15-441 Computer Networking

Multimedia

Outline

- Multimedia requirements
- Audio and Video Data
- Streaming
- Interactive Real-Time
- Recovering from Jitter and Loss

Application Classes

- Typically sensitive to delay, but can sometimes tolerate packet loss (would cause glitches that can be concealed somewhat)
- Data contains audio and video content ("continuous media"), three classes of applications:
  - Streaming
  - Unidirectional Real-Time
  - Interactive Real-Time

Application Classes (more)

- **Streaming**
  - Clients request audio/video files from servers and pipeline reception over the network and display
  - Interactive: user can control operation (similar to VCR: pause, resume, fast forward, rewind, etc.)
  - Delay: from client request until display start can be 1 to 10 seconds

Application Classes (more)

- **Unidirectional Real-Time:**
  - similar to existing TV and radio stations, but delivery on the network
  - Non-interactive, just listen/view
- **Interactive Real-Time** :
  - Phone conversation or video conference
  - More stringent delay requirement than Streaming and Unidirectional because of real-time nature
  - Video: < 150 msec acceptable
  - Audio: < 150 msec good, <400 msec acceptable

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 (transoceanic)
- 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?...

Challenges (more)

- Most router implementations use only First-Come-First-Serve (FCFS) packet processing and transmission scheduling
- To mitigate impact of "best-effort" protocols, we can:
  - Use UDP to avoid TCP and its slow-start phase…
  - Buffer content at client and control playback to remedy jitter
  - Adapt compression level to available bandwidth
  - Over-provision bandwidth, CDN, etc.
- Alternatively, we can change the network:
  - Resource reservations and guarantees and/or
  - Different classes of packets and services
  - Sufficient resources to meet promises

Outline

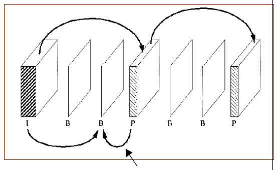
- Multimedia requirements
- Audio and Video Data
- Streaming
- Interactive Real-Time
- Recovering from Jitter and Loss

Audio Data

- Telephone system uses 8-bit samples at 8kHz: 64kbits/s.
- Further compression may be pointless given packet overhead.
- But much higher quality audio is possible, so why not?
- Modern compression achieves equivalent *perceptual* quality with about 1/10 to 1/5 of the bits.
- Most audio compression is performed in "blocks" of hundreds of original samples: adds latency.
- Audio compression is lossy: it encodes something *perceptually* similar but really different from the original.

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

Outline

- Multimedia requirements
- Video and Audio Data
- Streaming
- Interactive Real-Time
- Recovering from Jitter and Loss

Streaming

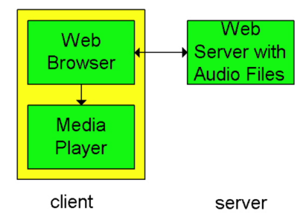
- Important and growing application due to reduction of storage costs, increase in high speed net access from homes, enhancements to caching
- Interactive control by user
  (but often with long response time)
- Ubiquitous on the web:
  - YouTube, Netflix, Vimeo
  - Television networks, Hollywood, etc.
  - Most local radio & TV stations
  - Virtually everywhere on websites

Helper Application

- Displays content, which is typically requested via a Web browser; typical functions:
  - Decompression
  - Jitter removal
  - Error correction: use redundant packets to be used for reconstruction of original stream
  - GUI for user control
- Examples:
  - RealPlayer
  - Adobe Flash Player
  - Windows Media Player
  - QuickTime
  - DivX Web Player

First Generation: HTTP Download

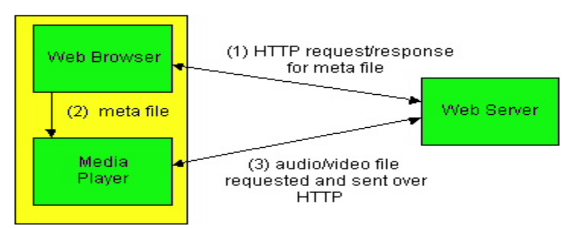
- A simple architecture is to have the Browser request the object(s) and after their reception pass them to the player for display
  - No pipelining



First Gen: HTTP 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

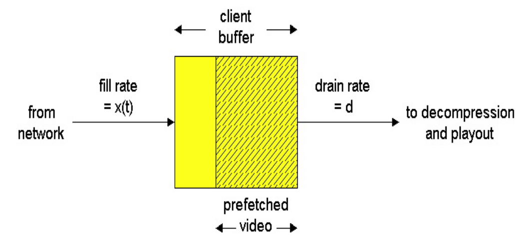


Buffering Continuous Media

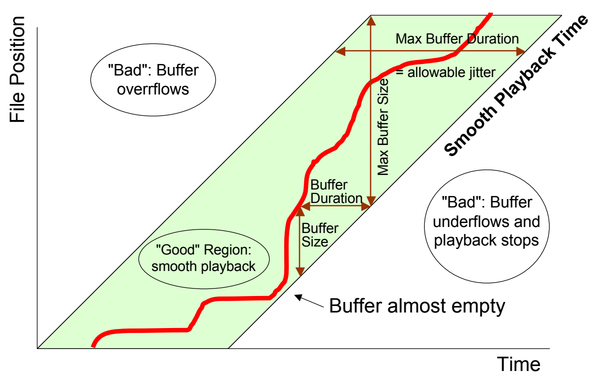
- Jitter = variation from ideal timing
- Media delivery must have very low jitter
  - Video frames every 30ms or so
  - Audio: ultimately samples need <1ns jitter
- But network packets have much more jitter that that!
- Solution: buffers
  - Fill them with best effort
  - Drain them via low-latency,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

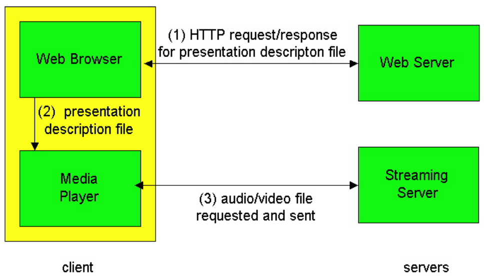


HTTP Progressive Download (2)

- HTTP connection keeps data flowing as fast as possible to user's local buffer
- May download lots of extra data if you do not watch the video
- TCP file transfer can use more bandwidth than necessary
- Mismatch between whole file transfer and stop/start/seek playback controls.
  - However: use file range requests to seek to video position
- Next, we'll see an approach that streams data into a buffer using only the bit rate of the video

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



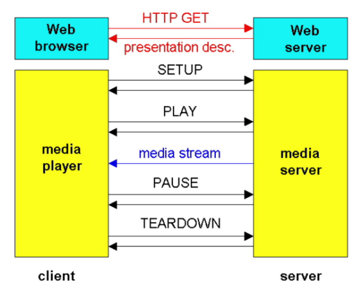
Real Time Streaming Protocol (RTSP)

- 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

Meta File Example

```
<title>Xena: Warrior Princess</title>
<session>
        <group language=en lipsync>
                <switch>
                    <track type=audio
                          e="PCMU/8000/1"
                          src = "rtsp://audio.example.com/xena/audio.en/lofi">
                    <track type=audio
                          e="DVI4/16000/2" pt="90 DVI4/8000/1"
                          src="rtsp://audio.example.com/xena/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/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

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

Credit: some content adapted from Alex Zambelli

RTMP - Real-Time Messaging Protocol

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

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

3rd Generation: HTTP Streaming

- Other terms for similar concepts: Adaptive Streaming, Smooth Streaming, HTTP Chunking
- Probably most important is *return to stateless server and TCP basis* of 1st generation
- Actually a series of small progressive downloads of chunks
- No standard protocol. Typically HTTP to download series of small files.
  - Apple HLS: HTTP Live Streaming
  - Microsoft IIS Smooth Streaming: part of Silverlight
  - Adobe: Flash Dynamic Streaming
  - DASH: Dynamic Adaptive Streaming over HTTP
- Chunks begin with keyframe so independent of other chunks
- Playing chunks in sequence gives seamless video
- Hybrid of streaming and progressive download:
  - Stream-like: sequence of small chunks requested/delivered as needed
  - Progressive download-like: HTTP transfer mechanism, stateless servers

HTTP Streaming (2)

- Adaptation:
  - 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 keyframes, so you can make smooth splices to chunks of higher or lower bit rates
- Evaluation:
  - + Easy to deploy: it's just HTTP, caches/proxies/CDN all work
  - + Fast startup by downloading lowest quality/smallest chunk
  - + Bitrate switching is seamless
  - - Many small files
- Chunks can be
  - Independent files -- many files to manage for one movie
  - Stored in single file container -- client or server must be able to access chunks, e.g. using range requests from client.

Example: Netflix

- Netflix servers allow users to search & select movies
- Netflix manages accounts and login
- Movie represented as an XML encoded "manifest" file with URL for each copy of the movie:
  - Multiple bitrates
  - Multiple CDNs (preference given in manifest)
- Microsoft Silverlight DRM manages access to decryption key for movie data
- CDNs do no encryption or decryption, just deliver content via HTTP.
- Clients use "`Range-bytes=`" in HTTP header to stream the movie in chunks.

Outline

- Multimedia requirements
- Audio and Video Data
- Streaming
- Interactive Real-Time
- Recovering from Jitter and Loss

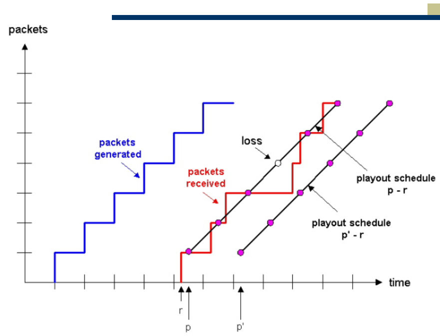
Interactive Real-Time (Phone) Over IP's Best-Effort

- Internet phone applications generate packets during talk spurts
- Bit rate is 8 KBytes, and every 20 msec, the sender forms a packet of 160 Bytes + a header to be discussed below
- The coded voice information is encapsulated into a UDP packet and sent out; some packets may be lost;
  - up to 20% loss is tolerable (but far from desirable)
  - using TCP eliminates loss but at a considerable cost: variance in delay;
  - FEC (forward error correction) is sometimes used to fix errors and make up losses

Interactive Real-Time (Phone) Over IP's Best-Effort (2)

- End-to-end delays above 400 msec cannot be tolerated; packets that are that delayed are ignored at the receiver
- Delay jitter is handled by using
  - timestamps, sequence numbers, and
  - delaying playout at receivers either a fixed or a variable amount
- With fixed playout delay, the delay should be as small as possible without missing too many packets; delay cannot exceed 400 msec

Internet Phone with Fixed Playout Delay



Adaptive Playout Delay

- Objective is to use a value for p-r that tracks the network delay performance as it varies during a phone call
- The playout delay is computed for each talk spurt based on observed average delay and observed deviation from this average delay
- Estimated average delay and deviation of average delay are computed in a manner similar to estimates of RTT and deviation in TCP
- The beginning of a talk spurt is identified from examining the timestamps in successive and/or sequence numbers of chunks

Real-Time Protocol (RTP)

- Provides standard packet format for real-time application
- Typically runs over UDP
- Specifies header fields below
- **Payload Type**: 7 bits, providing 128 possible different types of encoding; eg PCM, MPEG2 video, etc.
- **Sequence Number**: 16 bits; used to detect packet loss

RTP Header

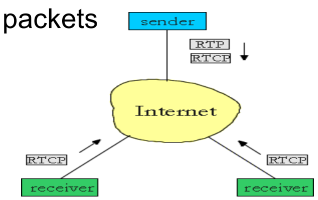
Real-Time Protocol (RTP)

- **Timestamp**: 32 bytes; gives the sampling instant of the first audio/video byte in the packet; used to remove jitter introduced by the network
- **Synchronization Source identifier (SSRC)**: 32 bits; an id for the source of a stream; assigned randomly by the source

RTP Header

RTP Control Protocol (RTCP)

- Protocol specifies report packets exchanged between sources and destinations of multimedia information
- Three reports are defined: Receiver reception, Sender, and Source description
- Reports contain statistics such as the number of packets sent, number of packets lost, inter-arrival jitter
- Used to modify sender transmission rates and for diagnostics purposes



RTCP Bandwidth Scaling

- If each receiver sends RTCP packets to all other receivers, the traffic load resulting can be large
- RTCP adjusts the interval between reports based on the number of participating receivers
- Typically, limit the RTCP bandwidth to 5% of the session bandwidth, divided between the sender reports (25%) and the receivers reports (75%)

Outline

- Multimedia requirements
- Audio and Video Data
- Streaming
- Interactive Real-Time
- Recovering from Jitter and Loss

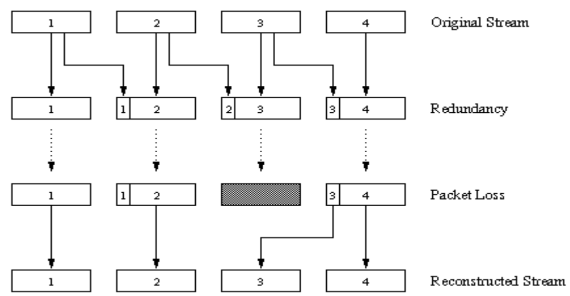
Recovery From Packet Loss

- Loss is in a broader sense: packet never arrives or arrives later than its scheduled playout time
- Since retransmission is inappropriate for Real Time applications, FEC or Interleaving are used to reduce loss impact.
  - Note: ping from CMU to west coast is 80ms
  - Retransmission seems feasible, so why "inappropriate"?
  - Retransmission may not be useful when there's no contention, but if there's contention, latency might be much higher
- FEC is Forward Error Correction
- Simplest FEC scheme adds a redundant chunk made up of
  - duplicate of previous chunk, redundancy is 1, or
  - exclusive OR of previous n chunks every n; redundancy is 1/n, or
  - there are other schemes that tolerate greater loss

Recovery From Packet Loss (2)

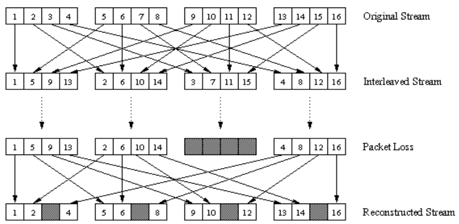
- Another approach:
  - mixed quality streams are used to include redundant duplicates of chunks;
  - upon loss, play out available redundant chunk, albeit a lower quality one
- With one redundant low quality chunk per chunk, scheme can recover from single packet losses

Piggybacking Lower Quality Stream



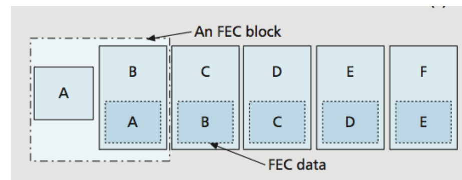
Interleaving

- Has no redundancy, but can trade off latency for smaller perceptual impact of a packet loss
- Divide 20 msec of audio data into smaller units of 5 msec each and interleave
- Upon loss, have a set of partially filled chunks



Example VOIP: Skype

- Peer-to-peer
  - Decentralized user directory
  - Supernodes (developed by the founders of KaZaA)
- Voice is via UDP between peers when possible
- Supernodes are used when necessary to get through firewalls
- Forward Error Correction: At around 4% packet loss, packets double in size and carry a copy of the previous block



Huang, Huang, Chen, & Wang. "Could Skype be more satisfying? a QoE-centric study of the FEC mechanism in an internet-scale VoIP system." *IEEE Network* **24**(2), 2010.

Summary

- Different classes of applications
  - Streaming
    - HTTP access to sequence of chunks - stateless servers
    - Adapt by selecting chunks with appropriate bit rate
  - Unidirectional Real-Time
  - Interactive Real-Time
    - Usually UDP to reduce latency
    - Forward Error Correction (FEC) rather than retransmission
- Buffering to reduce jitter
- Next: Can networks do better? Quality of Service.