بسم الله الرحمن الرحیم

Information Technology Engineering

Mohammad Hossein Manshaei

manshaei@gmail.com

1393

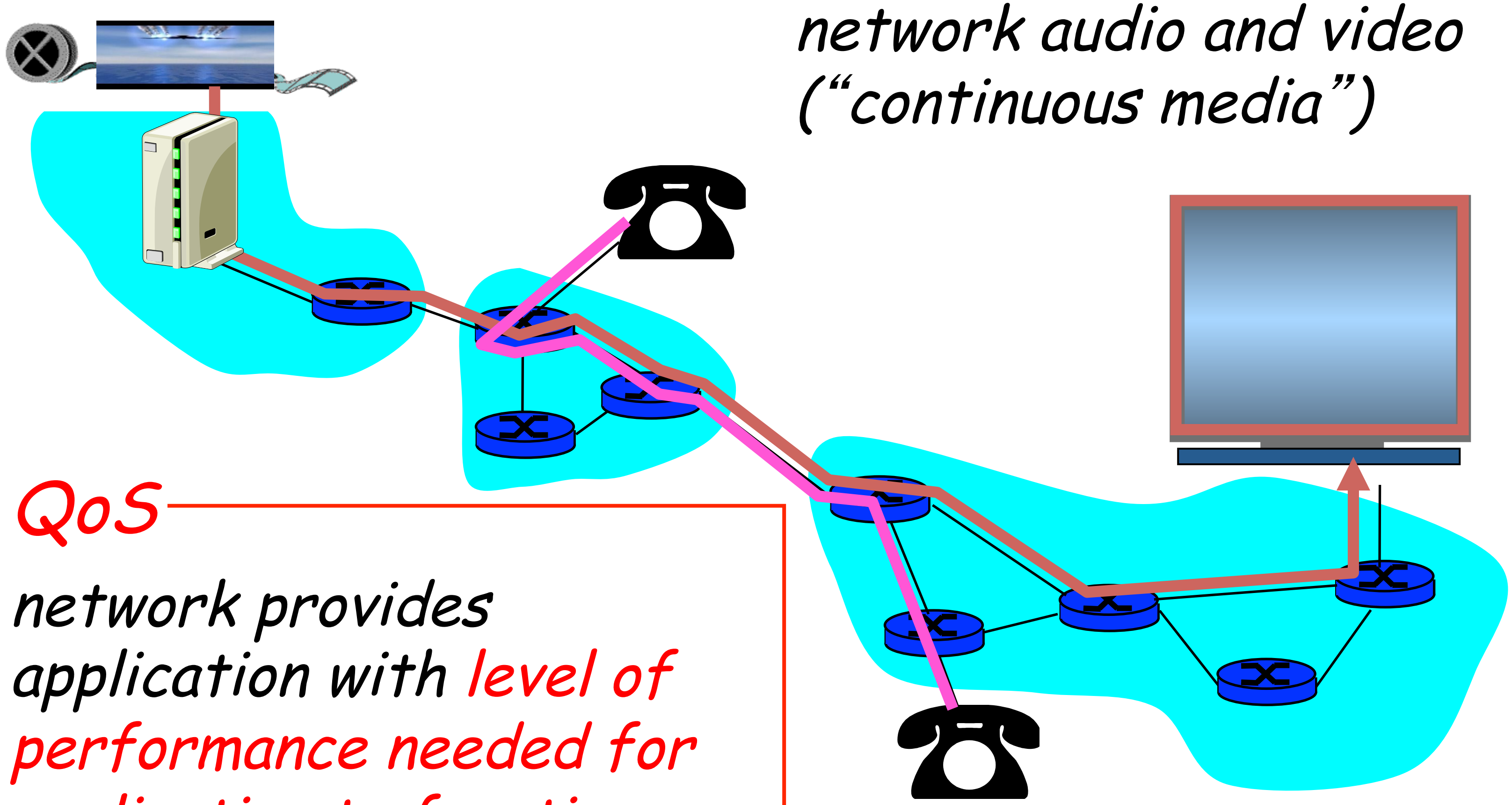
Voice and Video over IP

MULTIMEDIA NETWORKING

Slides derived from those available on the Web site of the book "Computer Networking", by Kurose and Ross, PEARSON

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

Multimedia networking: outline

7.1 Multimedia networking applications

7.2 Streaming *stored* video

7.3 Voice-over-IP

7.4 Protocols for *real-time* conversational applications

7.5 Network support for multimedia

Multimedia: video

- Video: sequence of images displayed at constant rate
  - e.g. 24 images/sec
- Digital image: array of pixels
  - each pixel represented by bits
- Coding: use redundancy *within* and *between* images to decrease # bits used to encode image
  - spatial (within image)
  - temporal (from one image to next)

spatial coding example: instead of sending *N* values of same color (all purple), send only two values: color value (*purple*) *and number of repeated values* (N)

frame *i*

temporal coding example: instead of sending complete frame at i+1, send only differences from frame i

frame *i+1*

Multimedia: video

- *CBR: (constant bit rate):* video encoding rate fixed
- *VBR: (variable bit rate):* video encoding rate changes as amount of spatial, temporal coding changes
- *examples:*
  - MPEG 1 (CD-ROM) 1.5 Mbps
  - MPEG2 (DVD) 3-6 Mbps
  - MPEG4 (often used in Internet, < 1 Mbps)

spatial coding example: instead of sending *N* values of same color (all purple), send only two values: color value (*purple*) *and number of repeated values* (N)

frame *i*

temporal coding example: instead of sending complete frame at i+1, send only differences from frame i

frame *i+1*

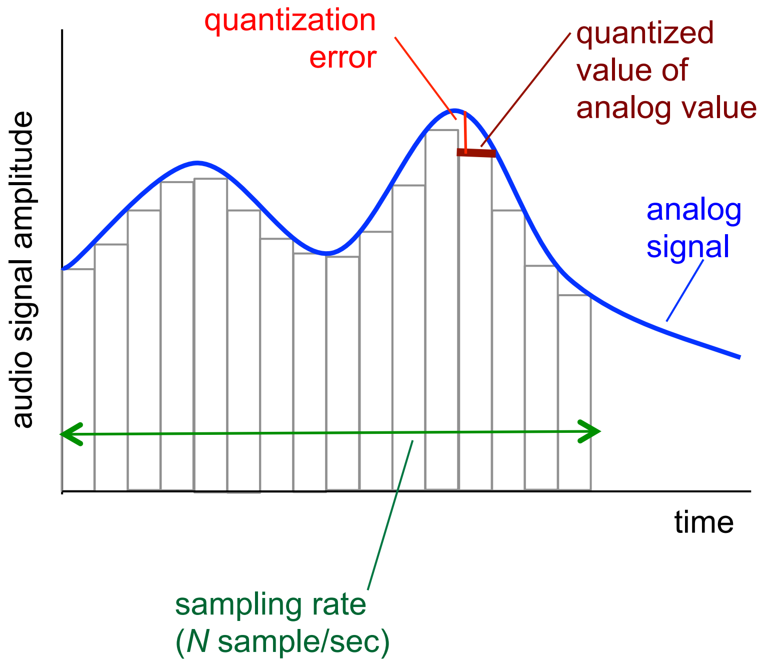
Bit-Rate Requirement for 3 Applications

	Bit rate	Bytes transferred in 67 min
Facebook Frank	160 kbps	80 Mbytes
Martha Music	128 kbps	64 Mbytes
Victor Video	2 Mbps	1 Gbyte

Cisco predicts that streaming and stored video will be **approximately 90 percent of global consumer** Internet traffic by 2015.

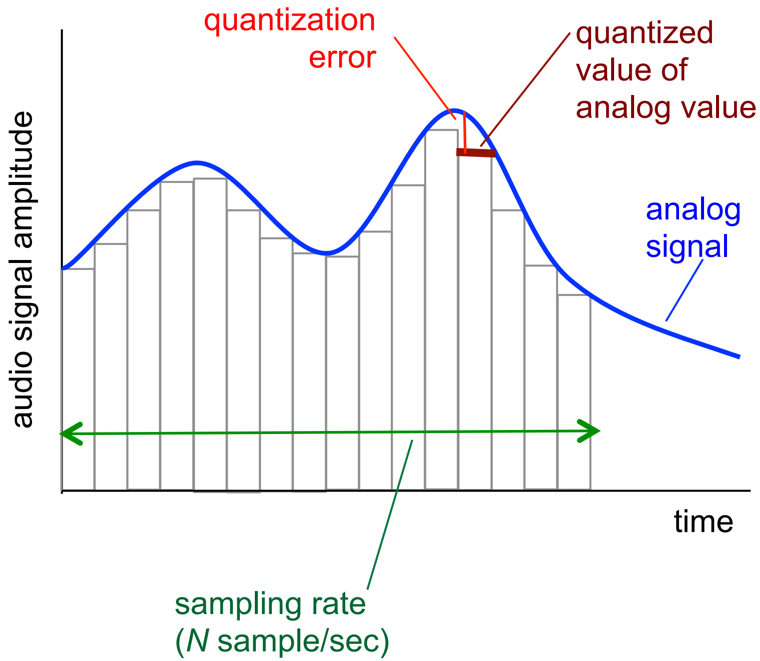
Multimedia: audio

- Analog audio 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, e.g., 8 bits for 256 values



Multimedia: audio

- 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
- Advanced Audio Coding (AAC): Apple standard
- Internet telephony: 5.3 kbps and up

MM Networking Applications

Fundamental characteristics:

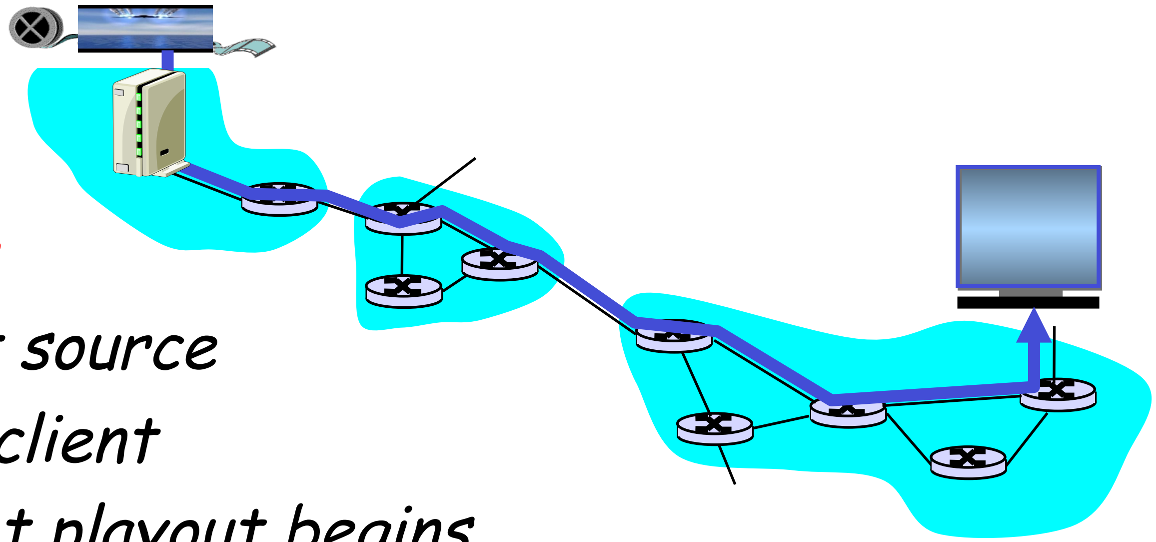
- Typically **delay sensitive**
  - end-to-end delay
  - delay jitter
- **Loss tolerant**: infrequent losses cause minor glitches
- Antithesis of data, which are loss *intolerant* but delay *tolerant*.

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

Multimedia networking: 3 application types

- *Streaming, stored* audio, video
  - *streaming:* can begin playout before downloading entire file
  - *stored (at server):* can transmit faster than audio/video will be rendered (implies storing/buffering at client)
  - e.g., YouTube, Netflix, Hulu
- *Conversational* voice/video over IP
  - interactive nature of human-to-human conversation limits delay tolerance
  - e.g., Skype
- *Streaming live* audio, video
  - e.g., live sporting event (football)

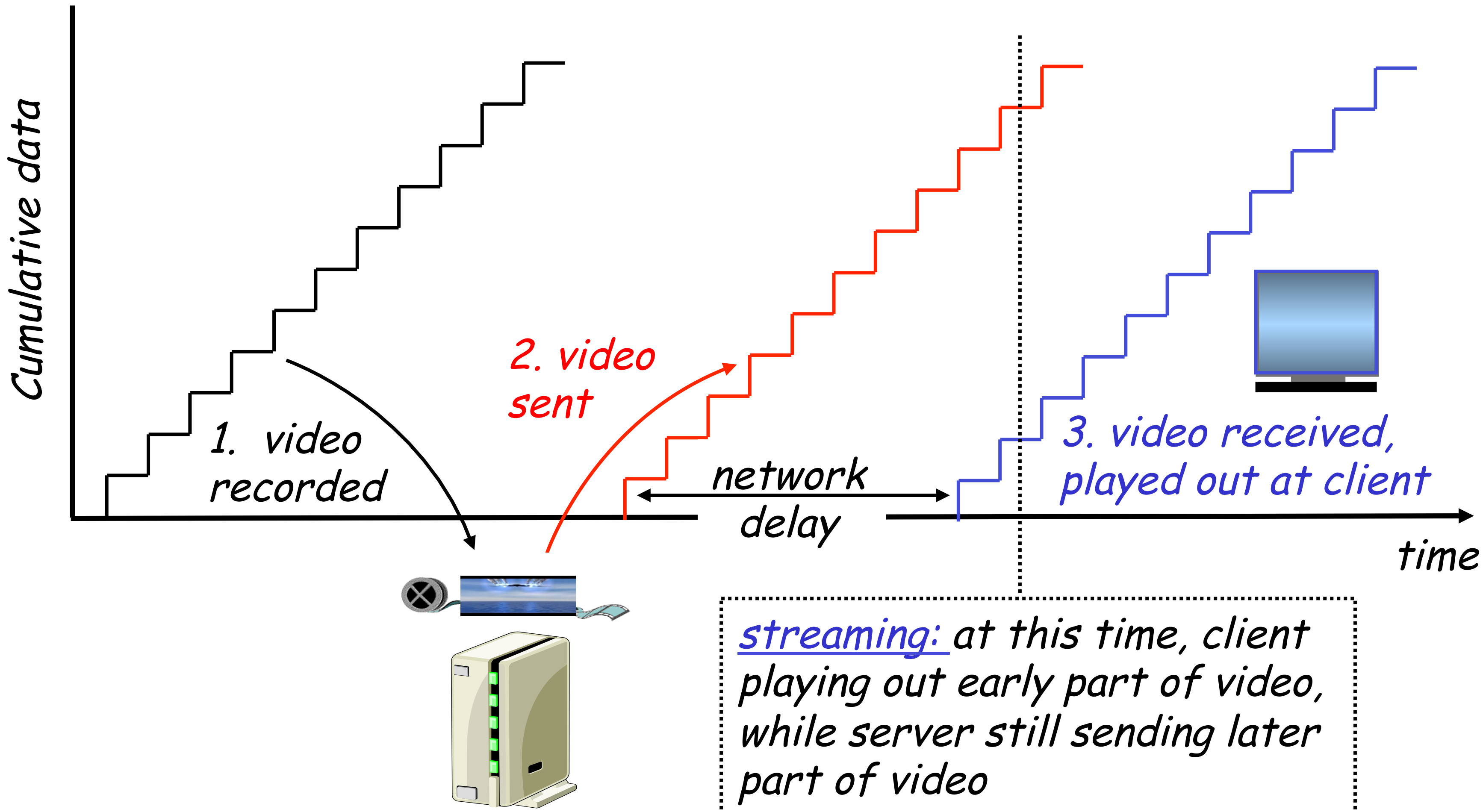
Streaming Stored Multimedia



Stored streaming:

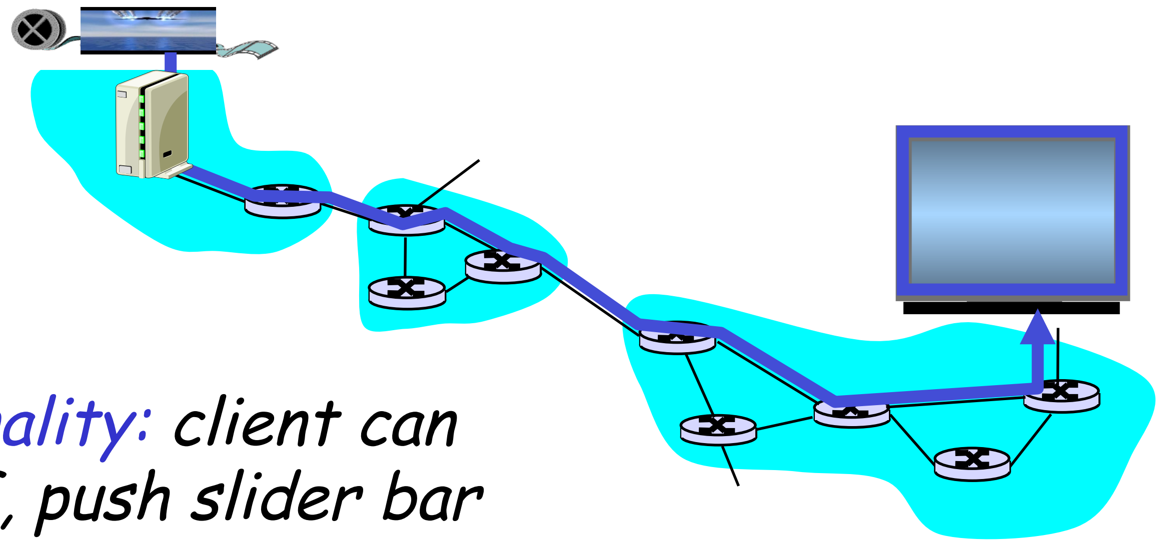
- *media stored at source*
- *transmitted to client*
- *streaming: client playout begins before all data has arrived*
  - *timing constraint for still-to-be transmitted data: in time for playout*

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

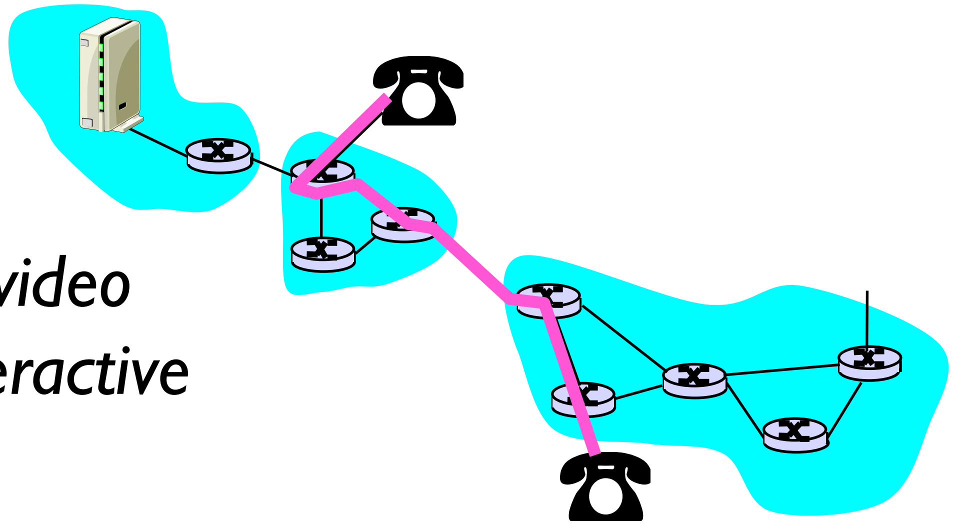
Streaming (as with streaming *stored* multimedia)

- 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



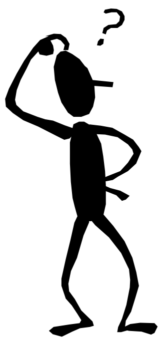
- *Applications: IP telephony, video conference, distributed interactive worlds*

  - 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?

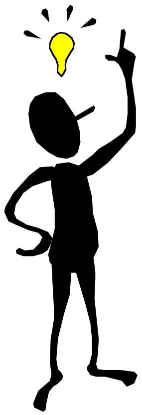
Multimedia Over Today's Internet

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:

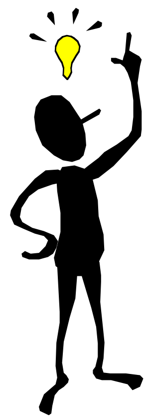
- fundamental changes in Internet so that apps can reserve end-to-end bandwidth
- requires new, complex software in hosts & routers

Laissez-faire

- no major changes
- more bandwidth when needed
- content distribution, application-layer multicast
  - In application layer and not IP layer multicast

Differentiated services philosophy:

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



What's your opinion?

Summary

Approach	Unit of Allocation	Guarantee	Deployment to date	Complexity	Mechanisms
Making the best of best-effort service	None	None, or soft	everywhere	minimal	Application-layer support, CDN, over-provisioning
Differential QoS	Classes of Flows	None, or Soft	some	medium	Policing, Scheduling
Guaranteed QoS	Individual Flows	Soft or hard, once a flow is admitted	little	high	Policing, Scheduling, call admission and signaling

Multimedia networking: outline

7.1 Multimedia networking applications

7.2 Streaming *stored* video

7.3 Voice-over-IP

7.4 Protocols for *real-time* conversational applications

7.5 Network support for multimedia

Streaming Stored Multimedia

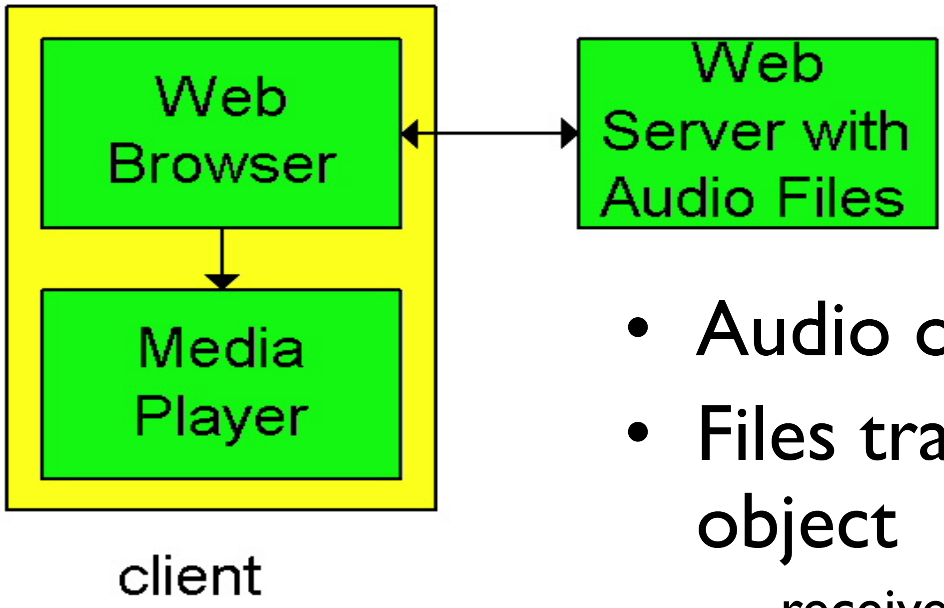
application-level streaming techniques for making the best out of best effort service:

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

Media Player

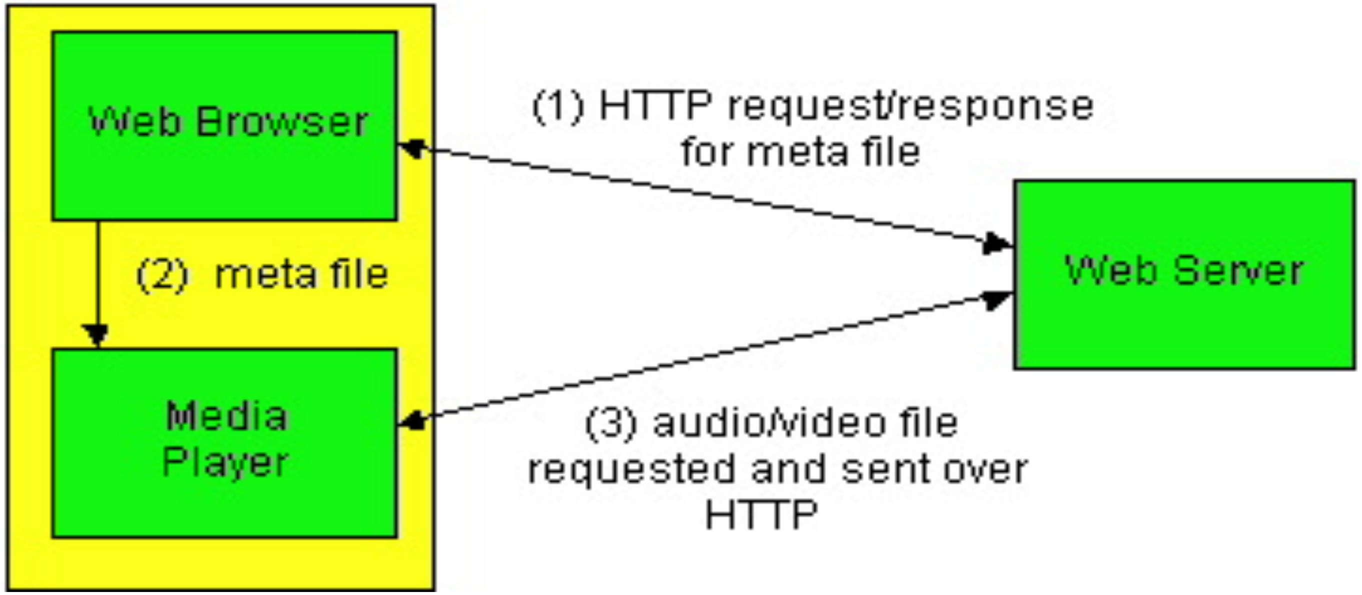
- jitter removal
- decompression
- error concealment
- graphical user interface w/ controls for interactivity

Internet multimedia: simplest approach



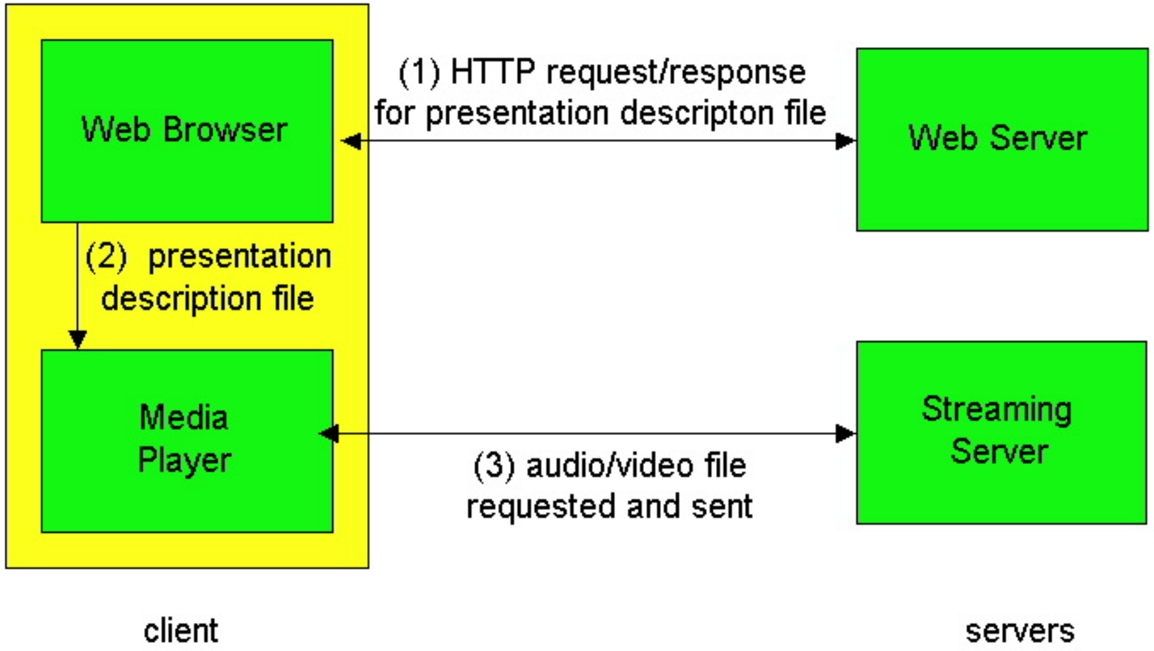
- Audio or video stored in file
- Files transferred as HTTP object
  - received in entirety at client
  - then passed to player

Internet Multimedia: Streaming Approach



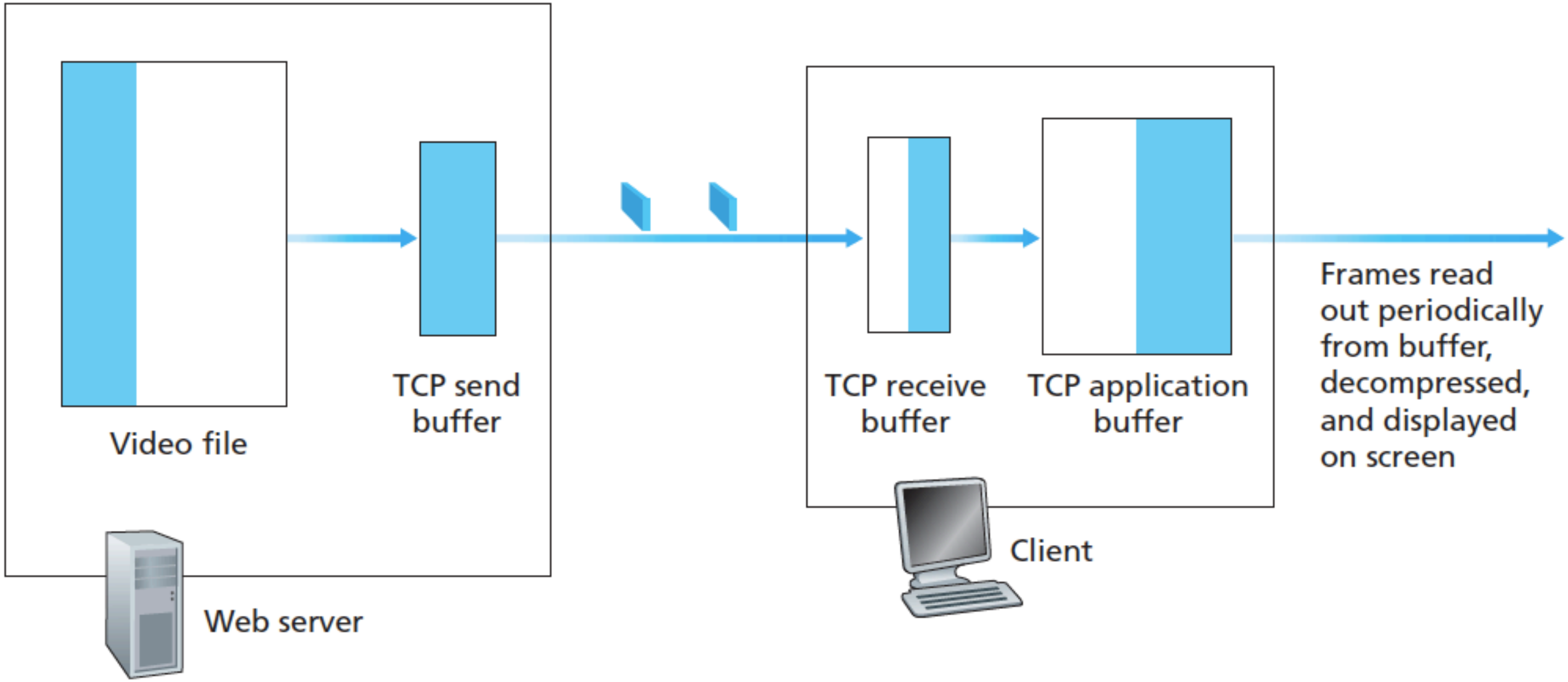
- browser GETs **metafile**
- browser launches player, passing metafile
- player contacts server
- server streams audio/video to player

Streaming from a Streaming Server

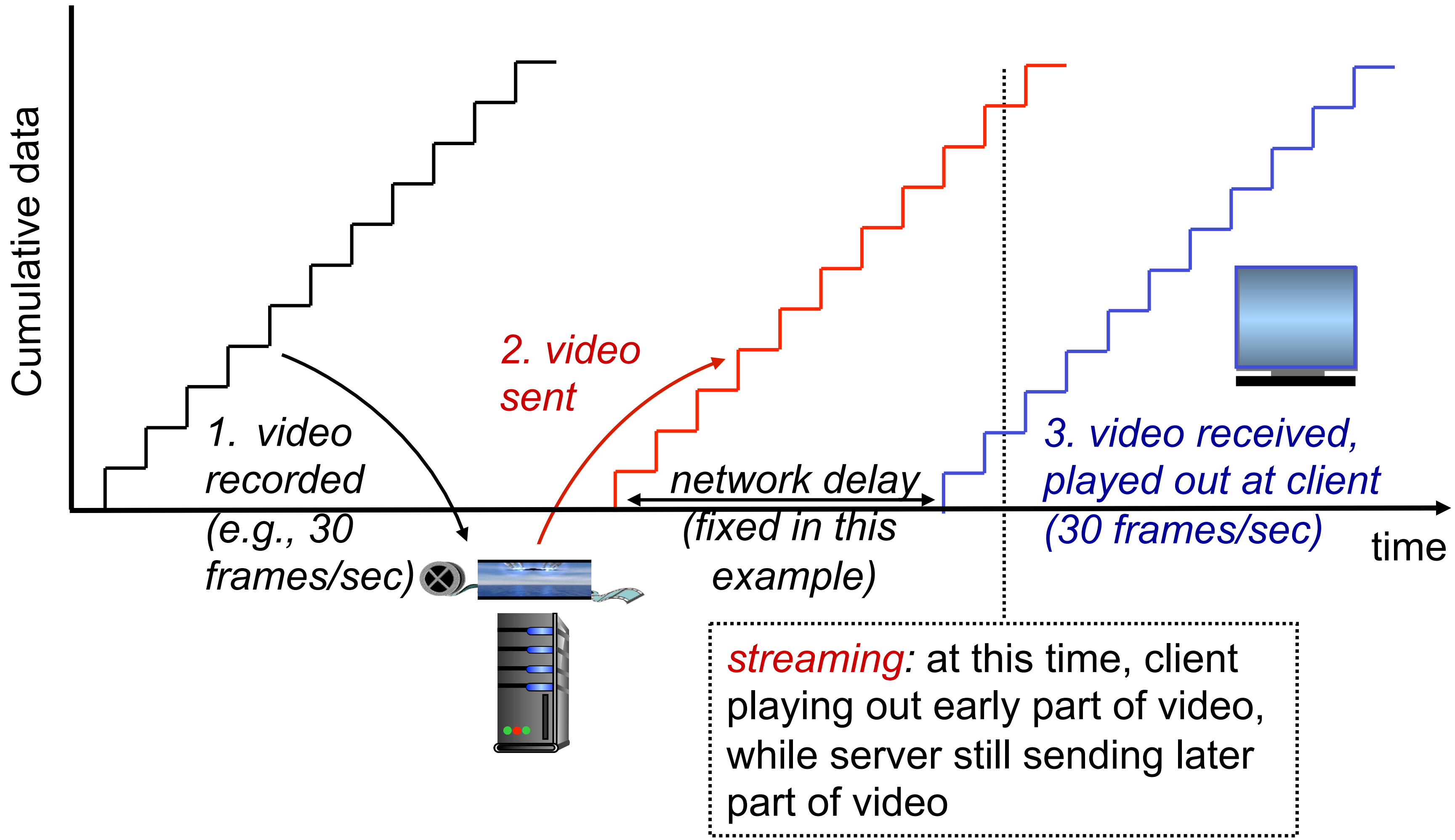


- Allows for non-HTTP protocol between server, media player
- UDP or TCP for step (3), more shortly

Streaming Stored Video over HTTP/TCP



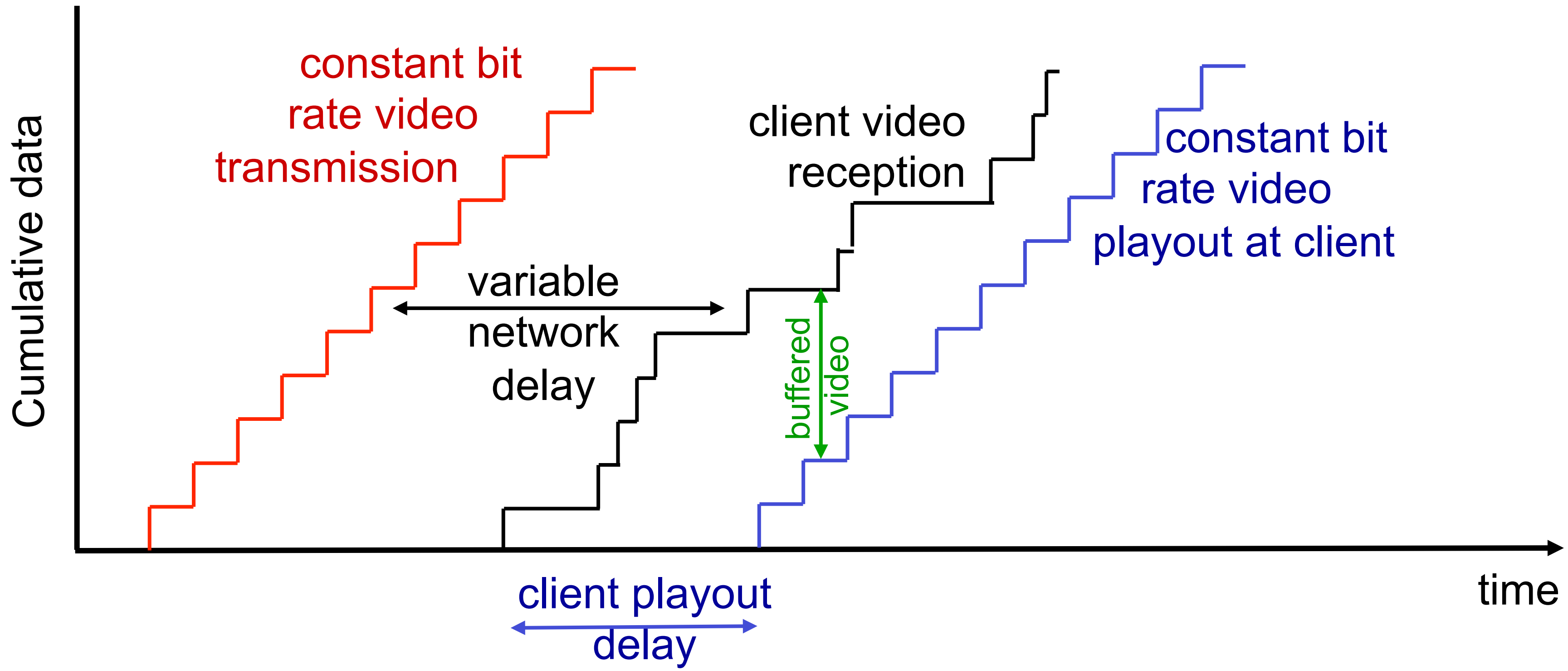
Streaming stored video:



Streaming Stored Video: Challenges

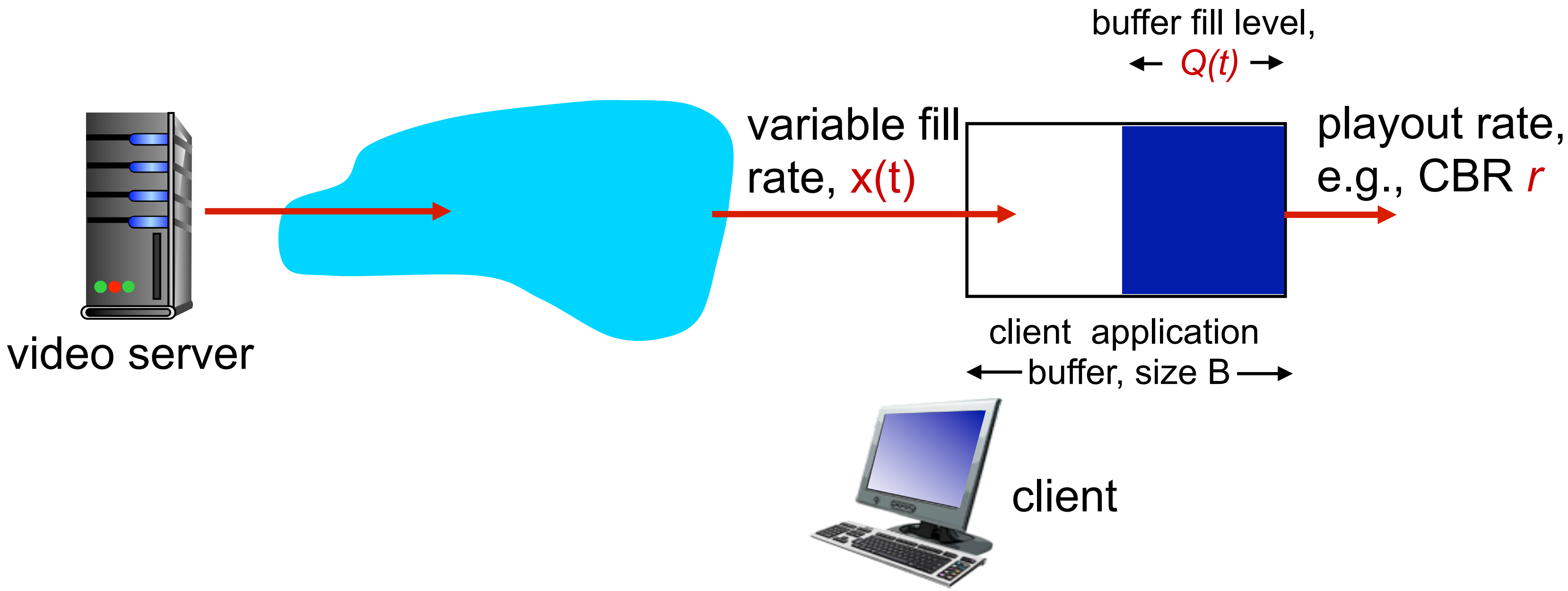
- *Continuous playout constraint*: once client playout begins, playback must match original timing
  - … but *network delays are variable* (jitter), so will need *client-side buffer* to match playout requirements
- Other challenges:
  - client interactivity: pause, fast-forward, rewind, jump through video
  - video packets may be lost, retransmitted

Streaming stored video: revisited

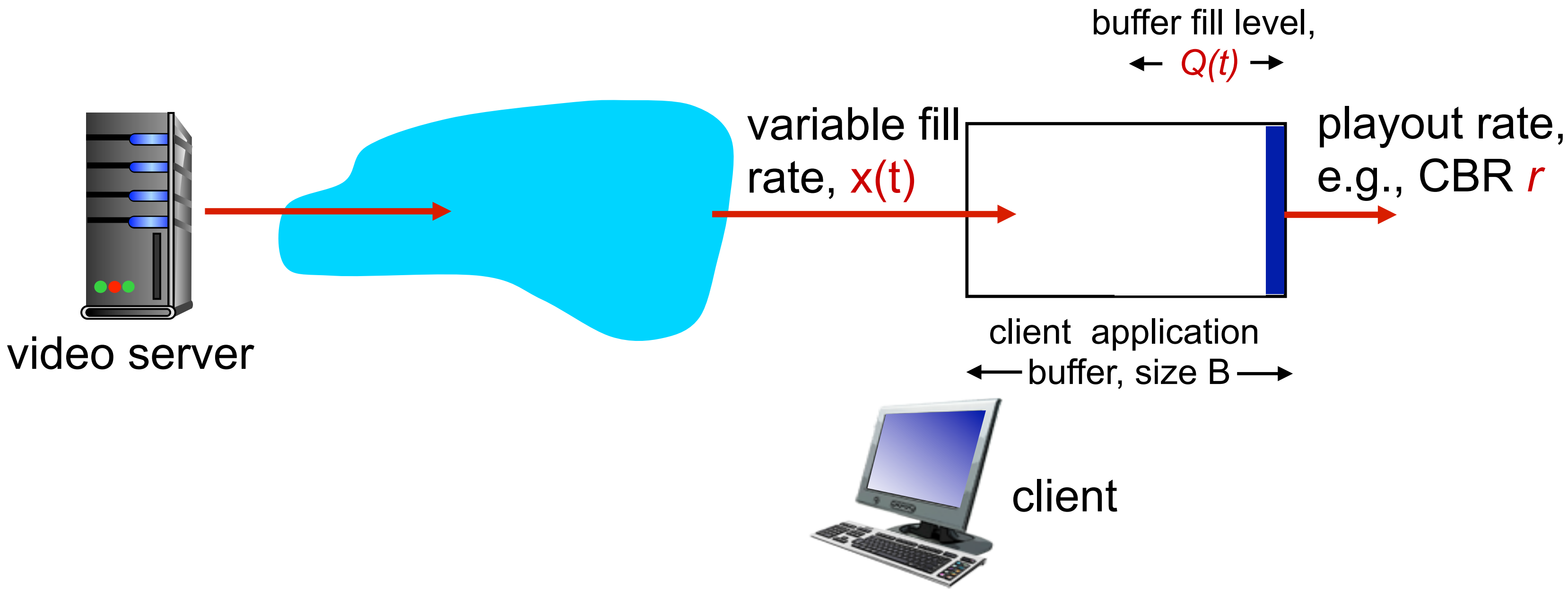


- *client-side buffering and playout delay:* compensate for network-added delay, delay jitter

Client-side buffering, playout

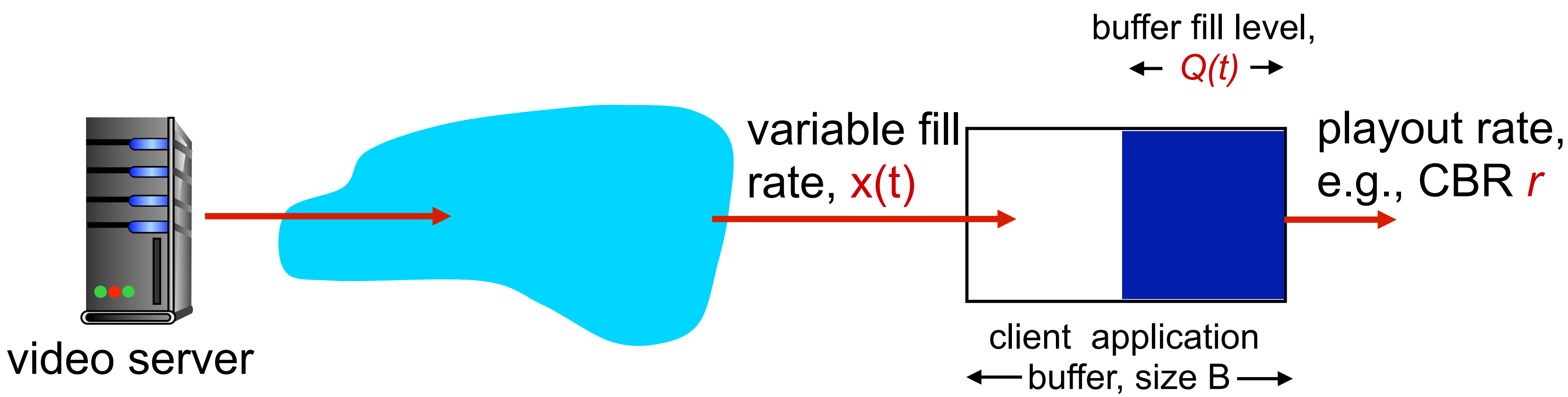


Client-side buffering, playout



1. Initial fill of buffer until playout begins at t_p
2. playout begins at t_p,
3. buffer fill level varies over time as fill rate *x(t)* varies and playout rate *r* is constant

Client-side buffering, playout



Playout buffering: average fill rate ($\bar{x}$), Playout rate (r):

- $\bar{x} < r$: buffer eventually empties (causing freezing of video playout until buffer again fills)
- $\bar{x} > r$: buffer will not empty, provided initial playout delay is large enough to absorb variability in x(t)
  - *Initial playout delay tradeoff:* buffer starvation less likely with larger delay, but larger delay until user begins watching

Streaming multimedia: UDP

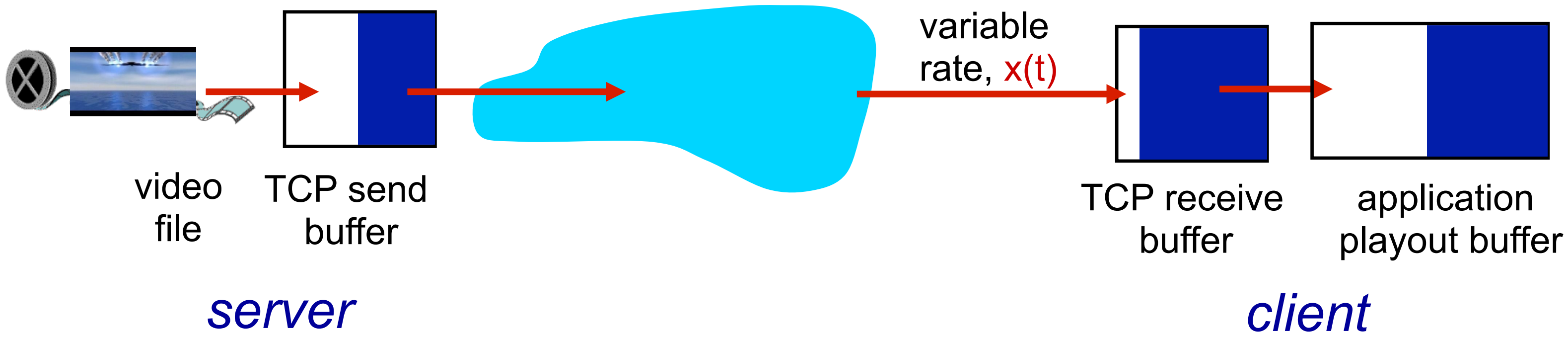
- Server sends at rate appropriate for client
  - often: send rate = encoding rate = constant rate
  - transmission rate can be oblivious to congestion levels
- Short playout delay (2-5 seconds) to remove network jitter
- Error recovery: application-level, time permitting
- RTP [RFC 3550]: Real-Time Transport Protocol (multimedia payload types)
- RTSP [RFC 2326] Real-Time Streaming Protocol – Maintain a control connection

UDP Streaming Drawbacks

- UDP may not go through firewalls
- Constant-rate UDP streaming can fail to provide continuous playout, given the network delay
- It requires a media control server, such as an RTSP server, to process client-to-server interactivity requests (Not scalable)

Streaming multimedia: HTTP

- multimedia file retrieved via HTTP GET
- send at maximum possible rate under TCP



- fill rate fluctuates due to TCP congestion control, retransmissions (in-order delivery)
- larger playout delay: smooth TCP delivery rate
- HTTP/TCP passes more easily through firewalls

Streaming Multimedia: DASH

- *DASH: D*ynamic, *A*daptive *S*treaming over *H*TTP
- *server:*
  - divides video file into multiple chunks
  - each chunk stored, encoded at different rates
  - *manifest file:* provides URLs for different chunks
- *client:*
  - periodically measures server-to-client bandwidth
  - consulting manifest, requests one chunk at a time
    - chooses maximum coding rate sustainable given current bandwidth
    - can choose different coding rates at different points in time (depending on available bandwidth at time)

Streaming Multimedia: DASH

- *DASH: Dynamic, Adaptive Streaming over HTTP*
- *"intelligence"* at client: client determines
  - *when* to request chunk (so that buffer starvation, or overflow does not occur)
  - *what encoding rate* to request (higher quality when more bandwidth available)
  - *where* to request chunk (can request from URL server that is "close" to client or has high available bandwidth)

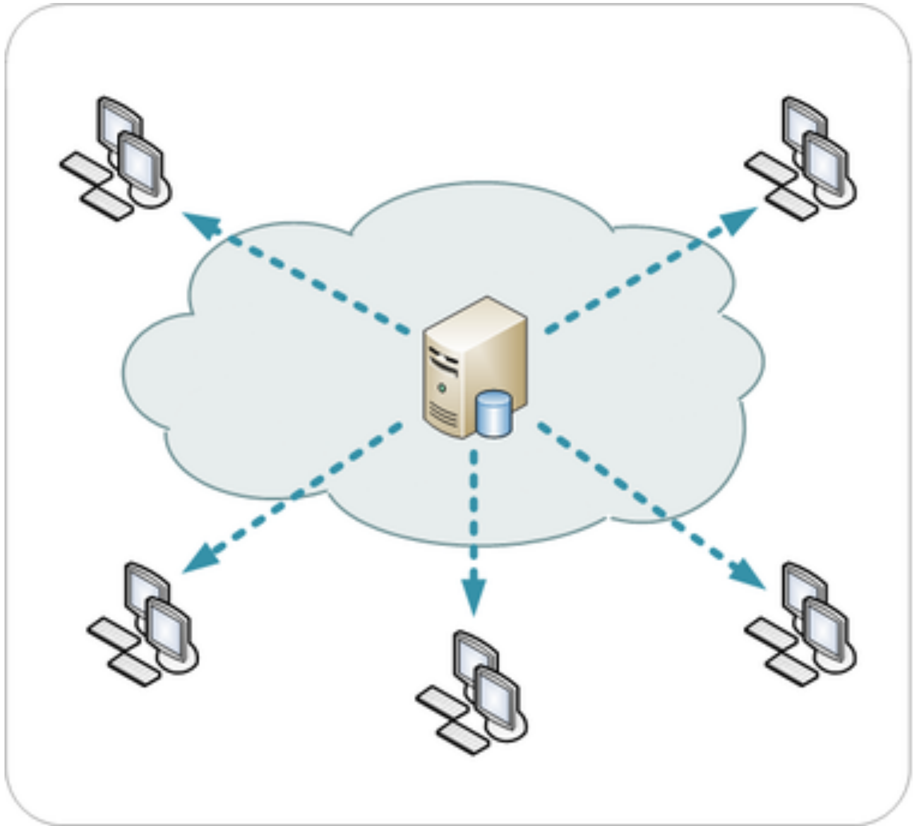
Content Distribution Networks

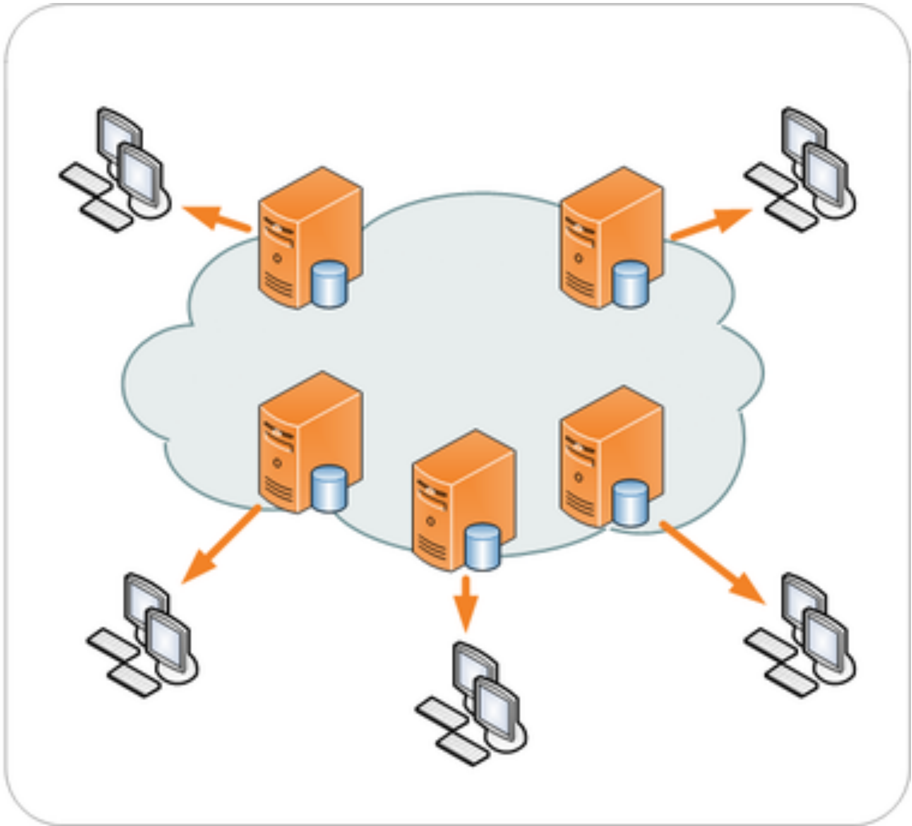
- *challenge:* how to stream content (selected from millions of videos) to hundreds of thousands of simultaneous users?

- *option 1:* single, large "mega-server"
  - single point of failure
  - point of network congestion
  - long path to distant clients
  - multiple copies of video sent over outgoing link

….quite simply: this solution *doesn't scale*

Traditional Scheme vs. CDN





Content Distribution Networks

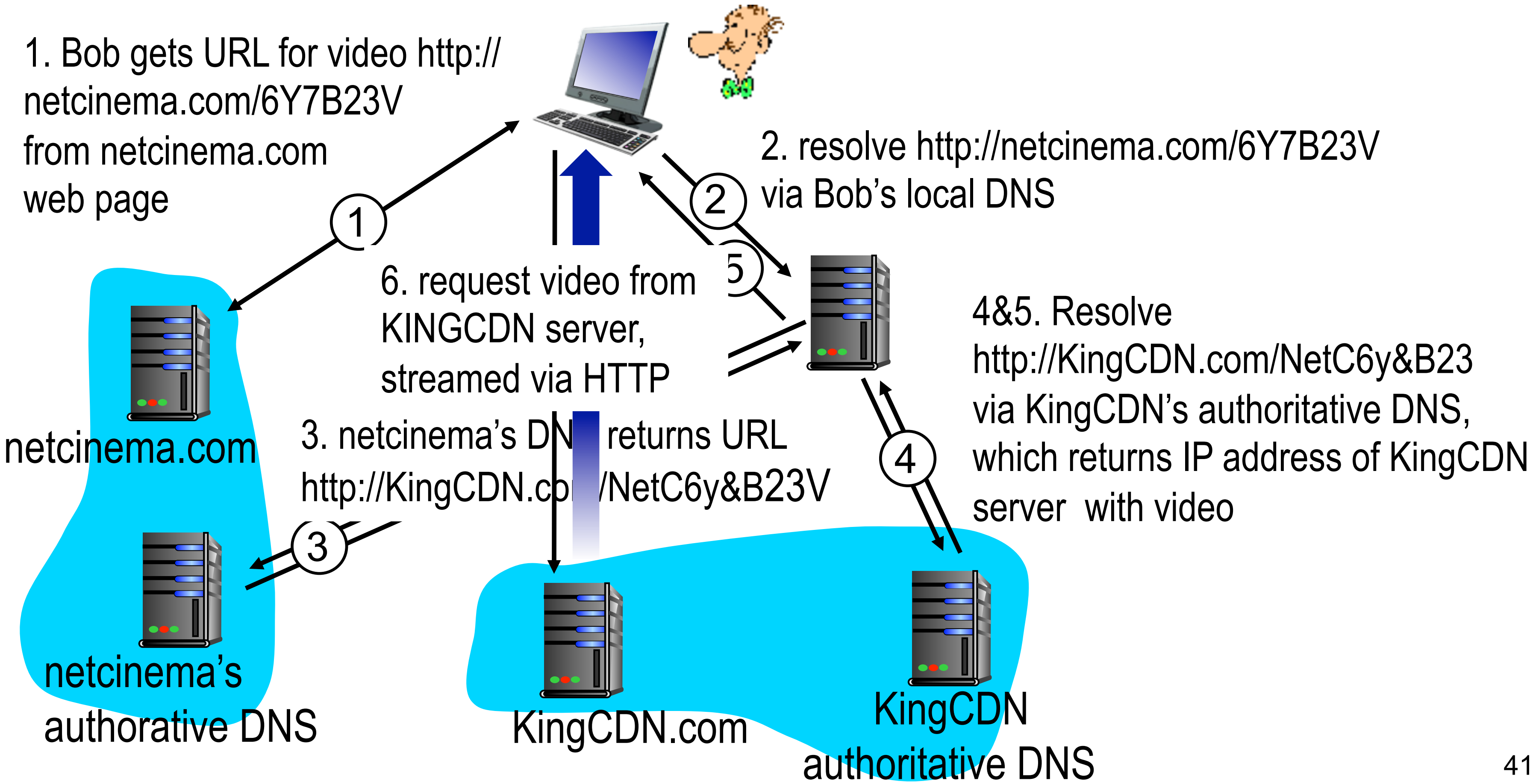
- *challenge:* how to stream content (selected from millions of videos) to hundreds of thousands of simultaneous users?

- *option 2:* store/serve multiple copies of videos at multiple geographically distributed sites *(CDN)*
  - *enter deep:* push CDN servers deep into many access networks
    - close to users
    - used by Akamai, 1700 locations
  - *bring home:* smaller number (10's) of larger clusters in POPs near (but not within) access networks
    - used by Limelight

CDN: "simple" Content Access Scenario

Bob (client) requests video http://netcinema.com/6Y7B23V

- video stored in CDN at http://KingCDN.com/NetC6y&B23V



CDN Cluster Selection Strategy

- *challenge:* how does CDN DNS select "good" CDN node to stream to client
  - pick CDN node geographically closest to client
  - pick CDN node with shortest delay (or min # hops) to client (CDN nodes periodically ping access ISPs, reporting results to CDN DNS)
  - IP anycast

- *alternative:* let *client* decide - give client a list of several CDN servers
  - client pings servers, picks "best"
  - Netflix approach

Case study: Netflix

- 30% downstream US traffic in 2011
- owns very little infrastructure, uses 3rd party services:
  - own registration, payment servers
  - Amazon (3rd party) cloud services:
    - Netflix uploads studio master to Amazon cloud
    - create multiple version of movie (different endodings) in cloud
    - upload versions from cloud to CDNs
    - Cloud hosts Netflix web pages for user browsing
  - *three* 3rd party CDNs host/stream Netflix content: Akamai, Limelight, Level-3

Case study: Netflix