# Chapter 9 – Internet and Higher Layer Protocols

* Multimedia Applications
  + Streaming/ Stored - Audio/ Video
    - Streaming – It can begin playout before downloading entire file
    - Stored (server) – It can transmit faster than audio/video rendering (it requires storing/ buffering over client system)
  + Conversational over IP – Voice/ Video
  + Streaming Live - Audio/ Video
* **Streaming Stored Video**
  + Challenges faced
    - Continuous Playout constraint – Once Client playout begins, playback must match original timing. But N/w delay are variable… so it will need client-side buffer to match the playout requirements.
    - Other challenges are Play, pause, fast forward, rewind which may lead to loss or retransmission of the packets
  + Client-side buffering and playout delay compensates for the network added delay and jitter.
    - Buffer-fill level varies over time as fill rate x(t) varies while the playout rate r is constant
    - If Average fill rate x < r; buffer eventually empties, video pauses for buffering.
    - If Average fill rate x > r; buffer will not be empty, provided initial playout delay is large to absorb variability in x(t)
  + **UDP**
    - Server encapsulates, maintains parallel connection ie. play, pause, resume
    - Transmission rates are oblivious of congestion levels
    - Short Playout delay (2-5 seconds), error recovery is at application level
  + **HTTP**
    - Video is stored in HTTP server, requires TCP Connection
    - After predefined threshold client begins playback
    - Fill rates may fluctuate, it is good to have larger playout delay. It can easily pass through firewalls.
* **Voice-Over-IP (VoIP)**
  + Types
    - VoIP end-end delay requirement:
      * < 150 msecs: good
      * > 400 msecs: bad – can be noticeable
    - Session Initialization:
    - Value-added Services
    - Emergency Services
  + Characteristics
    - 64 kbps during talk spurt
    - 20 msecs chunks are transferred at 8 Kbytes/sec
  + VoIP: packet loss, delay
    - Network loss – IP datagram lost due to N/w congestion
    - Delay loss – IP datagram arrives too late for playout at R/c  
      Max tolerable delay: 400 ms
    - Loss tolerance: 1- 10% of packet loss rate can be tolerated
    - Delay Jitter – e-e delay can be more or less than 20 msecs even after constant bit rate transmission.
  + VoIP: fixed playout delay
    - Receiver attempts to playout chunk.
    - Chunks when transmitted have timestamp t and can only be played after t +q, so if data arrives too late then the data can be lost.  
      Hence, if we have large q, then packet loss is reduced  
      and if we have small q, we can have better interaction
  + Adaptive Playout Delay
    - It allows the user to have a low playout delay as well as it reduces the loss of packet due to late transmission.
    - It estimates the N/w delay and adjusts accordingly
    - Silent periods are elongated still the chunks are played out at 20 msecs
    - Adaptive EPD - **di = (1-a) \* di-1 + a (ri – ti)**
    - Average Deviation of delay - **vi = (1-b) \* vi-1 + b |ri – ti – di|**
    - Playout time = **ti + di + Kvi**
* **Real-Time Protocol (RTP)**
  + It specifies packet structure for packets carrying A/V data. These packets are encapsulated in UDP. It runs in the end systems only.
  + Timestamp field (32 bit long), SSRC field (32 bit long) it identifies sampling instant of 1st byte and Source of RTP stream respectively.
* **Real-Time Control Protocol (RTCP)**
  + It works in conjunction of RTP.
  + Each participant in RTP session passes RTCP control packet to other participant.
  + Every packet contains sender /receivers report as well as provides feedback to control performance.
  + RTP, RTCP packets distinguishes from each other over a single multicast address using **PORT NO #**
  + Types of RTCP Packets
    - Receiver report packets
    - Source description packets
    - Sender report packets
  + RTCP can synchronize different media streams with in a RTP session.
  + RTCP attempts to limit the traffic to 5% of session B/W
* **Session Initiation Protocol (SIP)**
  + **SIP** to setup call to a known IP Address is a 4-step process

1. Caller sends SIP invite with its # Port, IP Addr., and the encoding tech
2. Callee sends an ACK 200 with # Port, IP Addr, preffered encoding tech
3. Messages can now be sent over TCP/UDP over default SIP Port#- 5060
   * **SIP server functions**
     + **Register –**

When callee starts SIP Client, Client sends SIP Register message to callee’s registrar services

* + - **Proxy –**

It is responsible for routing SIP messages. When a caller sends an SIP Invite to the callee using it’s proxy, then it can receive response thru’ one or multiple proxies

* + SIP proxy is analogous to DNS Server + TCP setup
* **H.323 Signaling Protocol**