IK2215 Advanced Internetworking

Lecture 7—Multimedia networking
Markus Hidell

Contents

Kurose & Ross, Chapter 9

From ”TCP/IP Protocol Suite”, 3rd edition, by Behrouz A. Fourouzan

A lot of useful information can be found by yourself (Wikipedia is usually a good start)

Make sure you learn to describe the protocols we discuss here yourself

Multimedia Applications

Ongoing widespread deployement:

- Sites with streaming audio and video
  - CNN, SVT,....
- On-demand video clips
  - YouTube
- Telephony and video conferencing
  - Skype

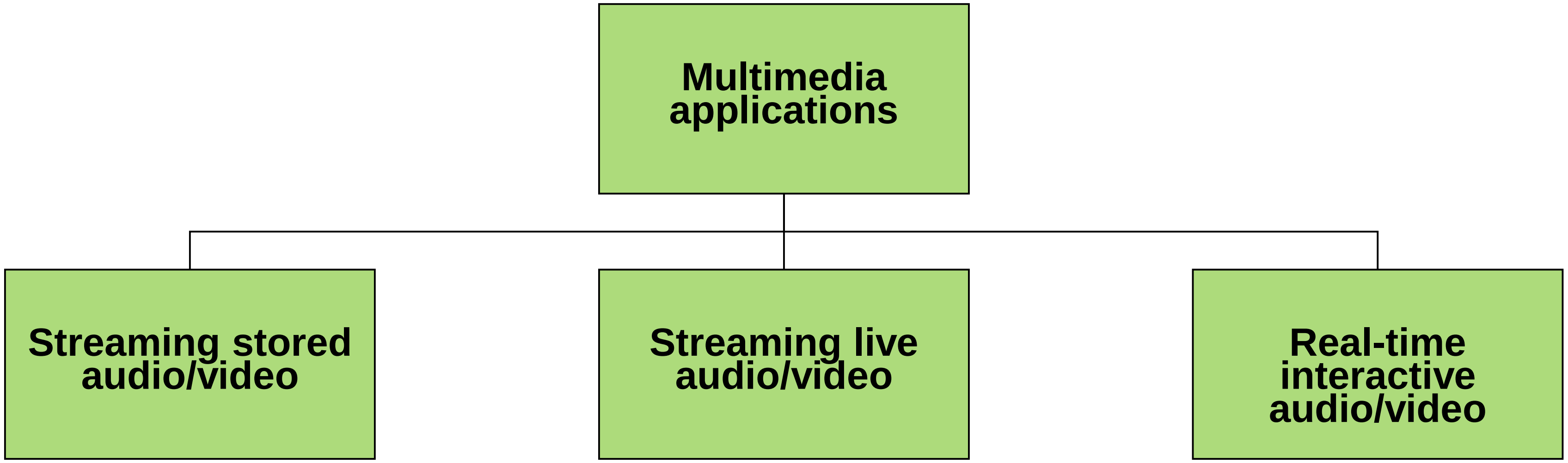
Thanks to

- Broadband resedential access (xDSL)
- High-speed wireless access (WiFi)
- Wireless Fidelity (IEEE 802.11 standards)

Multimedia Service Requirements

- Significantly different compared to traditional applications
  - Such as e-mail, file downloading, web browsing
- Multimedia applications are
  - Sensitivie to end-to-end delay
  - Sensitive to delay variation (delay jitter)
  - Less sensitive to occasional loss of data
- Network could make performance guarantees to applications
  - Qualite of service and Class of service....
  - ...but that is something we will cover later in the course
  - We will look at some other issues today

Classes of Multimedia Applications


Multimedia applications
Streaming stored audio/video
Streaming live audio/video
Real-time interactive audio/video

Download-and-then-play applications not covered today

- Pretty much like file transfers
- Peer-to-peer file sharing
  - P2P has been covered earlier in the course
- Client-server based file sharing
  - Traditional file transfer is prior knowledge for this course

Streaming Stored Audio/Video

Clients request on-demand compressed audio/video files

- From CNN, Microsoft Video, YouTube......

Three key features for this class of applications

- Stored media
  - Content prerecorded and stored on server
  - Pause, rewind, fast-forward, index through content
- Streaming
  - Client begins playout a few seconds after it begins receiving the file
  - Playing out one part while receiving later parts—streaming
  - RealPlayer, QuickTime, Windows Media
- Continuous playout
  - Once playout begins, original timing should be preserved
  - Delays can be tolereated (several seconds may be OK)
    - Interactive operations (pause, rwnd, play) must be quick though....

Streaming Live Audio/Video

Internet radio and IPTV

Content is not stored

- Can’t fast-forward ☺
- Pausing and rewinding can be possible

Many simlutaneous receivers

- Distribution through IP multicast (not so widely deployed)
- Distribution through application-layer multicast (overlay)
  - P2P or CDN (Content Distribution Networks)
- Multiple separate servers→client unicast streams

Continuous playout required

- Constraints less stringent than for real-time interactive applications
- Delays up to tens of seconds may be OK

Real-Time Interactive Audio/Video

For audio/video communication in real-time

- Internet telephony (Skype)
- Video conferencing (NetMeeting, Skype video)

More stringent requirements on delay and delay variations

- Delay should not be more than a few hundred milliseconds
- Voice
  - Delays below 150 ms not perceived
  - Delays 150-400 ms can be acceptable
  - Delays > 400 ms is frustrating

Multimedia in Today's Internet

IP gives a best-effort service

Changes to the Internet service model have been proposed

- Integrated services with bandwidth guarantees (covered later)
- Differentiated services with service classes (covered later)
- Intserv and diffserv have not been widely deployed!

What can we do, then?

- Tricks ot improve user-perceived quality
- Send audio/video over UDP
- Delay playback at the receiver
  - Diminish network-induced delay jitter
- Timestamp packets at the sender
- Prefetch data during playback
- Send redundant information to compensate for packet loss

Servers for Stored Streaming Audio/Video

Ordinary web servers

Special streaming servers

- Tailored for audio/video streaming applications

Both TCP and UDP can be used

- TCP most common
  - Firewalls often block UDP traffic
  - Reliable delivery→entire file transferred OK→allows for caching

User interactivity can be provided (pause/resume etc)

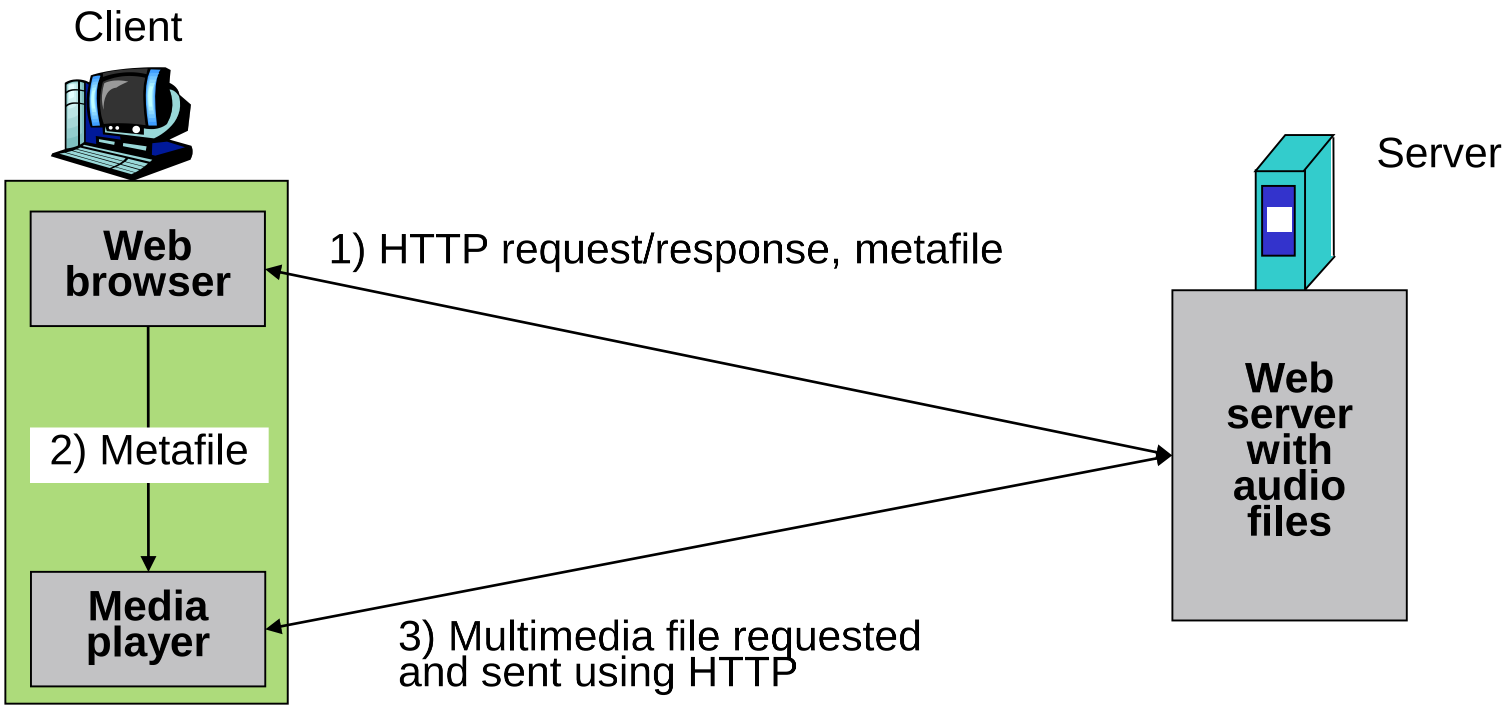
- Real-Time Streaming Protocol (RTSP)

Web client used for *requesting* audio/video streaming

Media Player used for *display and control* of audio/video

- Windows Media Player, Flash Player
- Decompress audio/video, remove packet jitter

Multimedia from Web Server

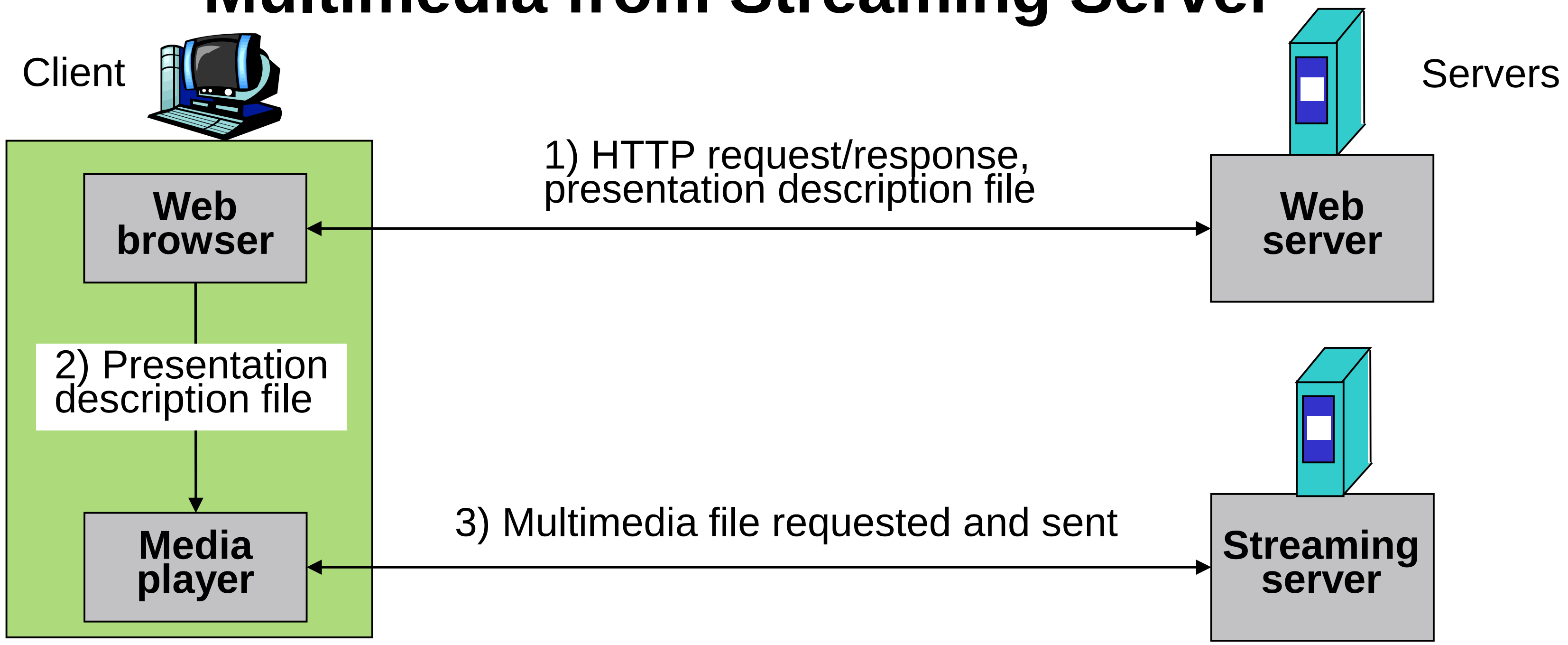

Client
Server
Web browser
1) HTTP request/response, metafile
2) Metafile
Media player
Web server with audio files
3) Multimedia file requested and sent using HTTP

Hyperlink for the audio/video file points to metafile

Metafile indicates content-type→launch specific application

Media player gets metafile so that it can fetch and stream out the audio/video file

Multimedia from Streaming Server


Client
Servers
Web browser
2) Presentation description file
Media player
1) HTTP request/response, presentation description file
3) Multimedia file requested and sent
Web server
Streaming server

Streaming server optimized for streaming

- Servers could be on two different end-systems or on the same

Steps for procedure similar to the previous slide

Media player←→streaming server: richer interaction

- Own protocols, many options for delivering the file

Streaming Server—Delivering Data

Many options for delivering audio/video

Send over UDP at a constant rate equal to the drain rate at the receiver

- Server clocks out data and client plays out as soon as data is decompressed

Same as above, but with delayed playback

- Eliminate network-induced jitter
- Will look more closely into this soon...

Send over TCP

- Place data in media player buffer, delayed playback
- TCP congestion control may lead to *starvation* (empty buffer)

Real-Time Streaming Protocol (RTSP)

IETF Internet RFC 2326

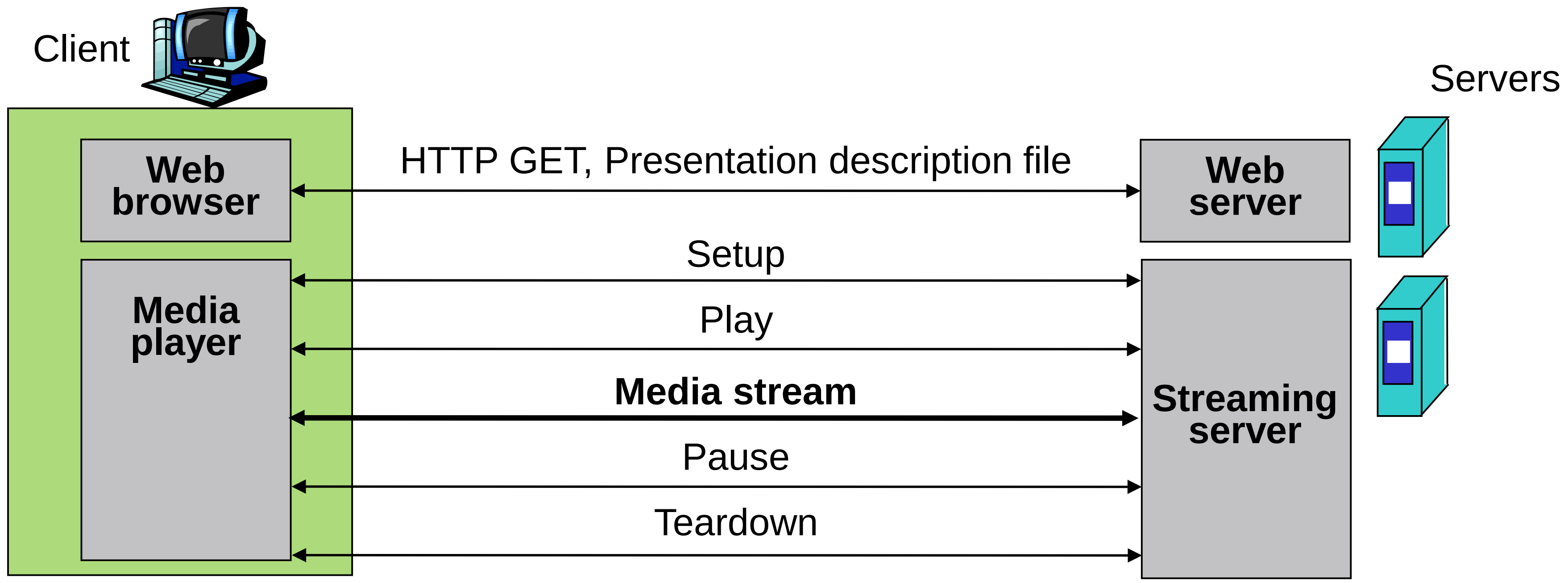
Protocol for exchanging playback control information

- Pausing, repositioning to future/past time, fast-forward, rewinding...

RTSP does not

- Define compression schemes
- Define how to encapsulate audio/video in packets
  - Done by e.g., RTP (later) or some proprietary protocol
- Restrict how streamed media is transported
  - Either TCP or UDP can be used
- Restrict how the media player buffers audio/video
  - Up to the media player

RTSP Client/Server Interaction



RTSP allows media player to *control* stream transmission

Out-of-band protocol (port 554)

- Media stream is sent ”in-band”, not defined by RTSP (other port)
- RTSP can use either UDP or TCP for its messages

Server keeps track of client’s state (session#, sequence#)

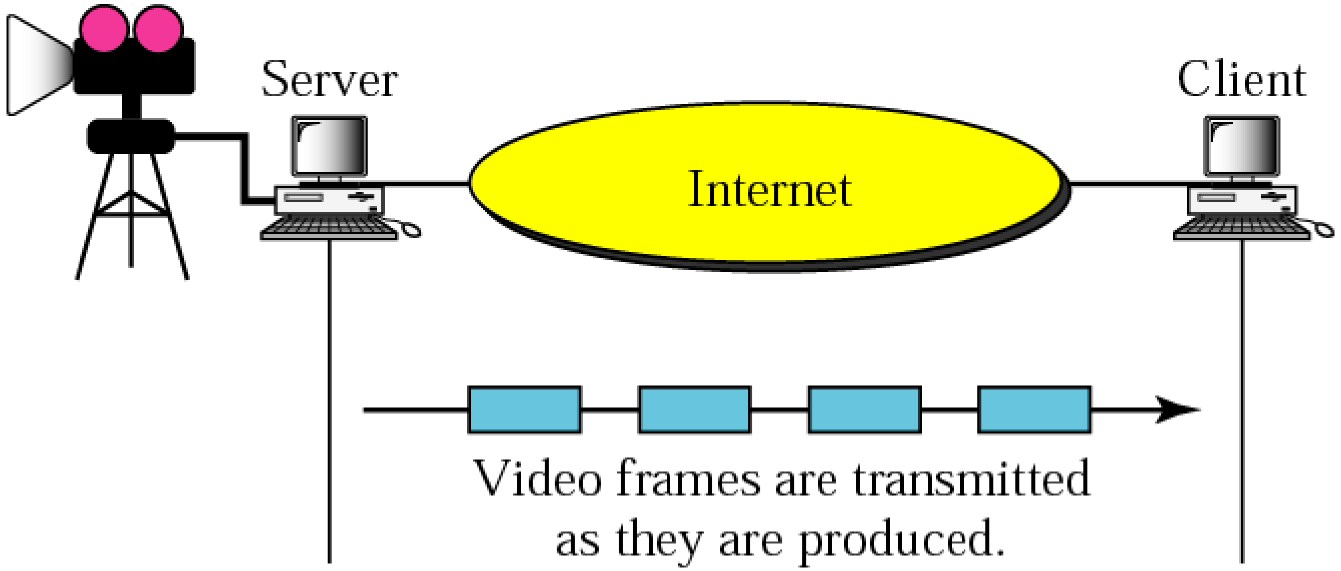
Real-Time Traffic

Real-time traffic needs to be delivered in timely manner

Specifically: interactive multimedia applications

- continuous and delay-sensitive

Example: video conference

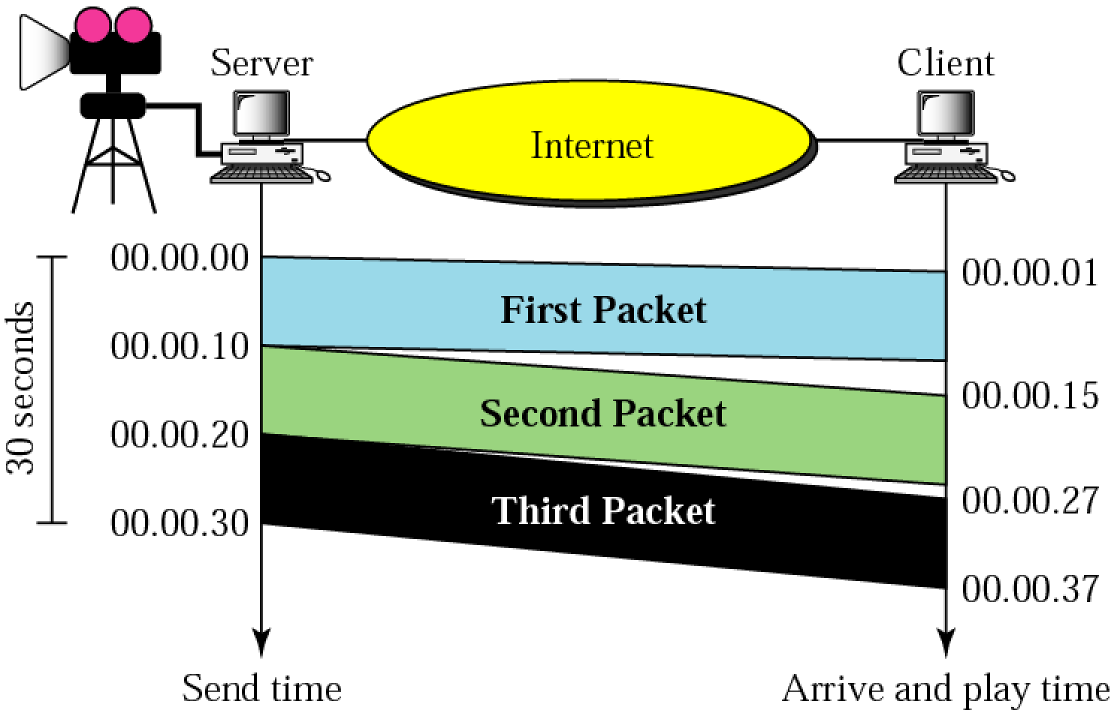


Delay Jitter

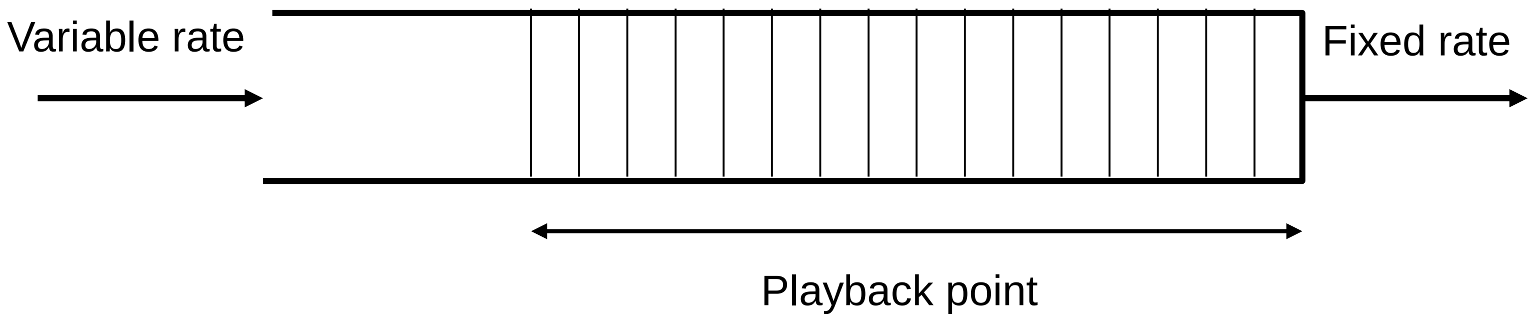
What happens if the packets arrive with different delays?

There is a gap between first and second packet

This phenomenon is called *jitter*



Playback Buffers



The playback buffer *absorbs* jitter in the network

Playback point

- Predetermined threshold
- The maximum jitter allowed
- Packets coming later are dropped
- All packets are delayed by this amount at the receiver
- Tradeoff: packet loss vs low latency

The playback point can be made adaptive

RTP: Real-time Transport Protocol

Designed to carry out variety of real-time data: e.g., audio and video.

Sequence number for receiver to detect out-of-order delivery

Timestamp allowing receiver to control playback

Typically run *on top of* UDP,

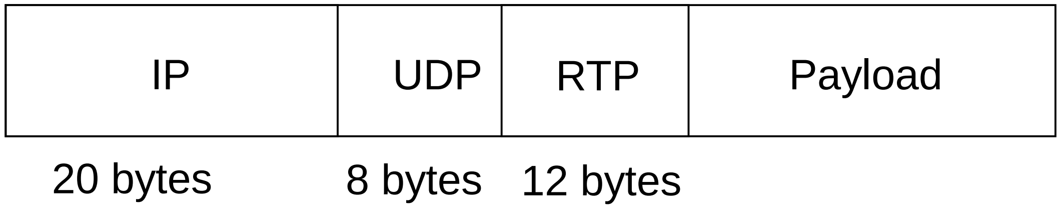
No mechanisms to ensure timely delivery

- Just provides the mechanisms to build a real-time service

RTP Example

Carry the speech frame inside an RTP packet

Typical packetization time of 20ms per audio frame



- Example: PCM audio (8 kHz, 8 bit) 64kbps
- Payload is 64000/50*8 = 160 bytes
    - Overhead is 40/160 = 25%
- Case for header compression
    - Especially for compressed audio and low-speed links

Recovering from Packet Loss

Packet considered lost if it arrives after its scheduled playout time—no use in retransmitting it

FEC—Forward Error Correction

- 1 redundant encoded chunk after every *n* chunks
  - Redundant chunk constructed by XOR-ing the *n* original chunks
  - If any one packet of the (n+1) chunks is missing, receiver can fully reconstruct the lost packet
  - Small *n*: good recovery probability, but large overhead

Interleaving

- Sender resequences audio units [1,5,9,13]...[2,6,10,14]...[3,7,11,12]
- Receiver put units in correct order before playout
- One lost packet→multiple small disturbances instead of one large gap

Content Distribution Networks

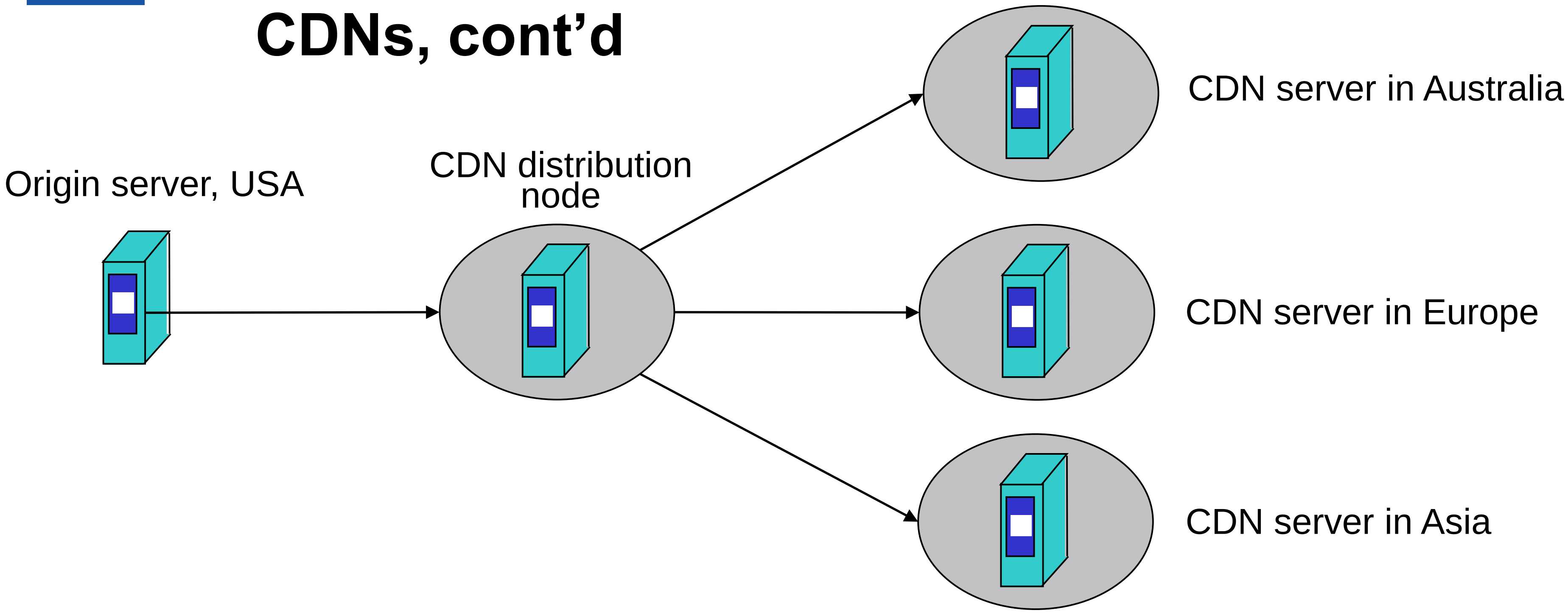
Single server (or server farm) for streaming is problematic

- Large number of geographically distributed users
- Client may be very far from the server→delay problems, packet loss
- Popular content→high bandwidth consumption

Content Distribution Network (CDN)

- Philosophy: bring content closer to the clients
- Content provider pays a CDN company (such as Akamai) to get its multimedia content to requesting users with the shortest possible delay
- CDN company installs 100s of CDN servers throughout the Internet
- CDN company replicates its conten provider’s content in CDN servers
- CDN company provides a mechanism to pick ”best” CDN server for a specific client

CDNs, cont'd



Interaction between content provider and CDN company

Content provider pushes its content to a CDN node

CDN node replicates and pushes content to selected CDN servers

When content provider modifies an object, CDN node immediately replicates the new object to the CDN servers

CDN—How to Find Best Server?

Typically, CDNs use DNS redirection to guide browsers to correct server

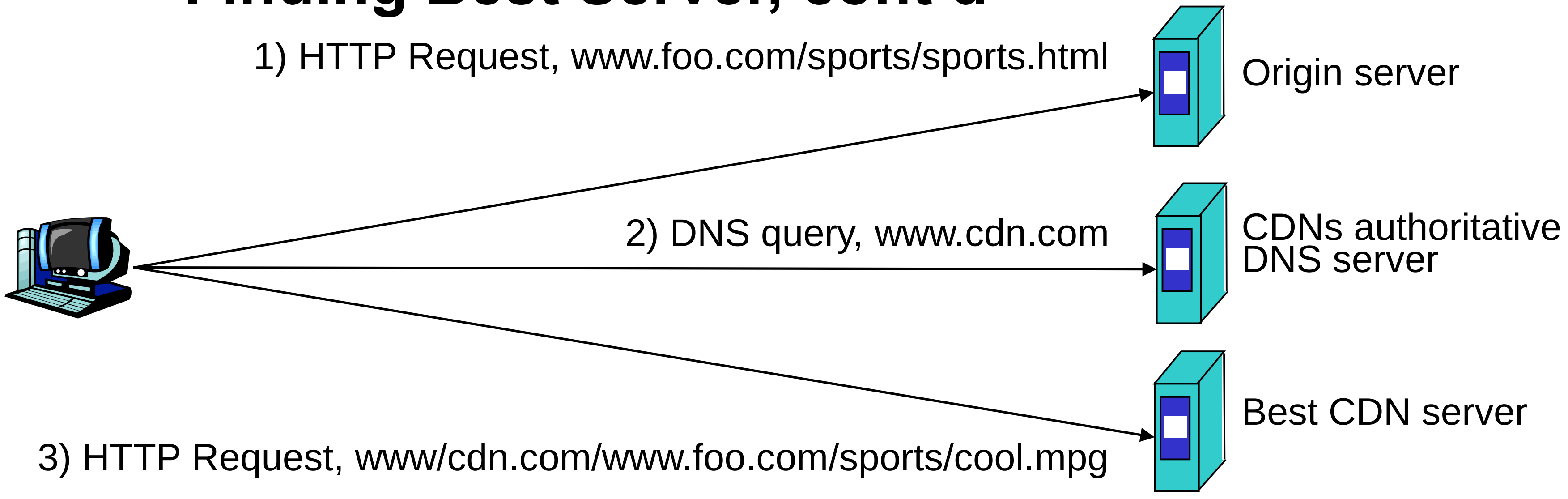
Scenario:

- Content provider is www.foo.com
- CDN company is cdn.com
- Content provider wants CDN to distribute video mpeg files
  - All other objects distributed directly by the content provider

Then,

- URLs of the video files are prefixed with http://www.cdn.com
- Object is then http://www.cdn.com/www.foo.com/sports/cool.mpg
- When browser requests page containing cool.mpg......
  - To be continued.....

Finding Best Server, cont'd


1) HTTP Request, www.foo.com/sports/sports.html
Origin server
2) DNS query, www.cdn.com
CDNs authoritative DNS server
Best CDN server
3) HTTP Request, www/cdn.com/www.foo.com/sports/cool.mpg

1. Browser requests base HTML object from origin server
   1. Receives reference to www.cdn.com/www.foo.com/sports/cool.mpg
2. Browser does DNS lookup on www.cdn.com
   1. DNS configured→ query sent to CDNs authoritative DNS server
   2. Use internal network map to return IP address to best CDN server
3. Browser sends HTTP request to the selected CDN server

Finding Best Server, cont’d

CDN has its own proprietary way to determine the most suitable CDN server for a specific client. Rough idea:

- For every access ISP in the Internet, keep track of the best CDN server for that access ISP
- Determine the best CDN server based on knowledge
  - Internet routing tables (specifically BGP tables)
  - Round-trip time estimates
  - Other measurement data
- CDN uses this info to configure the authoritative DNS server

Thank you!