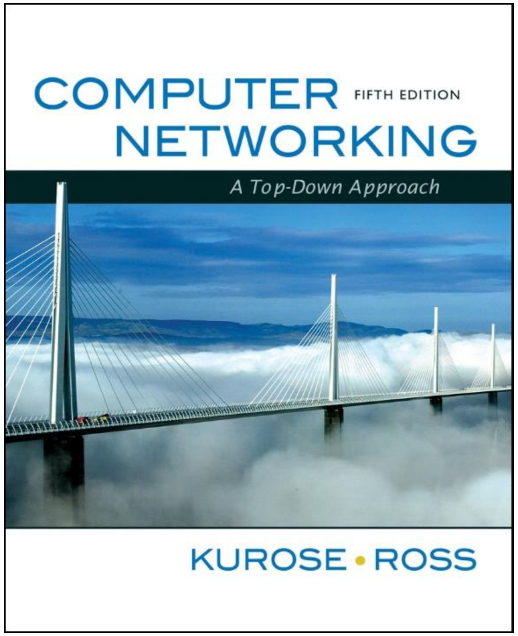
# Chapter 7
# Multimedia Networking




## A note on the use of these ppt slides:

We're making these slides freely available to all (faculty, students, readers). They're in PowerPoint form so you can add, modify, and delete slides (including this one) and slide content to suit your needs. They obviously represent a *lot* of work on our part. In return for use, we only ask the following:

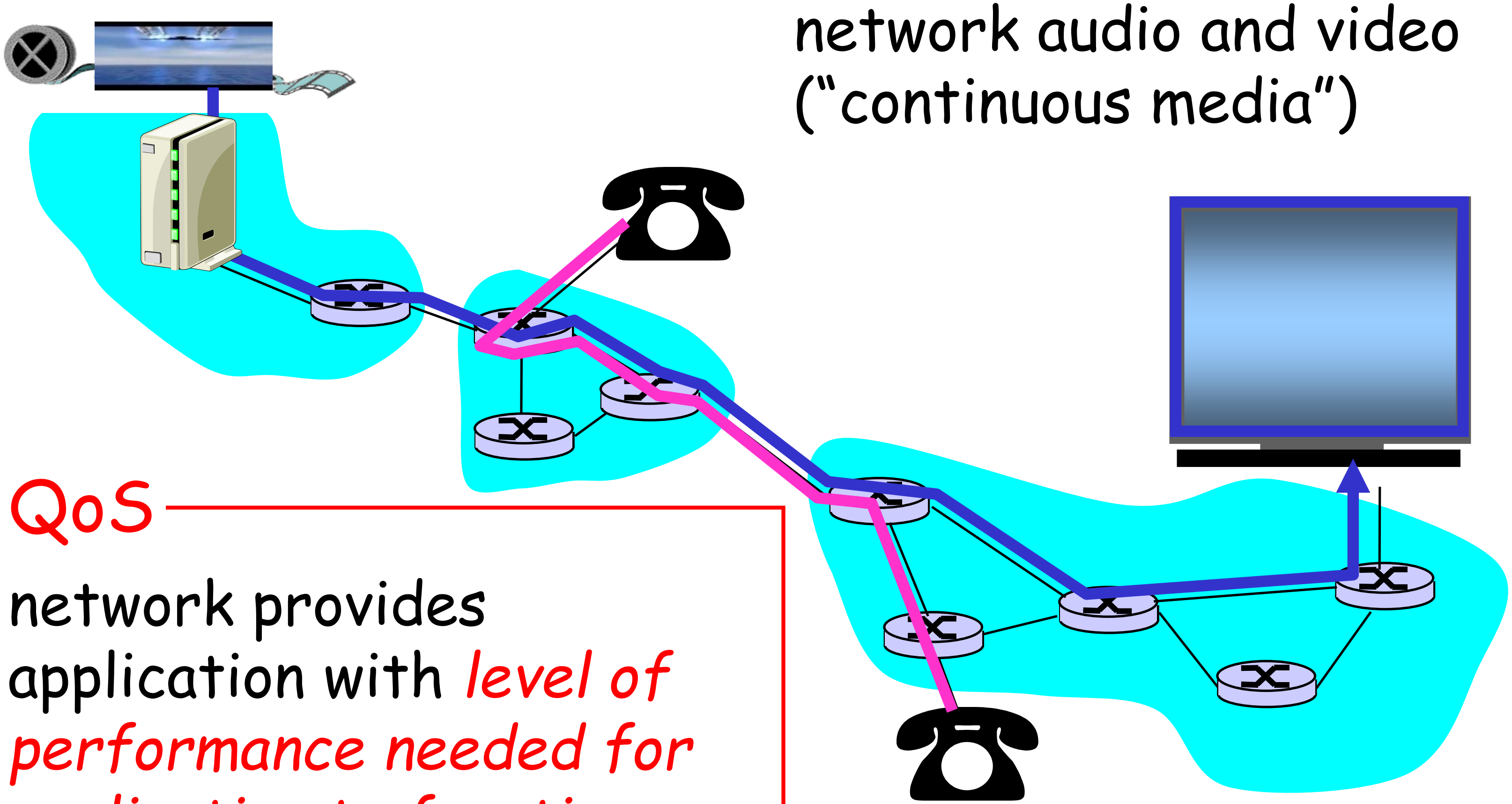
- If you use these slides (e.g., in a class) in substantially unaltered form, that you mention their source (after all, we'd like people to use our book!)
- If you post any slides in substantially unaltered form on a www site, that you note that they are adapted from (or perhaps identical to) our slides, and note our copyright of this material.

Thanks and enjoy!  JFK / KWR



*Computer Networking: A Top Down Approach*
5th edition.
Jim Kurose, Keith Ross
Addison-Wesley, April 2009.

# 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
- 尽全力而为

## Protocols and Architectures

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

# 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

# MM Networking Applications

**Classes of MM applications:**

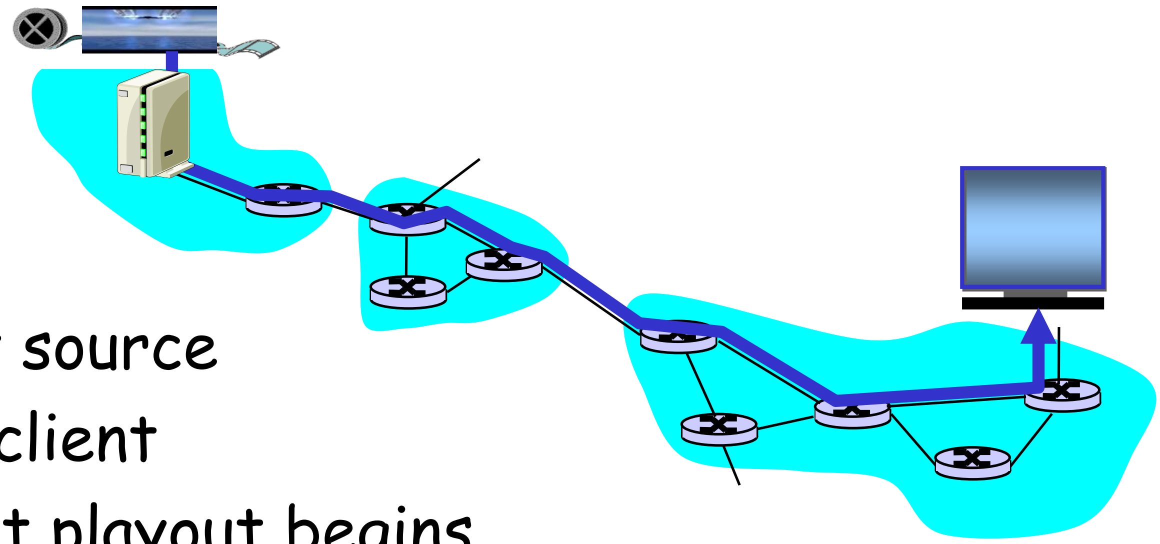
1) stored streaming
2) live streaming
3) interactive, real-time

**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

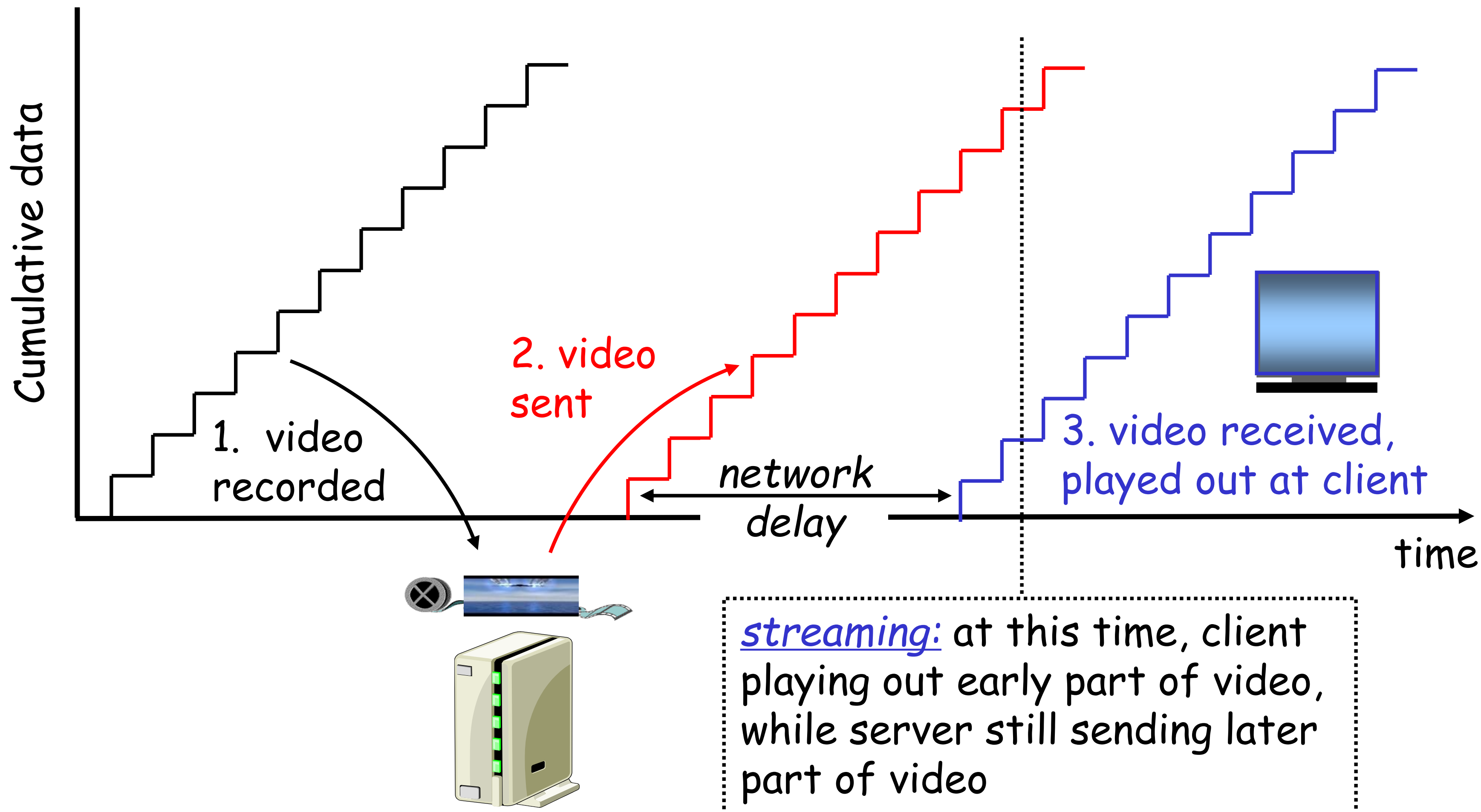
# Streaming Stored Multimedia



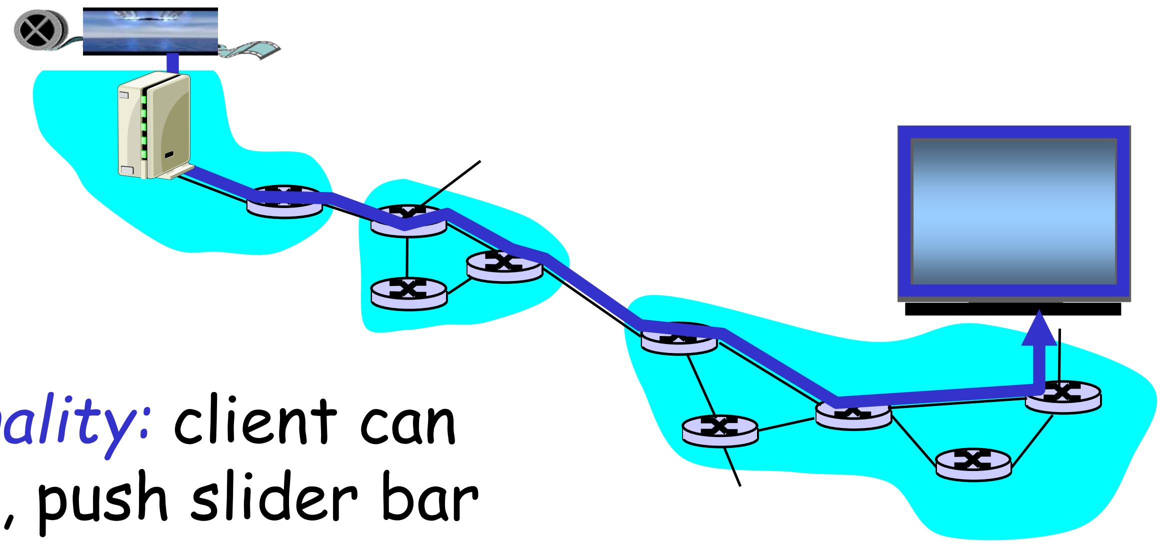
Stored streaming:

- 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 *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

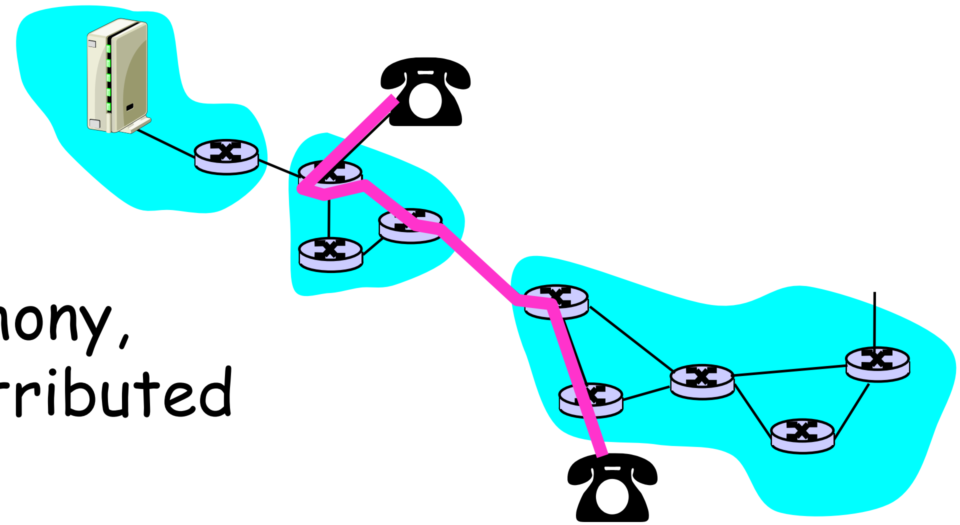
Streaming (as with streaming *stored* multimedia)

- 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

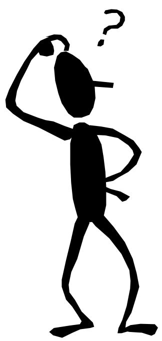


- applications: IP telephony, video conference, distributed interactive worlds
- end-end delay requirements:
  - audio: < 150 msec good, < 400 msec OK
    - includes application-level (packetization) and network delays
    - higher delays noticeable, impair interactivity
- session initialization
  - 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


Loading...
00:00 / 04:51


广告剩余 13 秒
静音
跳过广告
香辣鸡翅
详细了解


优酷
加载中...
02:40 / 03:31
标清

# Streaming Stored Multimedia

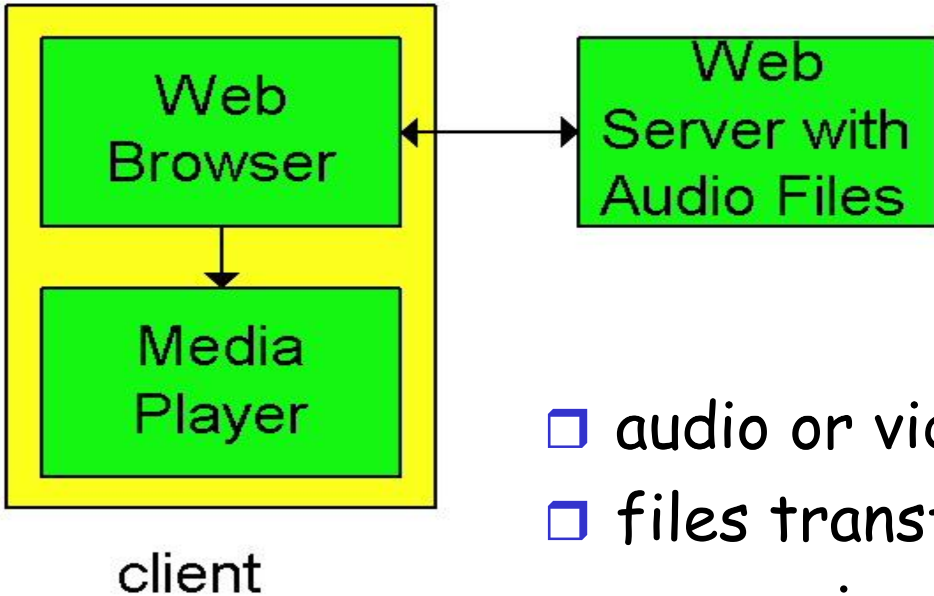
application-level streaming techniques for making the best out of best effort service:

- 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

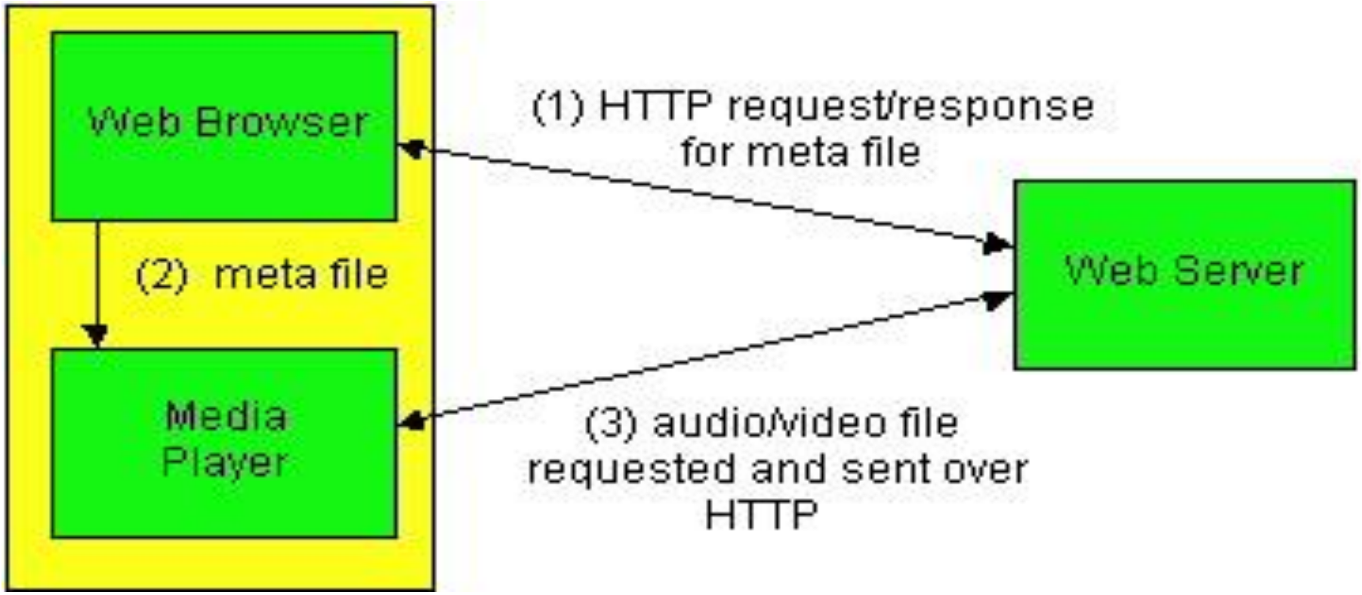



- audio or video stored in file
- files transferred as HTTP object
  - received in entirety at client
  - then passed to player

audio, video not streamed:

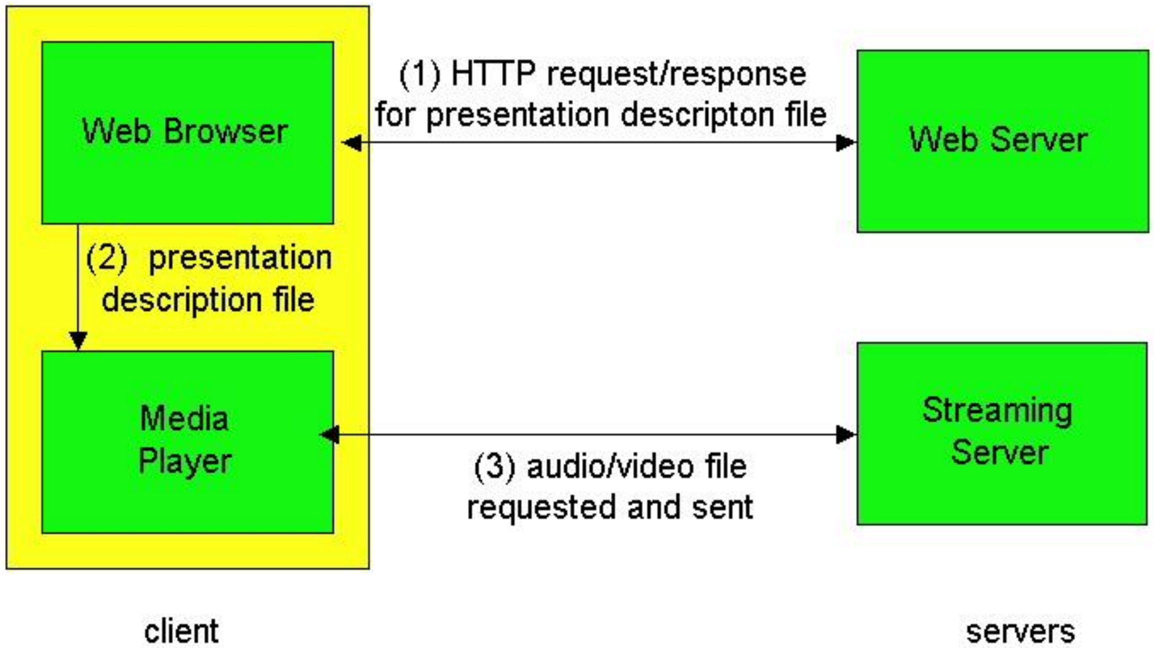
- no, "pipelining," long delays until playout!

# Internet multimedia: streaming approach



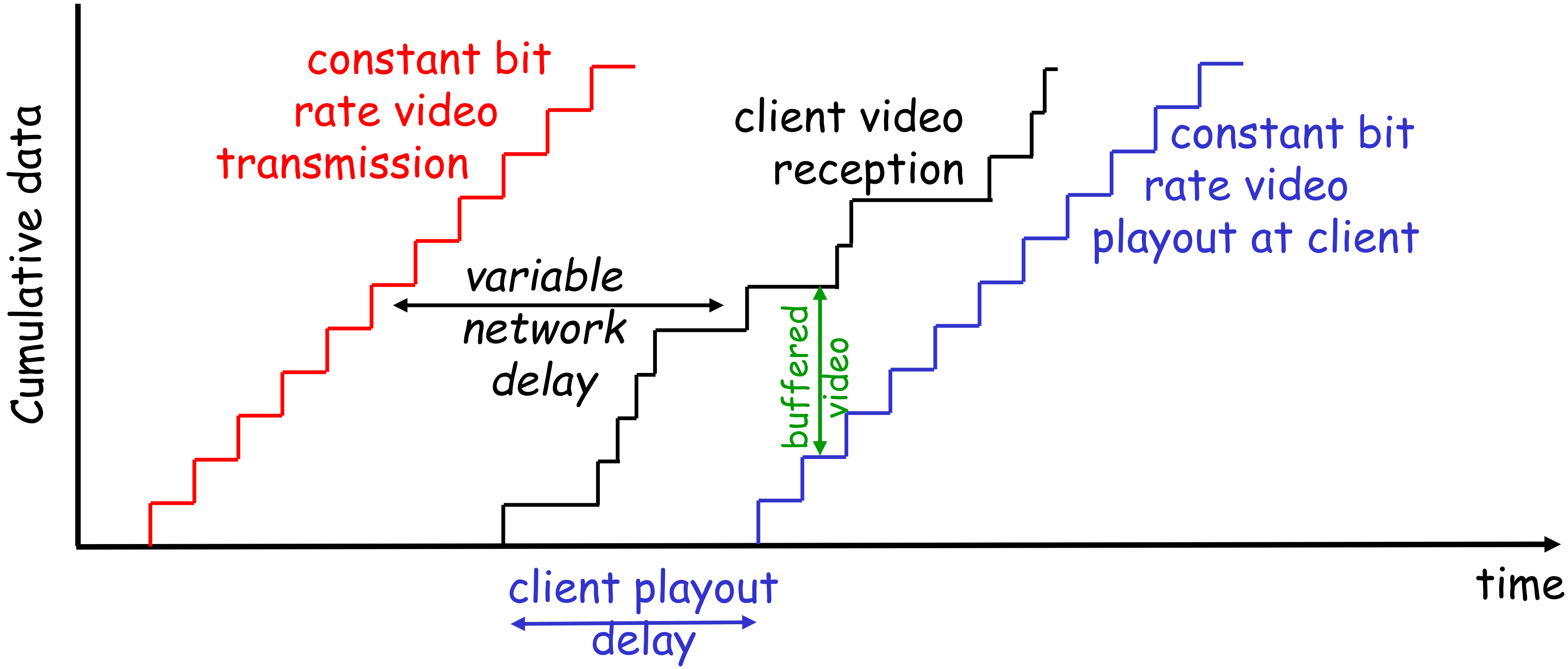

- browser GETs **metafile**
- browser launches player, passing metafile
- player contacts server
- server streams audio/video to player

# Streaming from a streaming server



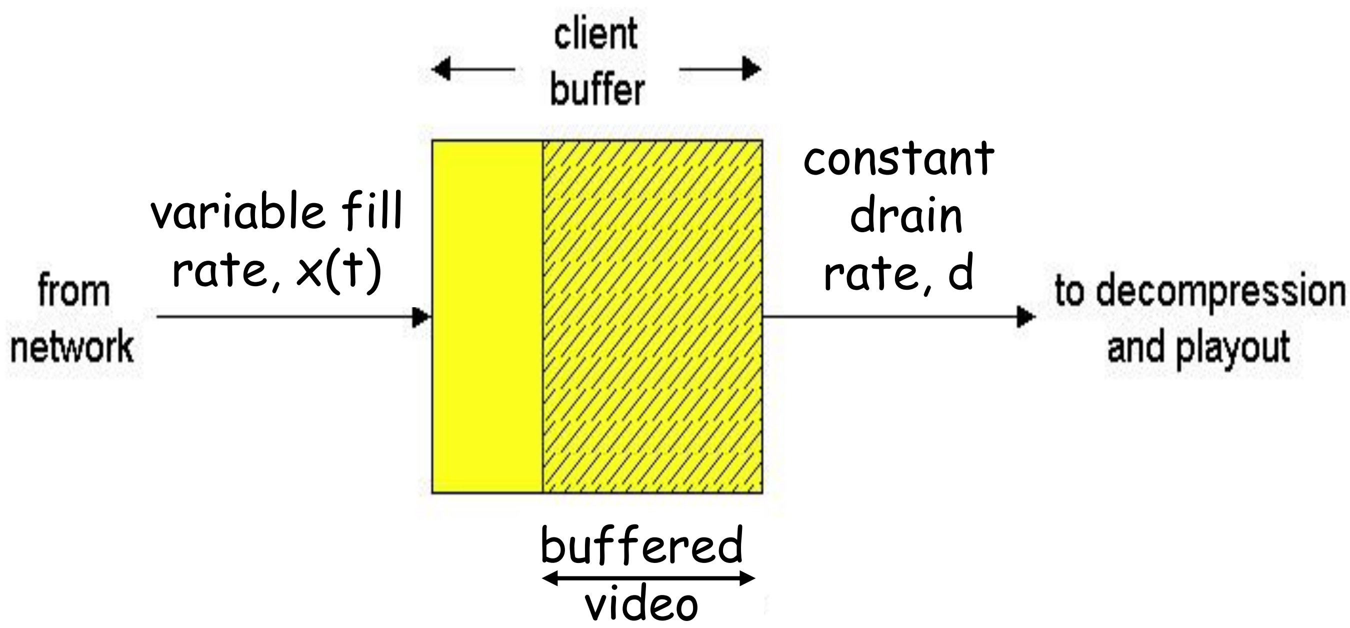

- allows for non-HTTP protocol between server, media player
- UDP or TCP for step (3), more shortly

# 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

# Streaming Multimedia: UDP or TCP?

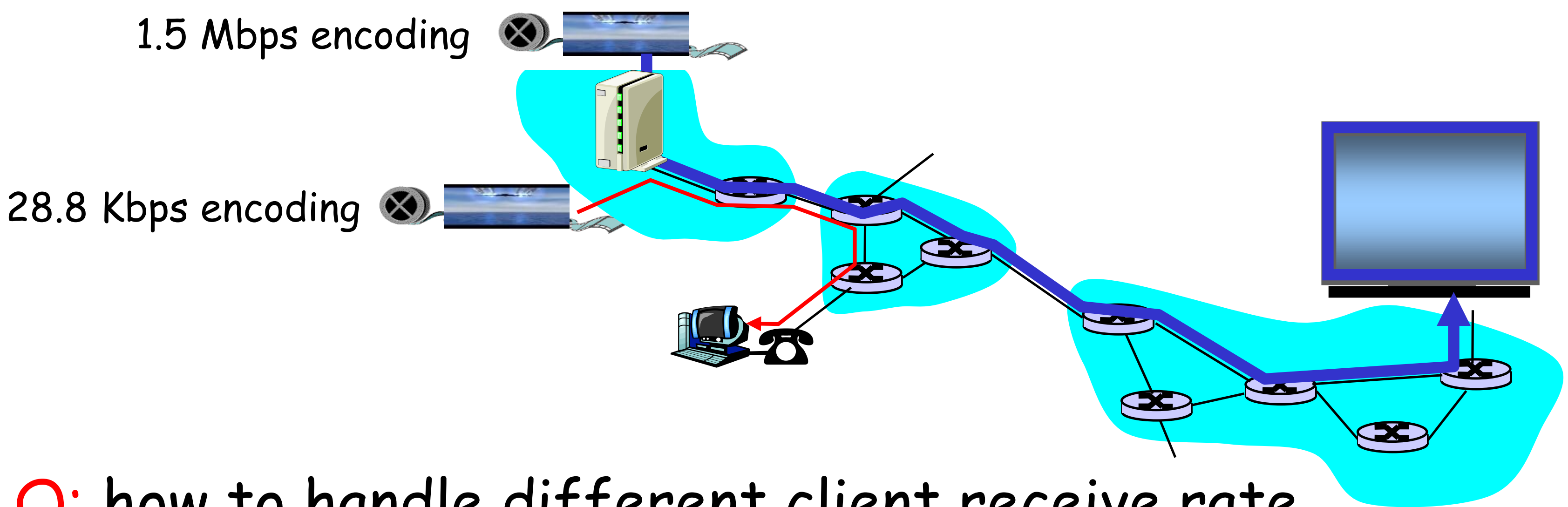
## UDP

- 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

# 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

# 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

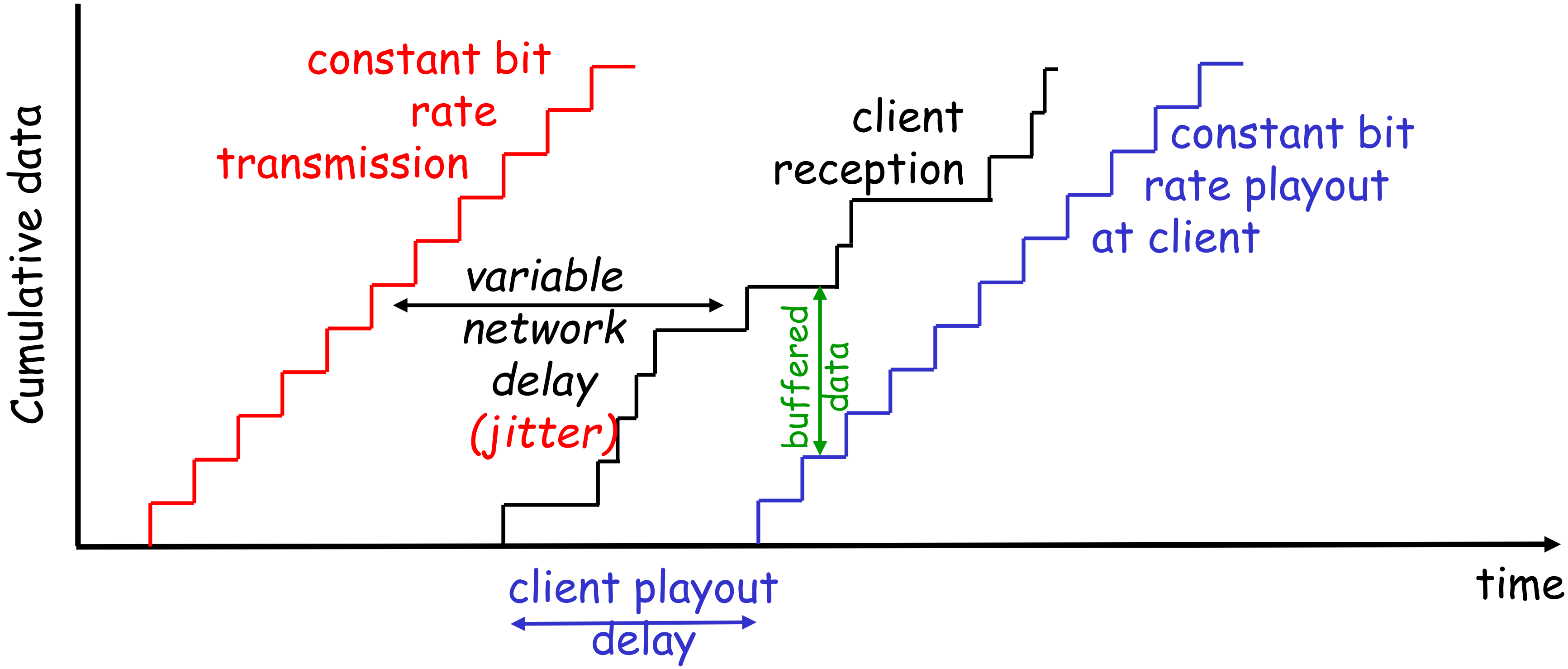
Introduce Internet Phone by way of an example

- 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



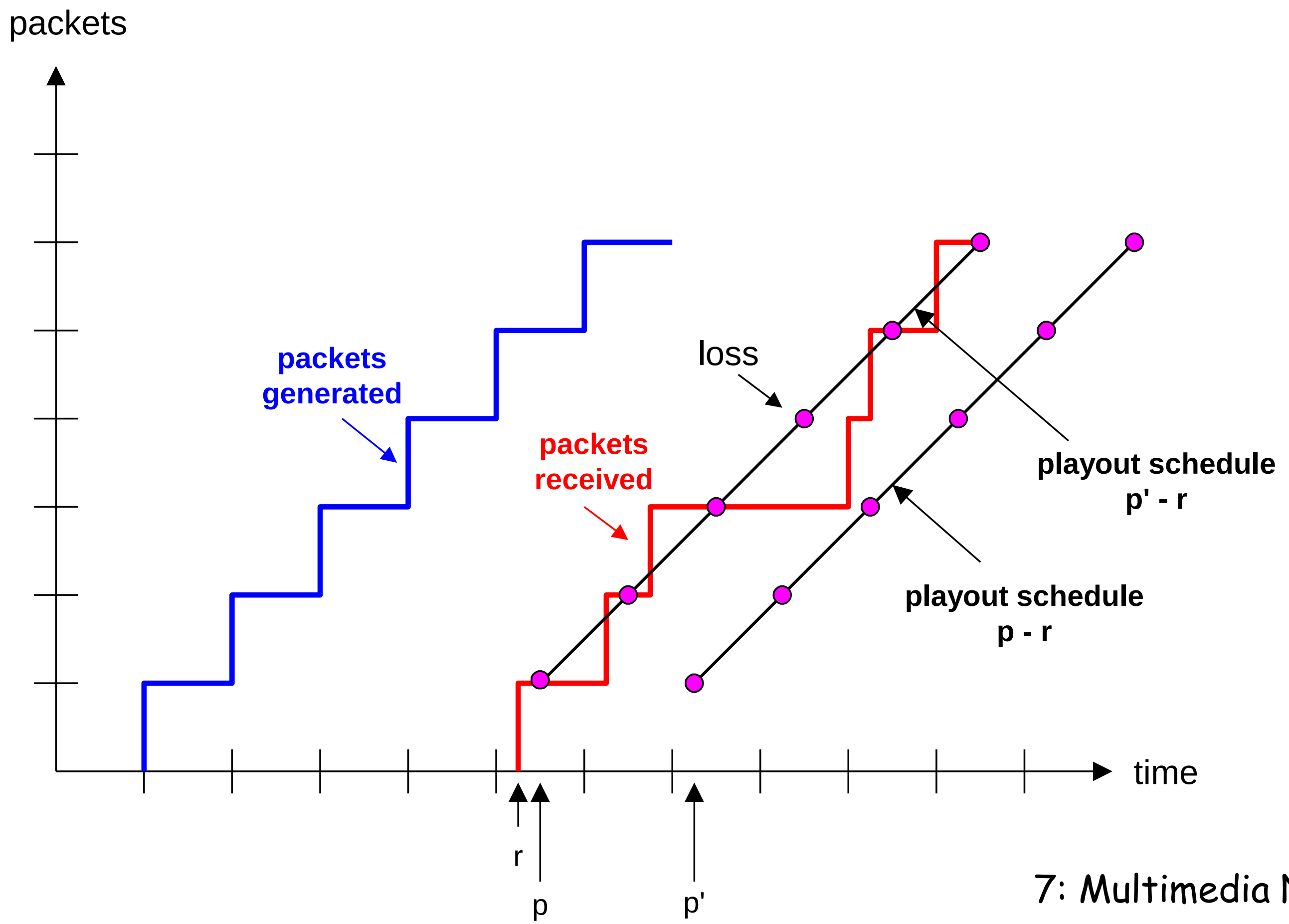

- consider end-to-end delays of two consecutive packets: difference can be more or less than 20 msec (transmission time difference)

# 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.

$t_i$ = timestamp of the ith 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 ith packet

$d_i$ = 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: 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.

# Recovery from packet loss (1)

**Forward Error Correction (FEC): simple scheme**

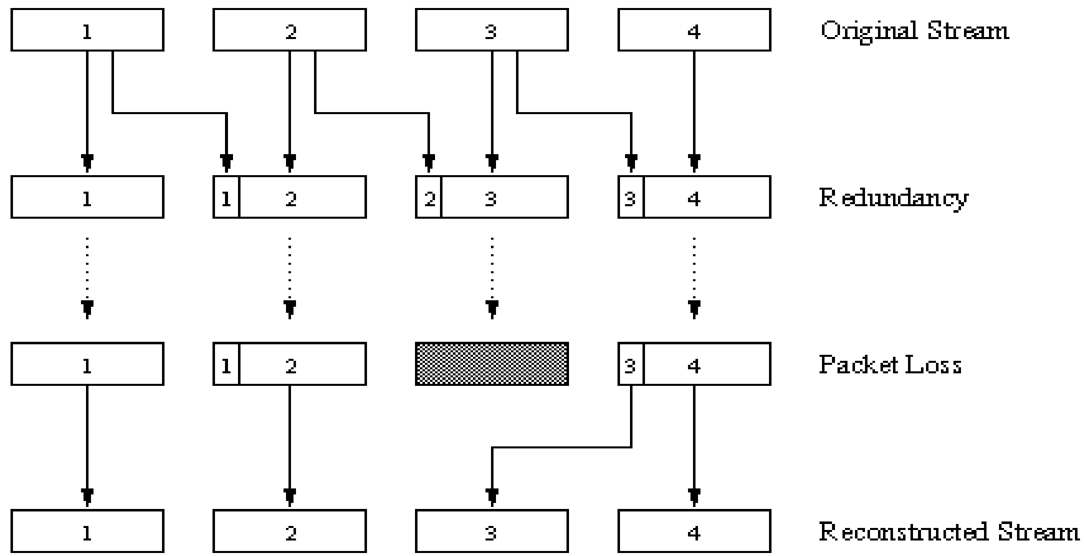
- for every group of *n* chunks create redundant chunk by exclusive OR-ing *n* original chunks
- send out *n+1* chunks, increasing bandwidth by factor *1/n.*
- can reconstruct original *n* chunks if at most one lost chunk from *n+1* chunks

- 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)

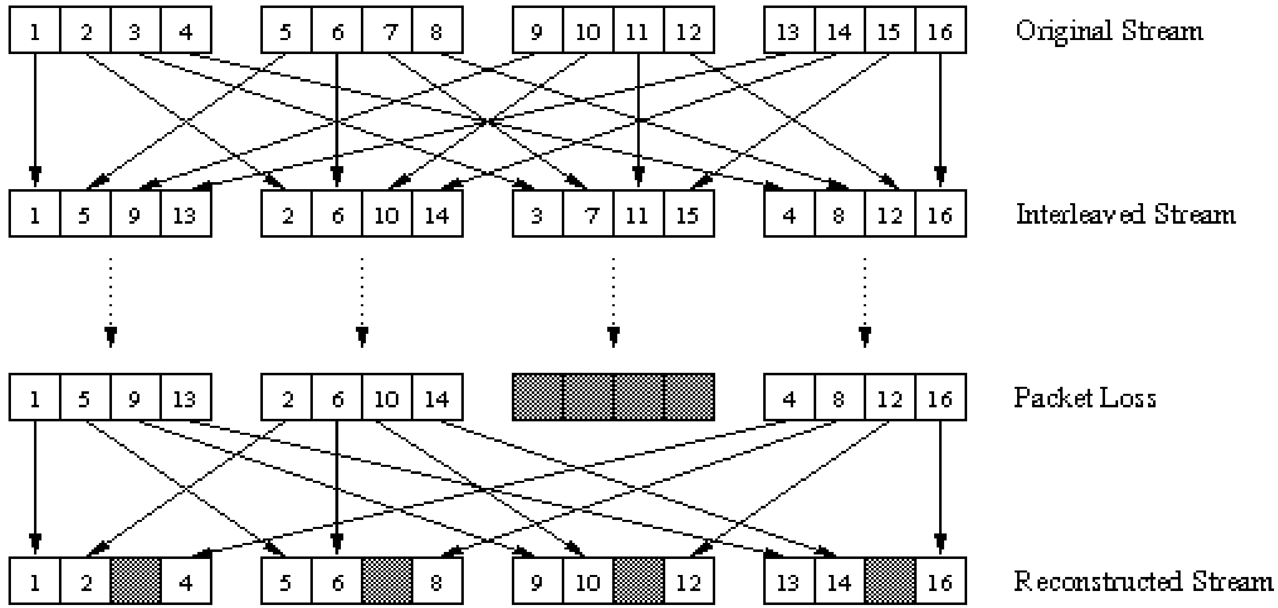
**2nd FEC scheme**

- "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)



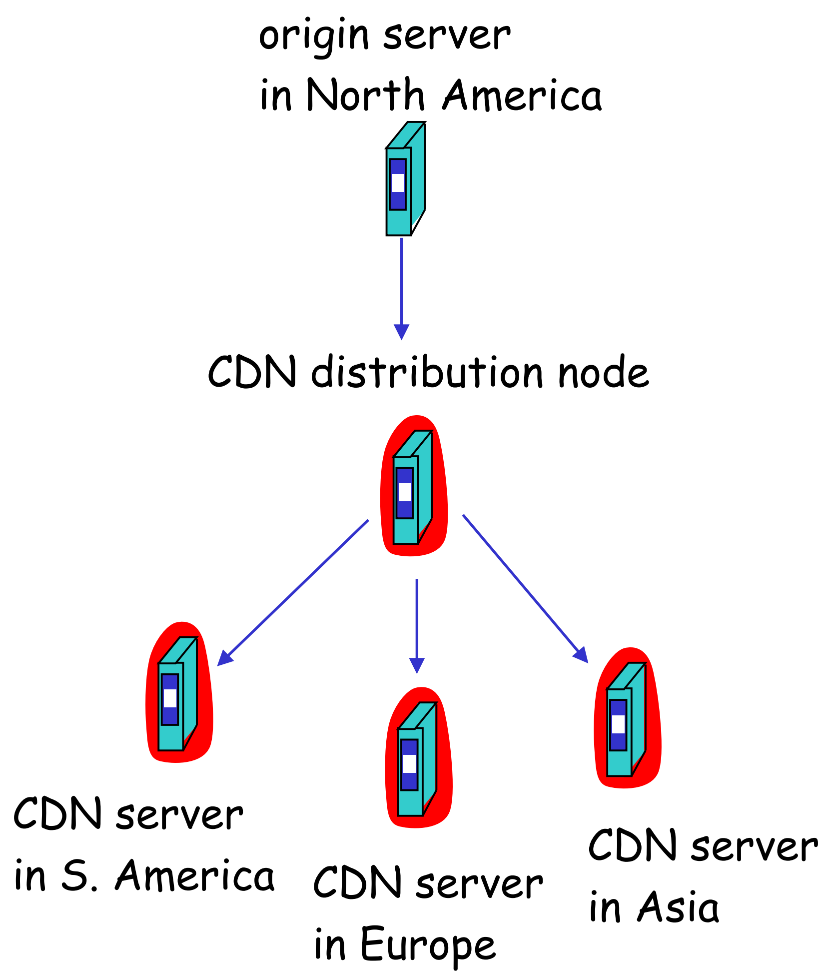

## Interleaving

- 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

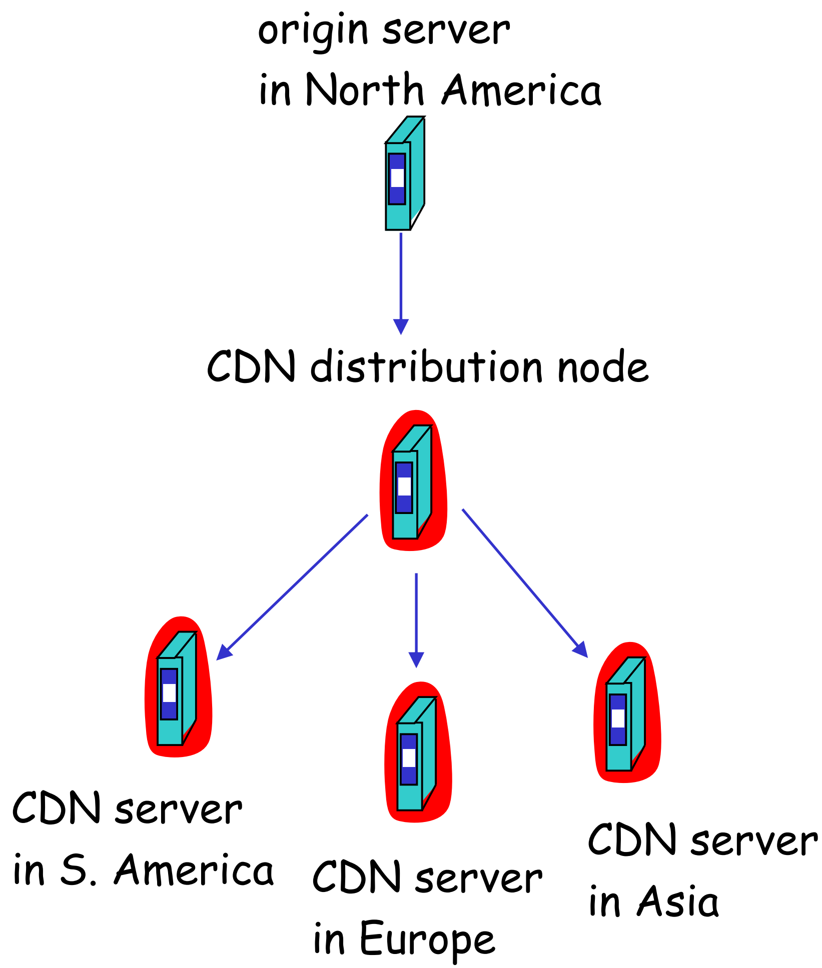
- challenging to stream large files (e.g., video) from single origin server in real time
- *solution:* replicate content at hundreds of servers throughout Internet
  - content downloaded to CDN servers ahead of time
  - *placing content "close" to user avoids impairments (loss, delay) of sending content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



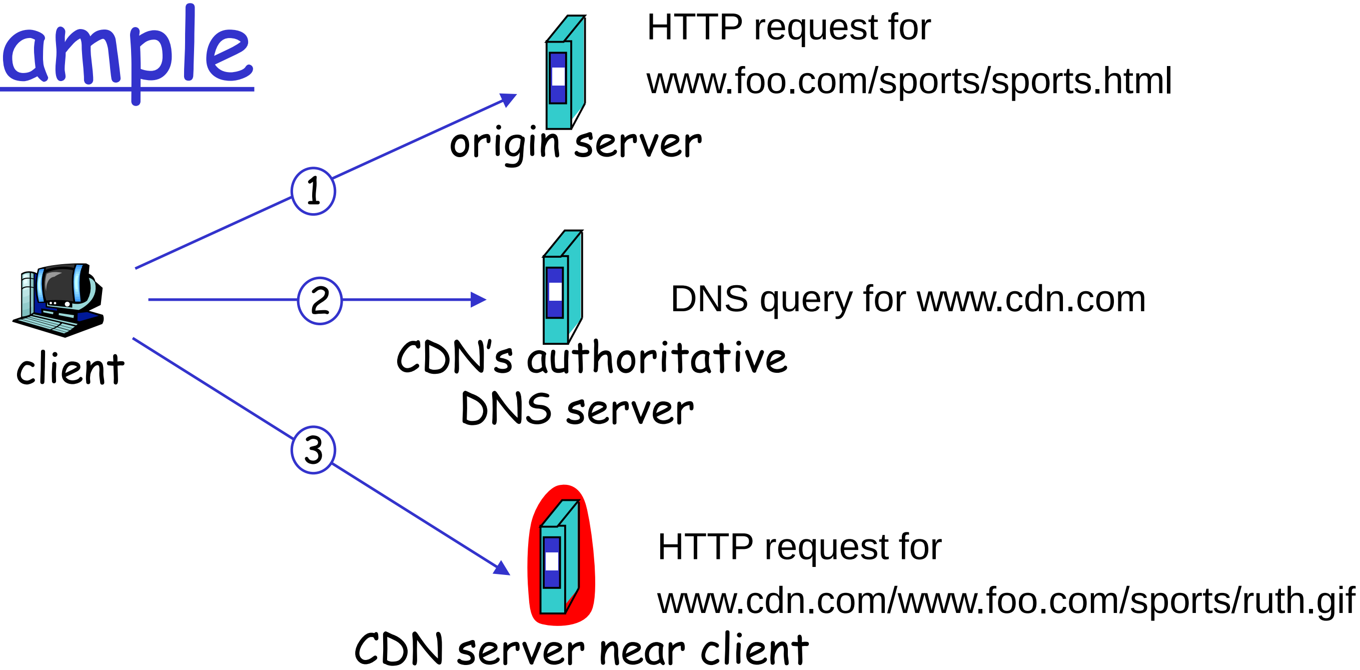
# Content distribution networks (CDNs)

## 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


东北电信客户
东北网通客户
东北电信节点
华北网通客户
华北电信客户
华北网通节点
东北网通节点
华北电信节点
西北客户
西北节点
西南客户
西南节点
华东节点
update
数据中心
备份数据中心
华东客户
华中节点
华南节点
另有:
美国节点
香港节点
铁通节点
教育网节点
华中客户
华南客户


Le:TV.com
乐视网
PPTV
PPS
www.pps.tv
网络电视
腾讯视频
V.QQ.COM
YOUKU
优酷
土豆网
每个人都是生活的导演
56.com
我乐
搜狐视频
tv.sohu.com
iQIYI
爱奇艺

# More about CDNs

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

# 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

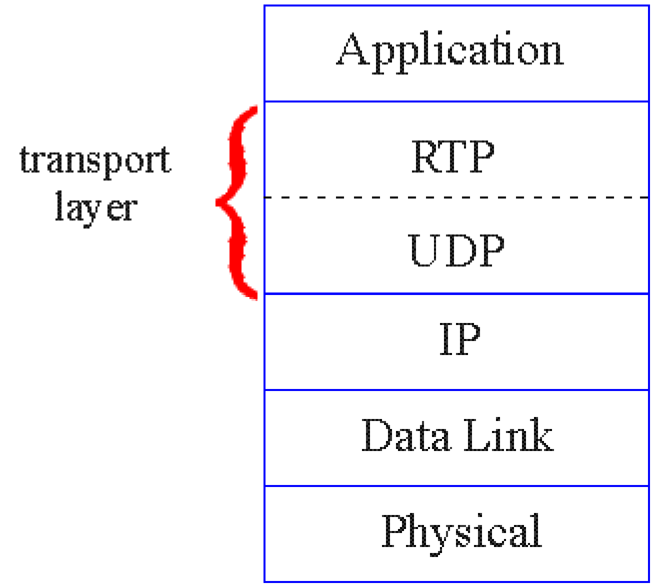
# Real-Time Protocol (RTP)

- 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 runs on top of UDP

RTP libraries provide transport-layer interface that extends 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 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 and QoS

- 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.

# 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

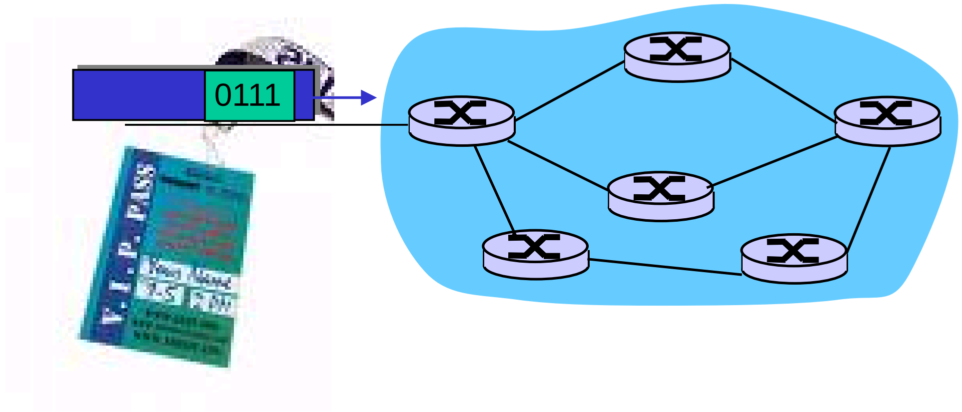
RTP, RTCP, SIP

7.5 providing multiple classes of service

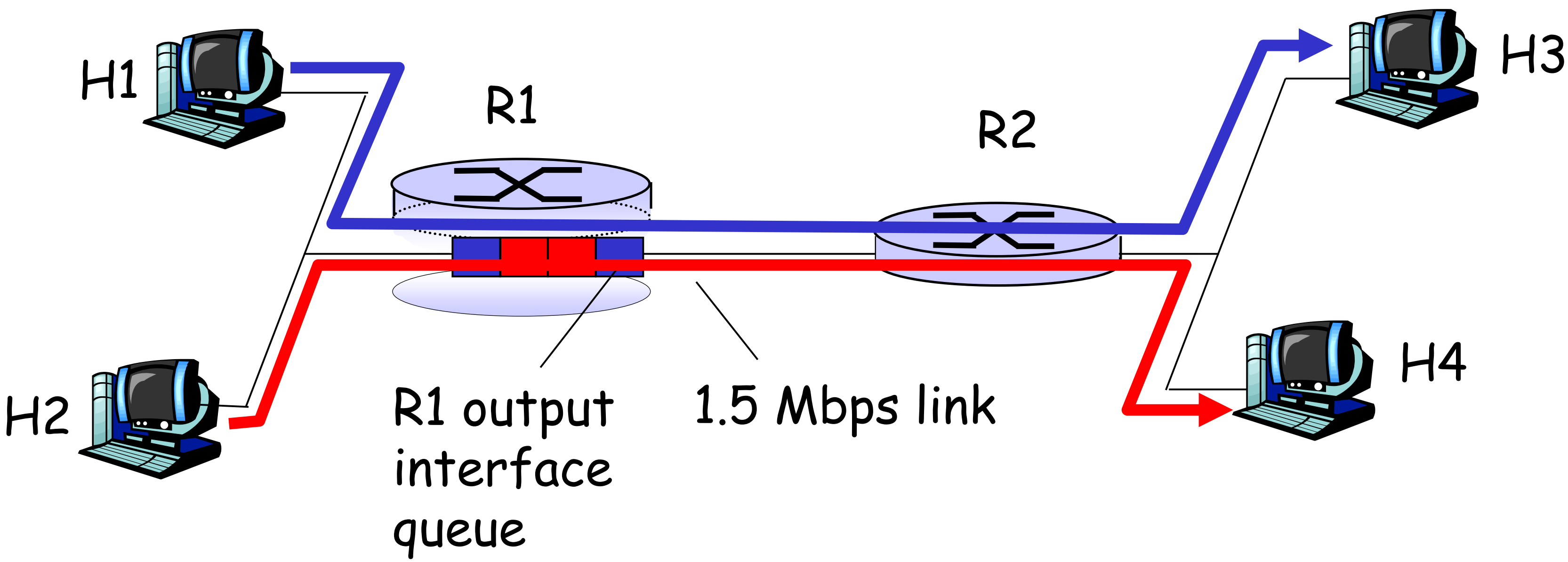
7.6 providing QoS guarantees

# 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

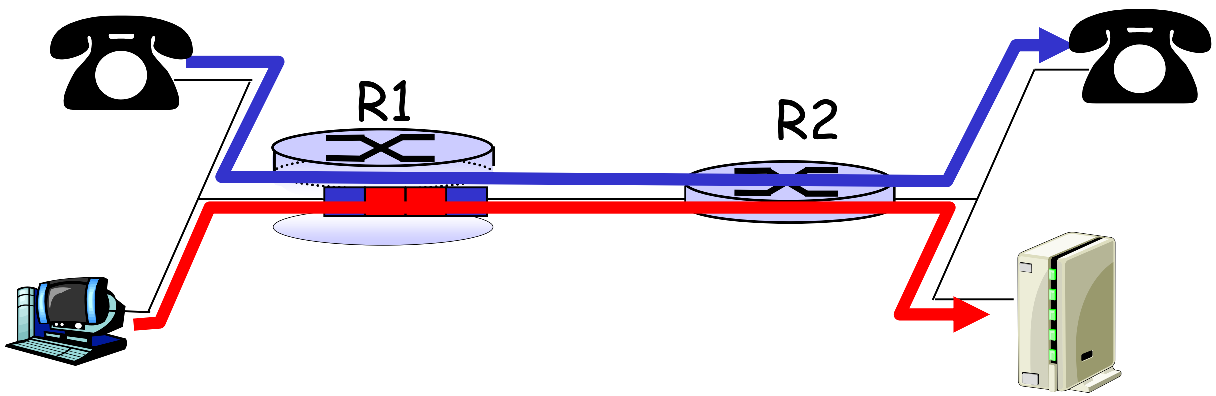


# Multiple classes of service: scenario



# 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

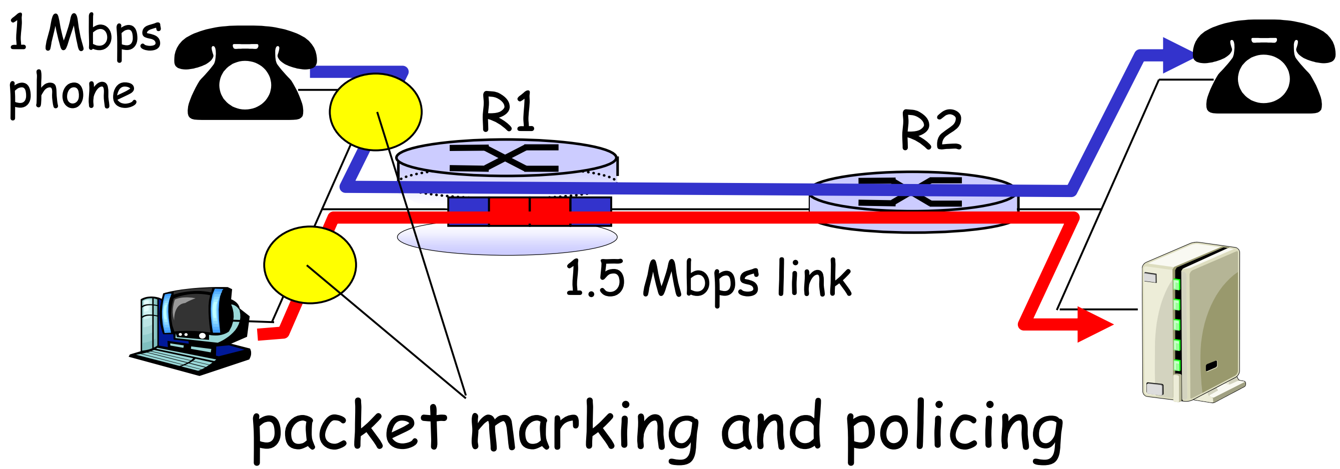



**Principle 1**

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)

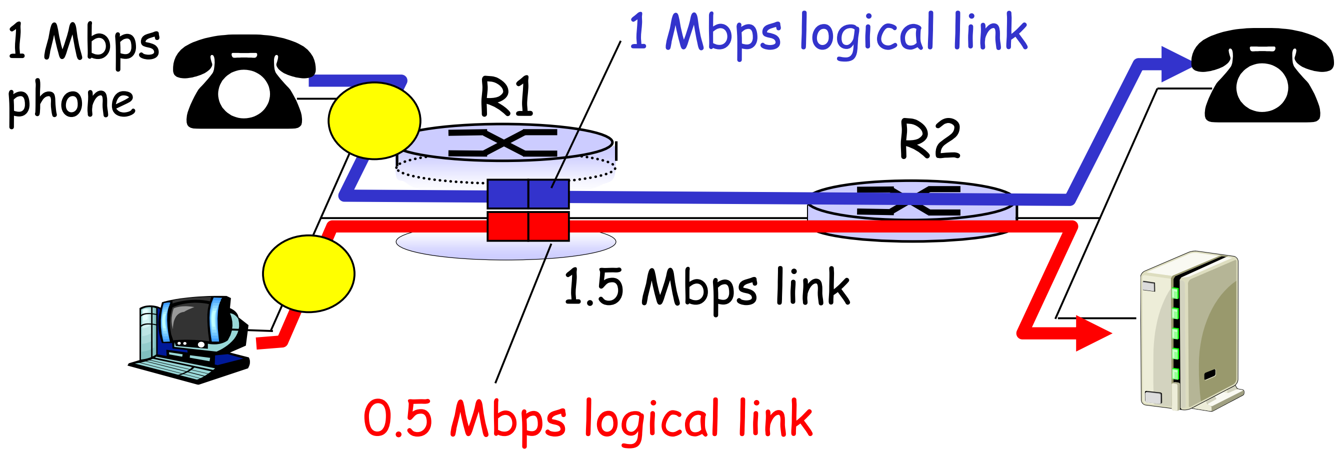



**Principle 2**

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

# Principles for QOS Guarantees (more)

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

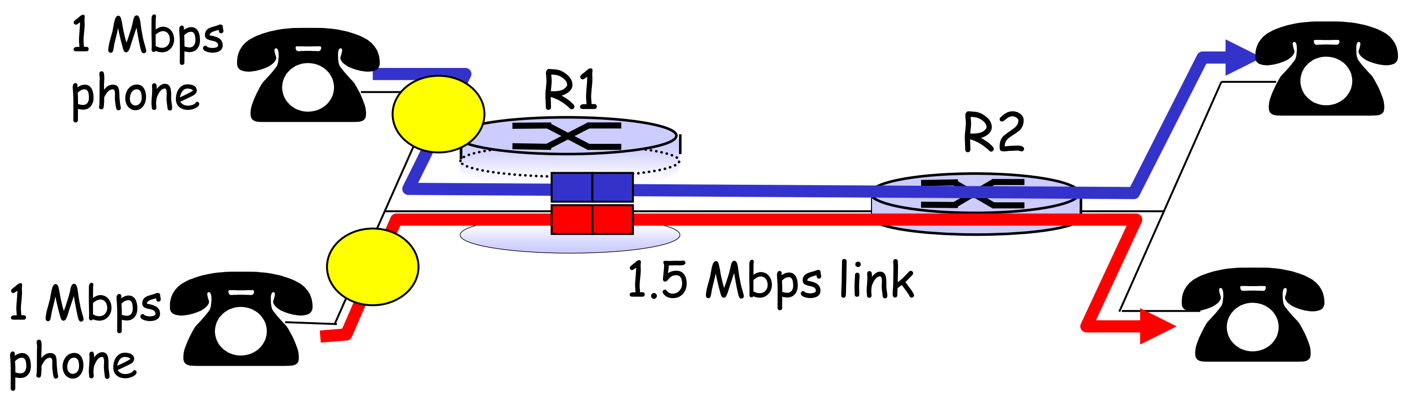



**Principle 3**

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

# Principles for QOS Guarantees (more)

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

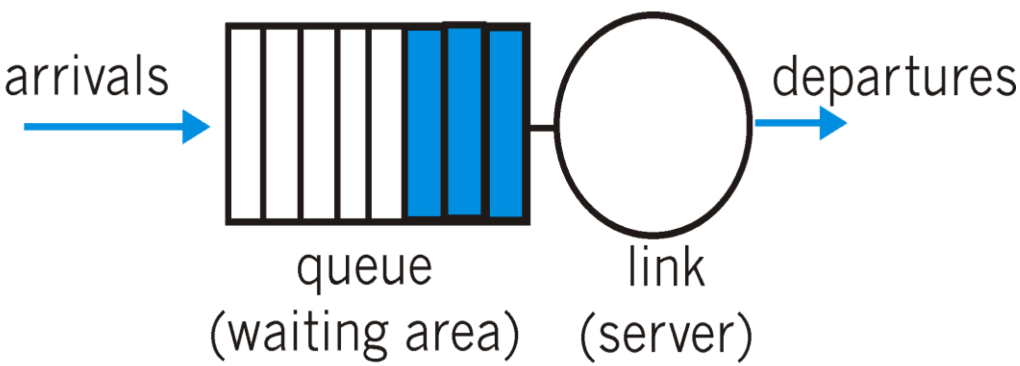



**Principle 4**

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

# Scheduling And Policing Mechanisms

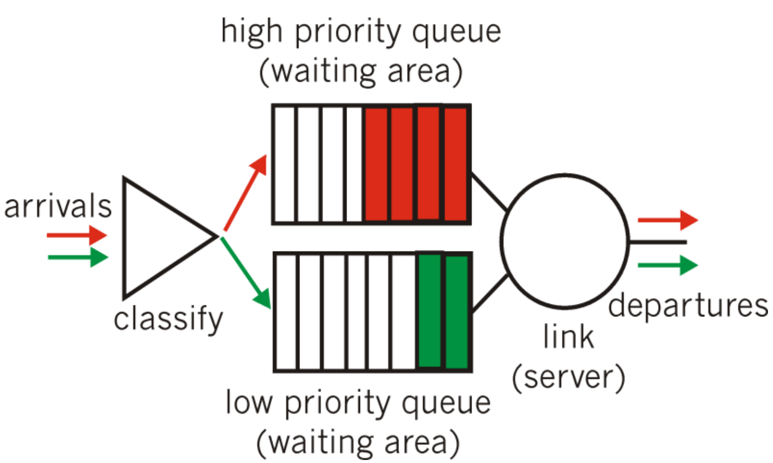
- **scheduling:** choose next packet to send on link
- **FIFO (first in first out) scheduling:** send in order of arrival to queue
  - real-world example?
  - **discard policy:** if packet arrives to full queue: who to discard?
    - Tail drop: drop arriving packet
    - priority: drop/remove on priority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

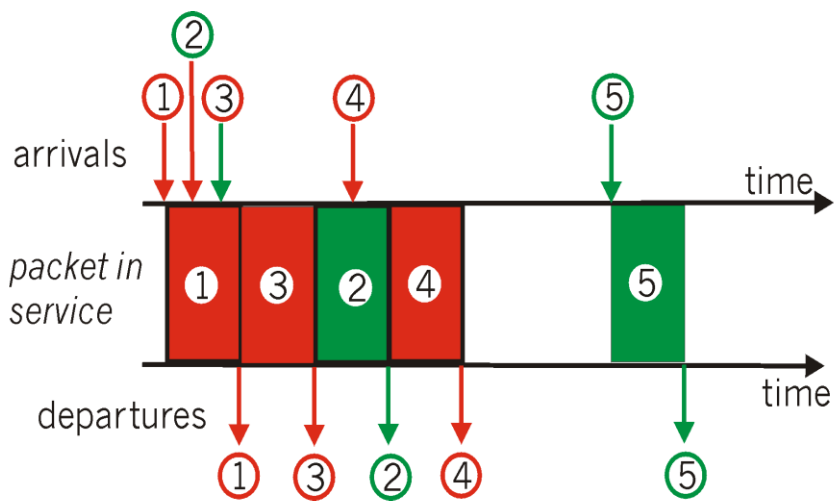


# Scheduling Policies: more

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?

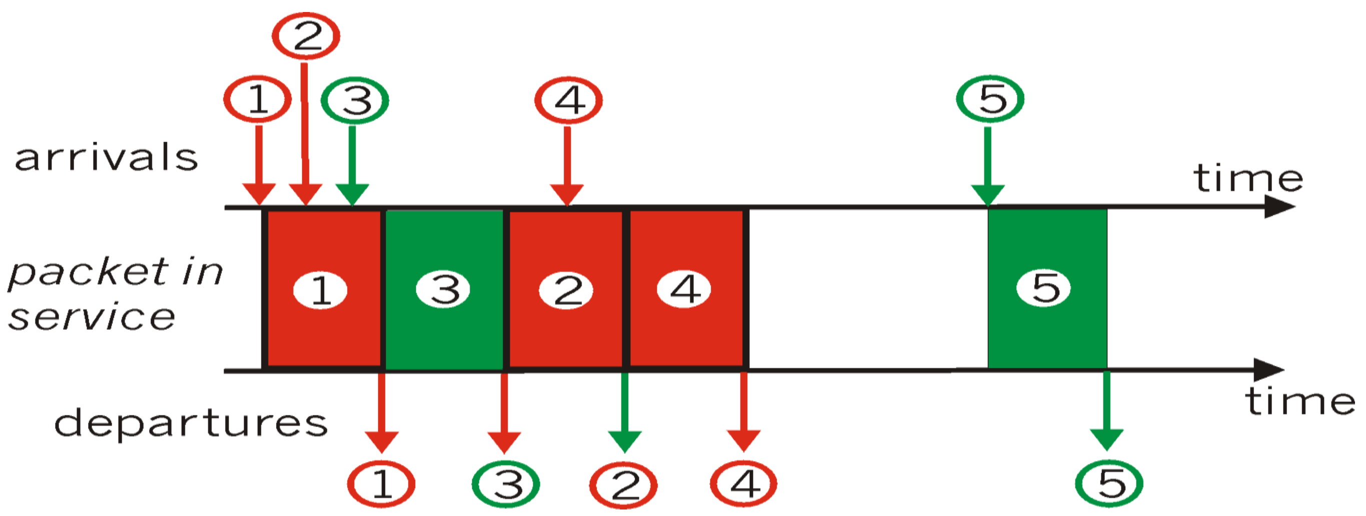


# Scheduling Policies: still more

round robin scheduling:

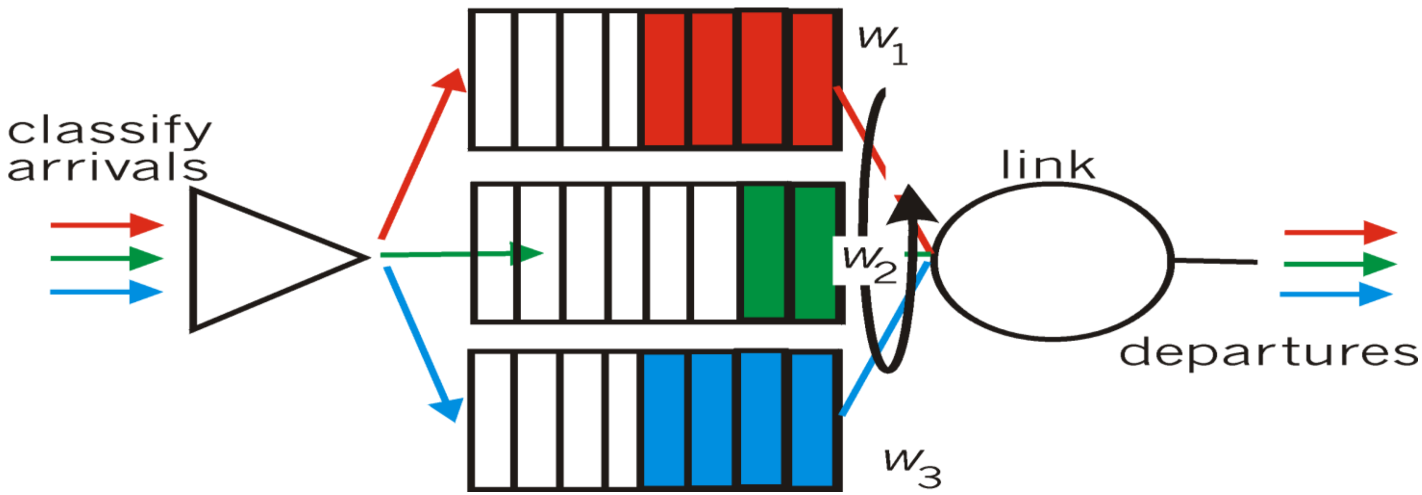
- multiple classes
- cyclically scan class queues, serving one from each class (if available)
- real world example?



# Scheduling Policies: still more

Weighted Fair Queuing:

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



# Policing Mechanisms

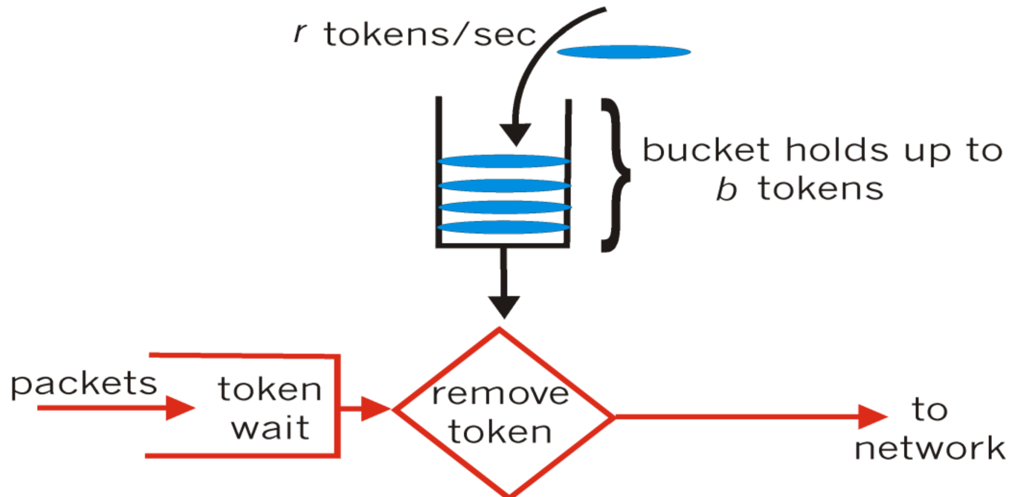
**Goal:** limit traffic to not exceed declared parameters

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)

# Policing Mechanisms

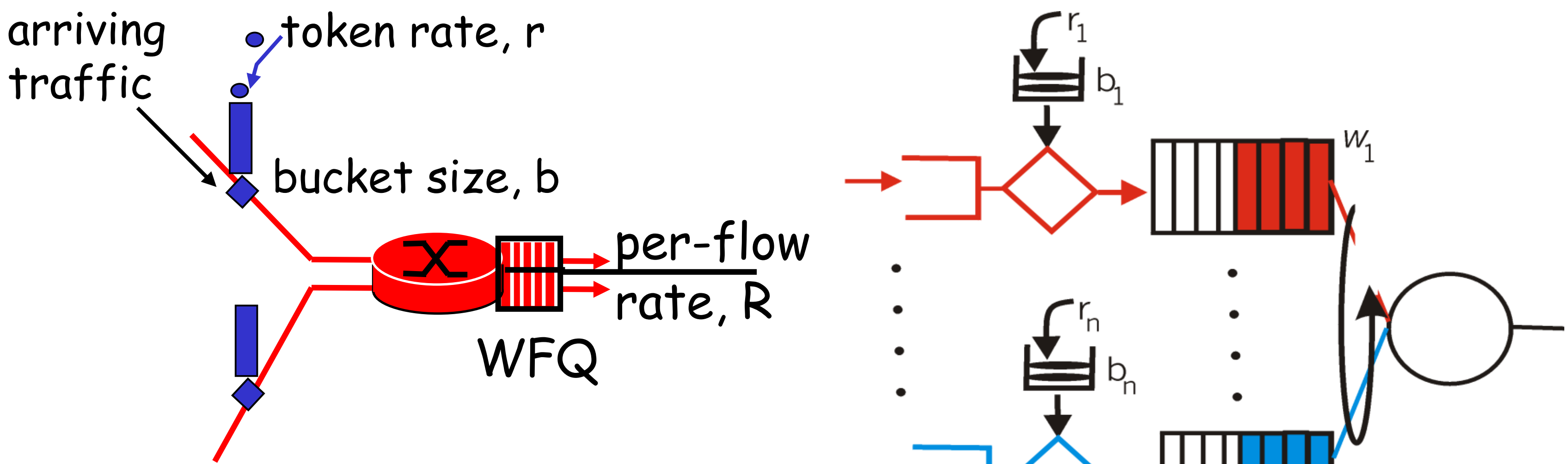
**Token Bucket:** limit input to specified Burst Size and Average Rate.




- 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).*

# Policing Mechanisms (more)

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




$D_{max} = b/R$

# IETF Differentiated Services

- 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 define service classes, provide functional components to build service classes

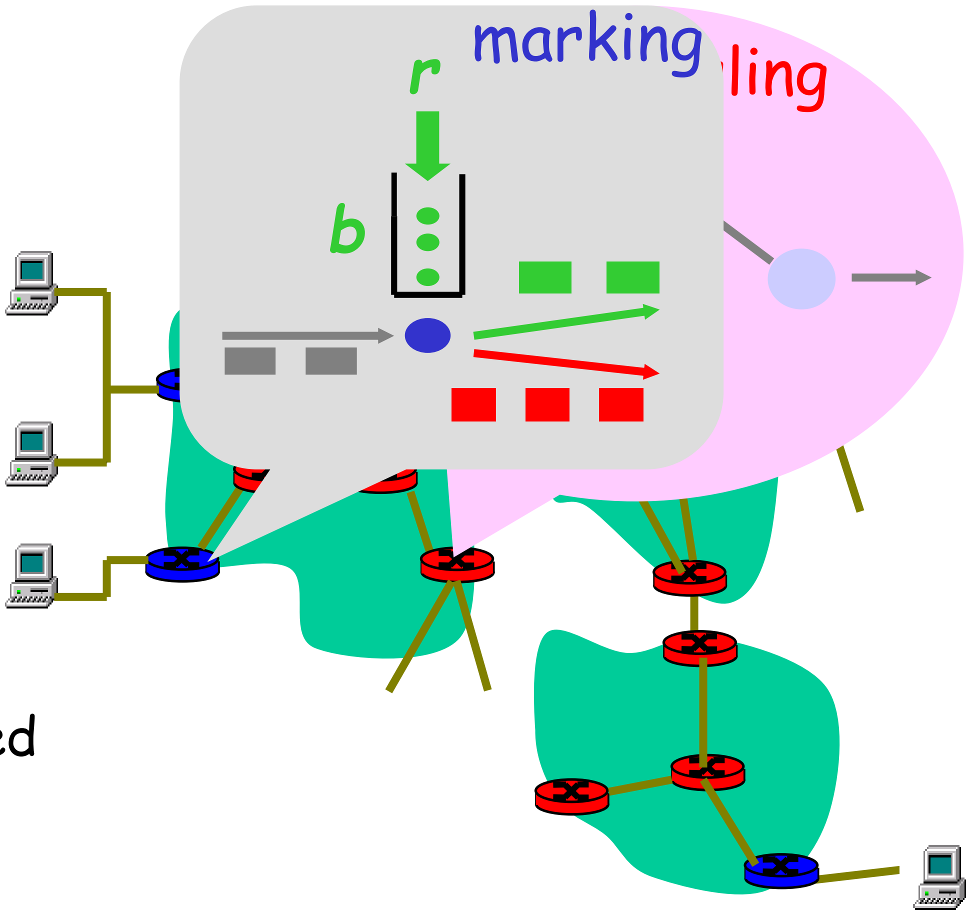
# Diffserv Architecture

## Edge router:

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

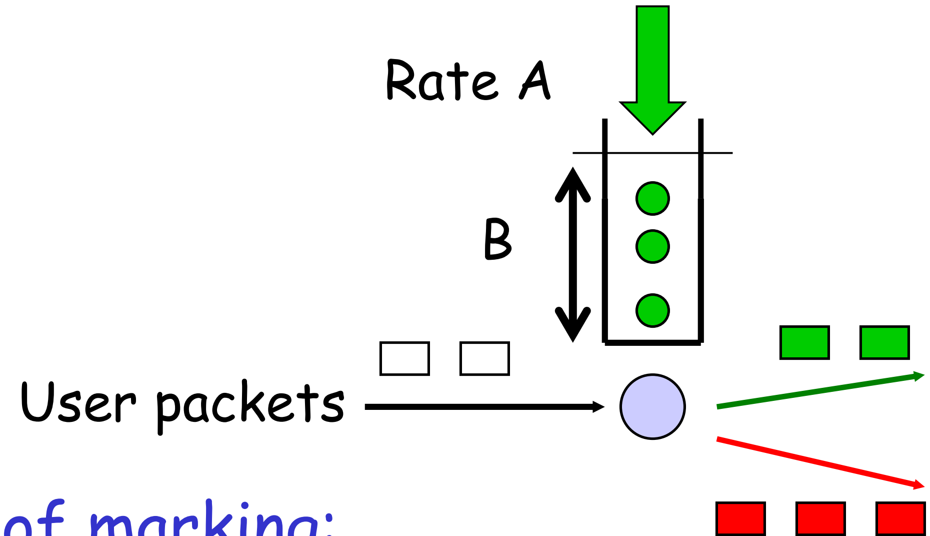
## Core router:

- 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

# 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?

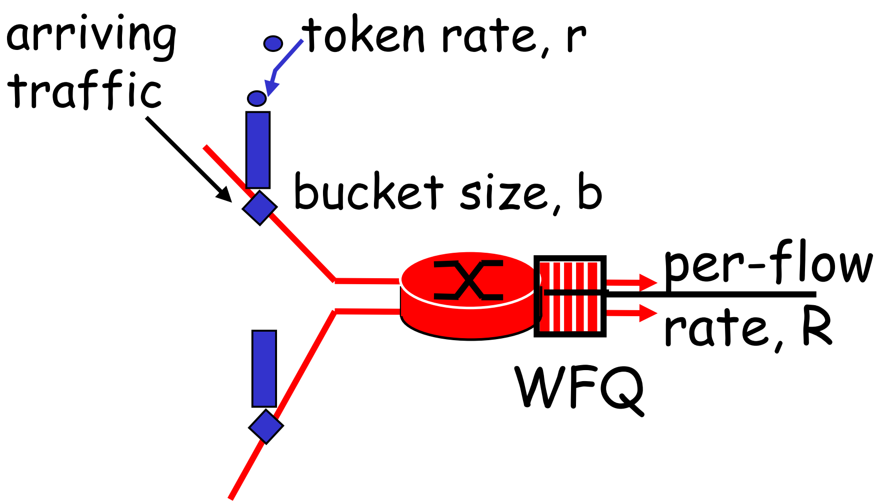
# Intserv QoS: Service models [rfc2211, rfc 2212]

## Guaranteed service:

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

## Controlled load service:

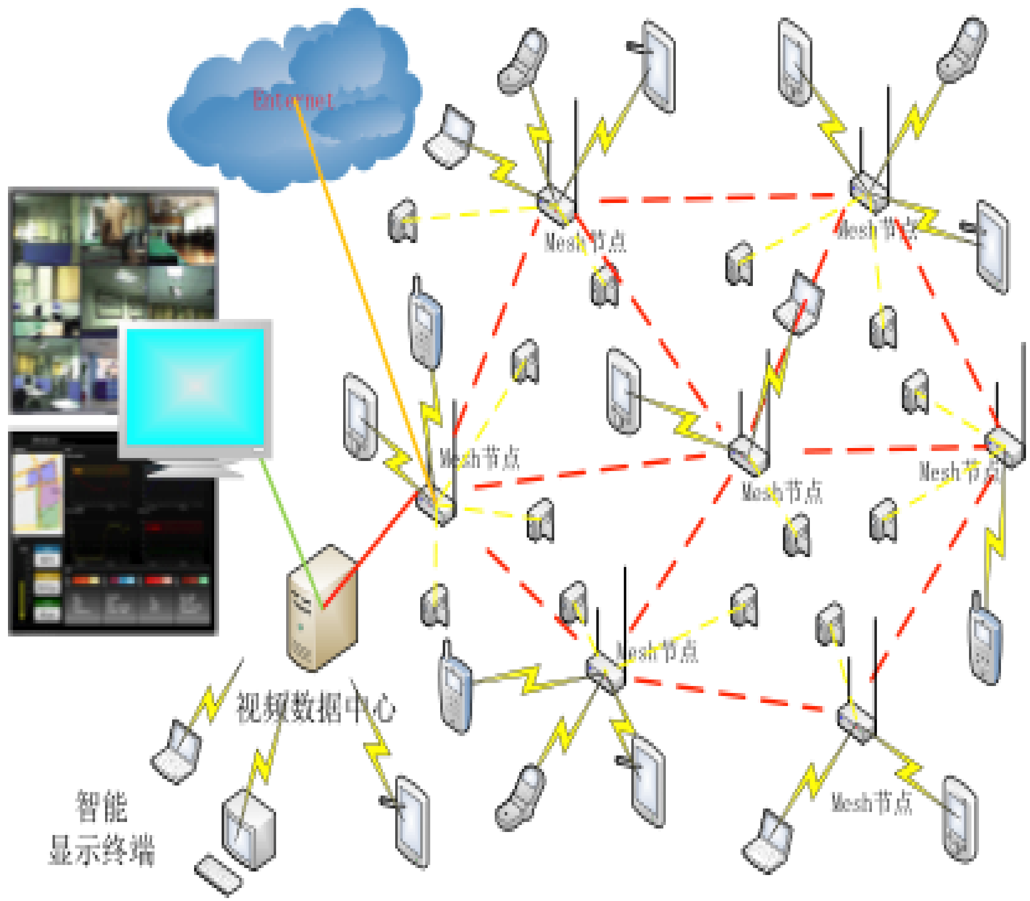
- "a quality of service closely approximating the QoS that same flow would receive from an unloaded network element."




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

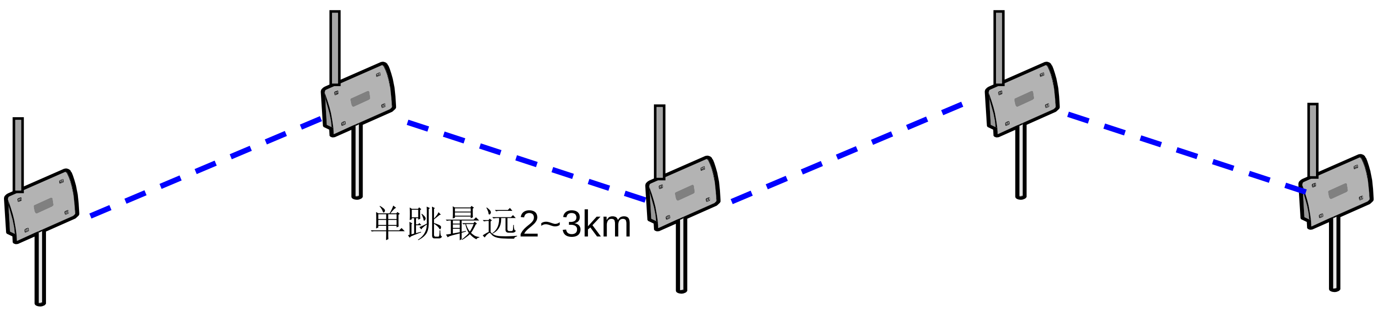
# 案例：利用无线Mesh网络传输视频

- 多跳自组织
  - 自愈合
  - 自组织
- 高带宽
  - 单跳最大150Mbps
- 多用途
  - 语音、视频通话
  - 视频监控、抓拍系统
  - 网络覆盖
- 方便部署
  - 多种供电
  - 加电自动入网



# 高带宽无线MESH网络—优势

- **优势1：多跳组网，通信距离长**



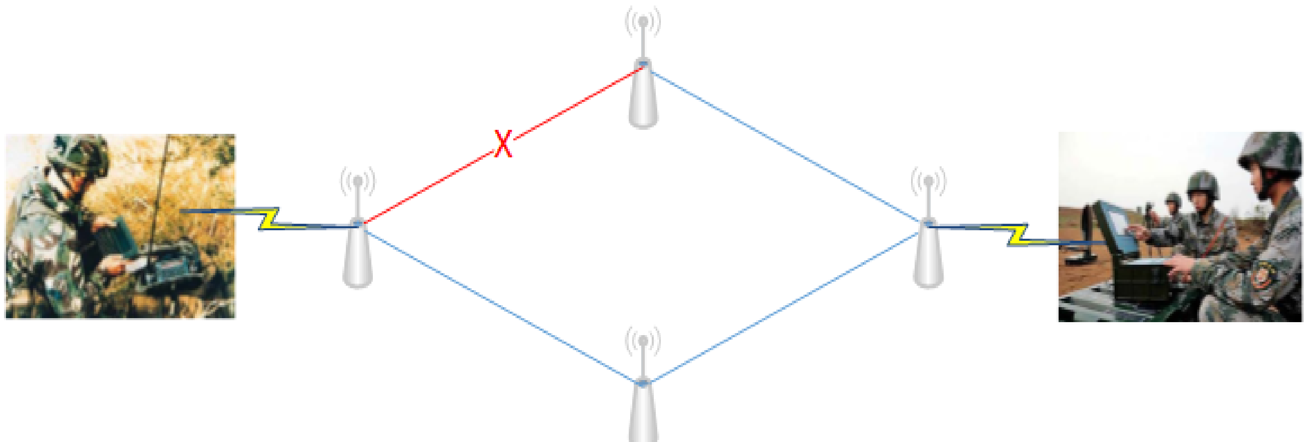

以多跳传输的方式与数十公里外的节点通信

# 高带宽无线MESH网络—优势

- **优势2：带宽高**，支持多路视频传输
  - 最高传输速率：**150Mbps**

# 高带宽无线MESH网络—优势

- **优势3：网络自愈合**



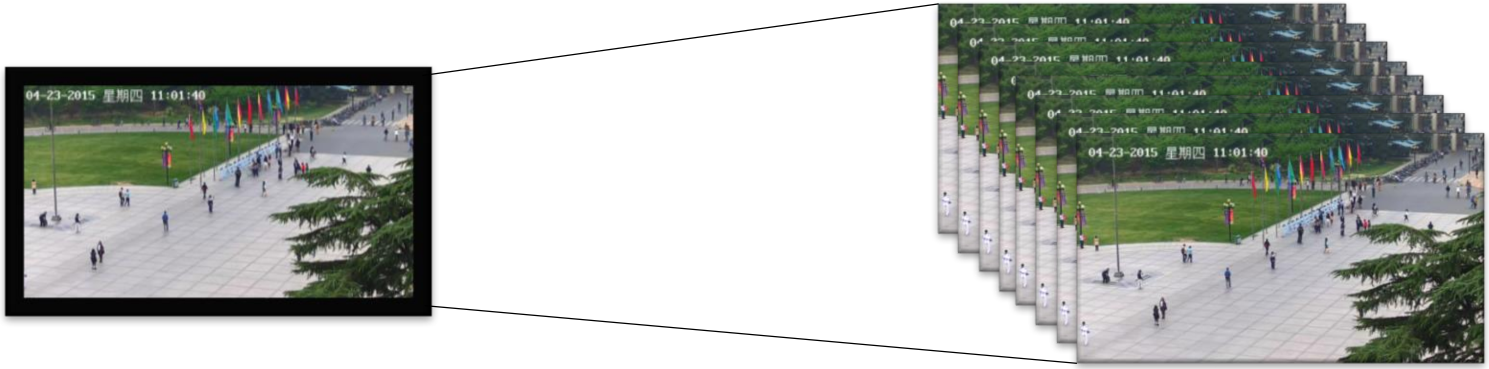

**当一条线路断路时，该骨干网络会自动寻找其他最优路径，保证数据传输的通畅**

# 高带宽无线MESH网络—优势

- **优势4：** 网络自动搭建，**部署、维护简单**
  - 第一步：固定节点
  - 第二步：通电启动
  - 第三步：完成！

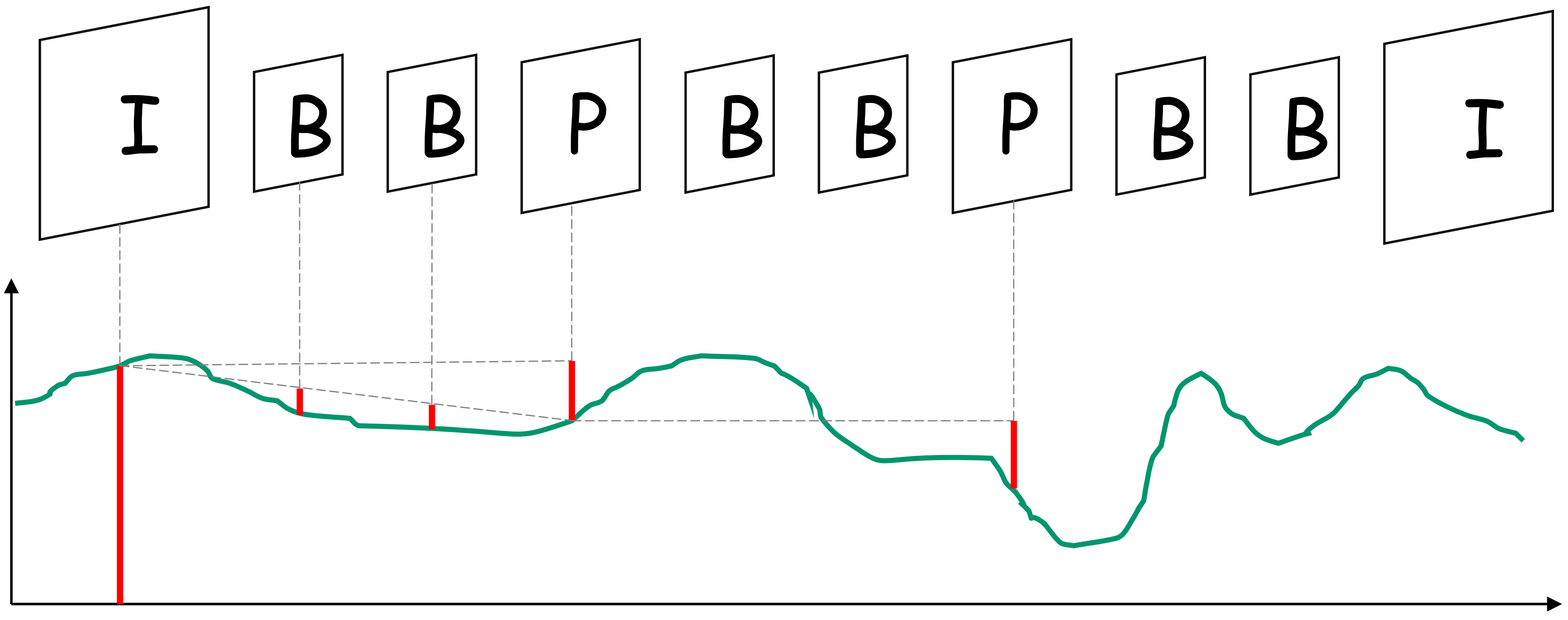
# H.264

- **H.264** is a video compression format that is currently one of the most commonly used formats.
- Video -> Frames -> Groups of Frames



# GOP of H.264

- Three Types of Frames: I, P and B

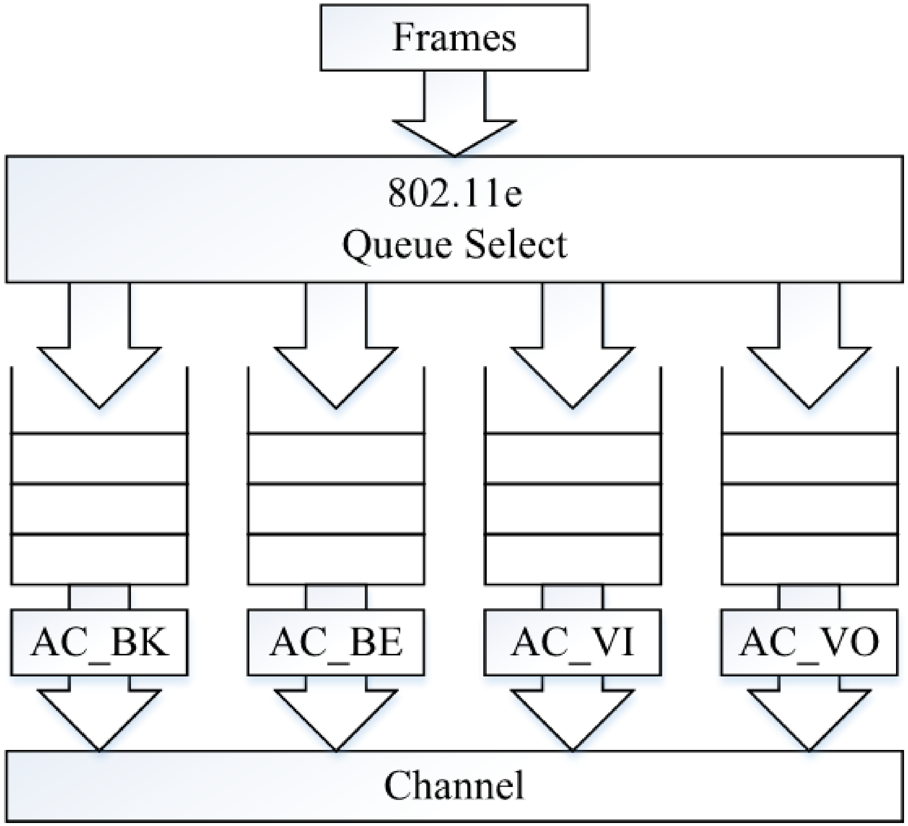


# Importance of H.264 Frames

- Considering Video Disruption Caused by Frame Loss

# 802.11e EDCA

- 802.11 -> DCF
  - 1 Queue
  - Best Effort
- 802.11e -> EDCA
  - 4 Queue
  - Different Priorities



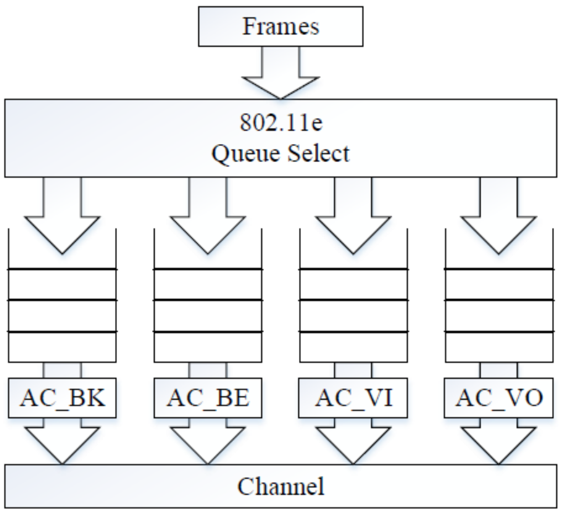
# Mapping

- Higher Importance -> Higher Priority

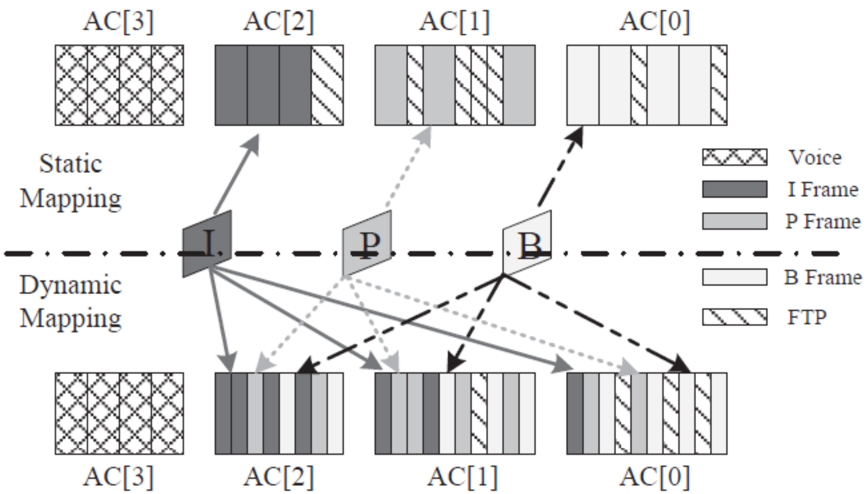
| | | | |
|---|---|---|---|
| I | I | B | B |
| I | P | P | B |
| P | P | B | B |
| I | P | B | B |

IPIIPIPPPBPBBPBB…

# 工作原理

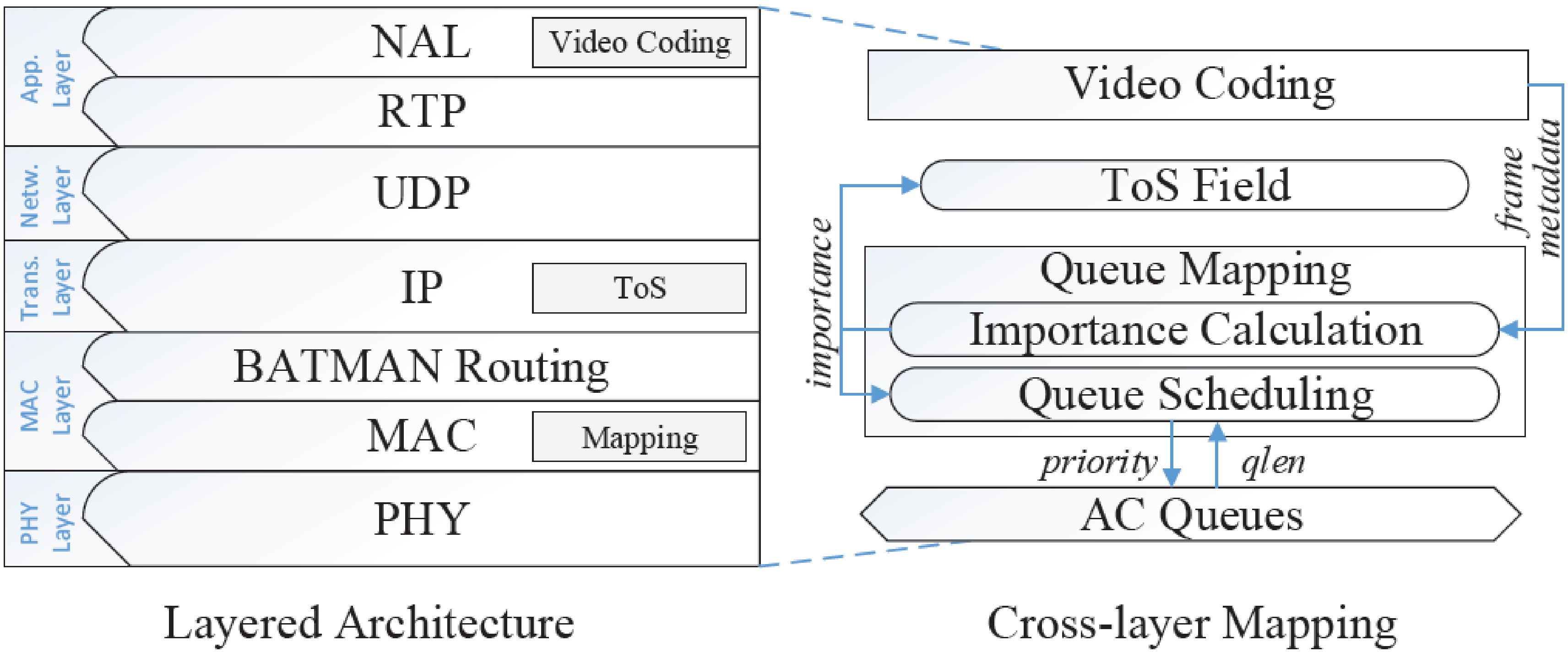



多优先级传输



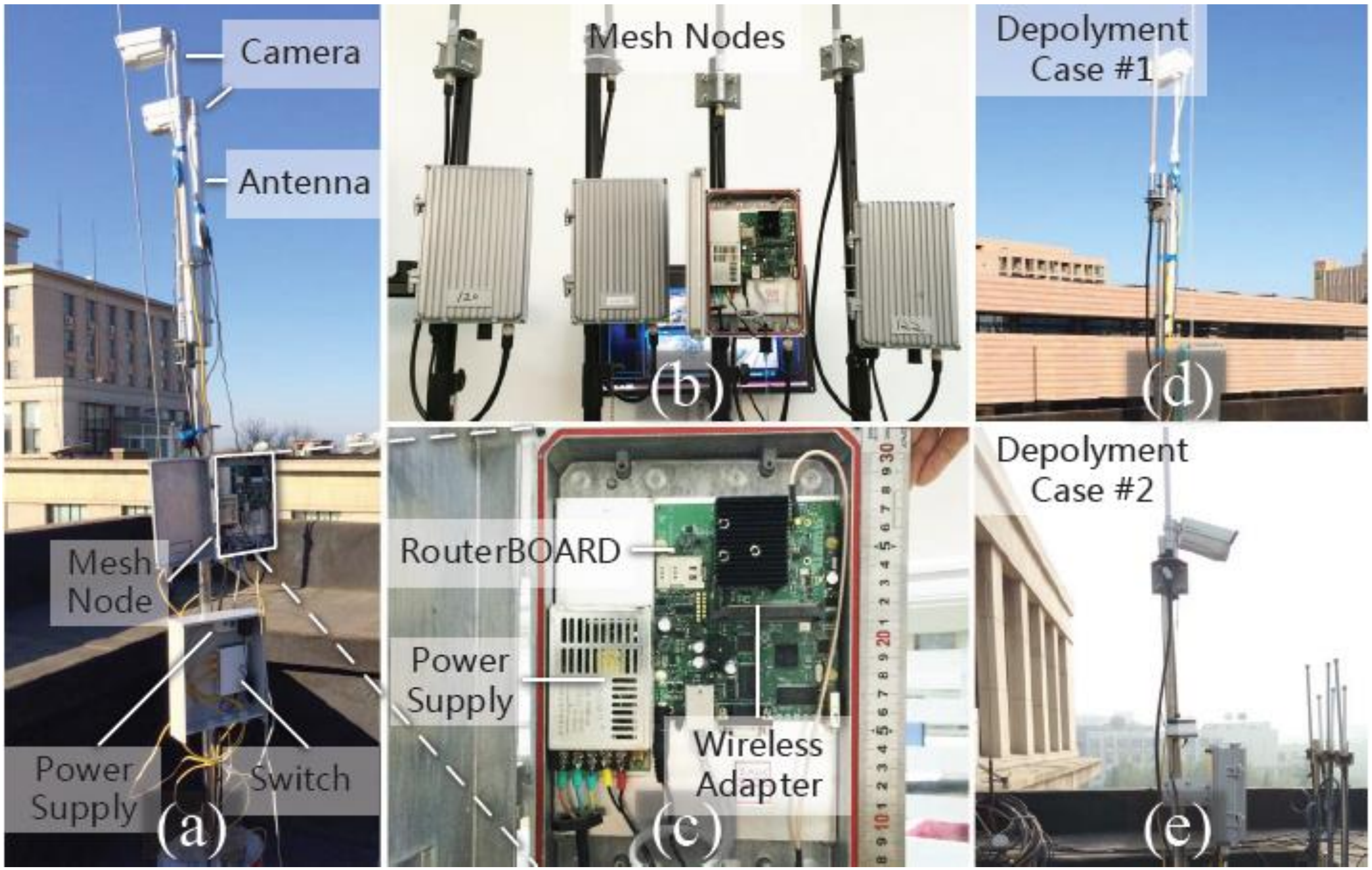

动态数据映射

# 实现方式



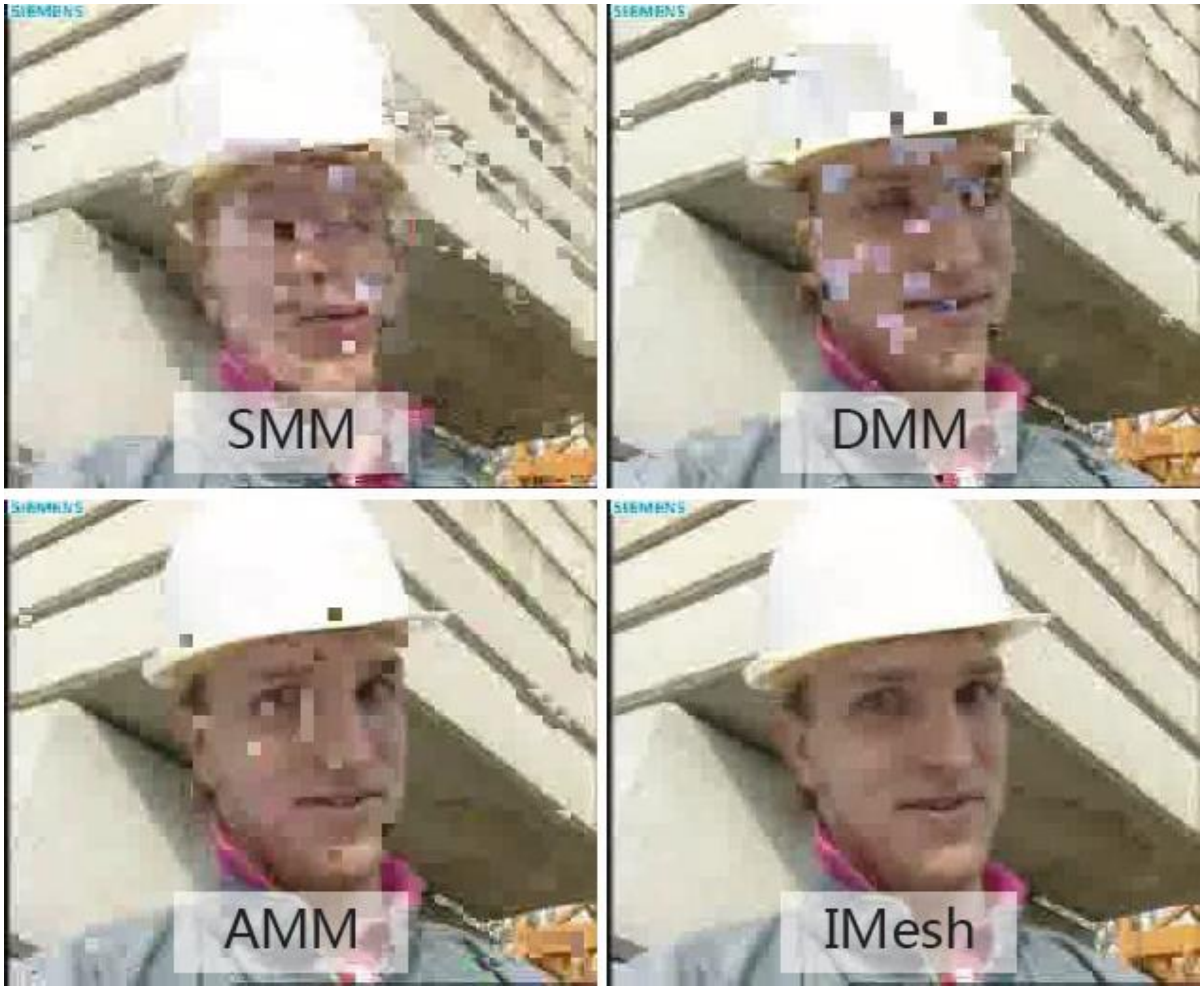

(b) Cross-layer design

# 部署


Camera
Antenna
Mesh Node
Power Supply
Switch
(a)
Mesh Nodes
(b)
RouterBOARD
Power Supply
Wireless Adapter
(c)
Depolyment Case #1
(d)
Depolyment Case #2
(e)

# 部署

# 效果



# 技术特点

多优先级传输

- 在无线自组织网络中实现多优先级队列，不同权重的数据包可以通过不同优先级的队列传输，协议实现与802.11e协议兼容，具有高效、鲁棒等特性

动态数据映射

- 实时探测网络传输质量并针对网络状况动态调整数据包映射策略，确保权重更大的数据包具有更高的链路占用概率，充分利用无线信道容量

跨层协议设计

- 针对主流视频编码协议（例如H.264）设计高效的数据链路层与网络层协议，统一优化多层协议的运行机制和参数，保障多媒体应用在极端恶劣的网络条件下流畅运行。

# Chapter 7: 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