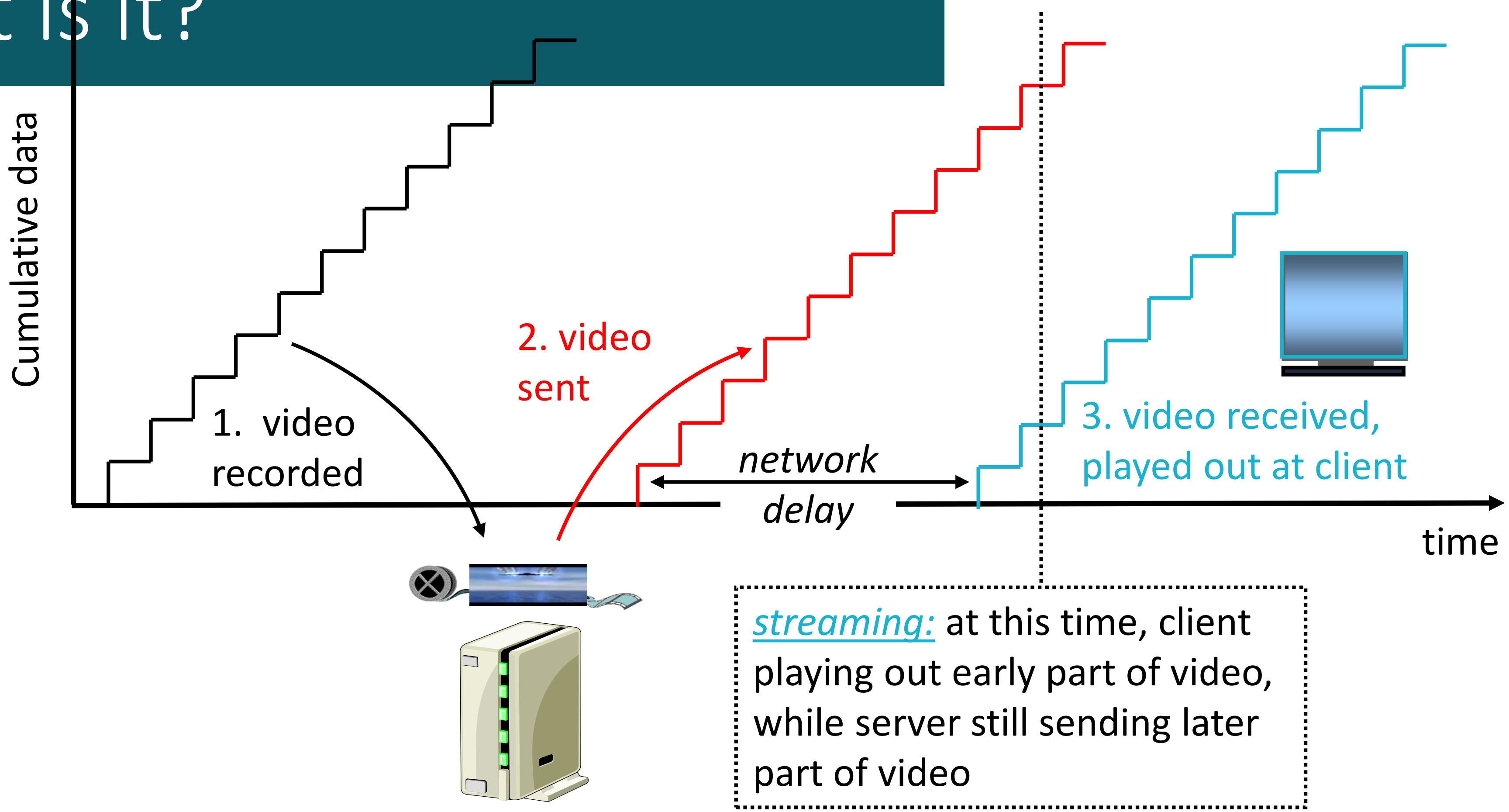
Streaming Stored Multimedia



Stored streaming:

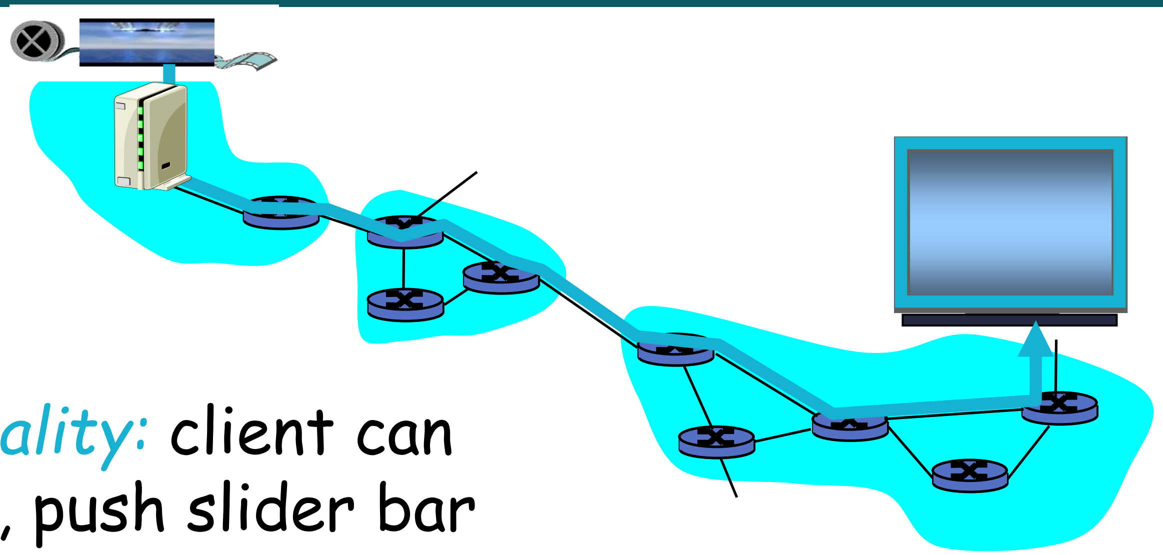
- media stored at source
- transmitted to client
- *streaming:* client playout begins *before* all data has arrived
  - timing constraint for still-to-be transmitted data: in time for playout

Streaming Stored Multimedia: What is it?



streaming: at this time, client playing out early part of video, while server still sending later part of video

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

Streaming *Live* Multimedia

Examples:

- Internet radio talk show
- live sporting event

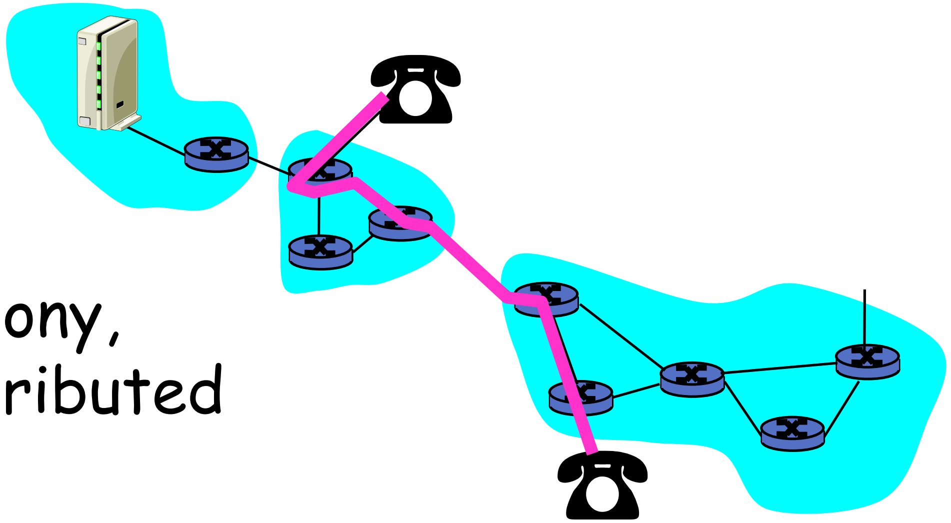
Streaming (as with streaming *stored* multimedia)

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

Interactivity

- fast forward impossible
- rewind, pause possible!

Real-Time Interactive Multimedia

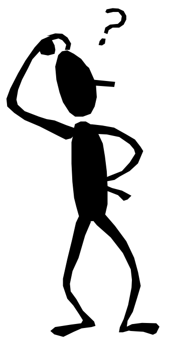


- applications: IP telephony, video conference, distributed interactive worlds
- end-end delay requirements:
  - audio: < 150 msec good, < 400 msec OK
    - includes application-level (packetization) and network delays
    - higher delays noticeable, impair interactivity
- session initialization
  - how does callee advertise its IP address, port number, encoding algorithms?

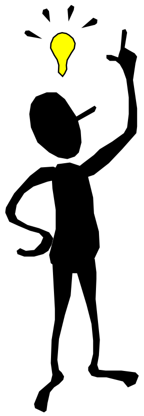
Multimedia Over Today's Internet

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

Chapter 7 outline

7.1 multimedia networking applications

7.2 streaming stored audio and video

7.3 making the best out of best effort service

7.4 protocols for real-time interactive applications

RTP,RTCP,SIP

7.5 providing multiple classes of service

7.6 providing QoS guarantees

Streaming Stored Multimedia

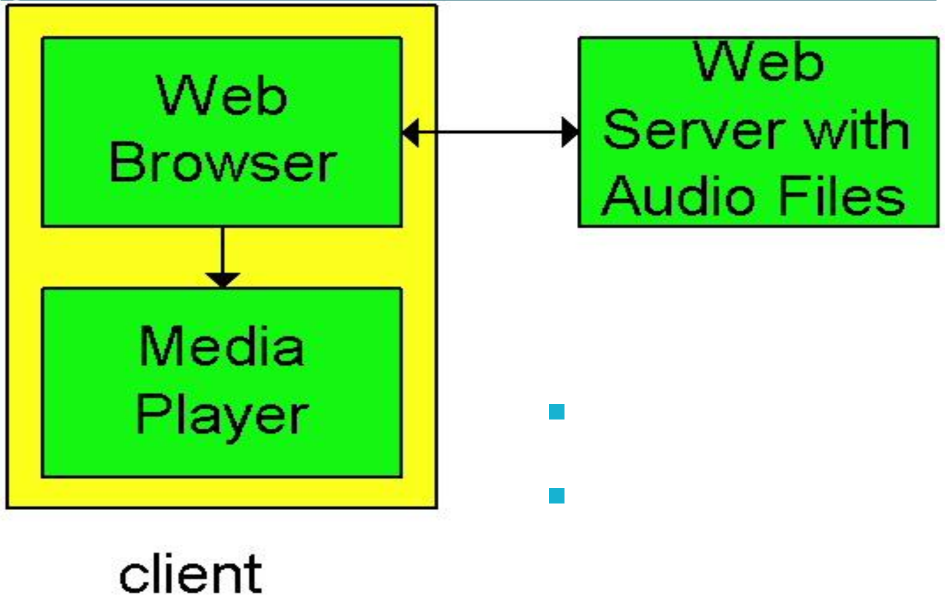
application-level streaming techniques for making the best out of best effort service:

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

Media Player

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

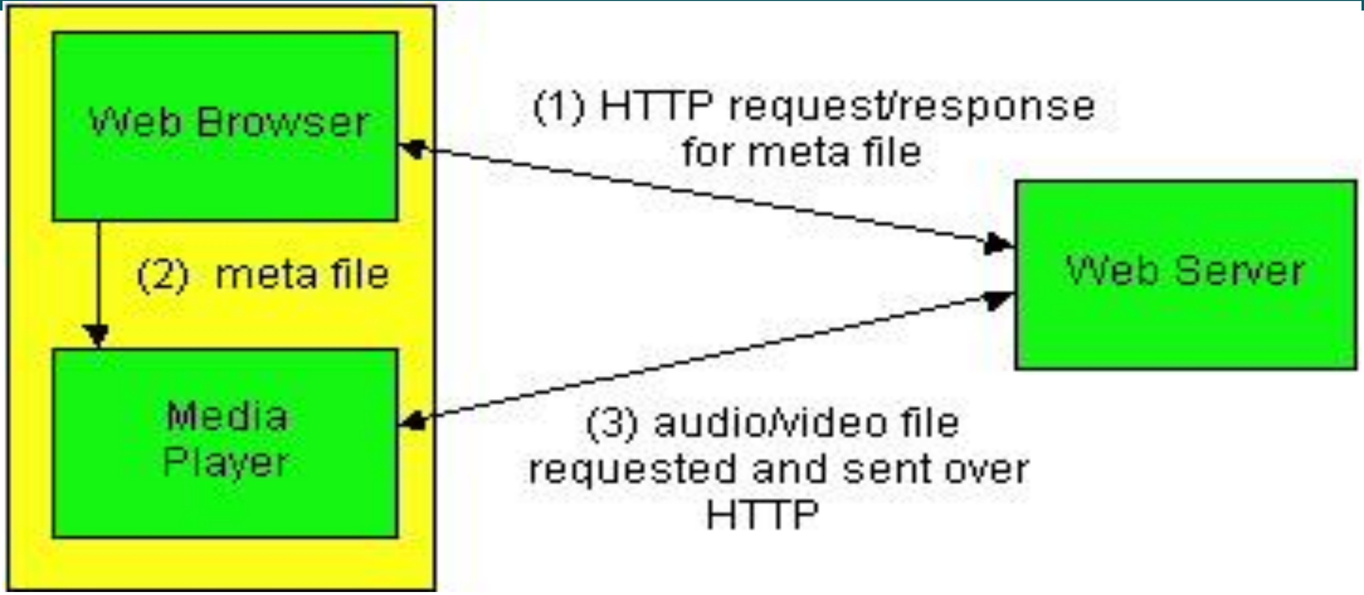
Internet multimedia: simplest approach



audio, video not streamed:

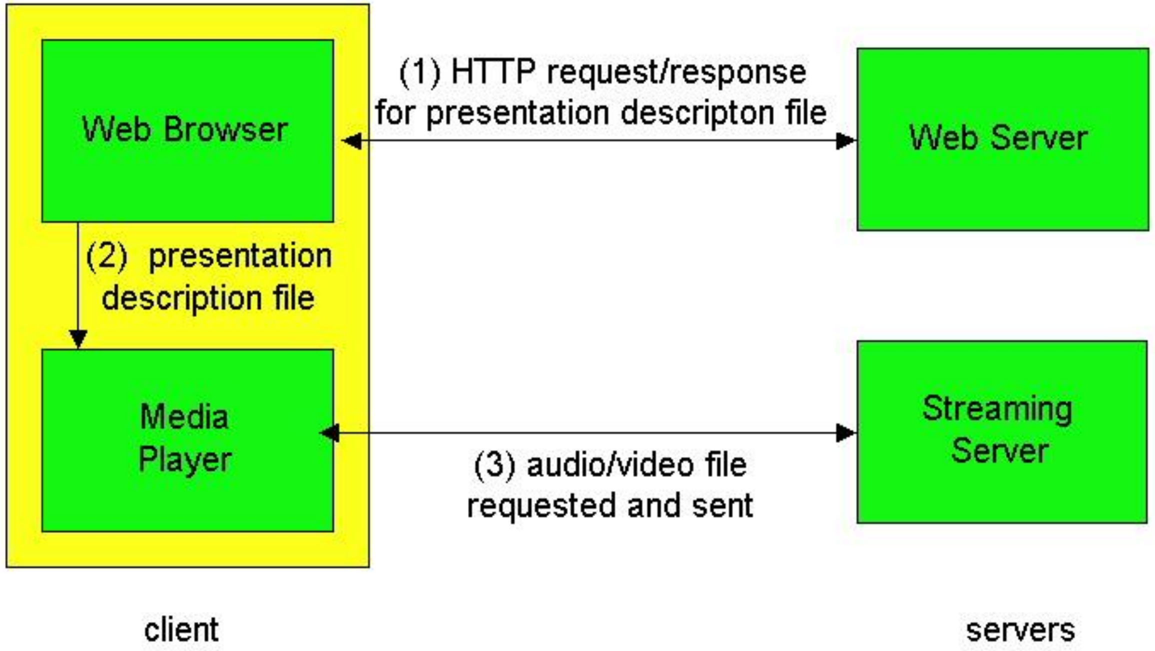
- no, "pipelining," long delays until playout!

Internet multimedia: streaming approach

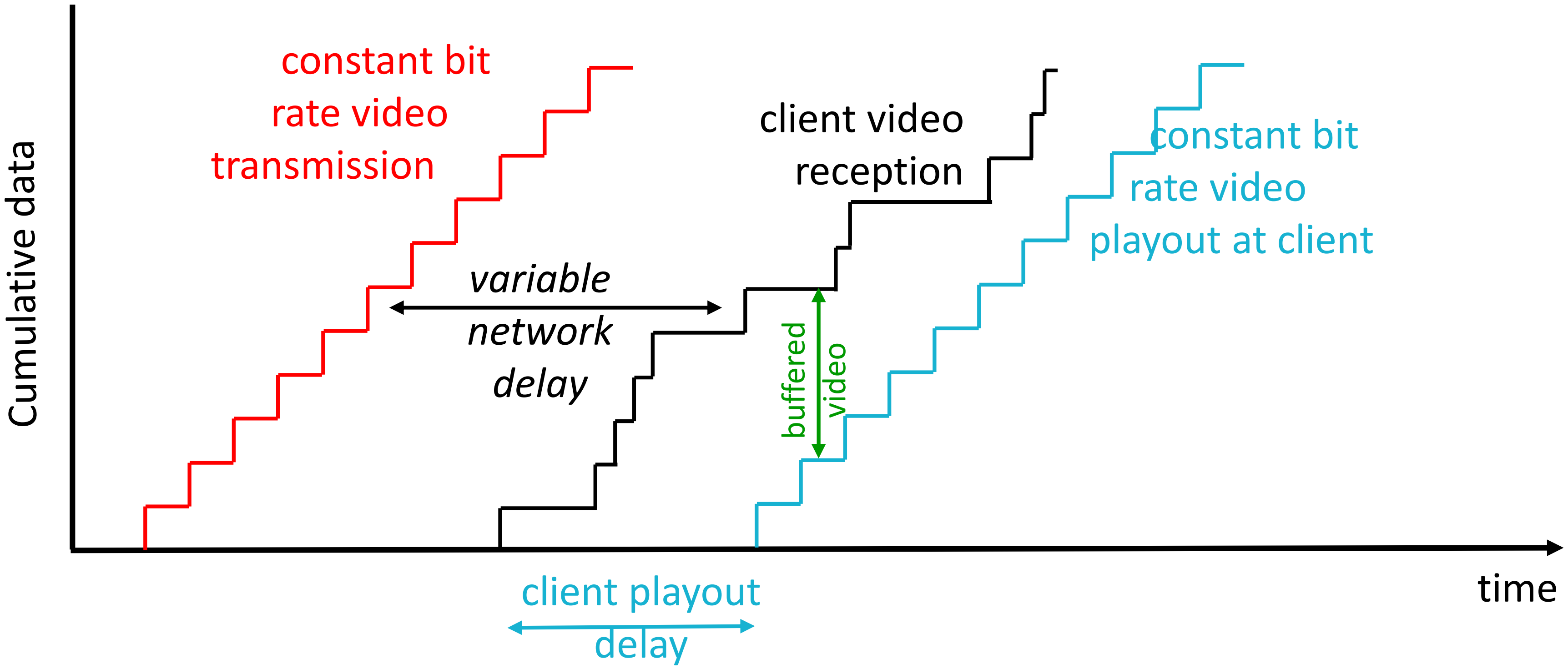


- browser GETs **metafile**
- browser launches player, passing metafile
- player contacts server
- server streams audio/video to player

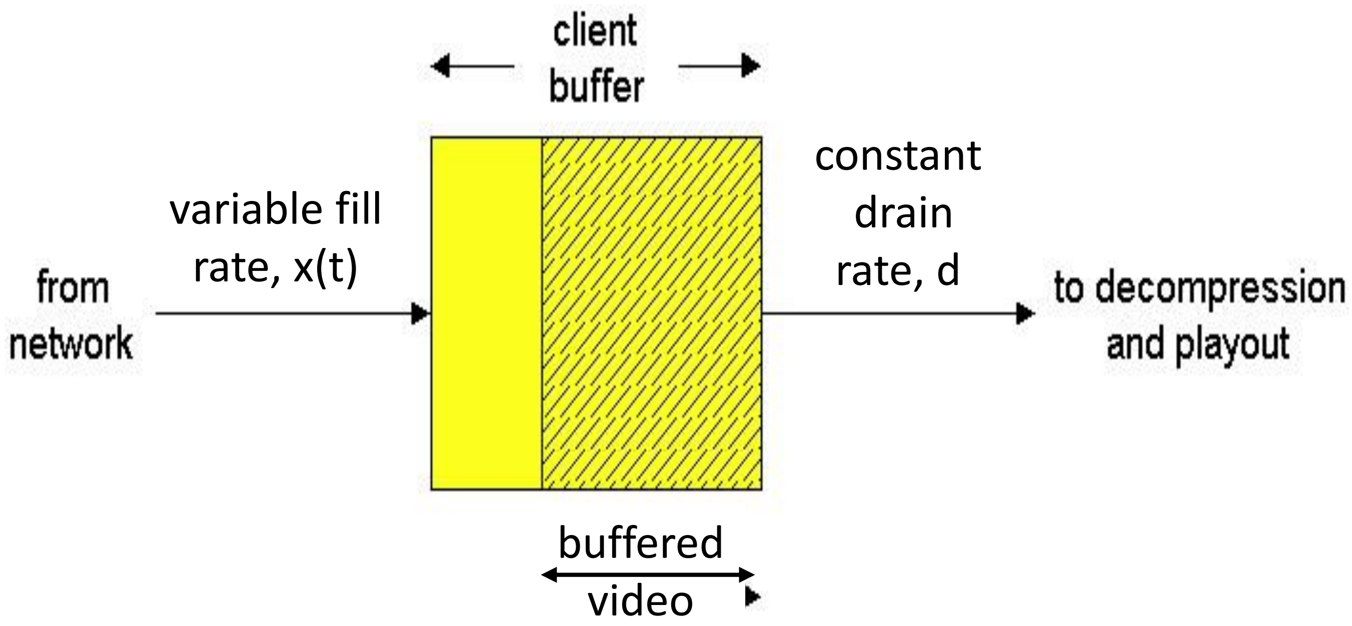
Streaming from a streaming server



Streaming Multimedia: Client Buffering



Streaming Multimedia: Client Buffering



Streaming Multimedia: UDP or TCP?

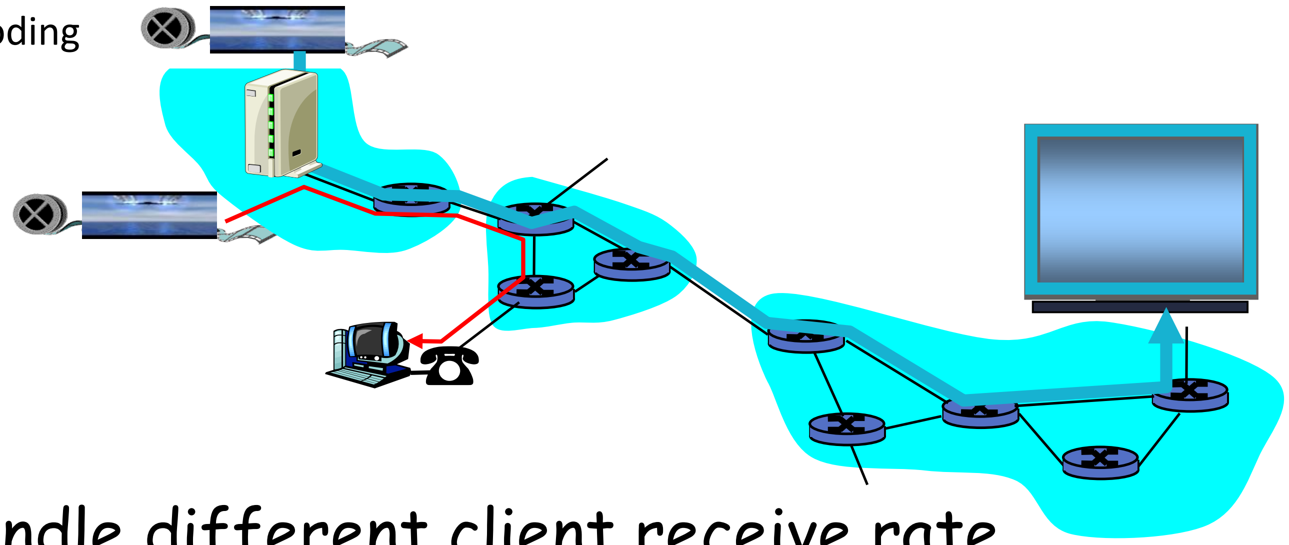
UDP

- 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 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

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
- 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

RTSP: out of band control

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".

RTSP Example

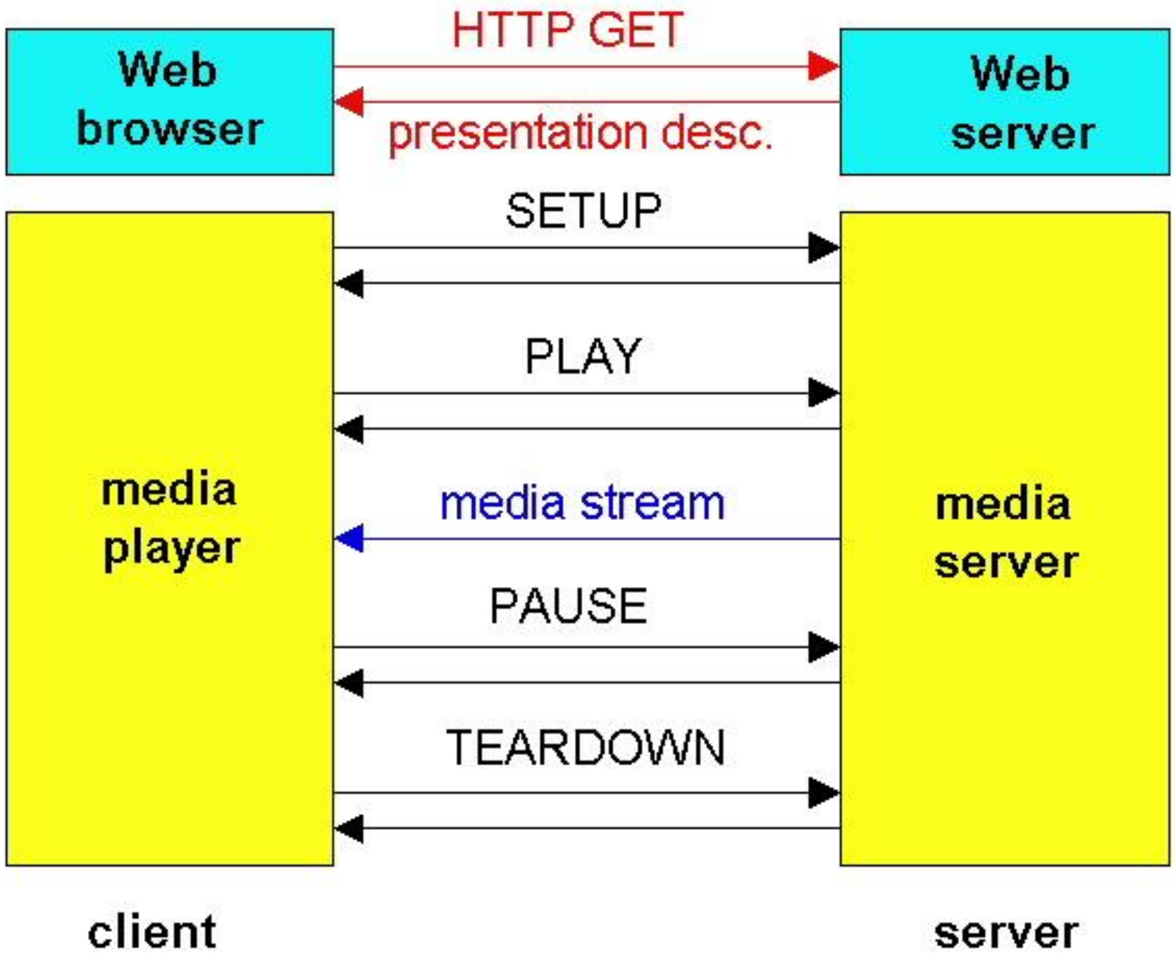
Scenario:

- 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

Real-time interactive applications

- 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

Interactive Multimedia: Internet Phone

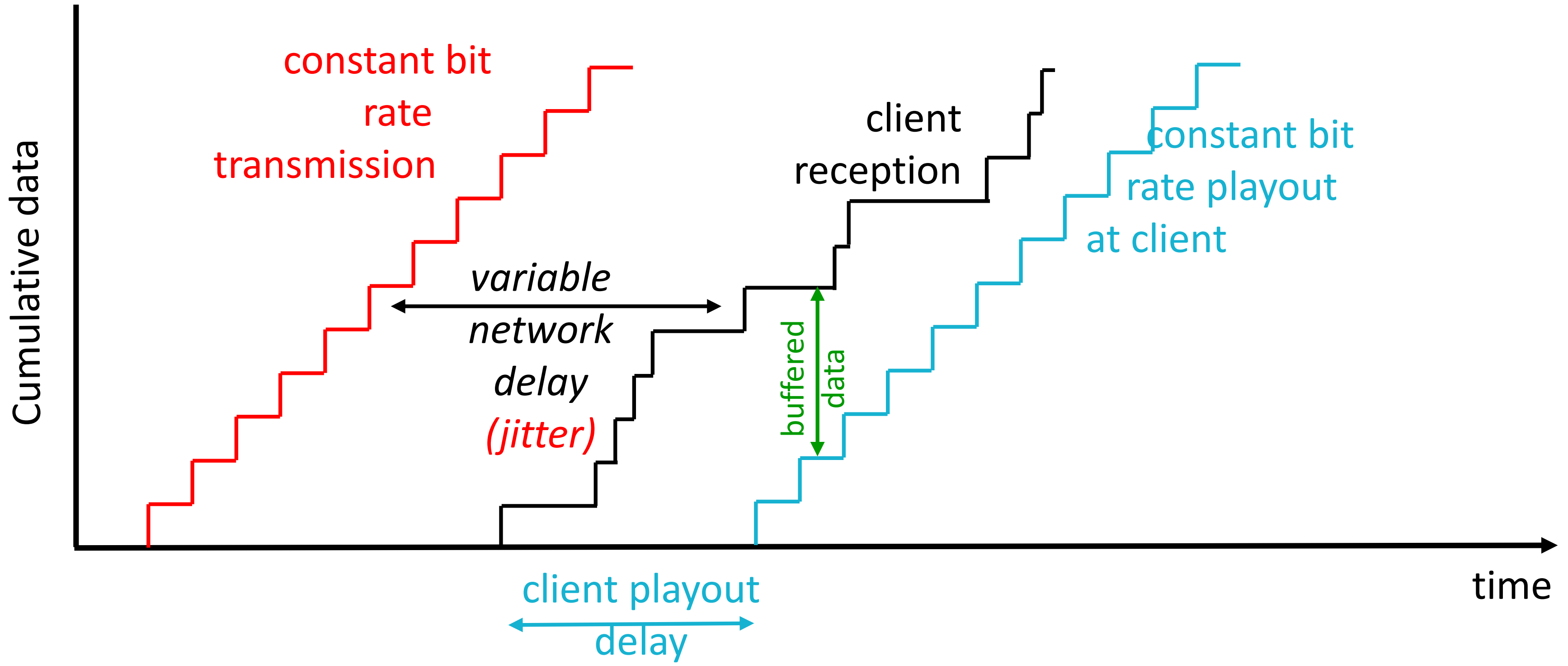
Introduce Internet Phone by way of an example

- 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

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.

Delay Jitter

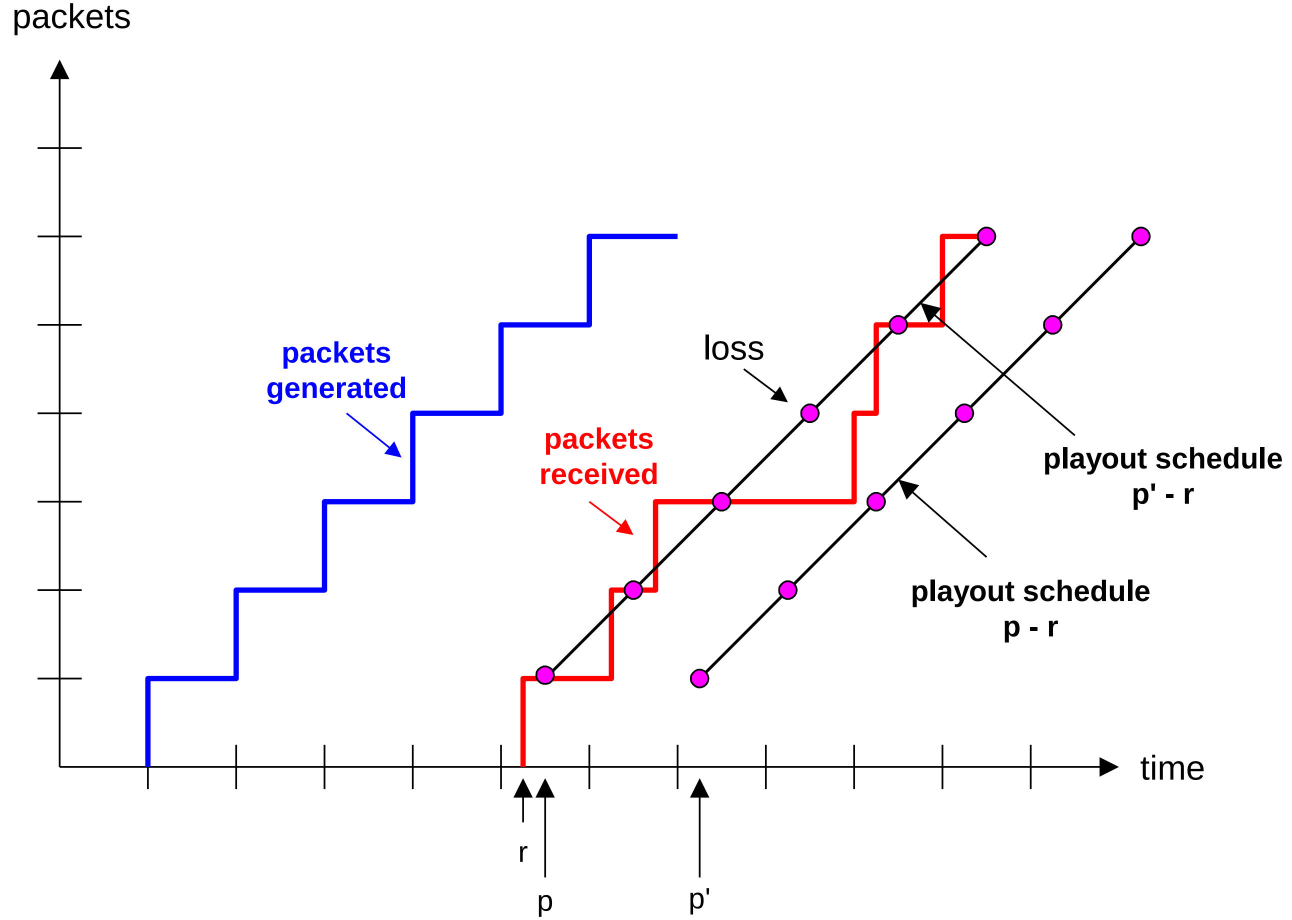


Internet Phone: Fixed Playout Delay

- 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 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.

$$\mathrm{t_i} = \text{timestamp of the ith packet}$$
$$\mathrm{r_i} = \text{the time packet i is received by receiver}$$
$$\mathrm{p_i} = \text{the time packet i is played at receiver}$$
$$\mathrm{r_i} - \mathrm{t_i} = \text{network delay for ith packet}$$
$$\mathrm{d_i} = \text{estimate of average network delay after receiving ith packet}$$

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

Adaptive Playout (3)

Q:

-

-

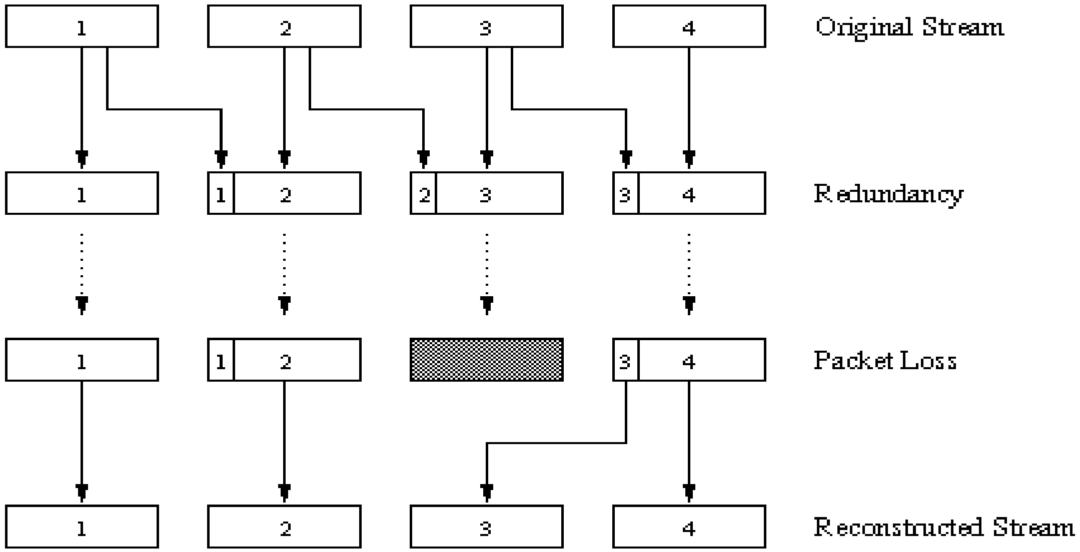
and

Forward Error Correction (FEC): simple scheme

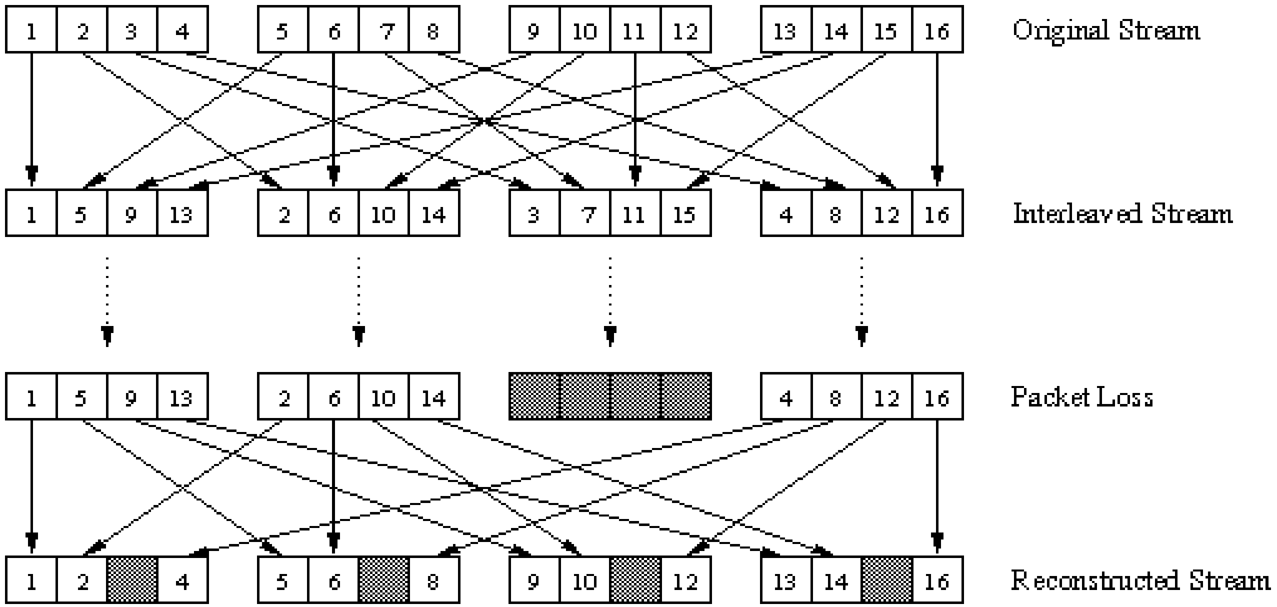
Recovery from packet loss (2)

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.



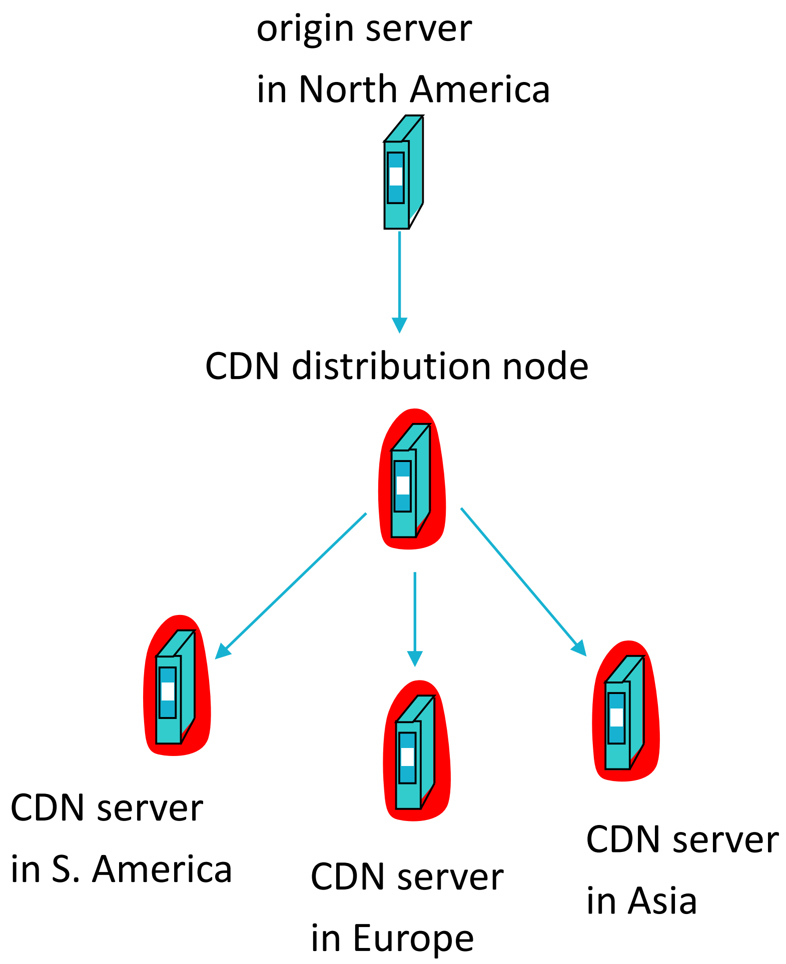
- whenever there is non-consecutive loss, receiver can conceal the loss.
- can also append (n-1)st and (n-2)nd low-bit rate chunk



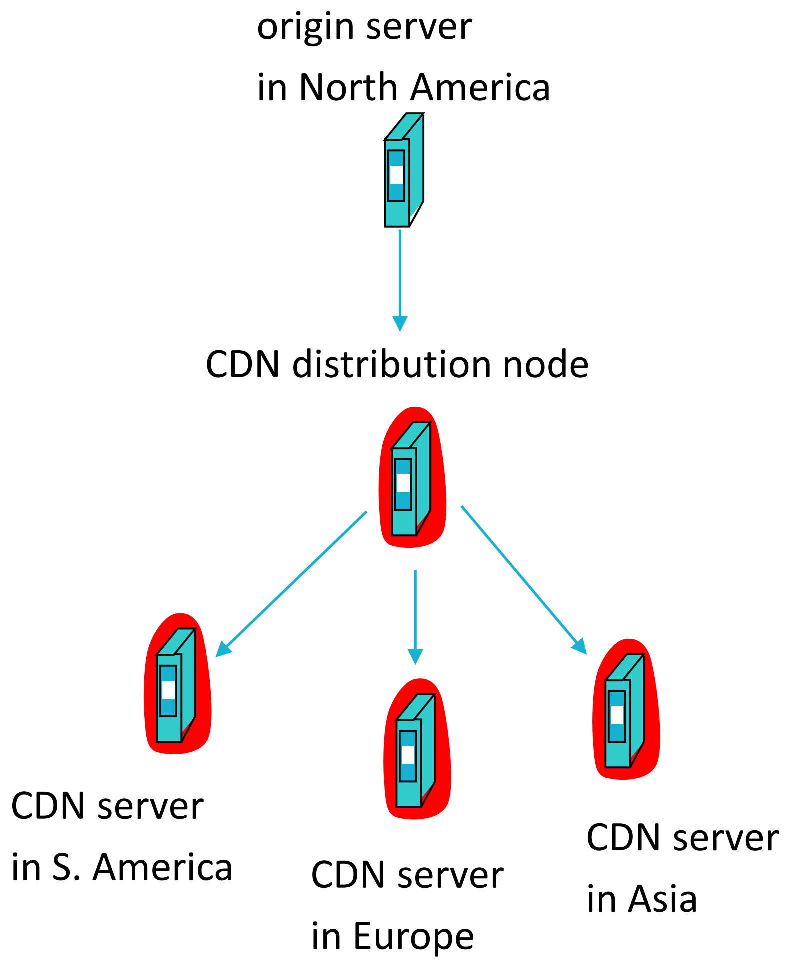
Interleaving

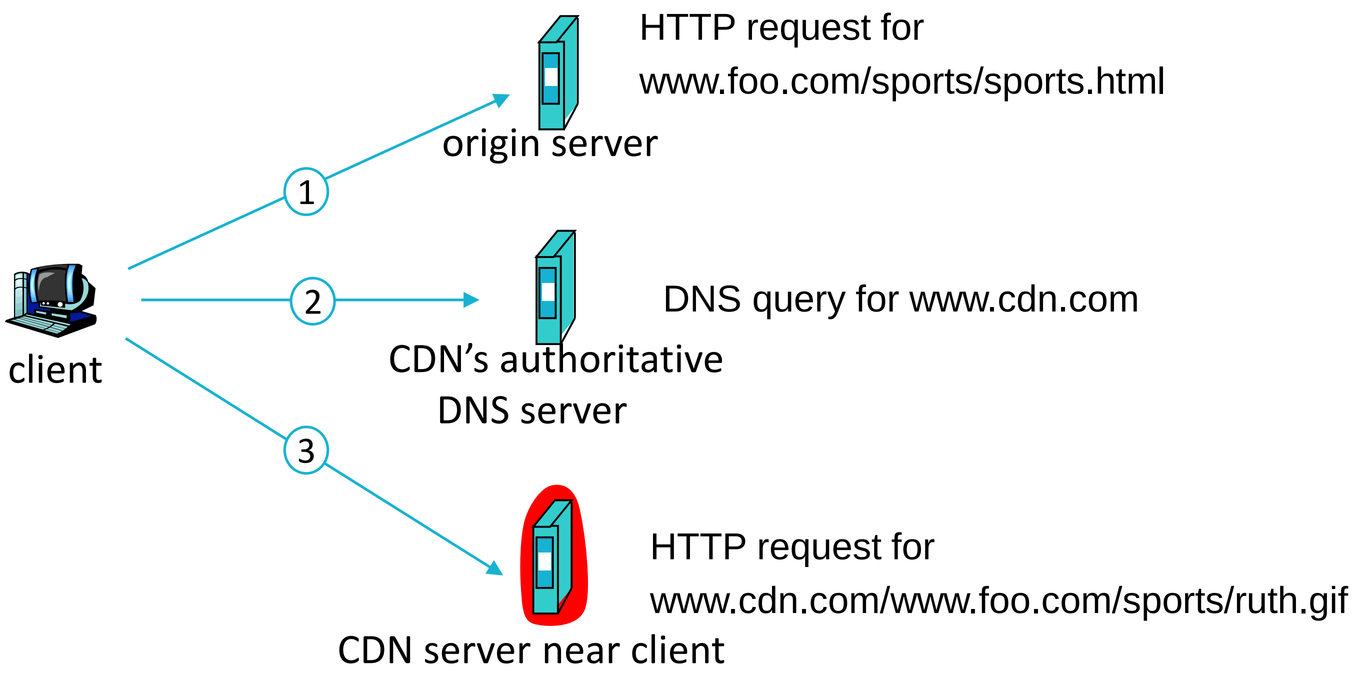
Content replication

- *placing content "close" to user avoids impairments (loss, delay) of sending content over long paths*



Content replication





origin server (www.foo.com)

-
-

CDN company (cdn.com)

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

routing requests

Summary: Internet Multimedia: bag of tricks

- use UDP
- adaptive playout delay
- matches stream bandwidth
-
-

7.1

7.2

7.3

7.4 protocols for real-time interactive applications

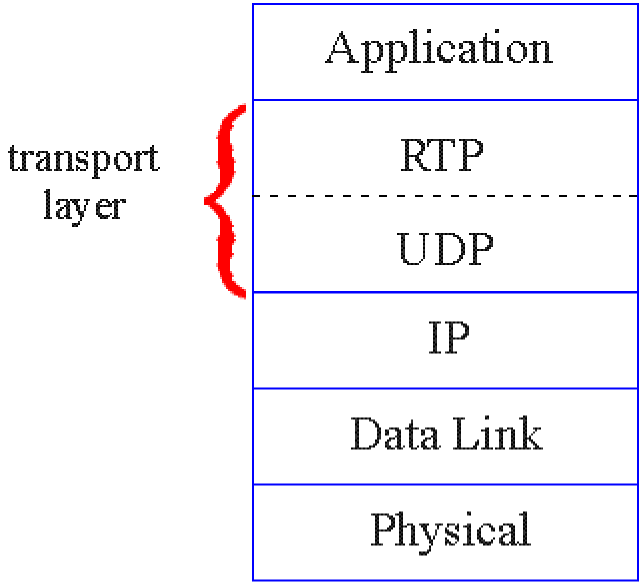
RTP, RTCP, SIP

7.5

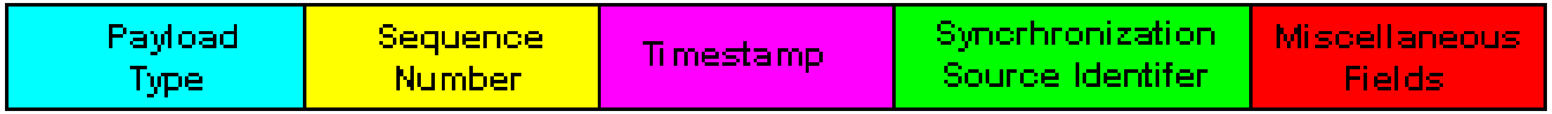
7.6

RTP libraries provide transport-layer interface that extends UDP:

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



RTP and QoS



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.

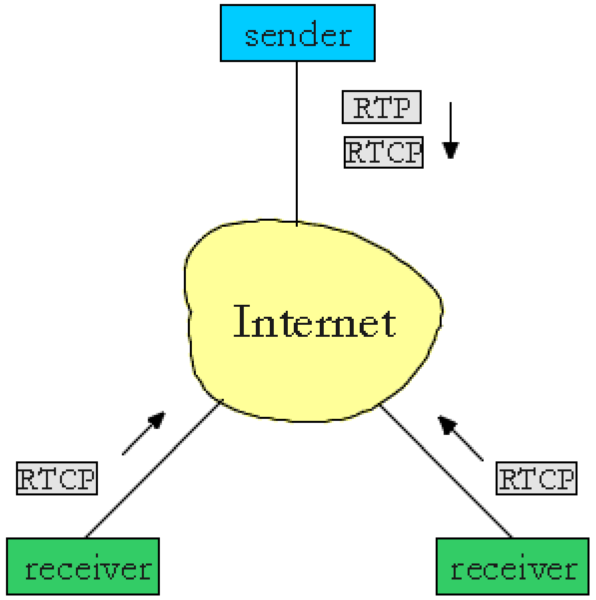
RTP Header (2)

- ***Timestamp field (32 bytes long):***

- ***SSRC field (32 bits long):***

RTSP/RTP Programming Assignment

-
-



- 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:

-

Sender report packets:

-

Source description packets:

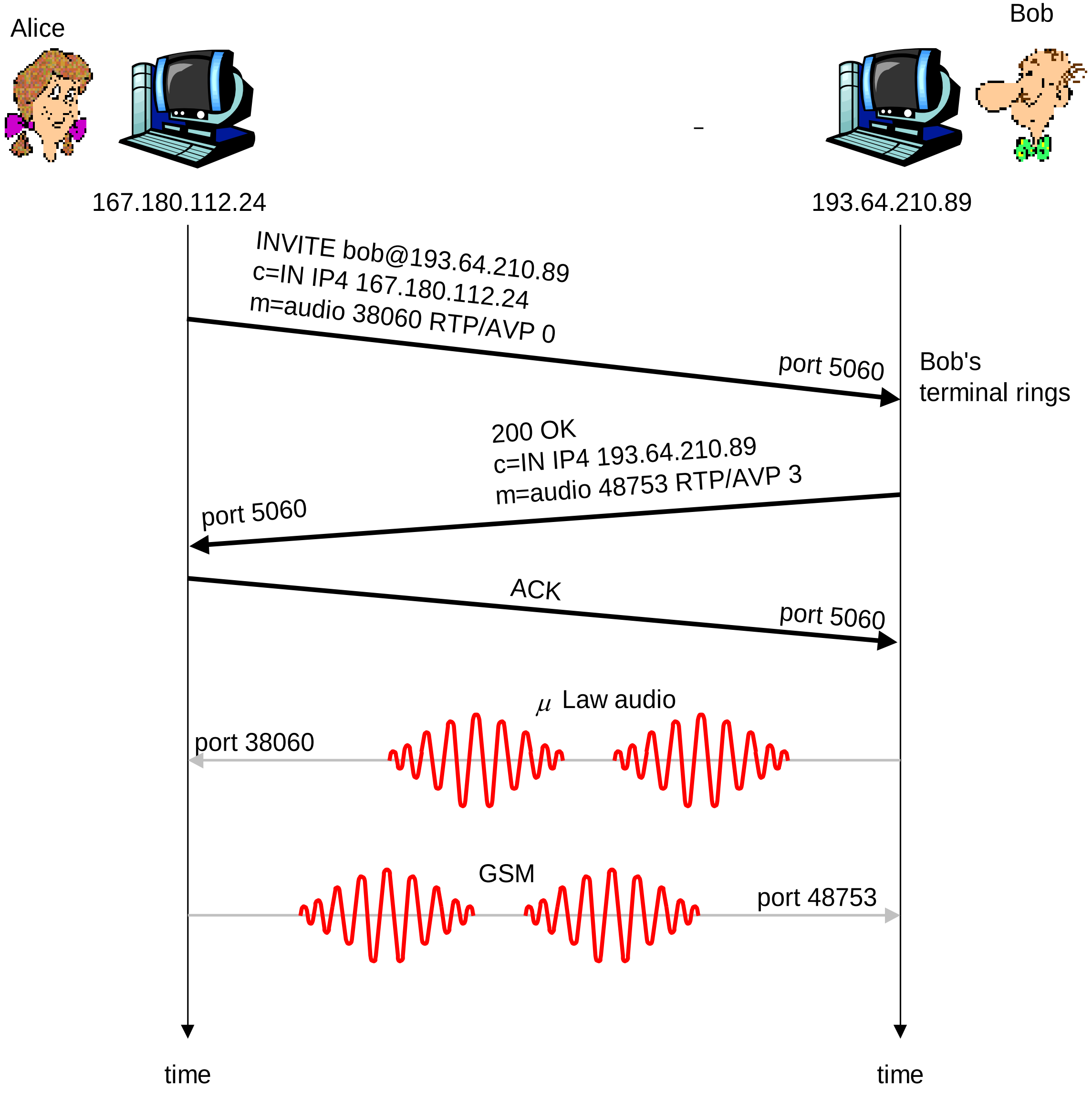
-
-

Example

SIP: Session Initiation Protocol [RFC 3261]

SIP long-term vision:

-
-
-



- 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.

Example of SIP message

- 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

Service provided by SIP servers:

SIP Registrar

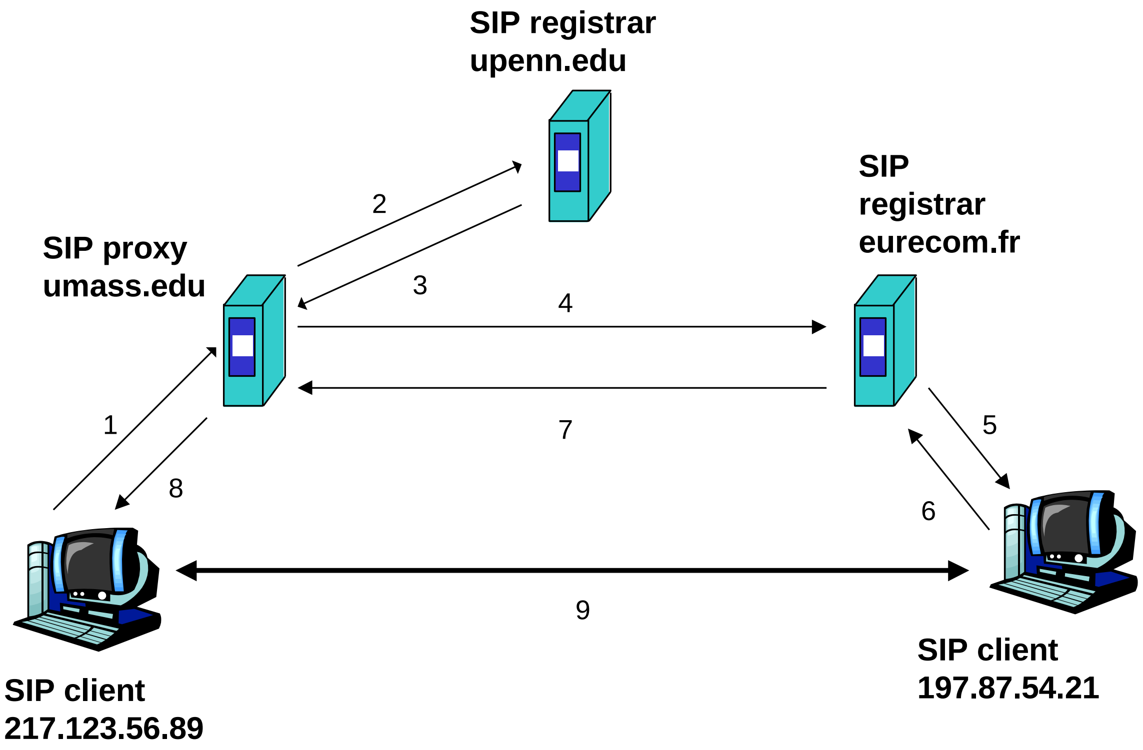
- when Bob starts SIP client, client sends SIP REGISTER message to Bob's registrar server (similar function needed by Instant Messaging)

Register Message:

SIP Proxy

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.

7.1

7.2

7.3

7.4

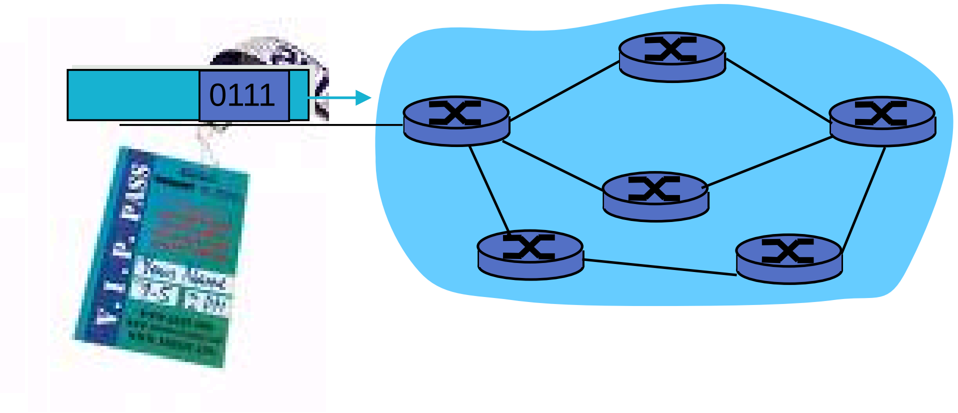
7.5 providing multiple classes of service

7.6

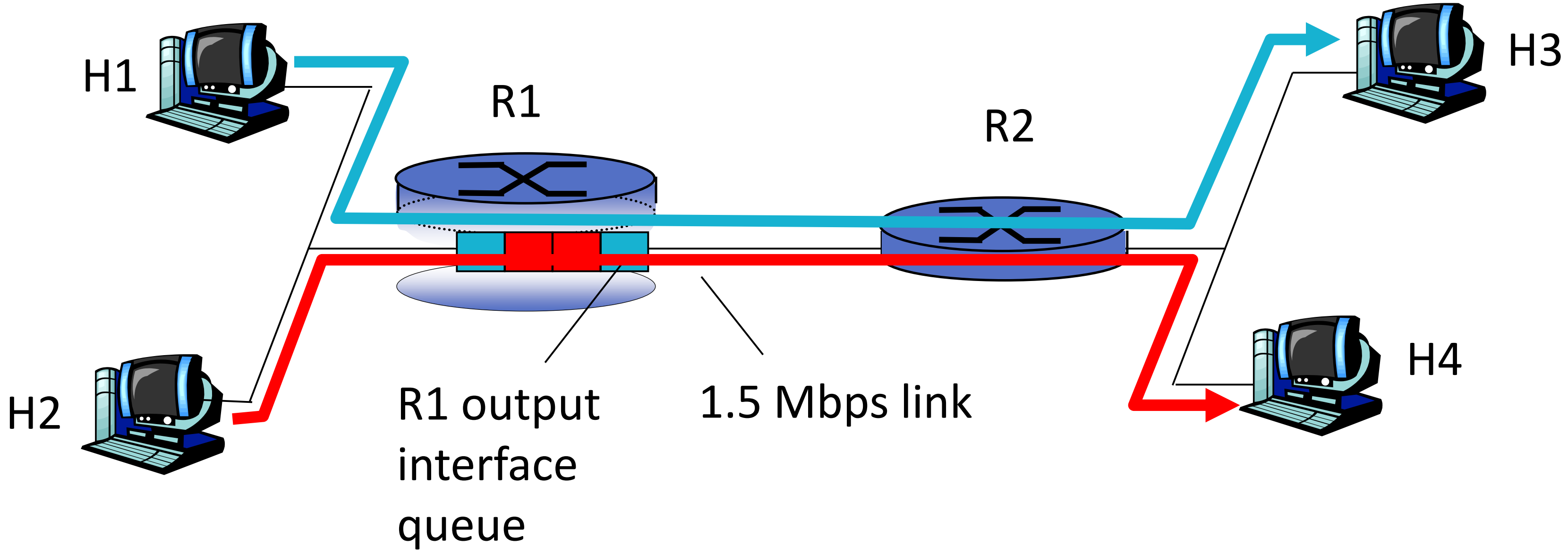
Providing Multiple Classes of Service

- thus far: making the best of best effort service
  - one-size fits all service model

- granularity: differential service among multiple classes, not among individual connections
- history: ToS bits

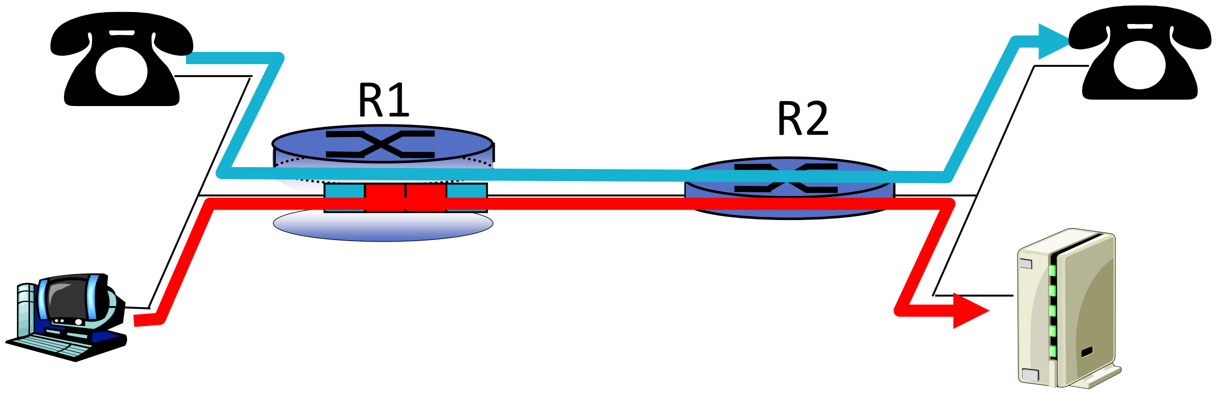


Multiple classes of service: scenario



Scenario 1: mixed FTP and audio

- Example: 1Mbps IP phone, FTP share 1.5 Mbp
  - bursts of FTP can congest router, cause audio l

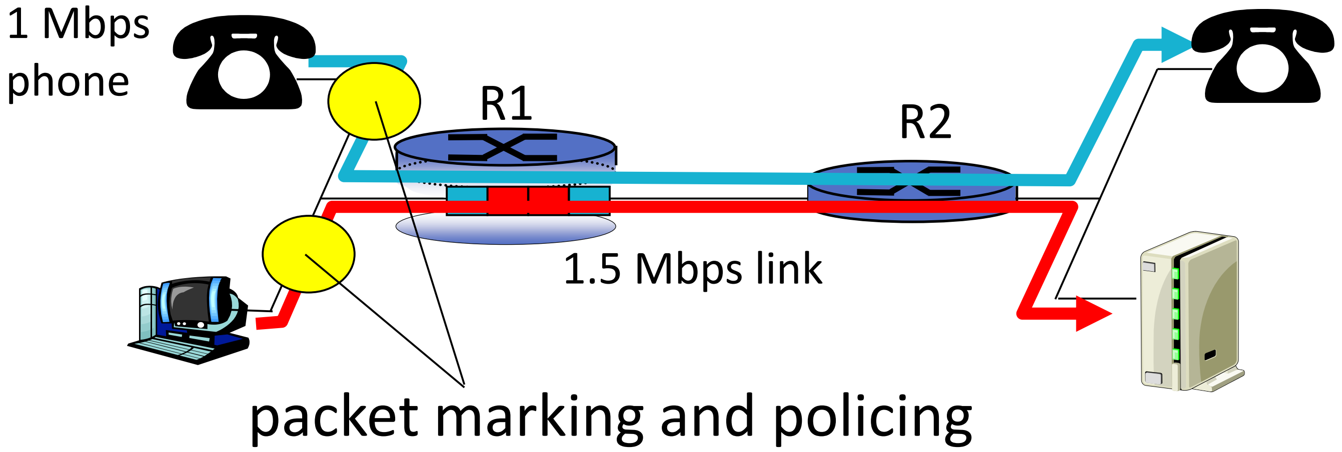


Principle 1

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 h
  - policing: force source adherence to bandwidt
-

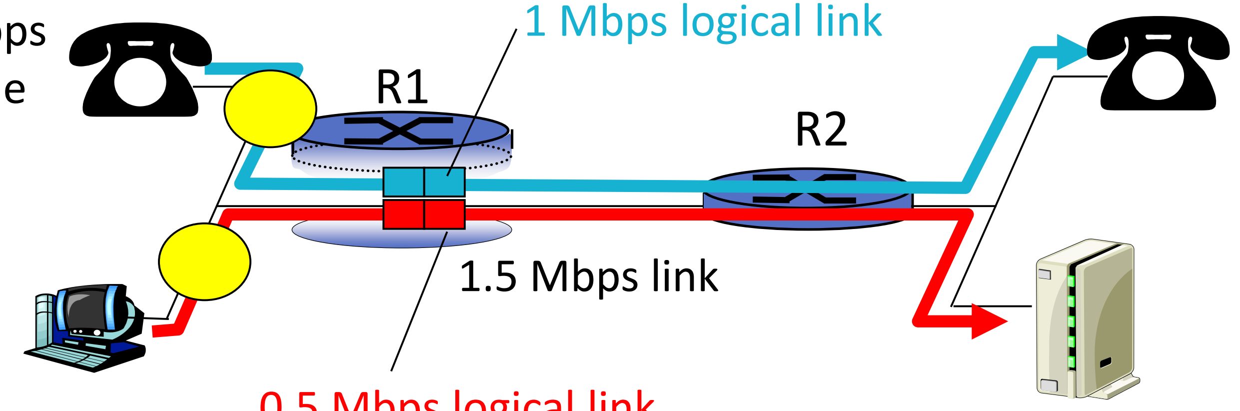


Principle 2

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

Principles for QOS Guarantees (more)

- Allocating *fixed* (non-sharable) bandwidth to f
bandwidth if flows doesn't use its allocation

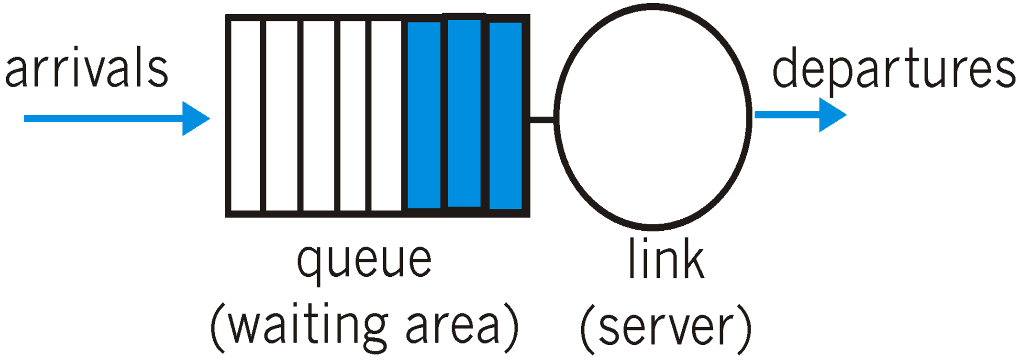


Principle 3

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

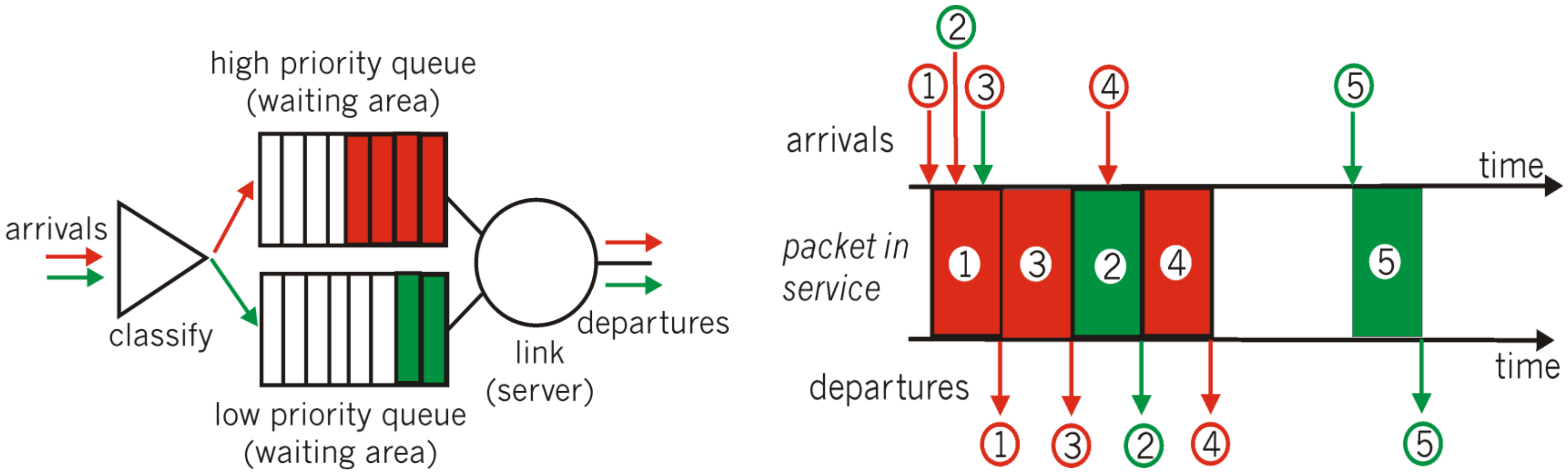
Scheduling And Policing Mechanisms

- scheduling: choose next packet to send on link
- FIFO (first in first out) scheduling: send in order of arri
    - discard policy:



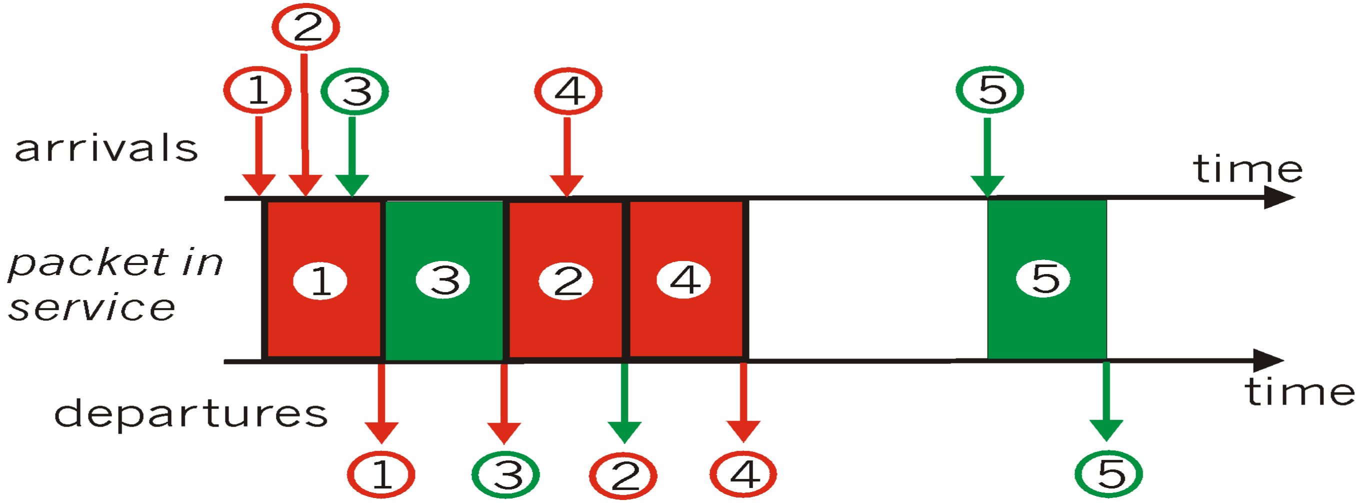
Scheduling Policies: more

Priority scheduling: transmit highest priority qu



Scheduling Policies: still more

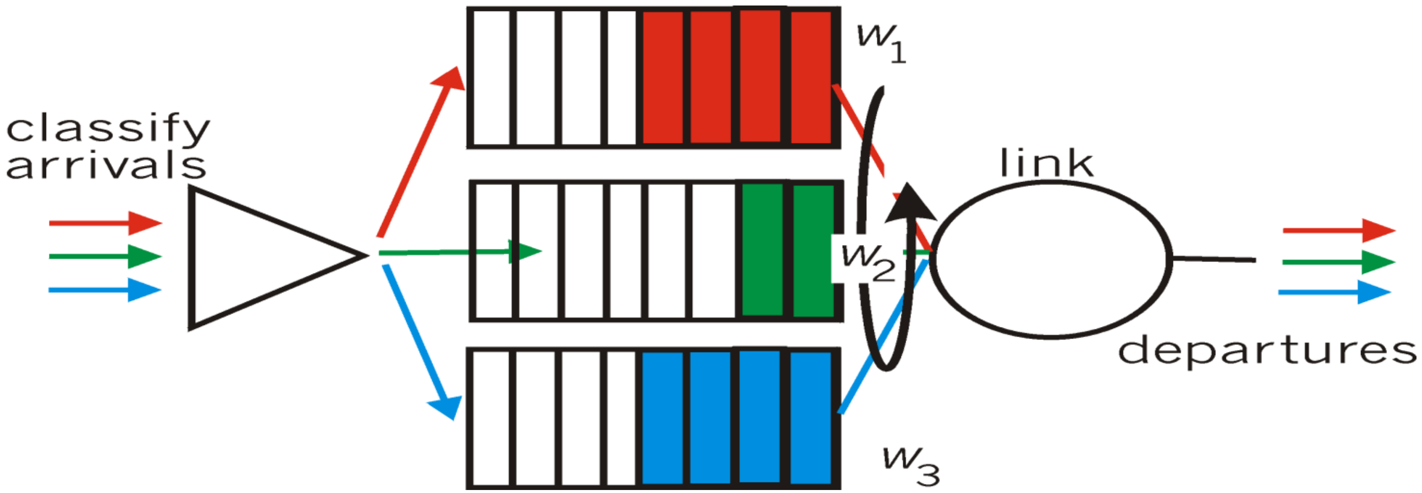
round robin scheduling:



Scheduling Policies: still more

Weighted Fair Queuing:

-
-
-



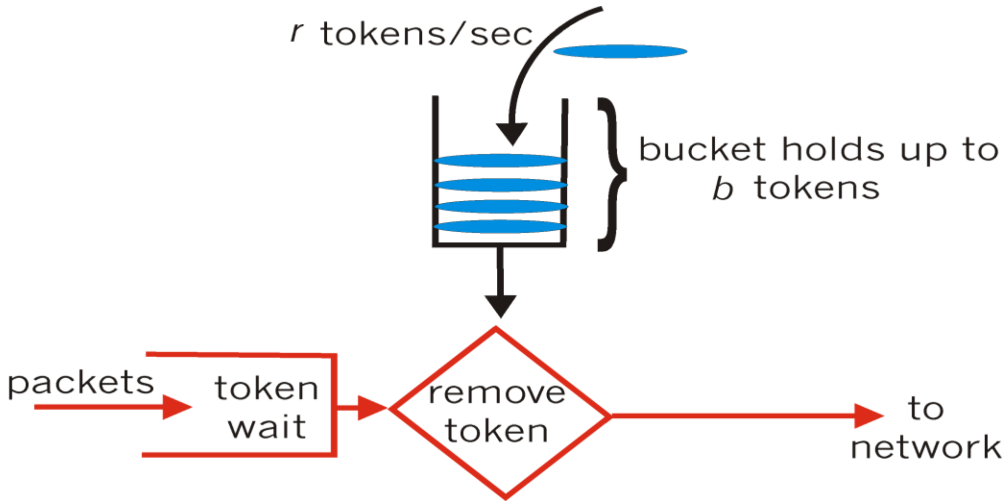
Policing Mechanisms

Goal:

- *(Long term) Average Rate:*
- *Peak Rate:*
- *(Max.) Burst Size:*

Policing Mechanisms

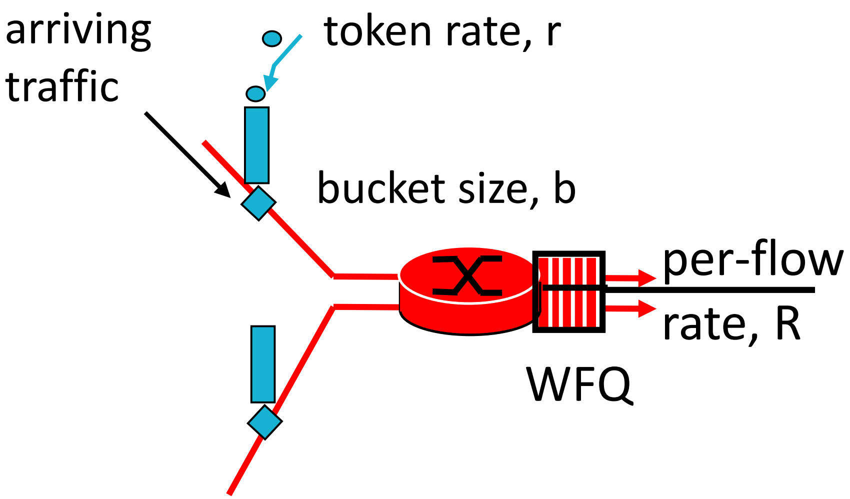
Token Bucket:



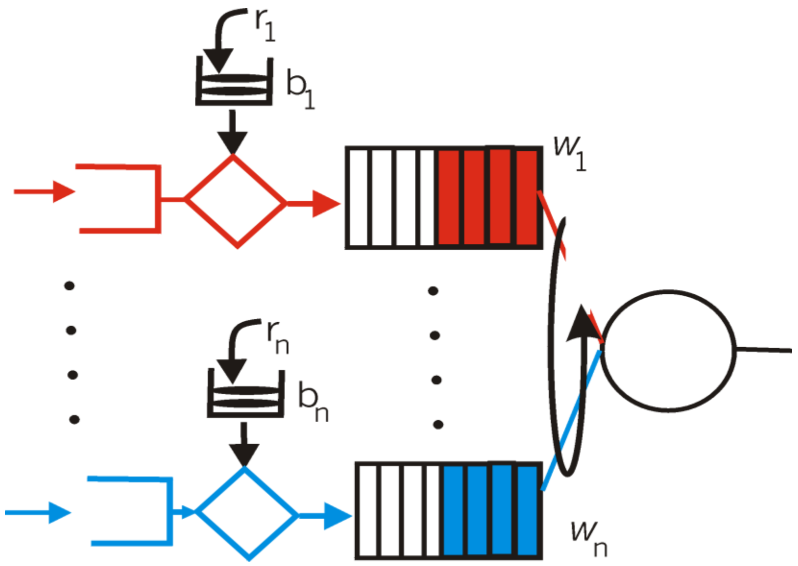
-
-
- *over interval of length t: number of packets admitted less than or equal to (r t + b).*

Policing Mechanisms (more)

- *QoS guarantee*



$D_{max} = b/R$



IETF Differentiated Services

- want "qualitative" service classes
- *scalability:*
-

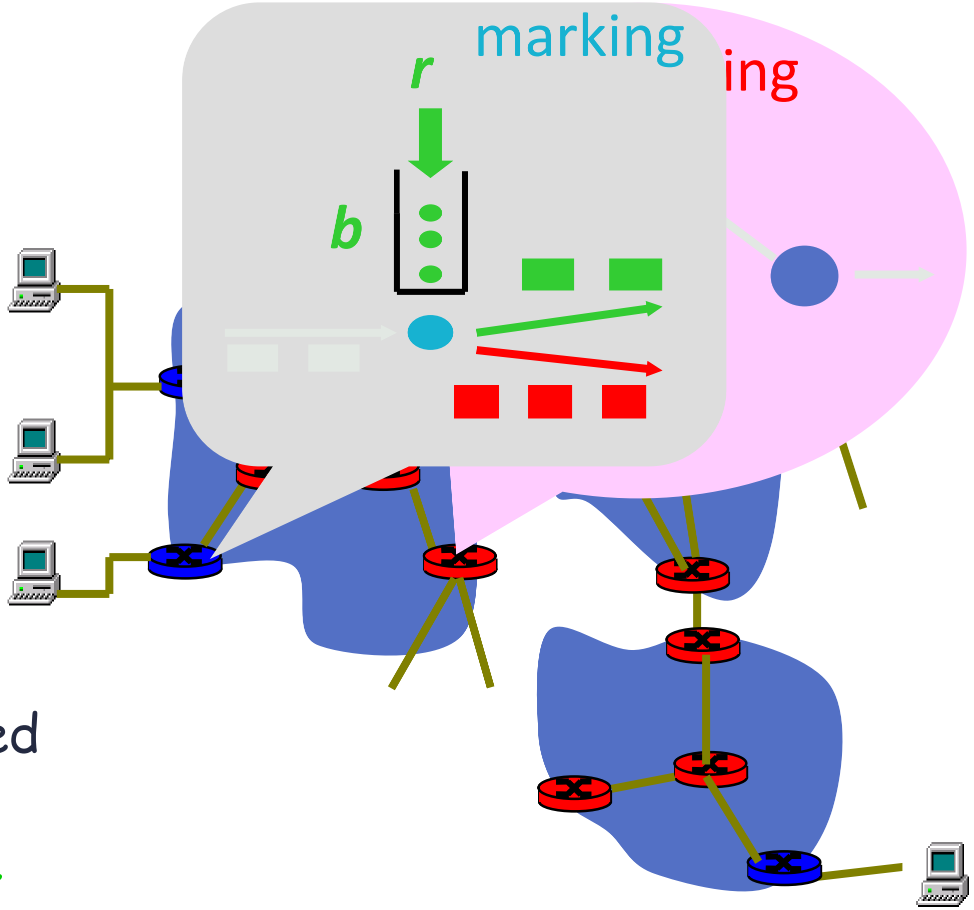
Diffserv Architecture

Edge router:

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

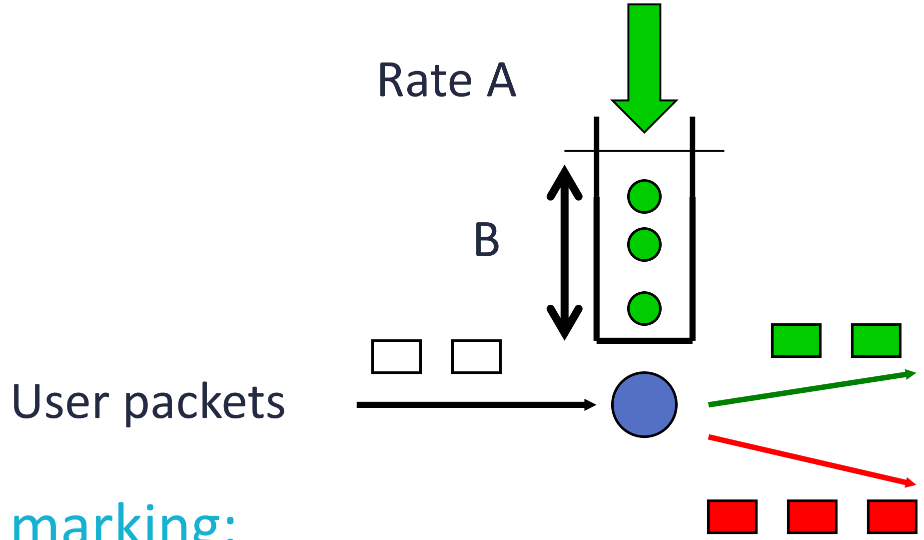
Core router:

- 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

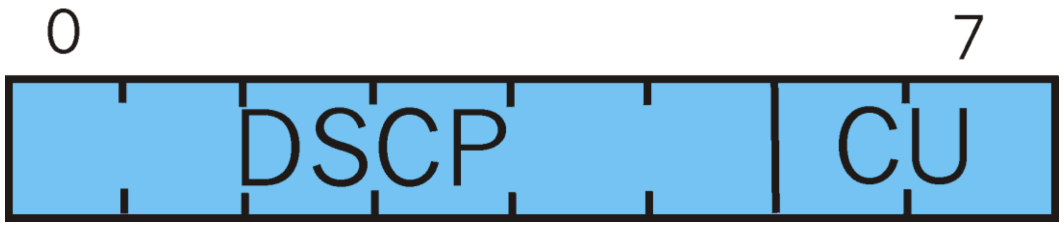


Possible usage of marking:

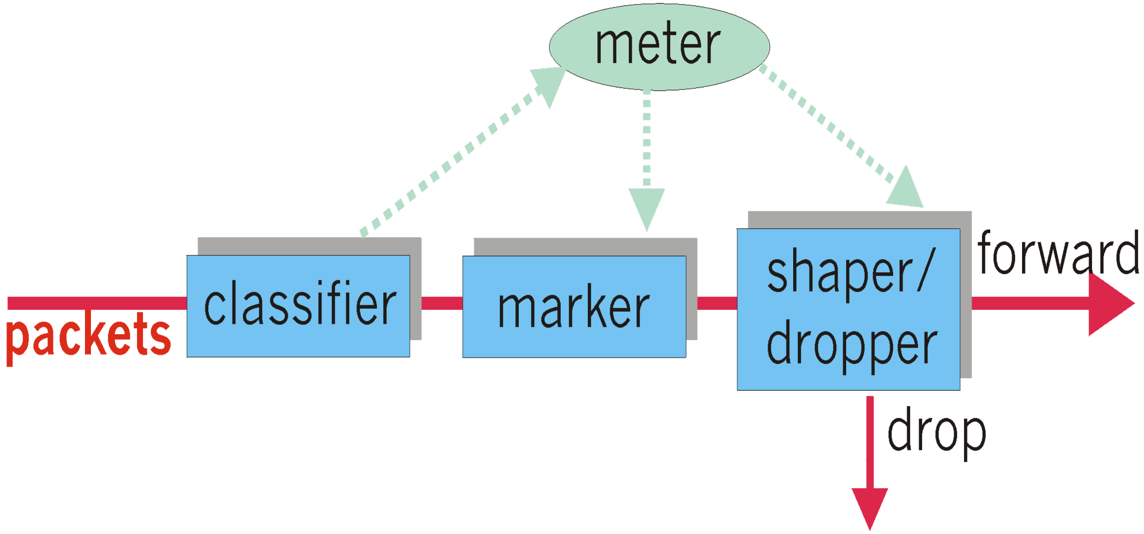
- class-based marking: packets of different classes marked differently
- intra-class marking: conforming portion of flow marked differently than non-conforming one

Classification and Conditioning

- Packet is marked in the Type of Service (TOS) i
in IPv6



Classification and Conditioning



Forwarding (PHB)

Forwarding (PHB)

- Expedited Forwarding:
- Assured Forwarding:

7.1

7.2

7.3

7.4

7.5

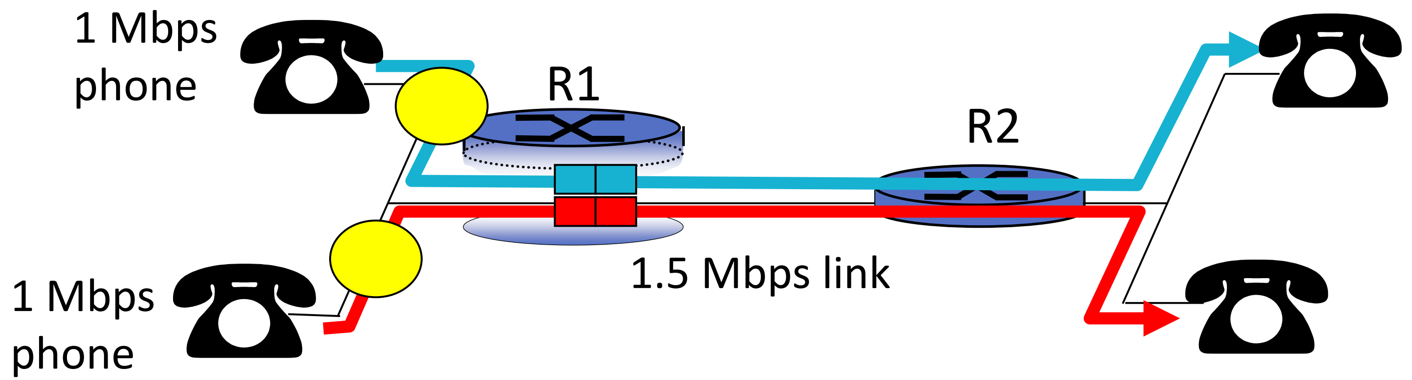
7.6 providing QoS guarantees

Multimedi
a
Networkin

- 7.9 RSVP

Principles for QOS Guarantees (more)

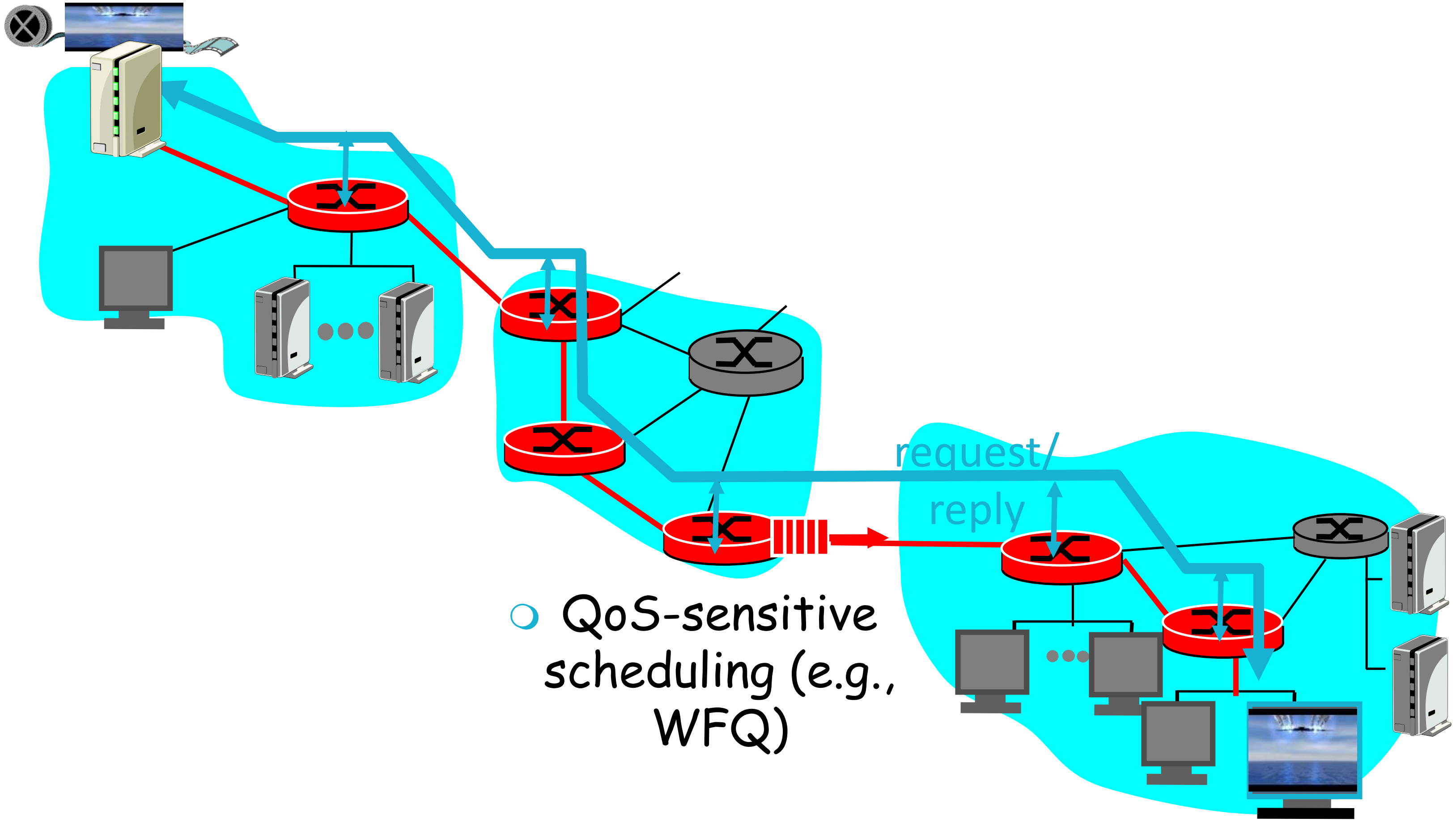
- *Basic fact of life:* can not support traffic demar



Principle 4

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

- Resource reservation



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

IETF Integrated Services

-
-
-

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

Call Admission

-
  - R-spec:
-
  - T-spec:
-
  - RSVP

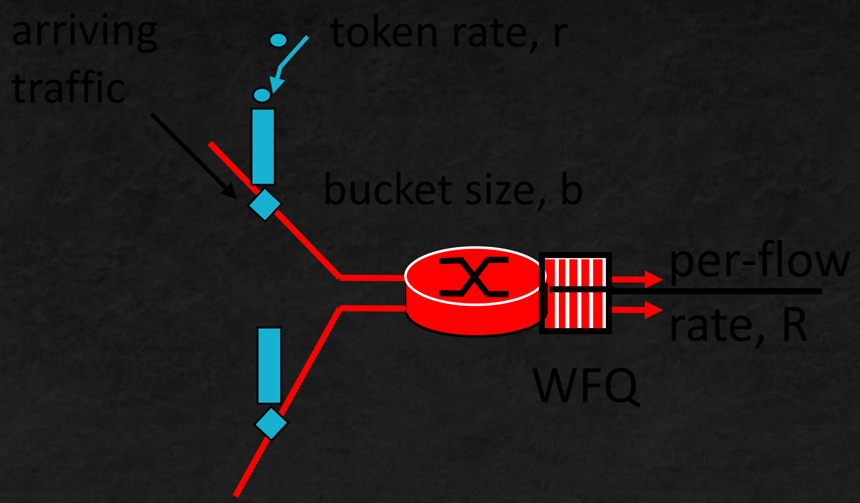
Intserv QoS: Service models [rfc2211, rfc 2212]

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."



$$D_{max} = b/R$$