15-441/15-641 Computer Networking

Lecture 20 – Internet Video Delivery
Peter Steenkiste
Slides by Professor Hui Zhang

Fall 2014
www.cs.cmu.edu/~prs/15-441-F14

Bad Things to Avoid in Streaming Video

1990 – 2004: 1st Generation Commercial PC/Packet Video Technologies

- Simple video playback, no support for rich app
- Not well integrated with Web browser
- No critical mass of compelling content over Internet
- No enough broadband penetration

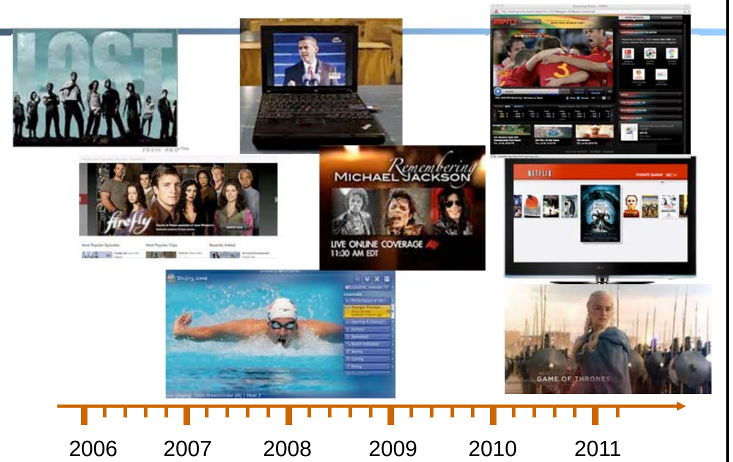
2005: Beginning of Internet Video Era

100M streams first year

BitTorrent

Premium Sports Webcast on Line

2006 – 2013: Internet Video Going Prime Time

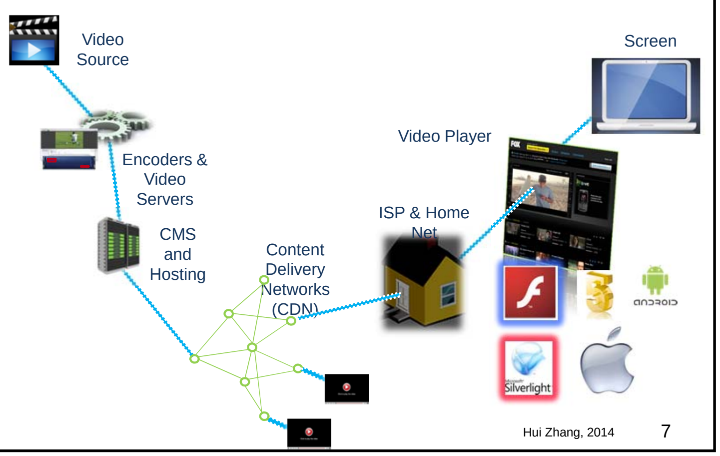


Hui Zhang, 2014 5

Internet Video on Multiple Devices

Hui Zhang, 2014 6

Internet Video Data-plane



Hui Zhang, 2014 7

Internet Video Requirements

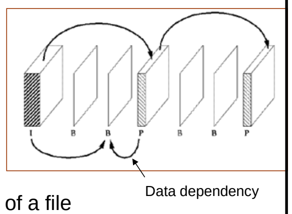
- Smooth/continuous playback
- Elasticity to startup delay: need to think in terms of RTTs
- Elasticity to throughput
  - Multiple encodings: 200Kbps, 1Mbps, 2 Mbps, 6 Mbps, 30Mbps
- Multiple classes of applications with different requirements

	Delay	Bandwidth	Examples
2, N-way conference	< 200 ms	4 kbps audio only, 200 kbps – 5 Mbps video	Skype, Google hangout, Polycom, Cisco
Short form VoD	< 1-5s	300 kbps – 2 Mbps & higher	Youtube
Long form VoD	< 5-30s	500 kbps – 6 Mbps & higher	Netflix, Hulu, Qiyi, HBOGO
Live Broadcast	< 5-10s	500 kbps – 6 Mbps & higher	WatchESPN, MLB
Linear Channel	< 60s	500 kbps – 6 Mbps & higher	DirectTV Live

Hui Zhang, 2014 8

Video Data

- Unlike audio, video compression is essential:
  - Too much data to begin with, but
  - Compression ratios from 50 to 500
- Takes advantage of spatial, temporal, and perceptual redundancy
- Temporal redundancy: Each frame can be used to predict the next -> leads to data dependencies
- To break dependencies, we insert "I frames" or *keyframes* that are independently encoded.
  - Allows us to start playback from middle of a file
- Video data is highly structured



Credit: http://www.icsi.berkeley.edu/PET/GIFS/MPEG_gop.gif

Terminology

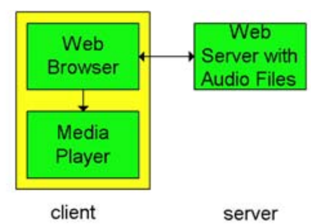
- Bitrate
  - Information stored/transmitted per unit time
  - Usually measured in kbps to mbps
  - Ranges from 200Kbps to 30 Mbps
- Resolution
  - Number of pixels per frame
  - 160x120 to 1920x1080 (1080p) to 4096x2160 (4K)
- FPS (frames per second)
  - 24, 25, 30, or 60

Challenges

- TCP/UDP/IP suite provides best-effort, no guarantees on expectation or variance of packet delay
- Streaming applications delay of 5 to 10 seconds is typical and has been acceptable, but performance deteriorate if links are congested
- Real-Time Interactive requirements on delay and its jitter have been satisfied by over-provisioning (providing plenty of bandwidth), what will happen when the load increases?

First Generation: HTTP Download

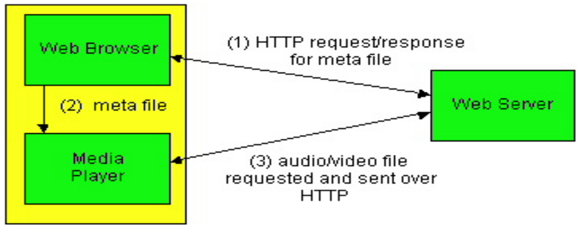
- A simple architecture is to have the Browser request the object(s) and after their reception pass them to the player for display
  - No pipelining



First Gen: HTTP Progressive Download (2)

- Alternative: set up connection between server and player; player takes over
- Web browser requests and receives a **Meta File** (a file describing the object) instead of receiving the file itself;
- Browser launches the appropriate Player and passes it the *Meta File*;
- Player sets up a TCP connection with Web Server and downloads or *streams* the file

Meta file requests

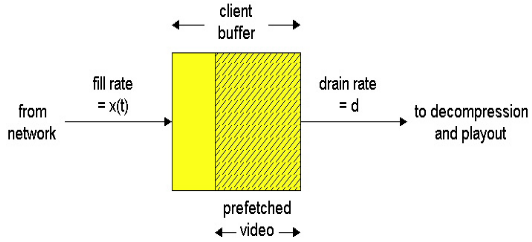


Buffering Continuous Media

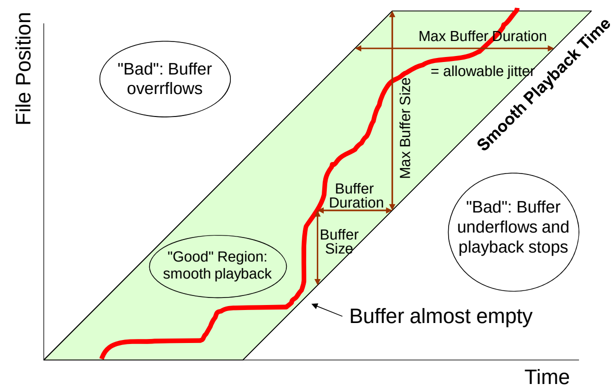
- Jitter = variation from ideal timing
- Media delivery must have very low jitter
  - Video frames every 30ms or so
  - Audio: ultimately samples need <1ns jitter
- But network packets have much more jitter that that!
- Solution: buffers
  - Fill buffer over the network with best effort service
  - Drain buffer via low-latency, local access

HTTP Progressive Download

- With helper application doing the download, playback can start immediately...
- Or after sufficient bytes are buffered
- Sender sends at maximum possible rate under TCP; retransmit when error is encountered; Player uses a much larger buffer to smooth delivery rate of TCP



Streaming, Buffers and Timing

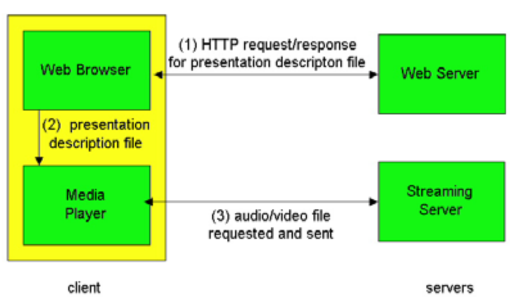


Drawbacks of HTTP Progressive Download

- HTTP connection keeps data flowing as fast as possible to user's local buffer
  - May download lots of extra data if user does not watch the entire video
  - TCP file transfer can use more bandwidth than necessary
- Mismatch between whole file transfer and stop/start/seek playback controls.
  - However: player can use file range requests to seek to video position
- Cannot change video quality (bit rate) to adapt to network congestion

2nd Generation: Real-Time Streaming

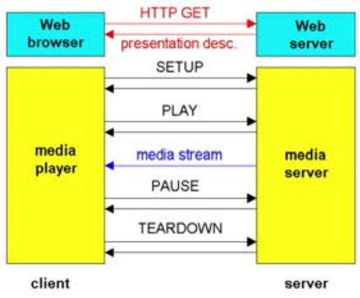
- This gets us around HTTP, allows a choice of UDP vs. TCP and the application layer protocol can be better tailored to Streaming; many enhancements options are possible



Example: Real Time Streaming Protocol (RTSP)

- For user to control display: rewind, fast forward, pause, resume, etc…
- Out-of-band protocol (uses two connections, one for control messages (Port 554) and one for media stream)
- RFC 2326 permits use of either TCP or UDP for the control messages connection, sometimes called the RTSP Channel
- As before, meta file is communicated to web browser which then launches the Player; Player sets up an RTSP connection for control messages in addition to the connection for the streaming media

RTSP Operation



Hui Zhang, 2014 21

RTSP Exchange Example

C: SETUP rtsp://audio.example.com/xena/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/xena/audio.en/lofi RTSP/1.0
Session: 4231
Range: npt=0 (npt = normal play time)

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

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

S: 200 3 OK

Hui Zhang, 2014 22

RTSP Media Stream

- *Stateful* Server keeps track of client's state
- Client issues Play, Pause, ..., Close
- Steady stream of packets
  - UDP - lower latency
  - TCP - may get through more firewalls, reliable

Hui Zhang, 2014 23

Example 2: RTMP - Real-Time Messaging Protocol

- Proprietary Adobe protocol
- Runs over TCP
- Manages audio, video, and other
- Multiplex multiple streams over TCP connection

Hui Zhang, 2014 24

Drawbacks of RTSP, RTMP

- Web downloads are typically cheaper than streaming services offered by CDNs and hosting providers
- Streaming often blocked by routers
- UDP itself often blocked by firewalls
- HTTP delivery can use ordinary proxies and caches
- Conclusion: rather than adapt Internet to streaming, adapt media delivery to the Internet

Hui Zhang, 2014 25

3rd Generation: HTTP Streaming

- Other terms for similar concepts: Adaptive Streaming, Smooth Streaming, HTTP Chunking
- Client-centric architecture with stateful client and stateless server
  - Standard server: Web servers
  - Standard Protocol: HTTP
  - Session state and logic maintained at client
- Video is broken into multiple chunks
- Chunks begin with a keyframe so each chunk is independent of other chunks
- A series of HTTP progressive downloads of chunks
- Playing chunks in sequence gives seamless video

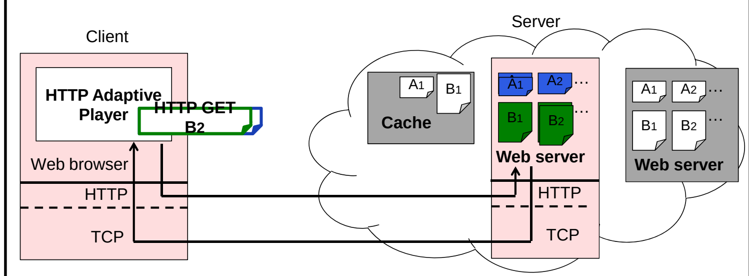
Hui Zhang, 2014 26

Adaptive Bit Rate with HTTP Streaming

- Encode video at different levels of quality/bandwidth
- Client can adapt by requesting different sized chunks
- Chunks of different bit rates must be synchronized
  - All encodings have the same chunk boundaries and all chunks start with key frames, so you can make smooth splices to chunks of higher or lower bit rates

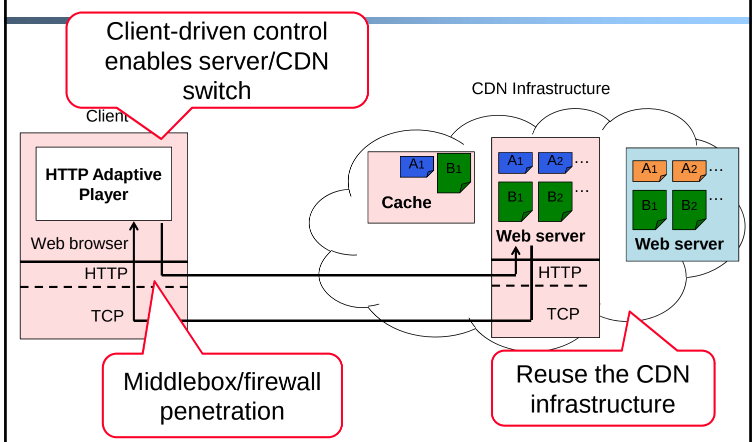
Hui Zhang, 2014 27

HTTP Chunking Protocol



Hui Zhang, 2014 28

Reasons for Wide Adoption



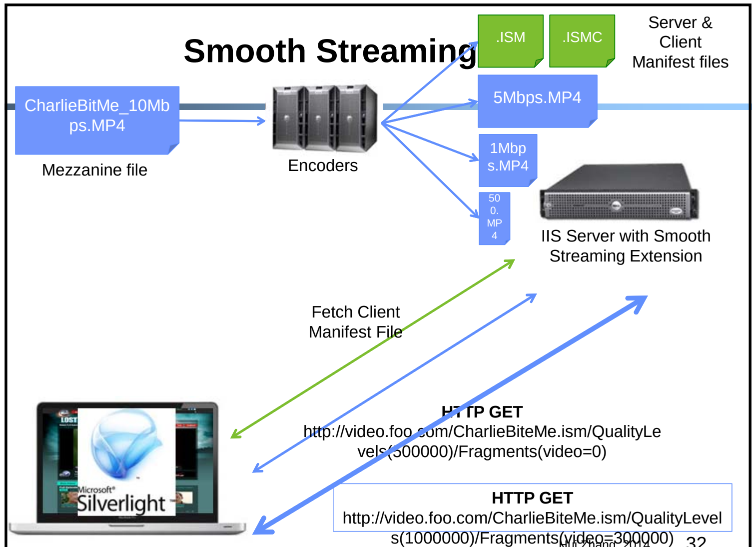
Advantages of HTTP Streaming

- Easy to deploy: it's just HTTP!
  - Work with existing caches/proxies/CDN/Firewall
- Fast startup by downloading lowest quality/smallest chunk
- Bitrate switching is seamless
- Many small files
  - Small with respect to the movie size
  - Large with respect to TCP
    - 5-10 seconds of 1Mbps – 3Mbps → 0.5MB – 4MB per chunk

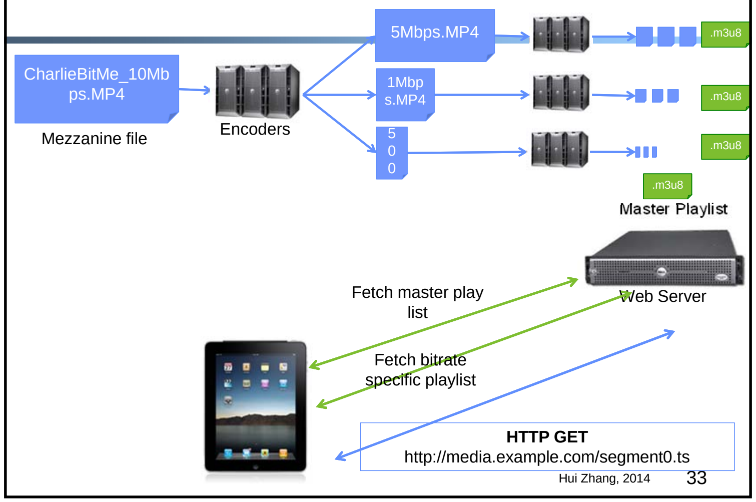
Example of HTTP Streaming Protocols

- Apple HLS: HTTP Live Streaming
- Microsoft IIS Smooth Streaming: part of Silverlight
- Adobe HDS: HTTP Dynamic Streaming
- DASH: Dynamic Adaptive Streaming over HTTP

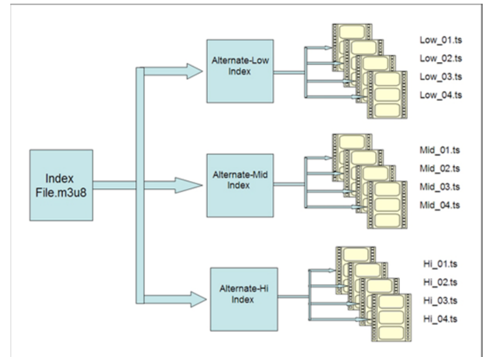
Smooth Streaming



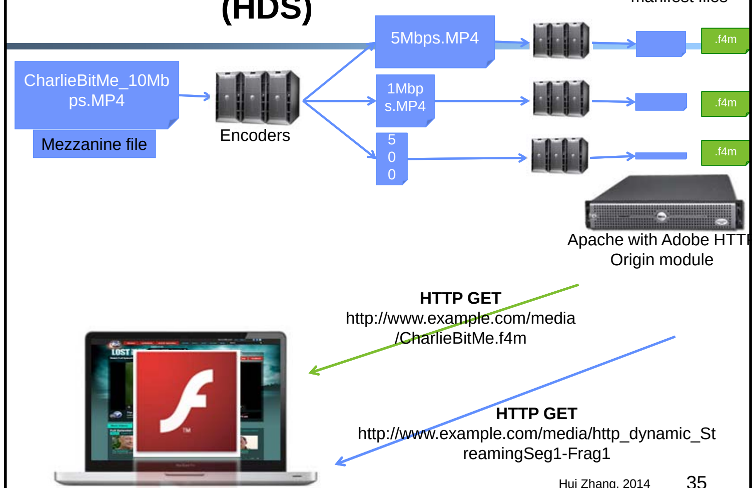
HTTP Live Streaming (HLS)



Example of HLS Meta Data



HTTP Dynamic Streaming (HDS)



Concluding Remarks

- NOT all contents are the same
- Video is fundamentally different from transaction traffic
- We are at the very beginning of Internet video revolution
  - video is more than 60% Internet traffic today,
  - video will be more than 90% Internet traffic in 2-3 years
- What is next?
  - Premium video on big screens → zero tolerance for poor quality: 4K + 3D video
  - Mobile video
- Technical challenges
  - Quality, scalability, mobility, security, usability