南京大学
NANJING UNIVERSITY

Computer Networks

Wenzhong Li, Chen Tian

Nanjing University

Material with thanks to James F. Kurose, Mosharaf Chowdhury, and other colleagues.

Outline

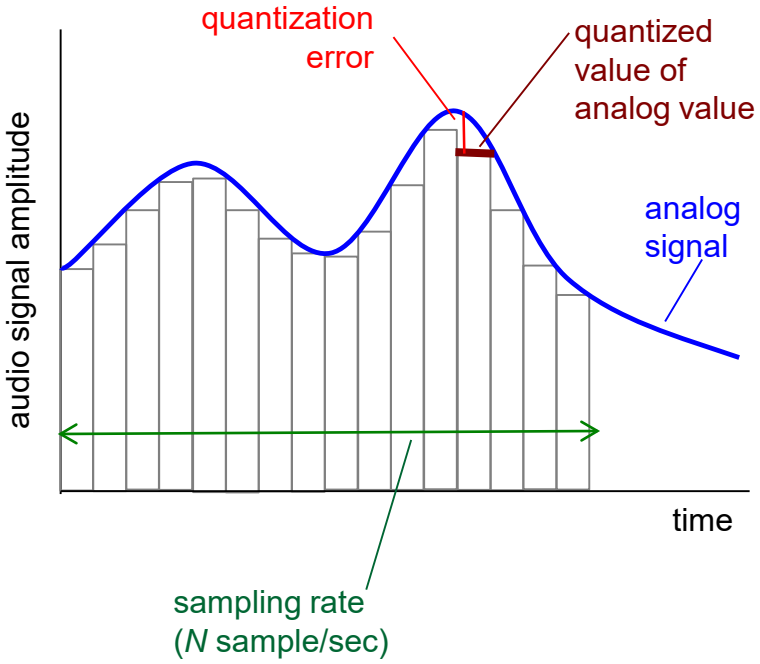
- Multimedia networking applications
- Real-time Transport Protocols
- Internet Quality of Service (QoS)

Multimedia networking applications

Multimedia Networking

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
- Example rates
  - CD: 1.411 Mbps
  - MP3: 96, 128, 160 kbps
  - Internet telephony: 5.3 kbps and up



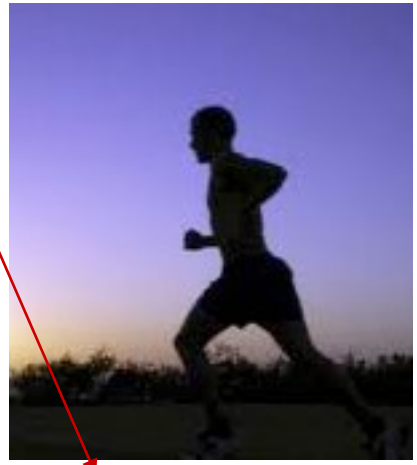
Multimedia: video

- 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)
- 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

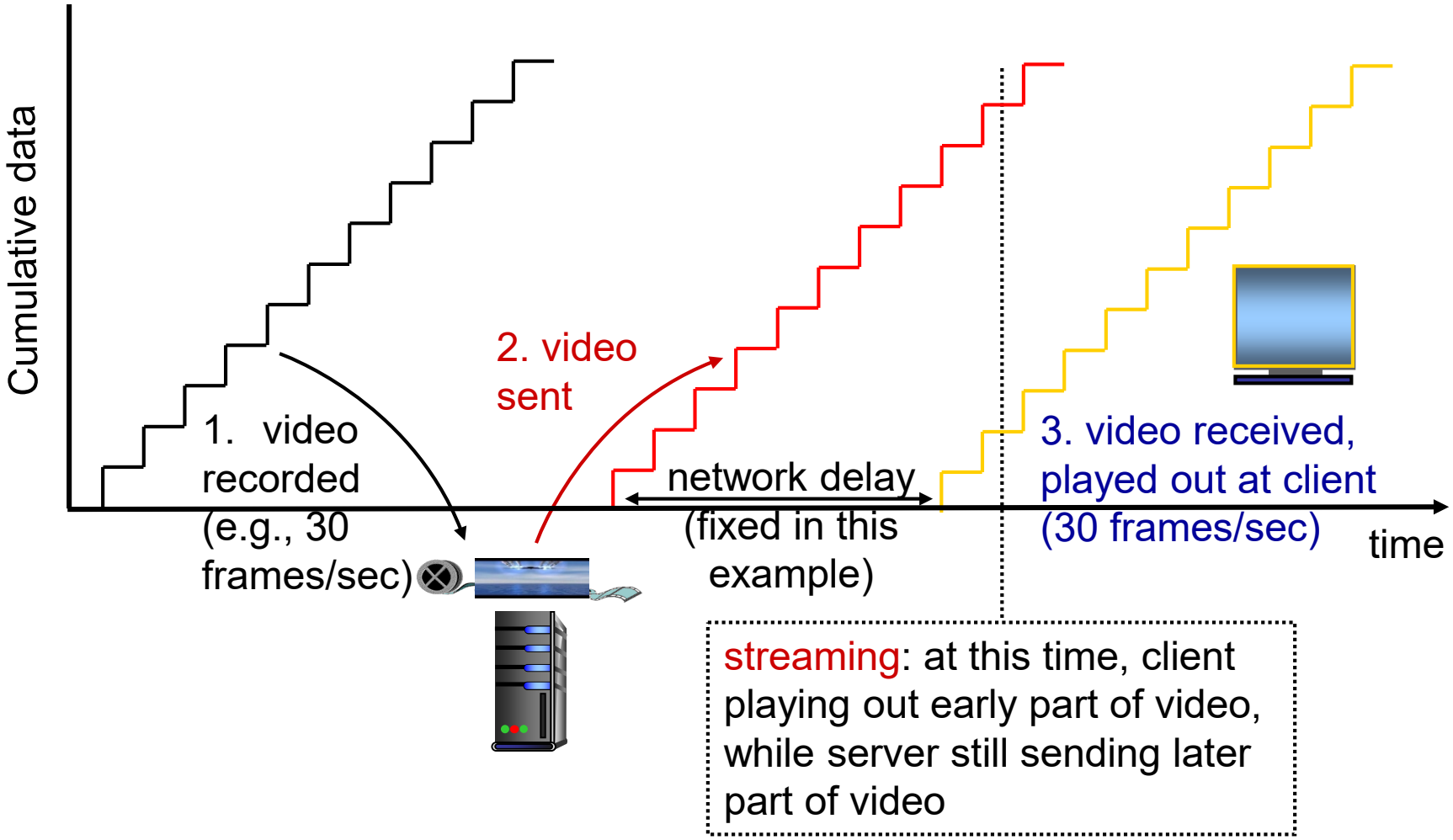


frame *i+1*

Three 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 (futbol)

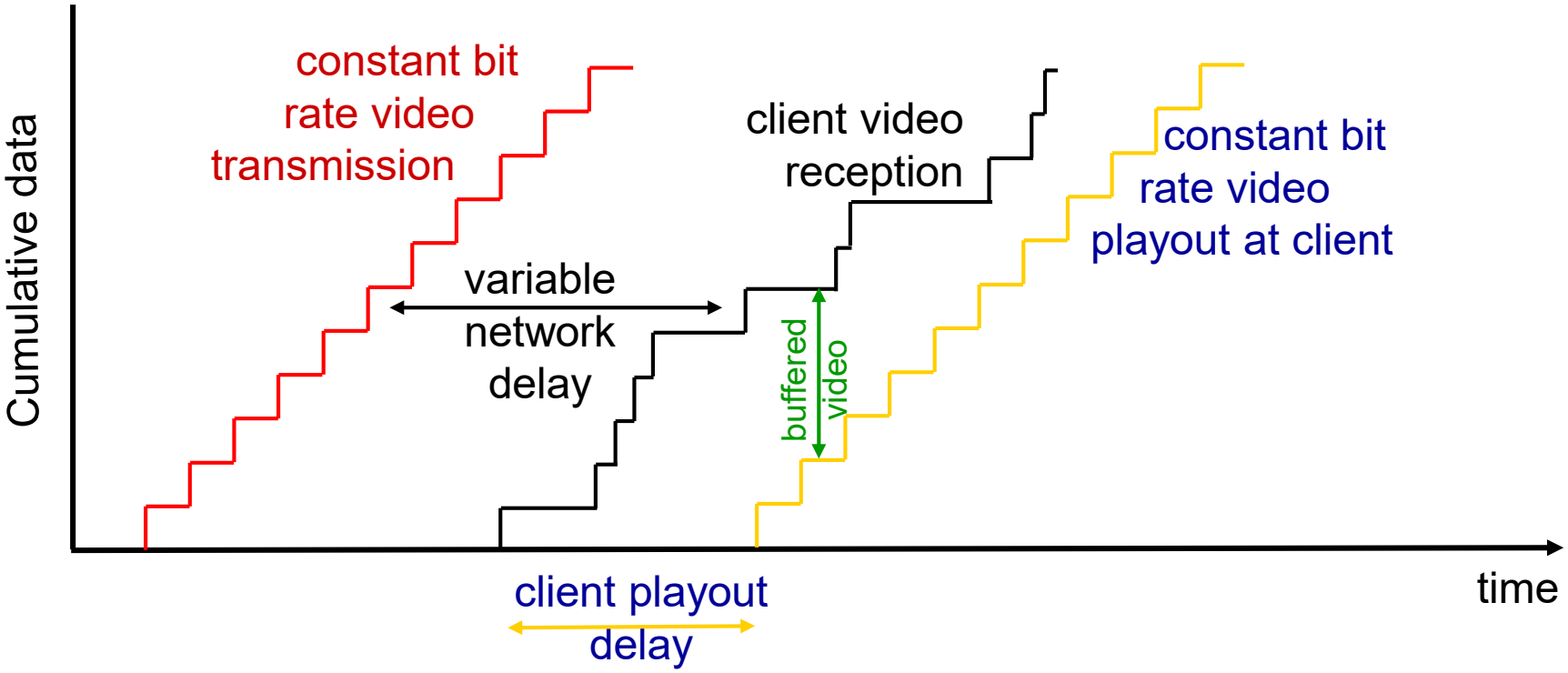
Streaming stored video:



Streaming stored video: challenges

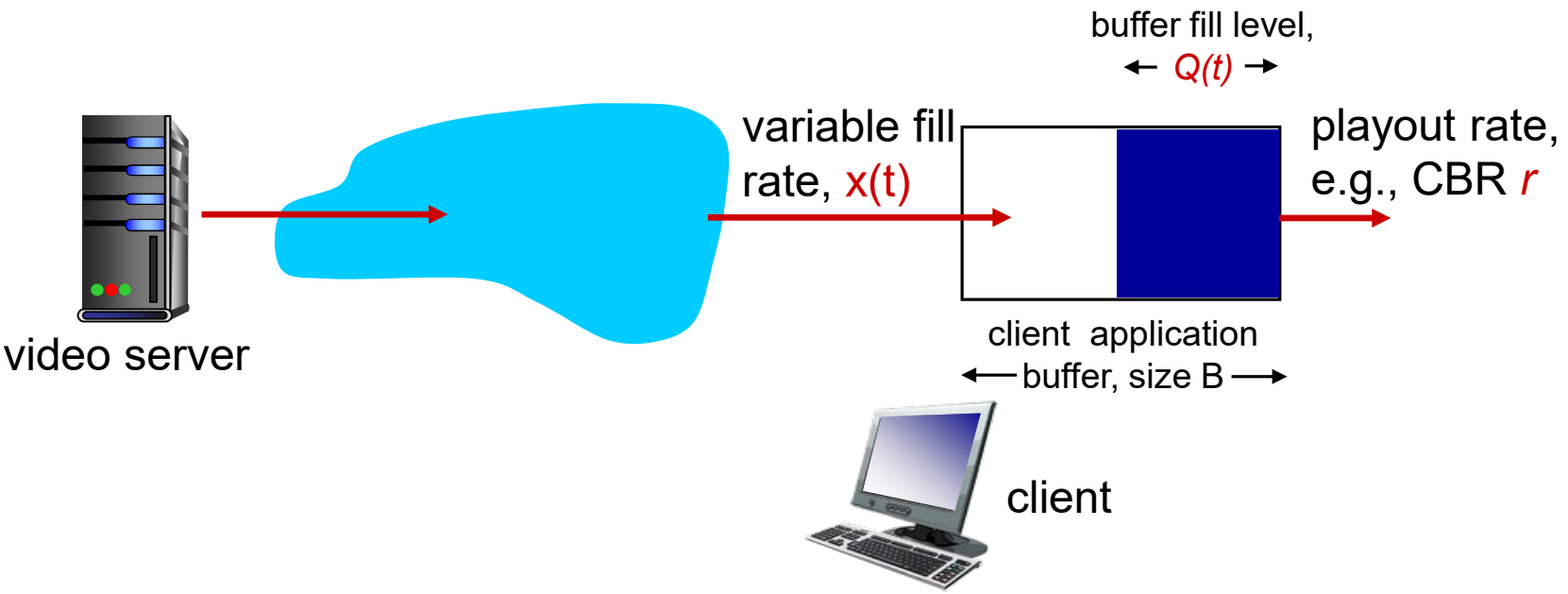
- *Continuous playout constraint*: once client playout begins, playback must match original timing
  - … but *network delays are variable* (jitter), so will need *client-side buffer* to match playout requirements
- Other challenges:
  - client interactivity: pause, fast-forward, rewind, jump through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

Streaming stored video: revisted

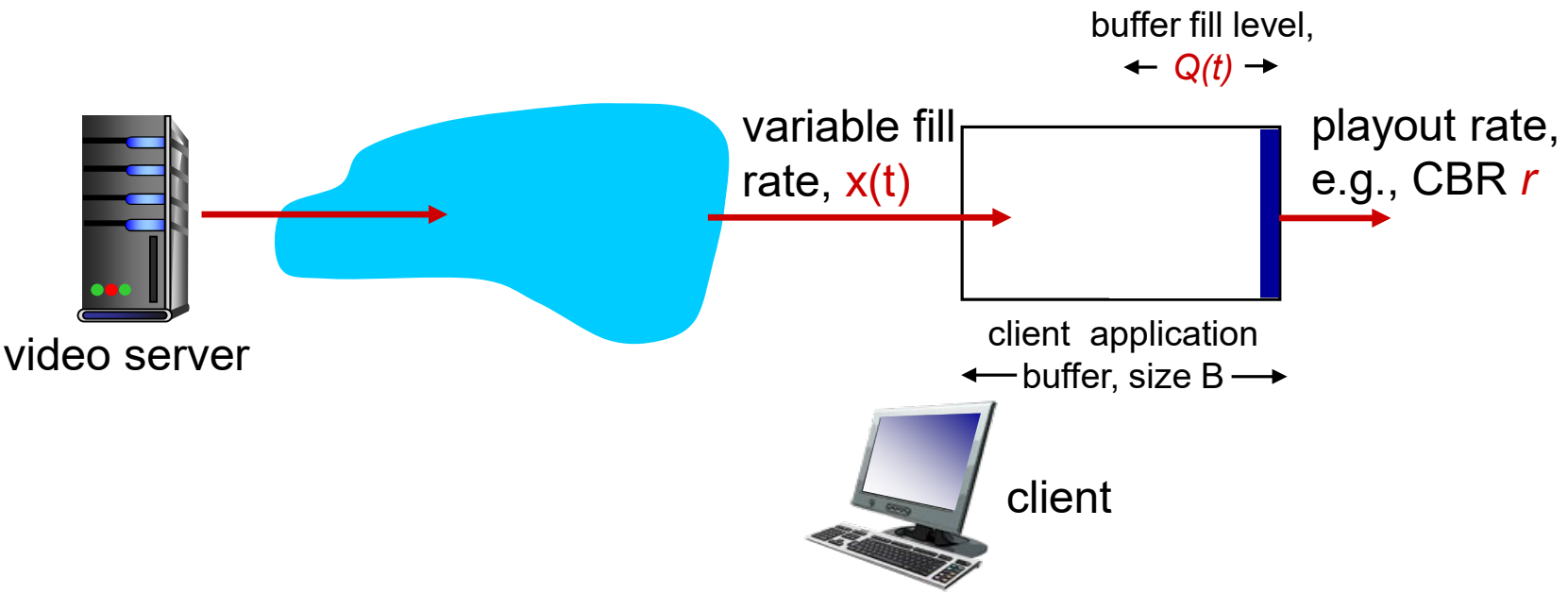


- *client-side buffering and playout delay:*
  compensate for network-added delay, delay jitter

Client-side buffering, playout

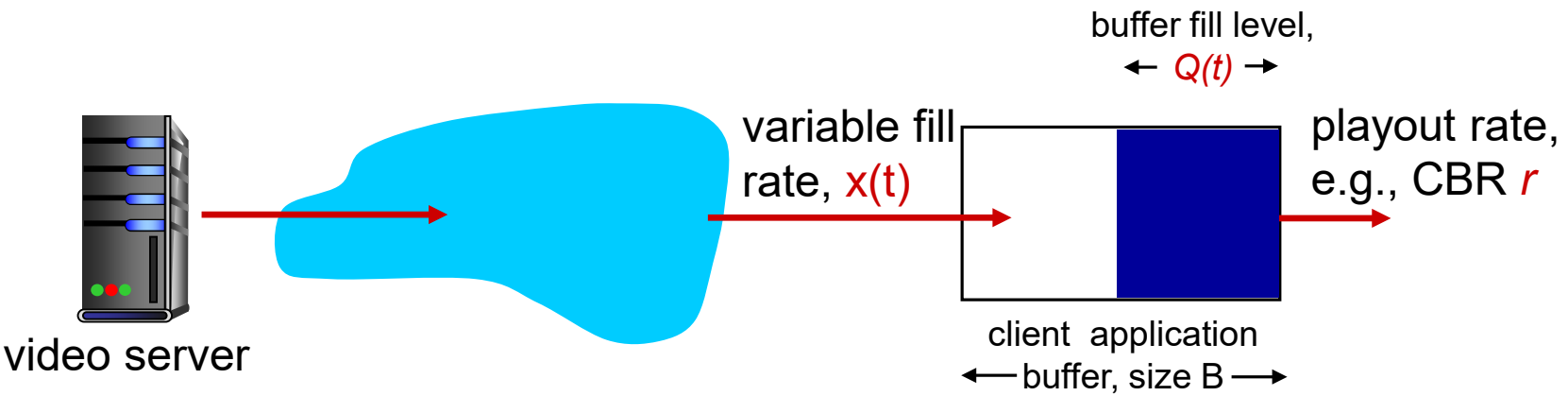


Client-side buffering, playout



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 (x), playout rate (r):

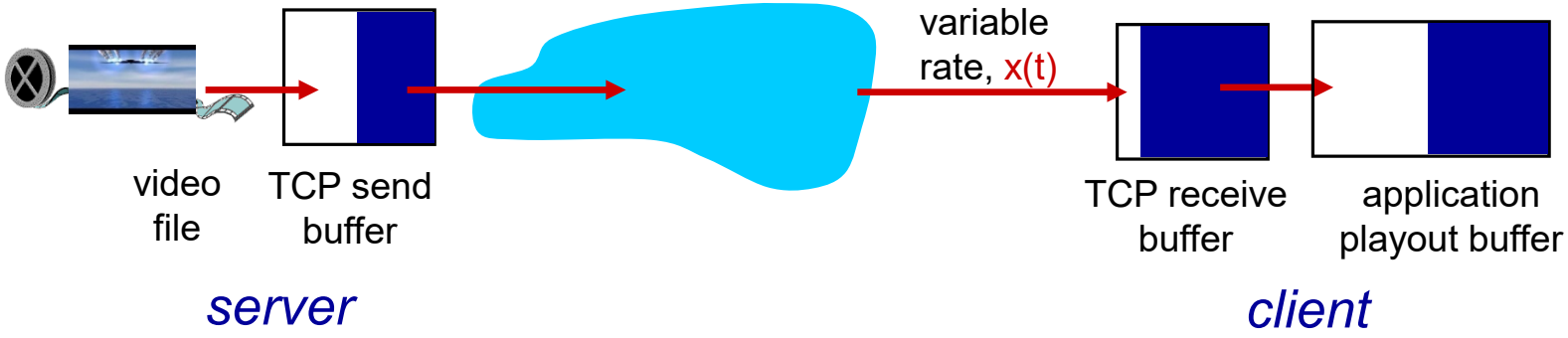
- x < r: buffer eventually empties (causing freezing of video playout until buffer again fills)
- x > r: buffer will not empty, provided initial playout delay is large enough to absorb variability in x(t)
  - *initial playout delay tradeoff:* buffer starvation less likely with larger delay, but larger 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, timeipermitting
- RTP [RFC 2326]: multimedia payload types
- UDP may *not* go through firewalls

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)

Voice-over-IP (VoIP)

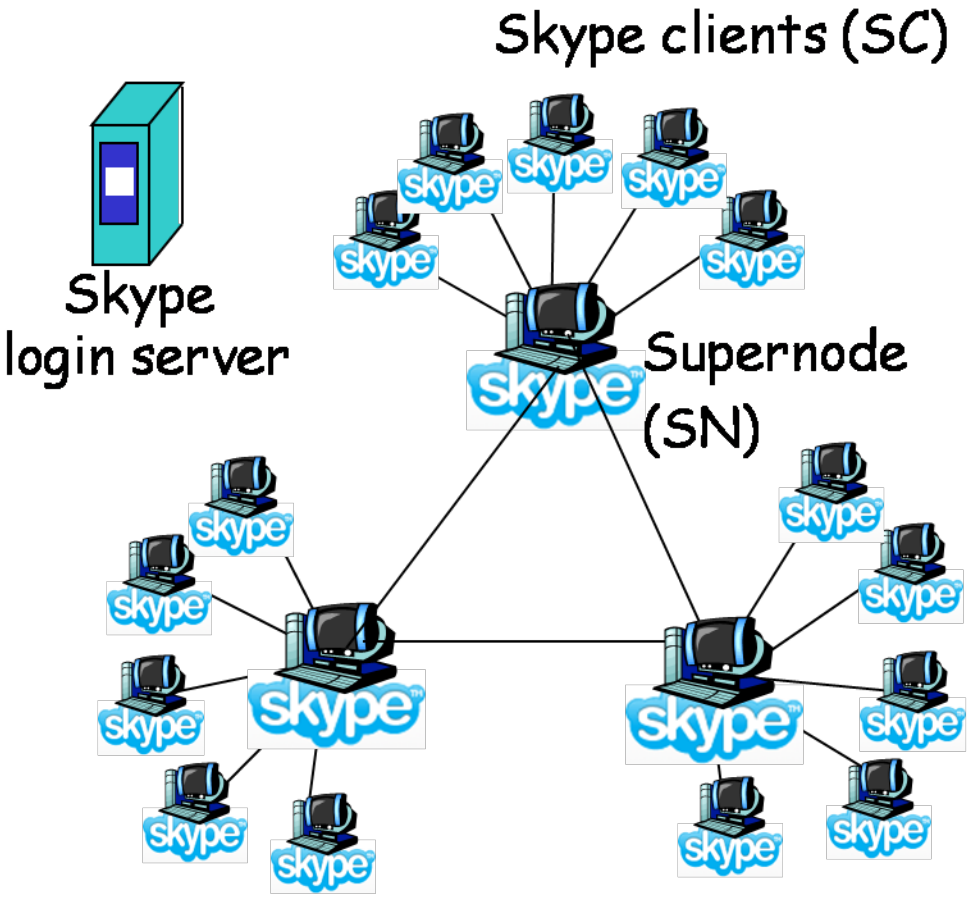
- *VoIP end-end-delay requirement*: needed to maintain "conversational" aspect
  - higher delays noticeable, impair interactivity
  - < 150 msec: good
  - > 400 msec bad
  - includes application-level (packetization,playout), network delays
- *session initialization:* how does callee advertise IP address, port number, encoding algorithms?
- *value-added services:* call forwarding, screening, recording
- *emergency services:* 911

VoIP: packet loss, delay

- *network loss:* IP datagram lost due to network congestion (router buffer overflow)
- *delay loss:* IP datagram arrives too late for playout at receiver
  - delays: processing, queueing in network; end-system (sender, receiver) delays
  - typical maximum tolerable delay: 400 ms
- *loss tolerance:* depending on voice encoding, loss concealment, packet loss rates between 1% and 10% can be tolerated

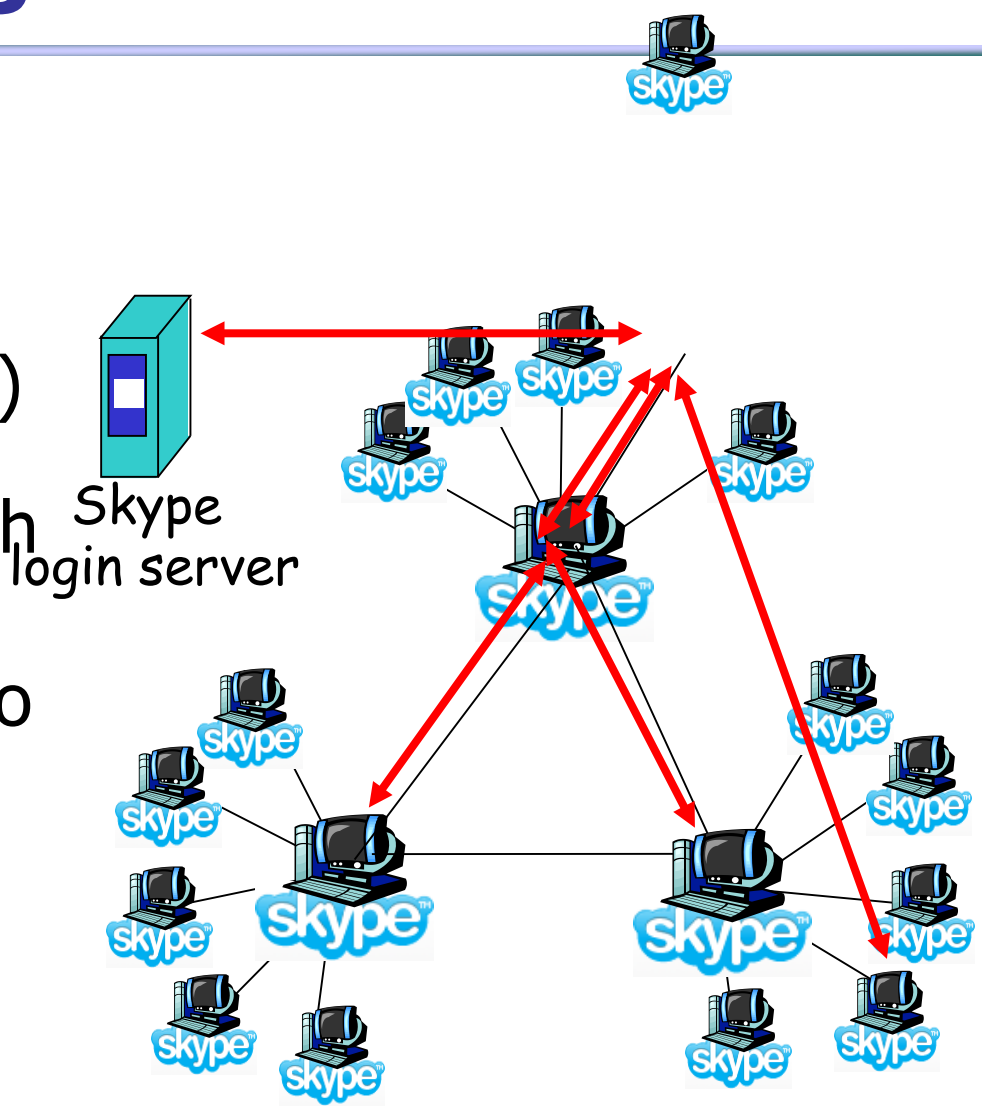
Skype

- P2P Voice-Over-IP (VoIP) application
  - pc-to-pc, pc-to-phone, phone-to-pc
- Proprietary application-layer protocol



Skype: Making a Call

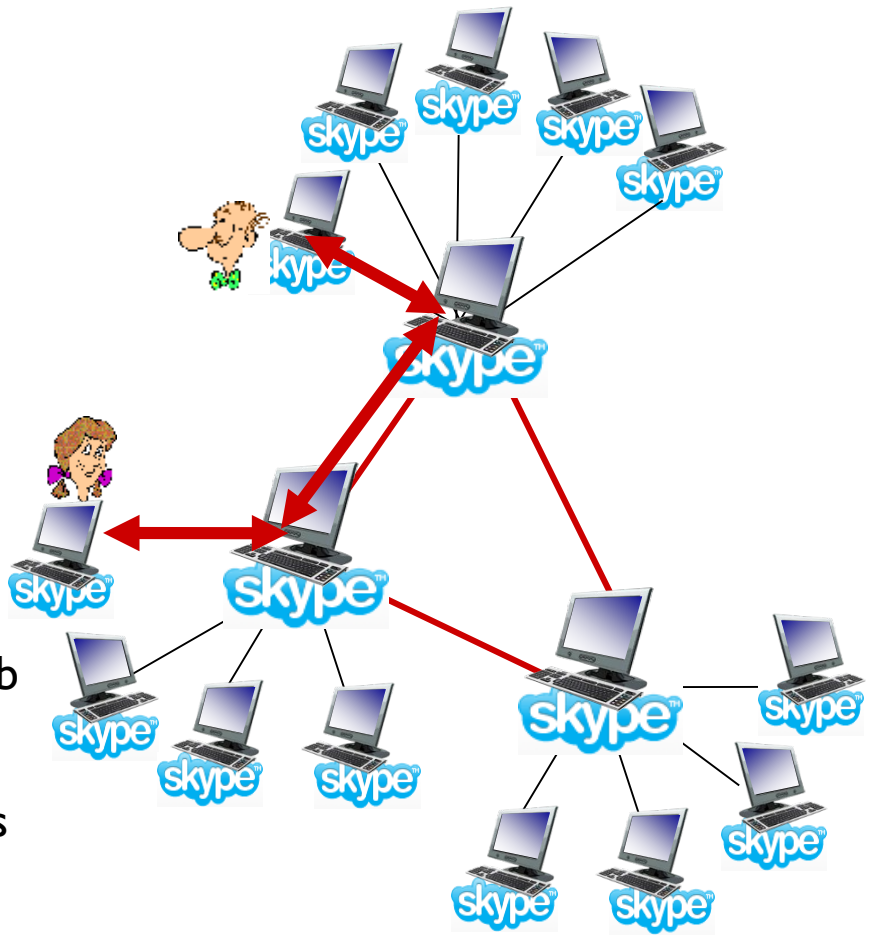
- User starts Skype
- SC registers with SN
- SC logs in (authenticate)
- Call: SC contacts SN with callee ID
- SN contacts other SNs to find address of callee
- SC directly contacts callee, over TCP



Skype Works with NAT

- *problem:* both Alice, Bob are behind "NATs"
  - NAT prevents outside peer from initiating connection to insider peer
  - inside peer *can* initiate connection to outside

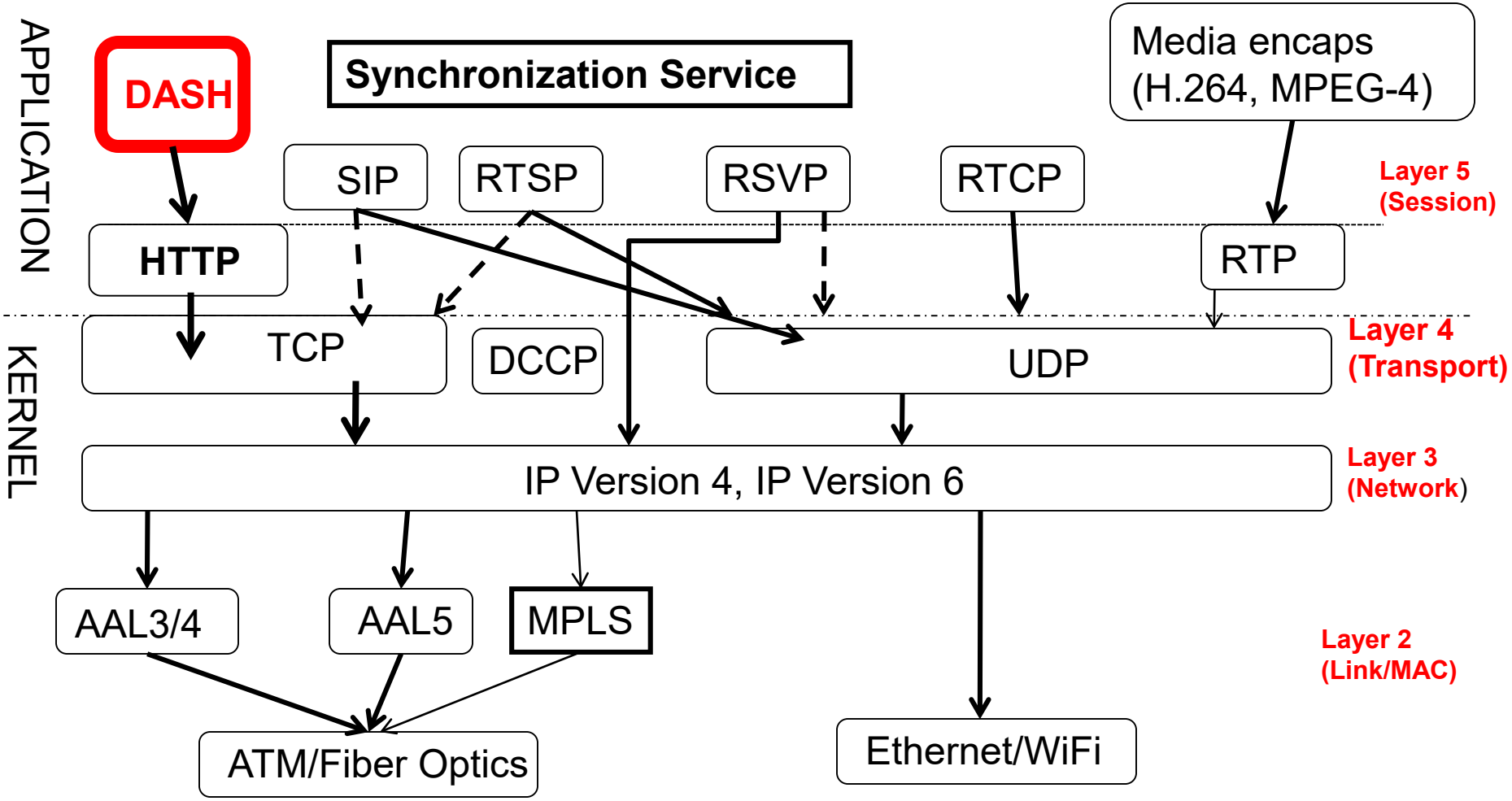
- *relay solution: Alice, Bob maintain open connection to their SNs*
  - Alice signals her SN to connect to Bob
  - Alice's SN connects to Bob's SN
  - Bob's SN connects to Bob over open connection Bob initially initiated to his SN



Real-time Transport Protocols

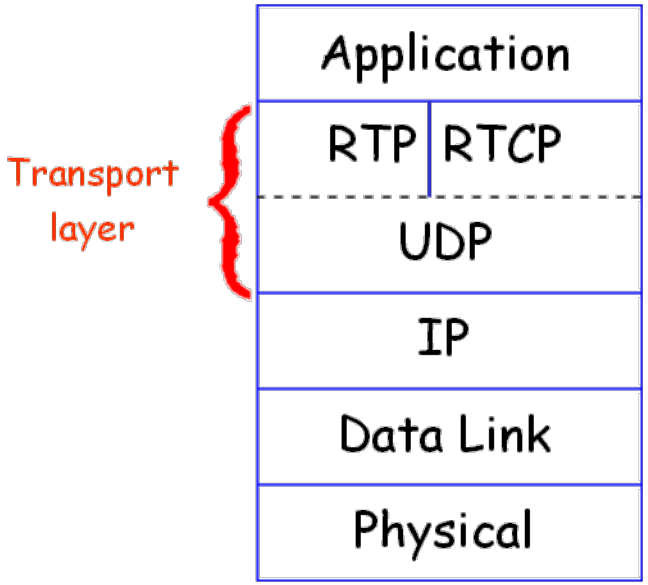
RTP/RTCP/RTSP/SIP

Internet Multimedia Protocol Stack

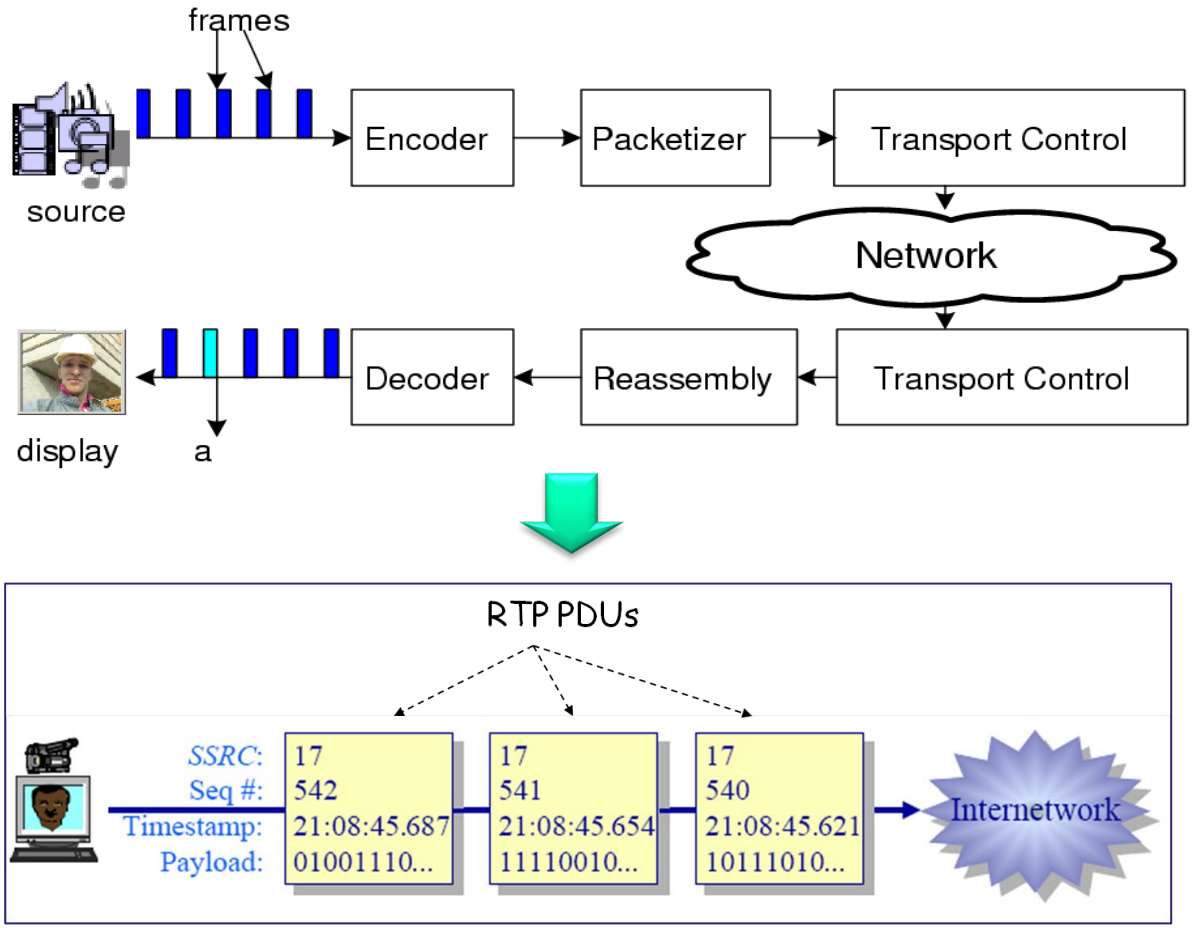


Real-time Transport Protocol (RTP)

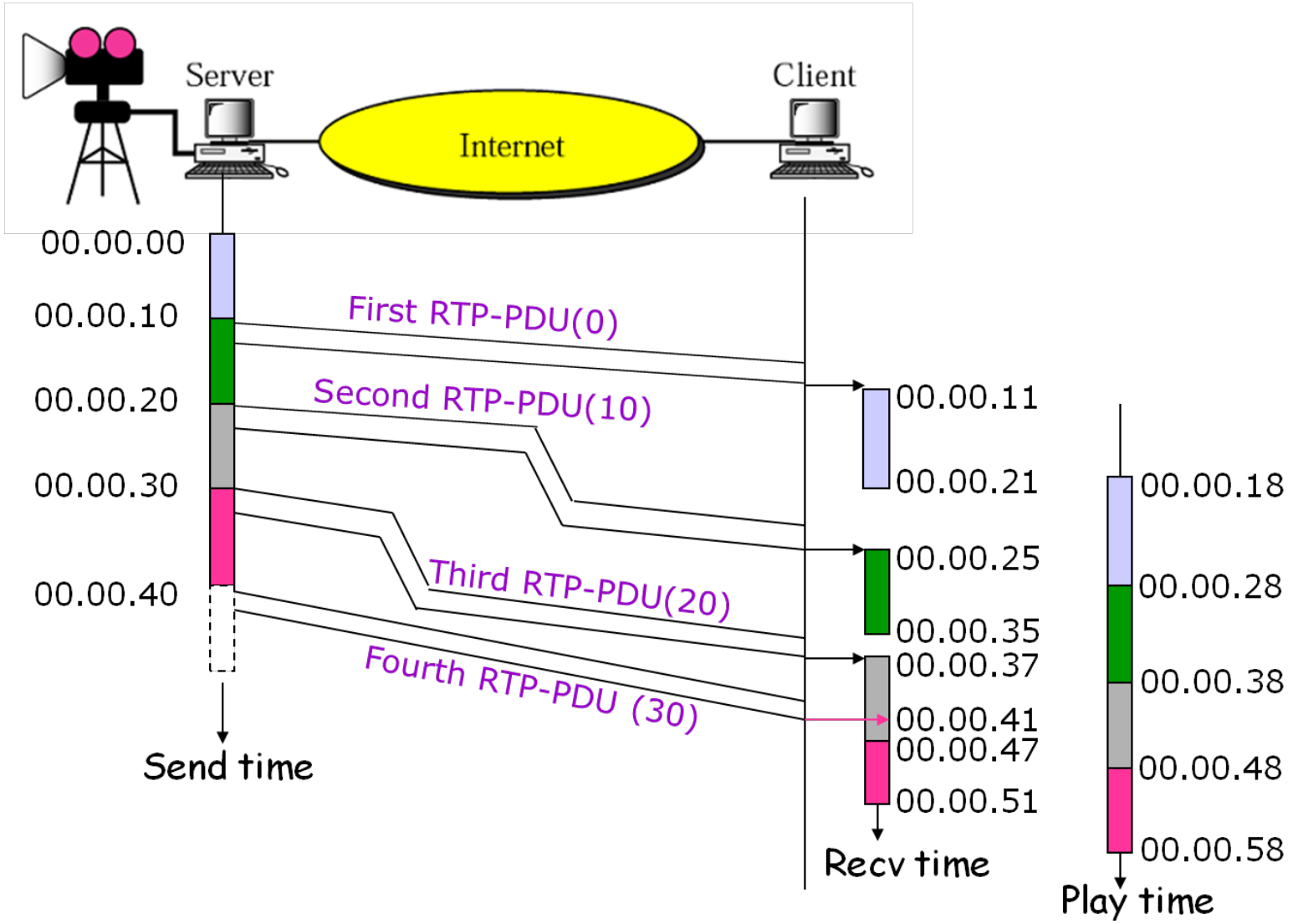
- RFC 3550
  - Built upon UDP, i.e. RTP packets encapsulated in UDP segments
  - Specifies packet structure for packets carrying audio, video data
- RTP packet provides
  - Payload type identification
  - Packet sequence numbering
  - Time stamping



Real-time Streaming



Real-time Streaming



How does RTP work

- Timestamping - most important information for real-time applications.
  - The sender timestamp according to the instant the first octet in the packet was sampled.
  - The receiver uses timestamp to reconstruct the original timing
  - Also used for synchronize different streams; audio an video in MPEG. (Application level responsible for the actual synchronization)

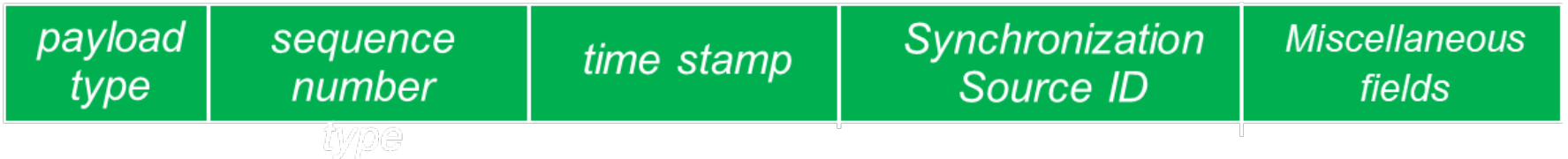
How does RTP work

- Payload type identifier
  - specifies the payload format as well as encoding/compression schemes
  - The application then knows how to interpret the payload
- Source identification
  - Audio conference

RTP and QoS

- RTP does not provide any mechanism to ensure timely data delivery
- RTP encapsulation is only seen at end systems, and unseen by intermediate routers
- Routers make no special effort for RTP packets

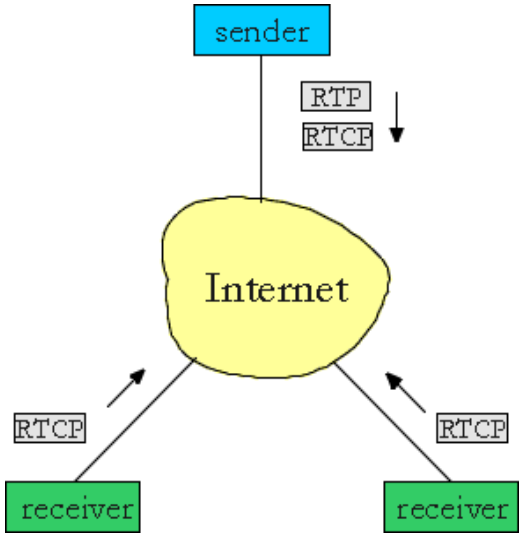
RTP Packets



- payload type (7 bits): indicates type of encoding currently being used.
- sequence # (16 bits): increment by one for each RTP packet sent
  - detect packet loss, restore packet sequence
- timestamp field (32 bits long): sampling instant of first byte in this RTP data packet
- SSRC field (32 bits long): identifies source of RTP stream.
  - Each stream in RTP session has distinct SSRC

RTP Control Protocol (RTCP)

- Specifies report PDUs exchanged between sources and destinations
- Works in conjunction with RTP
- Each participant in RTP session periodically sends RTCP control packets to all other participants



RTP Control Protocol (RTCP)

- Each RTCP packet contains sender and/or receiver reports
  - Receiver reception report
    - feedback of data delivery
    - Packet lost, jitter, timestamps
  - Sender report
    - Intermedia synchronization, number of bytes sent
  - Source description report
- Reports contain statistics such as the number of RTP-PDUs sent, number of RTP-PDUs lost, inter-arrival jitter
- Feedback used to control performance
  - Sender may modify transmission rates and for diagnostics purposes

RTCP provides the following services

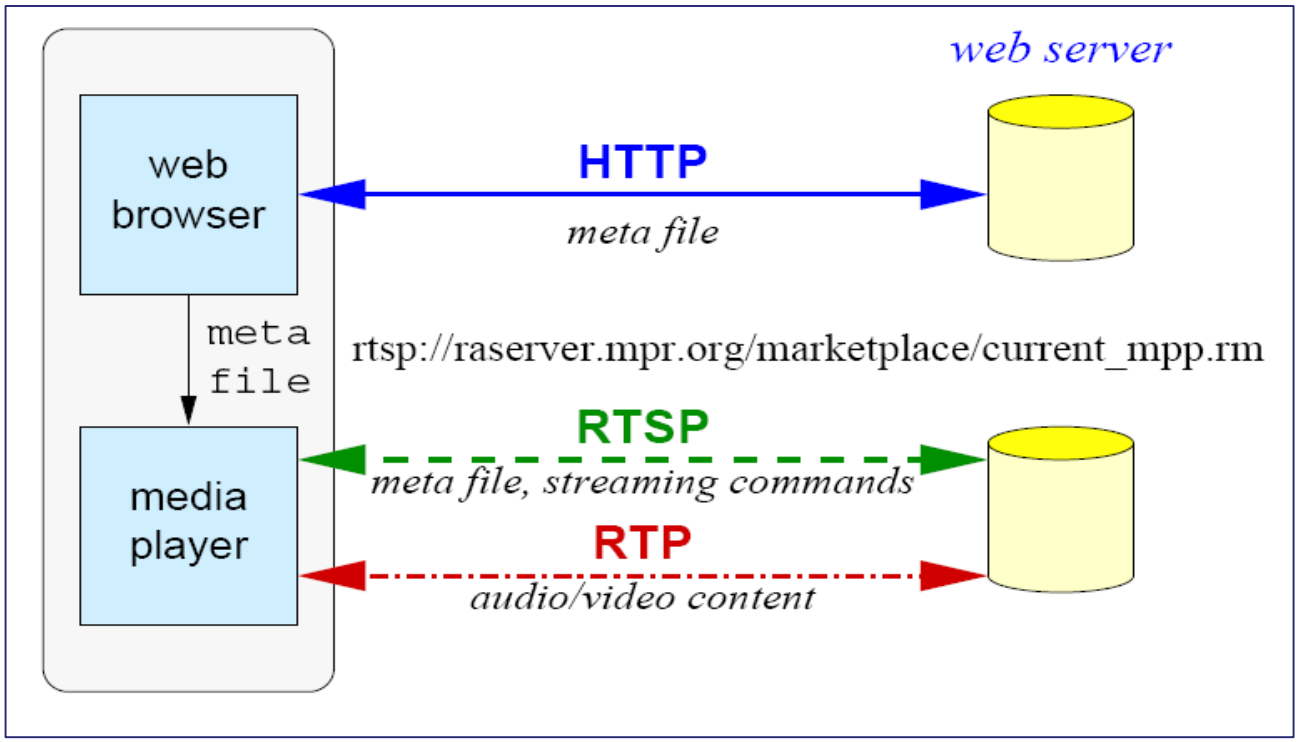
- QoS monitoring and congestion control
  - Primary function: QoS feedback to the application
  - The sender can adjust its transmission
  - The receiver can determine if the congestion is local, regional, or global
  - Network managers can evaluate the network performance for multicast distribution
- Source identification
- inter-media synchronization
- control information scaling
  - Limit control traffic (most 5 % of the overall session traffic)

Real Time Streaming Protocol (RTSP)

- RFC 2326
  - An app-level protocol that establishes and controls media sessions between end points
  - Support VCR commands: rewind, fast forward, pause, resume, repositioning, etc…
  - Can built upon UDP or TCP, commands sent in ASCII text
  - Integration with web architecture, separate stream channel and control channel

RTSP Scenario

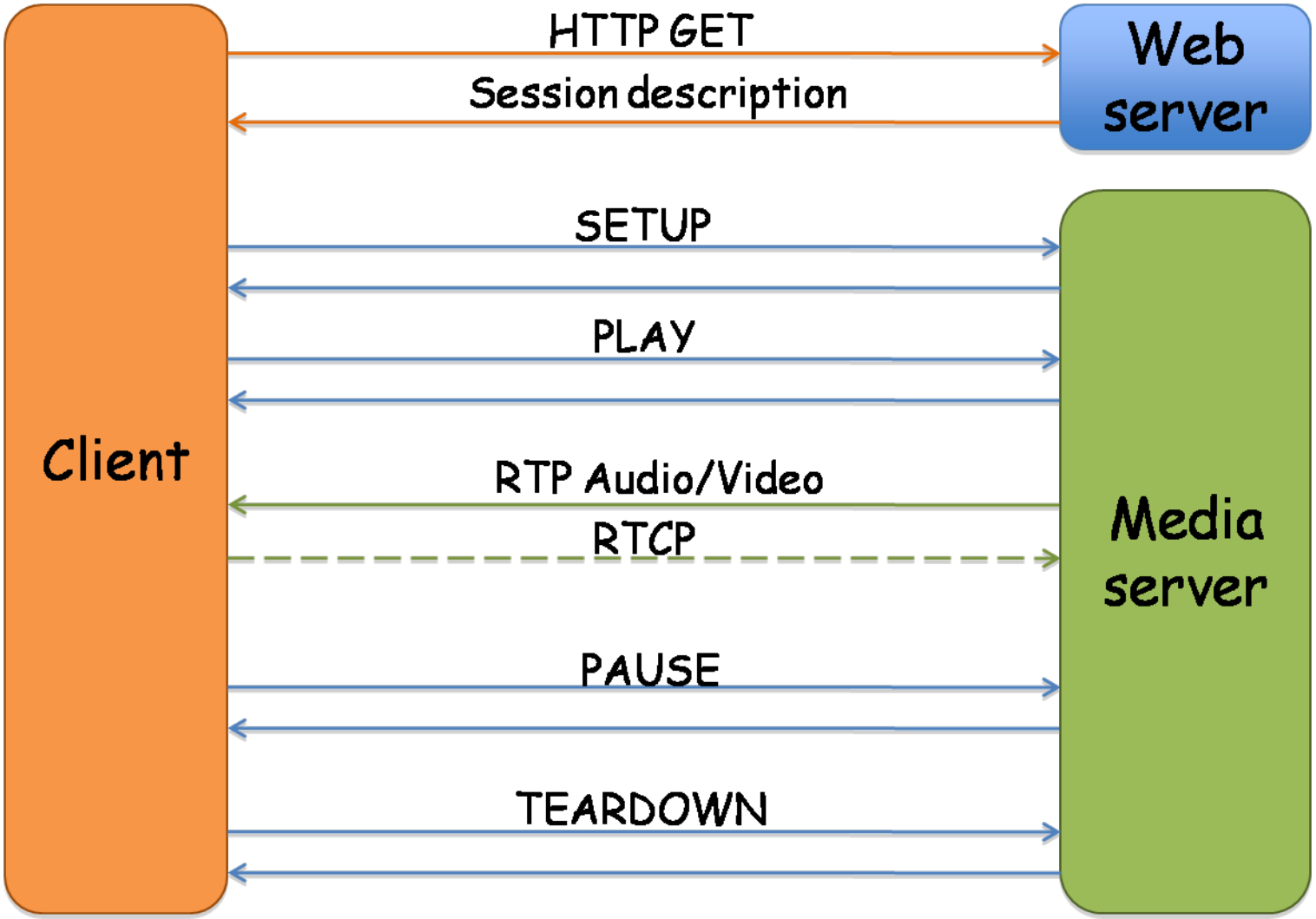
- Metafile communicated to web browser using HTTP
- Browser launches player
- Player sets up an RTSP control connection, data connection to streaming server



A Meta-File Example

```
<title>Twister</title>
<session>
    <group language=en lipsync>
        <switch>
            <track type=audio
                e="PCMU/8000/1"
                src = "rtsp://audio.example.com/twister/audio.en/lofi">
            <track type=audio
                e="DVI4/16000/2" pt="90 DVI4/8000/1"
                src="rtsp://audio.example.com/twister/audio.en/hifi">
        </switch>
        <track type="video/jpeg"
            src="rtsp://video.example.com/twister/video">
    </group>
</session>
```

RTSP Operation



Session Initiation Protocol (SIP)

- Long-term vision:
  - All telephone calls, video conference calls take place over Internet
  - People identified by names or e-mail addresses, rather than by phone numbers
  - Can reach callee (if callee so desires), no matter where callee roams, no matter what IP device callee is currently using

- Application examples
  - Video conferencing
  - Streaming multimedia distribution
  - Instant messaging
  - Online games

SIP Services

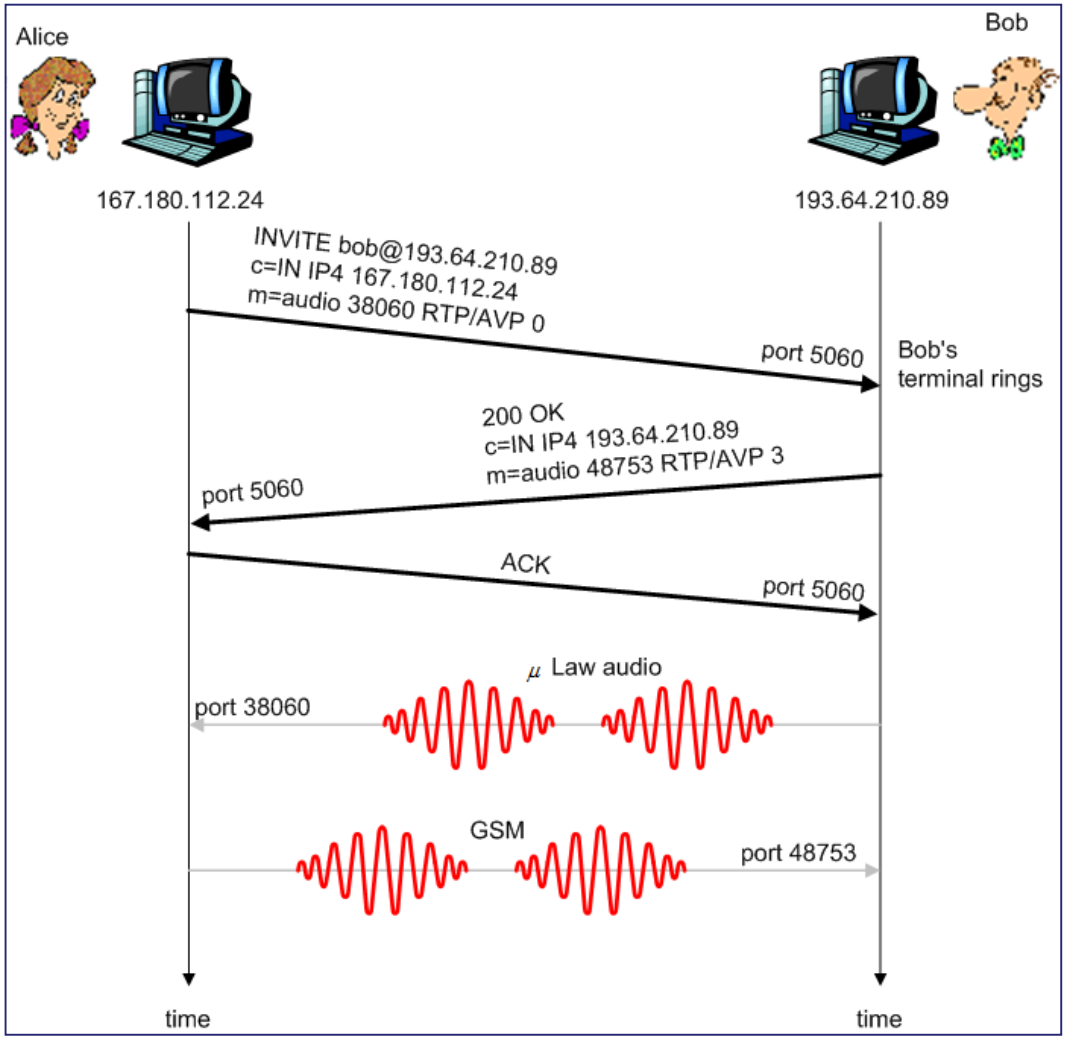
- Setting up a call
  - Determine current IP address of callee by SIP address
  - Let callee know caller wants to establish a call
  - Caller, callee agree on media type, encoding
- Call management
  - Add new media streams during call
  - Change encoding during call
  - Invite others, transfer and hold calls

SIP Addressing

- Uses Internet URLs
  - Uniform Resource Locators
  - Supports both Internet and PSTN addresses
  - General form is user@host
- Examples
  - sip:alan@wcom.com
  - sip:J.T. Kirk <kirk@starfleet.gov>
  - sip:+1-613-555-1212@wcom.com;user=phone
  - sip:guest@10.64.1.1
  - sip:790-7360@wcom.com;phone-context=VNET

Setting up a Call

- Alice's SIP agent *invite* msg, indicates her port, IP address, encoding she prefers to receive (PCM μlaw)
- Bob's agent 200 *OK* msg, indicates his port, IP address, preferred encoding (GSM)
- SIP msgs can be sent over TCP or UDP, here sent over RTP/UDP
- Default SIP port number is 5060



Other Possible Reply

- Rejecting a call
  - Bob' agent rejects with "600 busy", "503 unavailable", "302 gone", "401 unauthorized"
- Further negotiation
  - Bob replies with "606 Not Acceptable", listing his encoders
  - Alice can then send new "INVITE" message, advertising different encoder

A SIP Request Message

```
INVITE sip:bob@domain.com SIP/2.0
Via: SIP/2.0/UDP 167.180.112.24:5060
From: sip:alice@hereway.com
To: sip:bob@domain.com
Call-ID: a2e3a@pigeon.hereway.com
CSeq: 1 INVITE
Content-Type: application/sdp
Content-Length: 885
… … …
c=IN IP4 167.180.112.24
m=audio 38060 RTP/AVP 0
```

- Use HTTP message syntax
- Via: Shows route taken by request
- Call-ID: unique identifier generated by client
- CSeq: Command Sequence number, incremented for each successive request

A SIP Response Message

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP 167.180.112.24:5060
From: sip:alice@hereway.com
To: sip:bob@domain.com
Call-ID: a2e3a@pigeon.hereway.com
CSeq: 1 INVITE
```

- Via, From, To, Call-ID, and CSeq are copied exactly from Request
  - To and From are NOT swapped

Finding a Callee

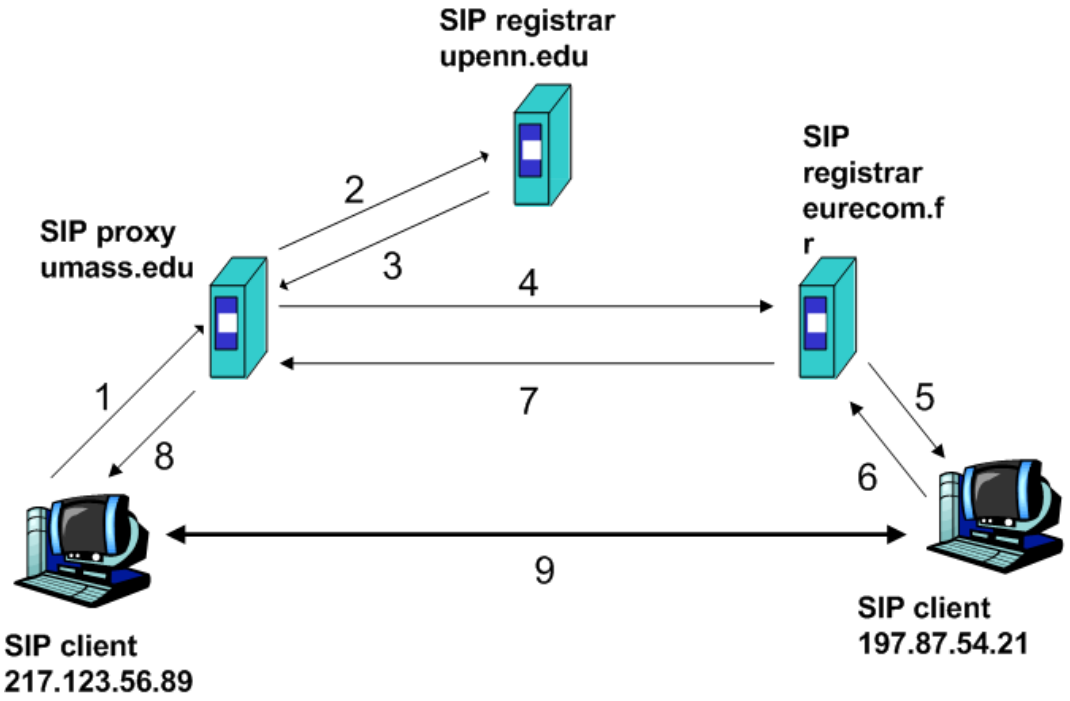
- SIP address must be transformed to IP address of callee's current host
  - Bob may move around, getting different IP addresses (using mobile devices)
- Different SIP servers to handle this
  - Registrar: Accepts REGISTER requests from clients (caller or callee)
  - Redirect: Sends address of next hop towards callee back to caller
  - Proxy: Decides next hop and forwards request (as a broker)

A More Complicated Example

Caller: Bob@umass.edu
Callee: Alice@upenn.edu

(1) Bob's agent sends INVITE message to umass SIP proxy
(2) Proxy forwards request to upenn registrar server

(3) upenn server returns redirect response, indicating that it should try Alice@eurecom.fr
(4) umass proxy sends INVITE to eurecom registrar



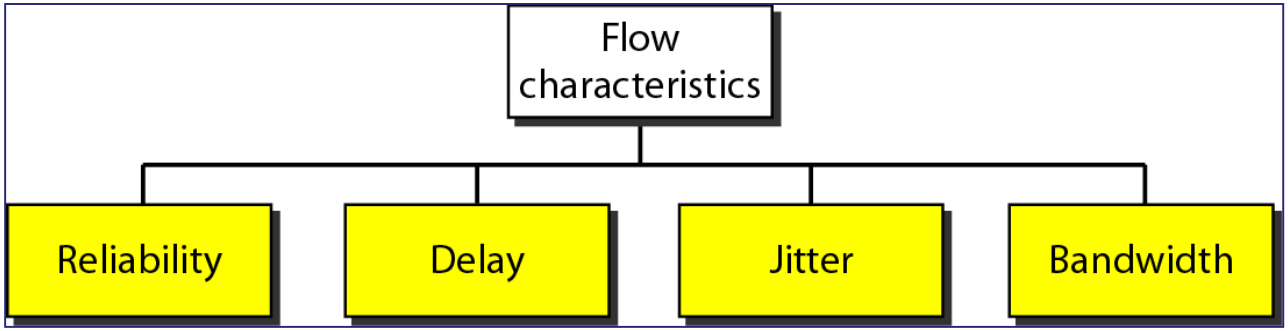
(5) eurecom registrar forwards INVITE to 197.87.54.21, which is running Alice's SIP agent
(6-8) SIP response sent back

(9) media sent directly between SIP agents

Internet QoS

Internet QoS

- New additions to Internet increasing traffic
  - High volume client/server application
  - Web with large amount of graphics
  - Real time voice and video
- Must support Quality of Service (QOS) within TCP/IP
  - In place of "best-effort"
  - Add traffic control to routers
  - Provide means of requesting QOS



Traffic Requirements of Internet Apps

Application	Data Loss (Reliability)	Throughput (Bandwidth)	Time Sensitive
File transfer	no loss	elastic	no
Email	no loss	elastic	no
Web documents	no loss	elastic	no
Real-time audio/video	loss-tolerant	audio: 5k~1Mbps video: 10k~5Mbps	100's msec
Stored audio/video	loss-tolerant	same as above	few secs
Interactive games	loss-tolerant	few kpbs up	100's msec
Instant messaging	no loss	elastic	nearly

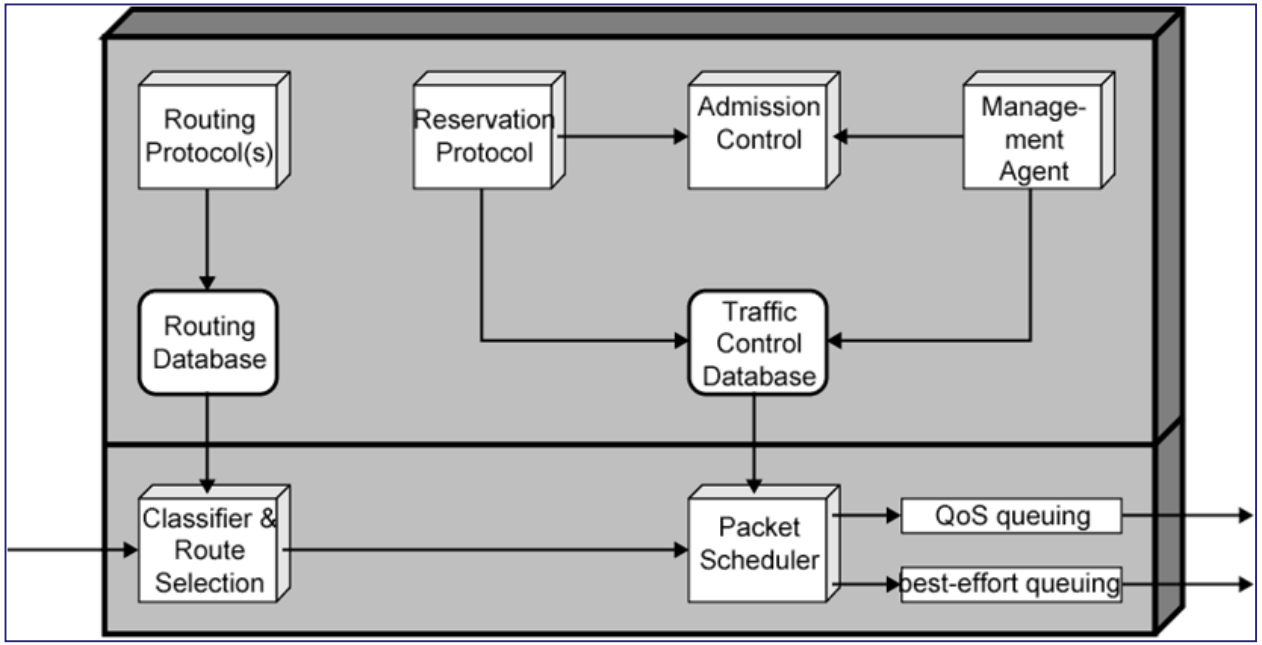
Two QoS Frameworks

- Integrated Services Architecture (ISA)
- Differentiated Services (DS)

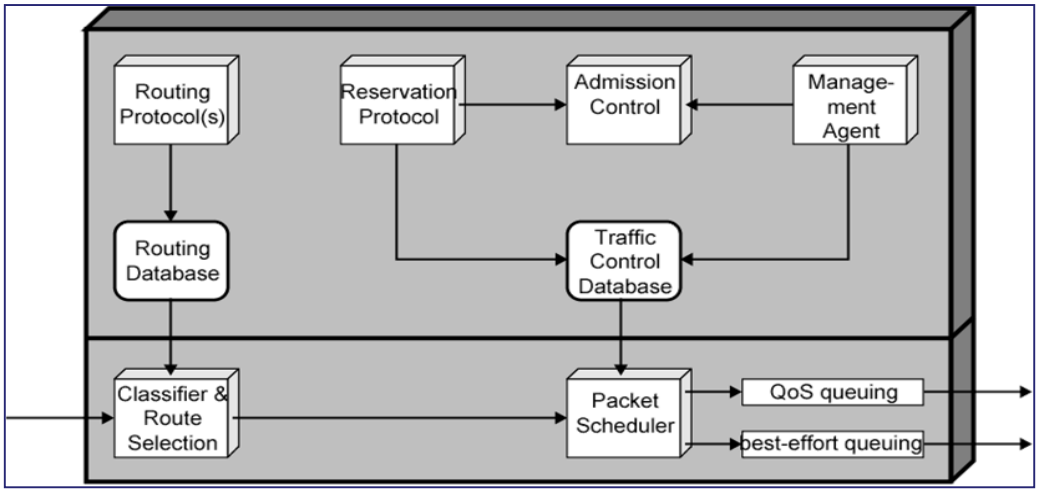
Integrated Services Architecture

- Associate a distinguishable stream of IP packets with a flow
  - With the same QOS parameters
  - Identified by source and destination IP address, port numbers, protocol type (TCP or UDP)
  - Unidirectional, Can be multicast

ISA Functions on a router

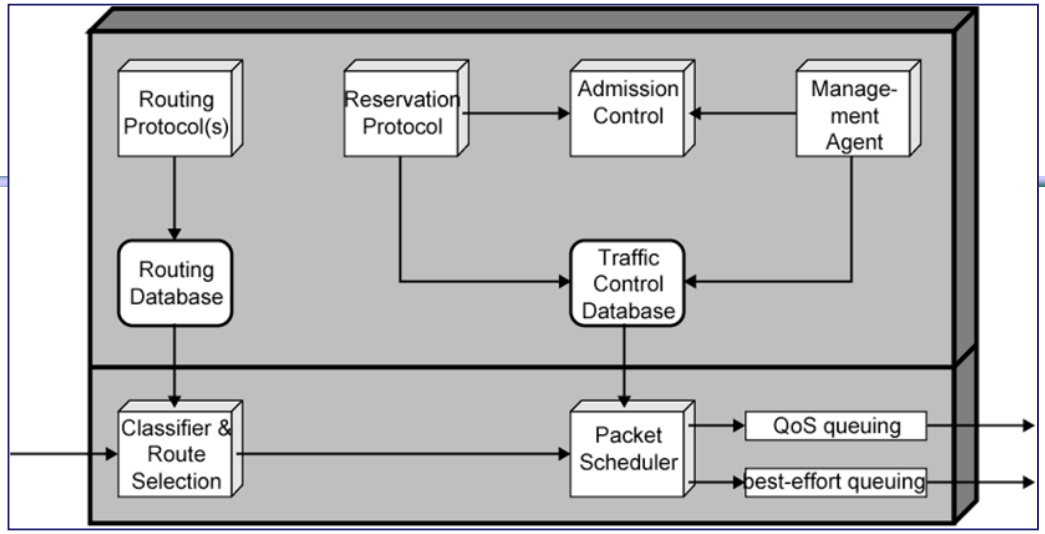


ISA Functions (1)



- Routing Algorithm
  - Link cost based on a variety of QOS parameters, not just delay
  - Routing / forwarding based on classes of flows with similar QoS
- Queuing discipline
  - Priority queuing
  - Multiple queues instead of one, taking account of different flow requirements
- Discard policy
  - Selective discard instead of just new comings

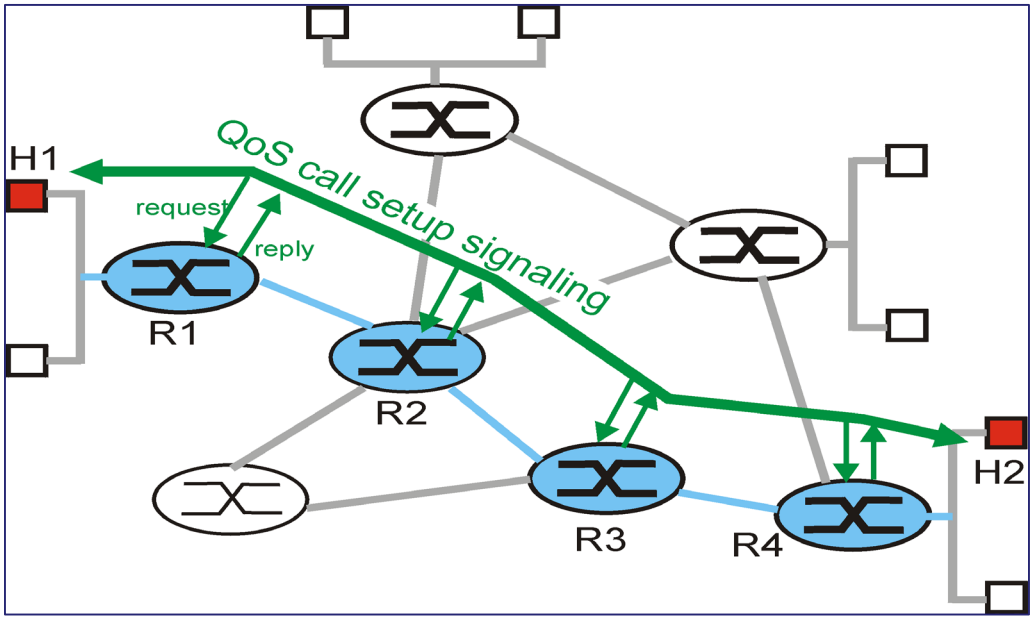
ISA Functions (2)



- Reservation protocol
  - RSVP, reserve resource for new flow at a given level of QOS
- Admission control
  - Determines if sufficient resources are available for the flow at the requested QOS
- Traffic control database
  - Parameters of traffic control
- Management agent
  - Modifies the traffic control database
  - Directs the admission control module to set policies

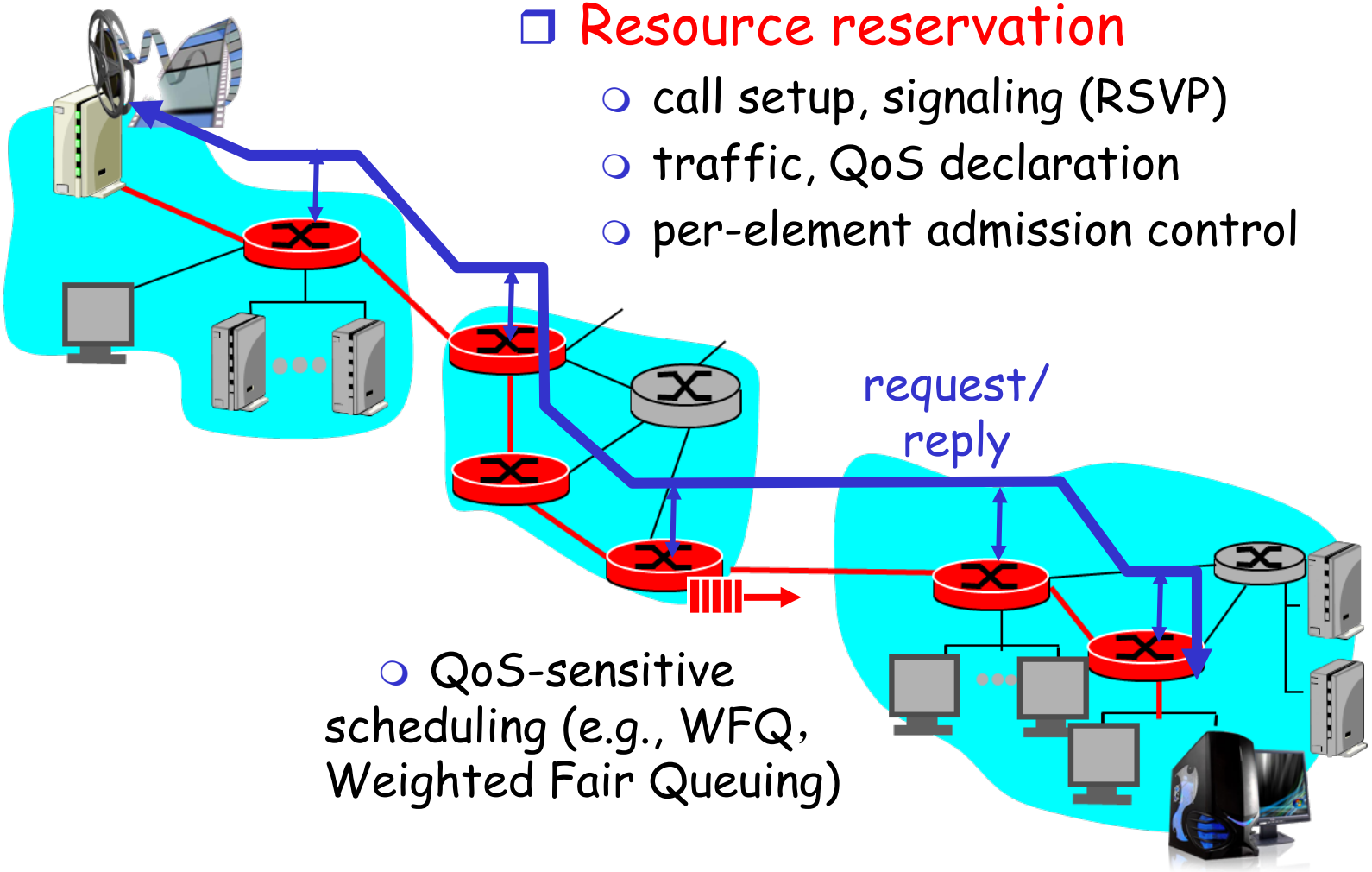
Resource Allocation and RSVP

- Providing QOS guarantees in IP networks for individual application sessions
- Resource reservation: routers maintain state info of allocated resources for each session
  - A virtual circuit like mechanism is needed
- Admit or deny call setup requests for a new session



RSVP Scenario

- Resource reservation
  - call setup, signaling (RSVP)
  - traffic, QoS declaration
  - per-element admission control



- QoS-sensitive scheduling (e.g., WFQ, Weighted Fair Queuing)

RSVP Reservation Process

- Receivers
  - Make RSVP msgs carrying reservation requests
  - Pass the msgs upstream towards the senders
- Scope of the request
  - The set of sender hosts to which a reservation request is propagated
- At each intermediate router
  - The RSVP module passes the request to Admission and Policy control and the check is executed
  - Maintain soft state (periodically renewed) for each flow
  - Reservation request is propagated upward

Disadvantage of ISA

- High cost: it need to maintain soft states for each flow, not scale well for large volumes of flows
- Complexity: the architecture is complicate, and it requires dedicate RSVP routers.
  - If there is a router in the path does not support RSVP, it will become best effort service
- Only very few service levels are defined, which is not flexible

Differentiated Services (DS)

- Providing multiple classes of service
  - Partition traffic into classes
  - Network treats different classes of traffic differently (analogy: VIP service versus regular service)

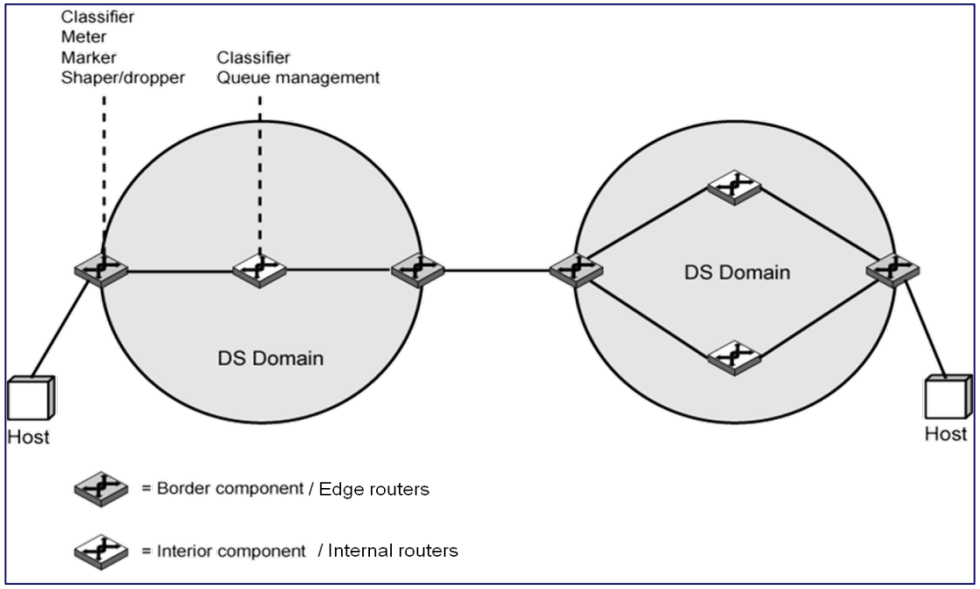
- RFC 2475
  - Provide simple, easy to implement, low overhead mechanism
  - Support range of network services differentiated on basis of performance
  - Simple functions in network core, relatively complex functions at edge routers

Characteristics of DS

- Use IPv4 header Type of Service or IPv6 Traffic Class field
- Define Service level agreement (SLA)
  - Established between provider (ISP) and customer prior to use of DS
  - Apps only need to select appropriate DS, and all traffic with same DS field treated same
  - e.g. multiple voice connections
- Classification and Conditioning
  - Per-Hop-Behavior (PHB, queuing and forwarding) determined based on DS field
  - Edge router classifies and shapes the non-conforming traffic

DS Domain

- Provided by singular ISP or group of ISPs
  - Contiguous portion of internet over which consistent set of DS policies administered
  - i.e. Similar explanation and handling of SLA parameters
- Service provider configures domain edge routers
  - Customer may be hosts or edge routers in other domain
  - Ongoing measure of performance provided for each class
  - Match the most appropriate service (class) for traffic from other domain



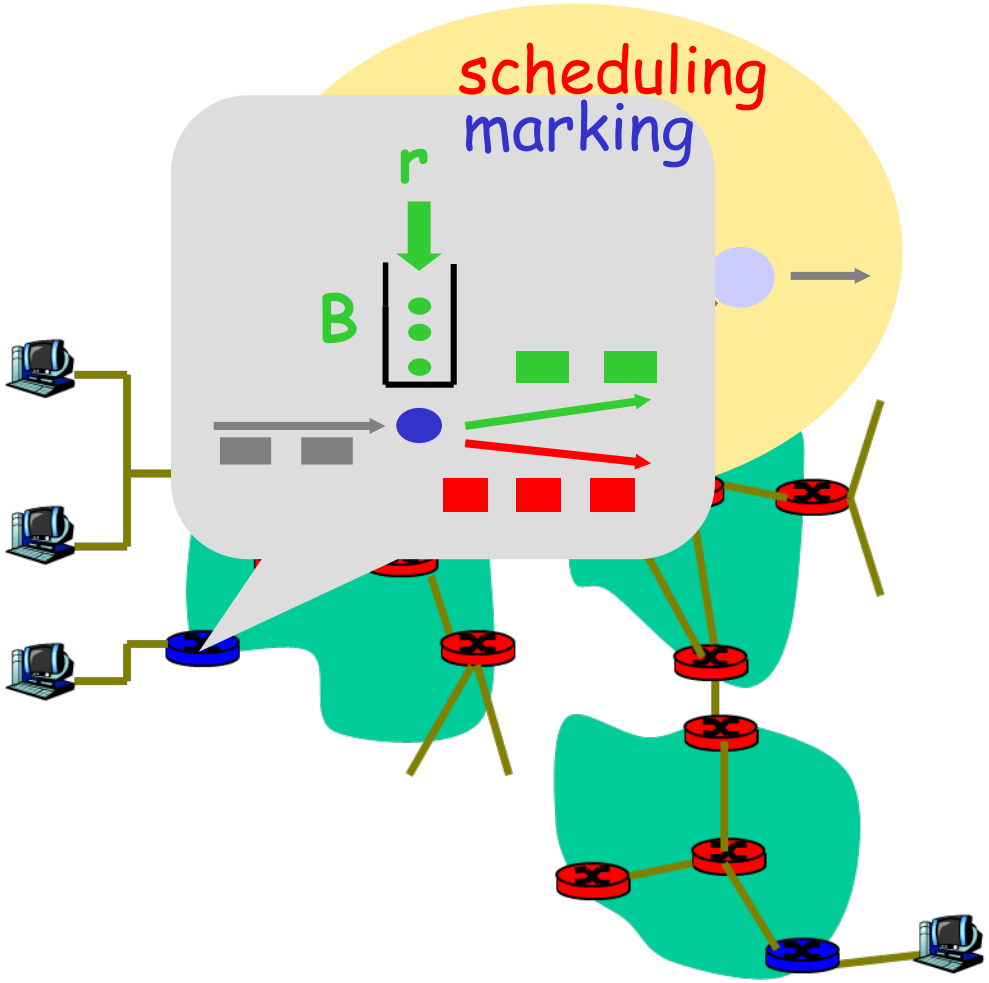
DS Architecture

Edge router

- Per-flow traffic management
- Marking and policing packets as in-profile and out-profile

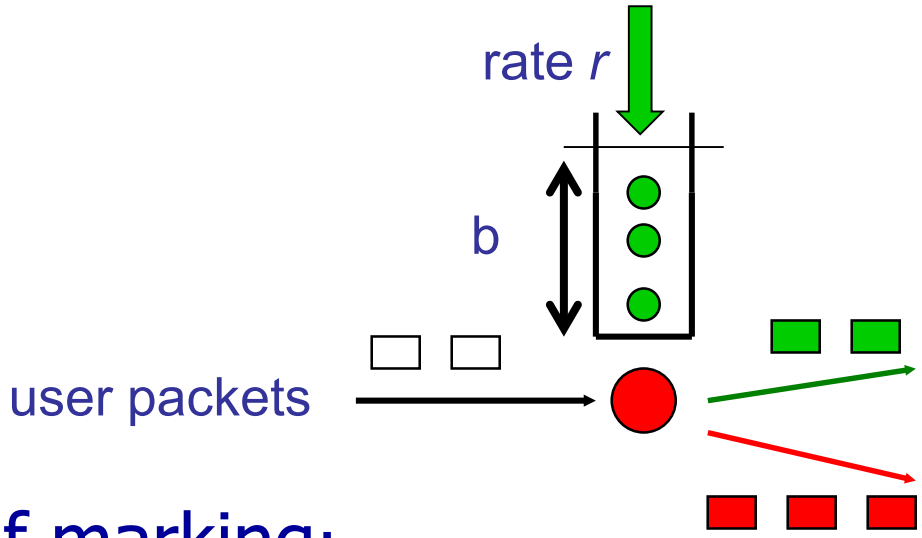
Internal router

- Per class traffic management
- Buffering and scheduling based on marking at edge
- Preference given to in-profile packets



Marking

- *profile:* pre-negotiated rate r, bucket size b
- packet marking at edge based on *per-flow* profile



possible use of marking:

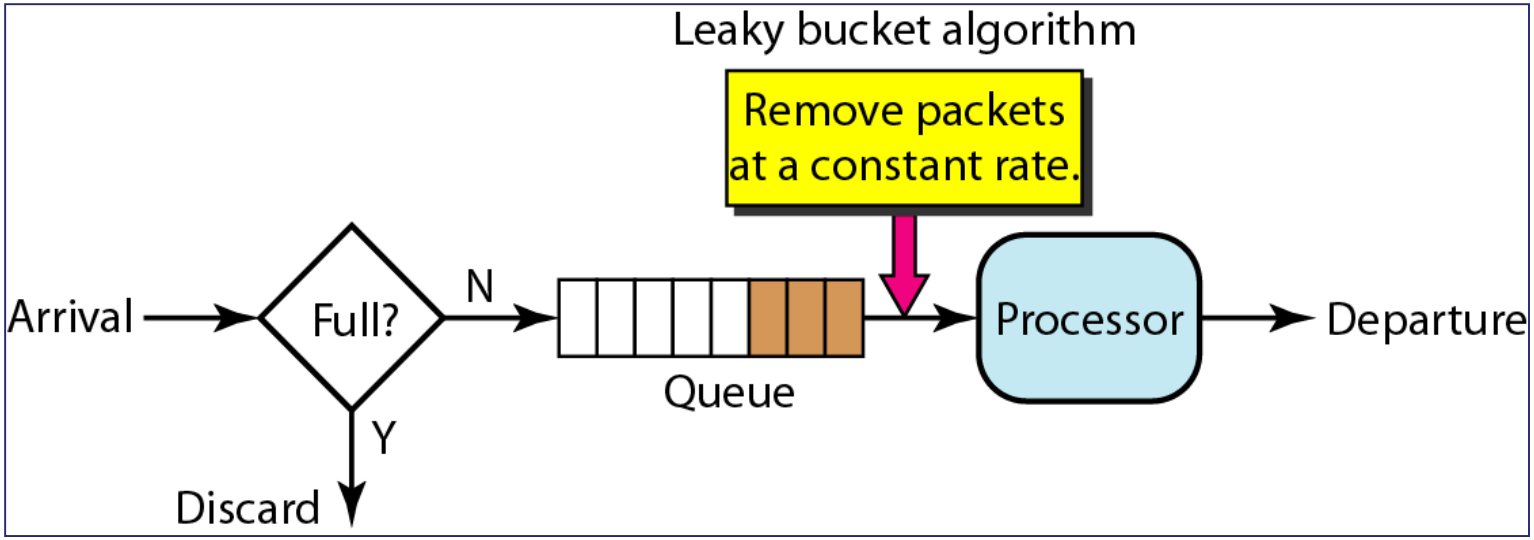
- class-based marking: packets of different classes marked differently
- intra-class marking: conforming portion of flow marked differently than non-conforming one

Policing

- Goal: limit traffic to not exceed declared parameters
- Three common-used criteria:
  - (long term) average rate: how many pkts can be sent per unit time (in the long run)
  - peak rate: e.g., 6000 pkts per min (ppm) avg.; 1500 ppm peak rate
  - (max.) burst size: max number of pkts sent consecutively (with no intervening idle)
- Shape the traffic (packet flow) before it enters the network
  - Control the rate at which packets are sent
- Two traffic shaping algorithms
  - Leaky Bucket
  - Token Bucket

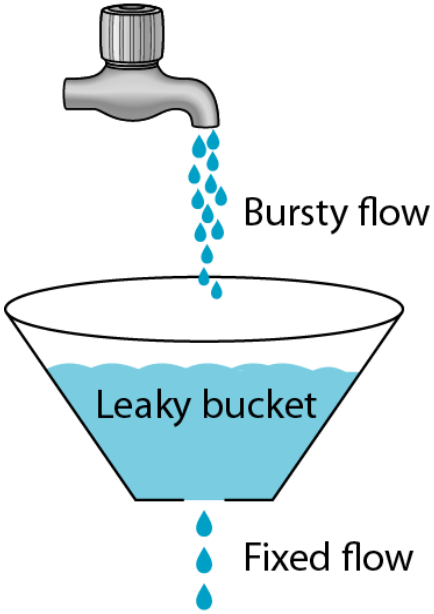
Leaky Bucket

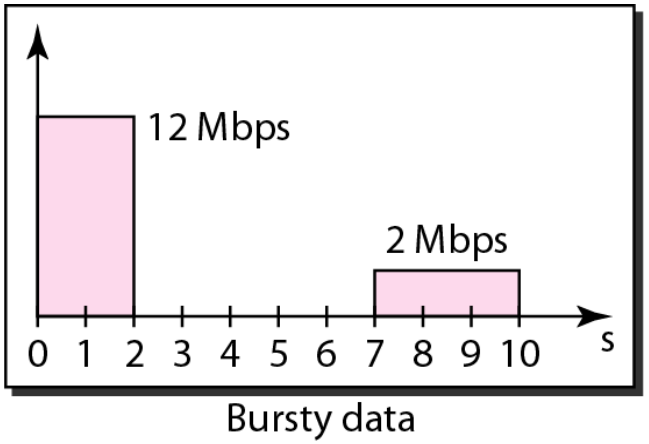
- Shape bursty traffic into fixed-rate traffic by averaging the data rate
- May drop the packets if the bucket is full



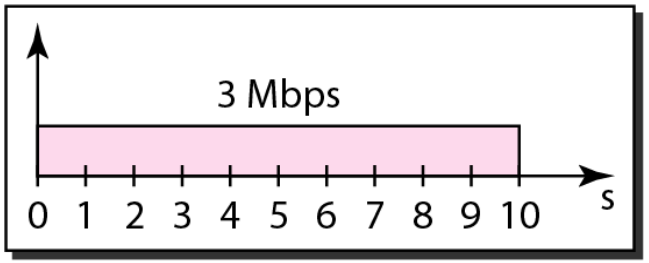
Leaky Bucket

- Do nothing when input is idle
- Packet output rate is fixed





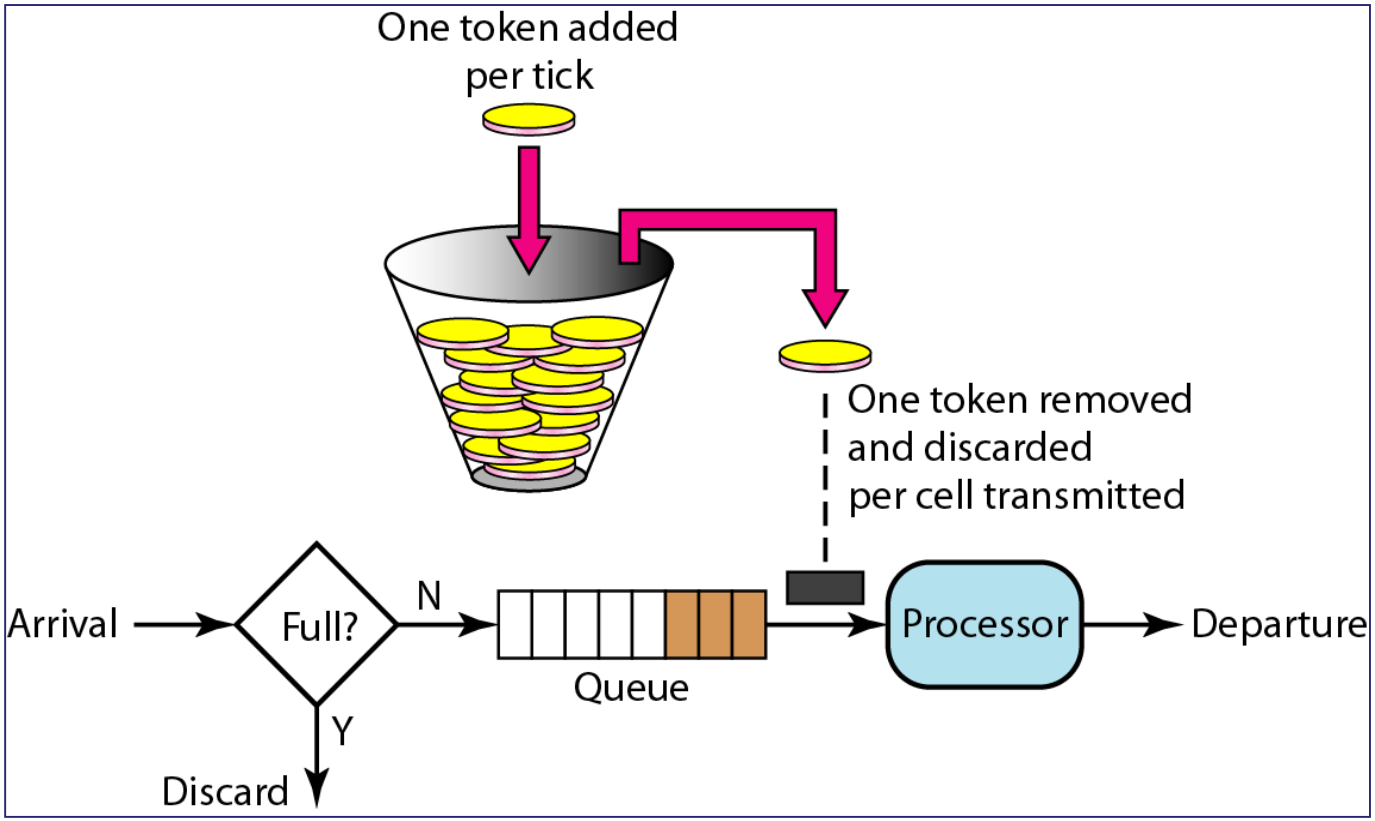
Bursty data



Fixed-rate data

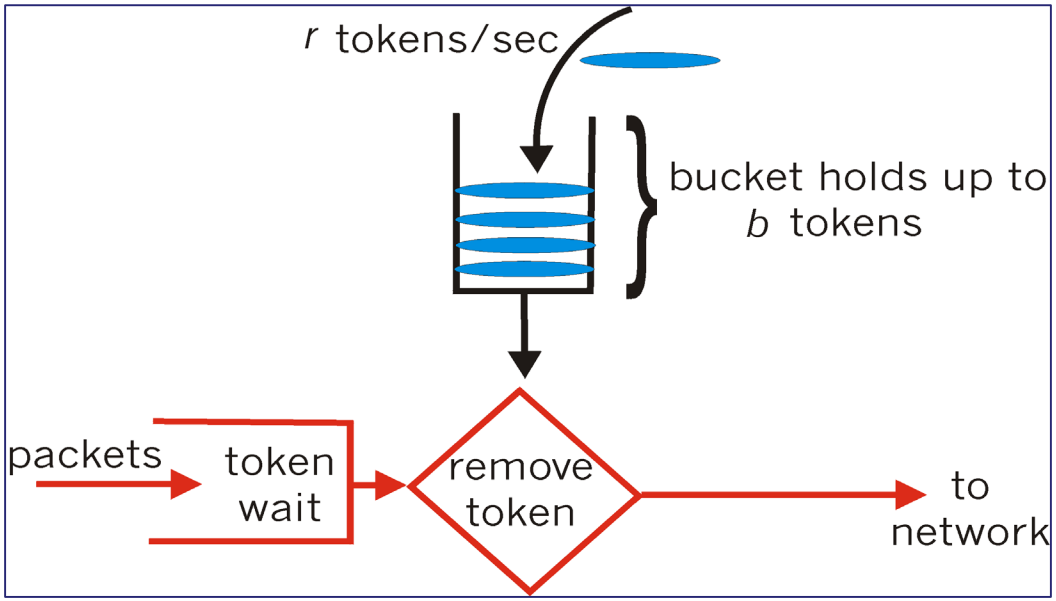
Token Bucket

- Use token to control the output traffic, allowing vary output rate
- Token generation rate is fixed, may drop token (not packet) when bucket full



Token Bucket

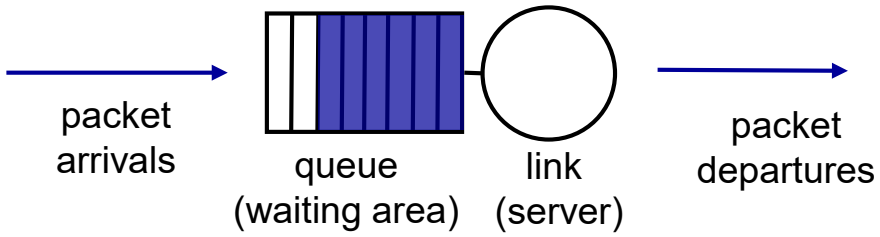
Limit input to specified *burst size* and *average rate*



- Bucket can hold b tokens
- Tokens generated at rate *r token/sec* unless bucket full
- *Over interval of length t: number of packets admitted less than or equal to (r t + b)*

Scheduling

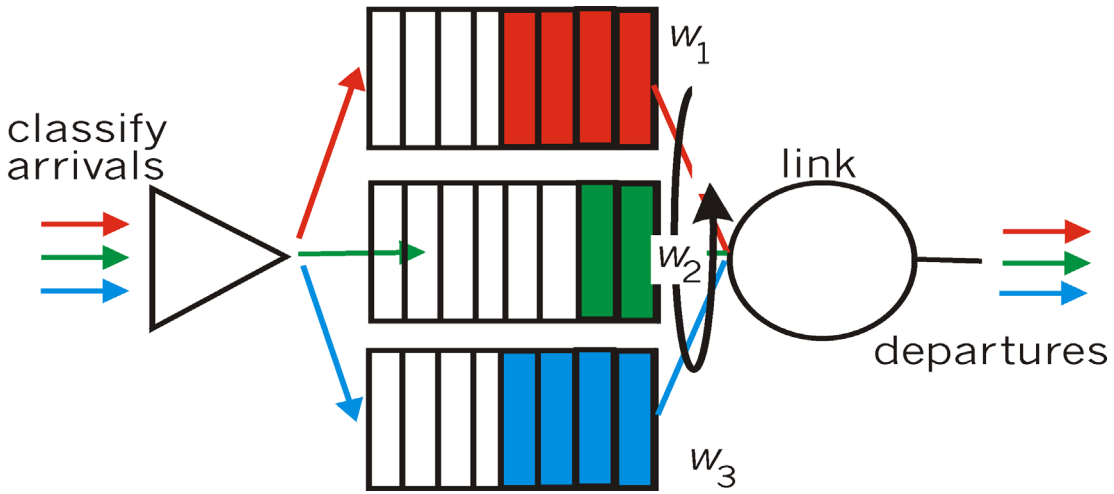
- *scheduling:* choose next packet to send on link
- *FIFO (first in first out) scheduling:* send in order of arrival to queue
  - *discard policy:* if packet arrives to full queue: who to discard?
    - *tail drop:* drop arriving packet
    - *priority:* drop/remove on priority basis
    - *random:* drop/remove randomly



Scheduling

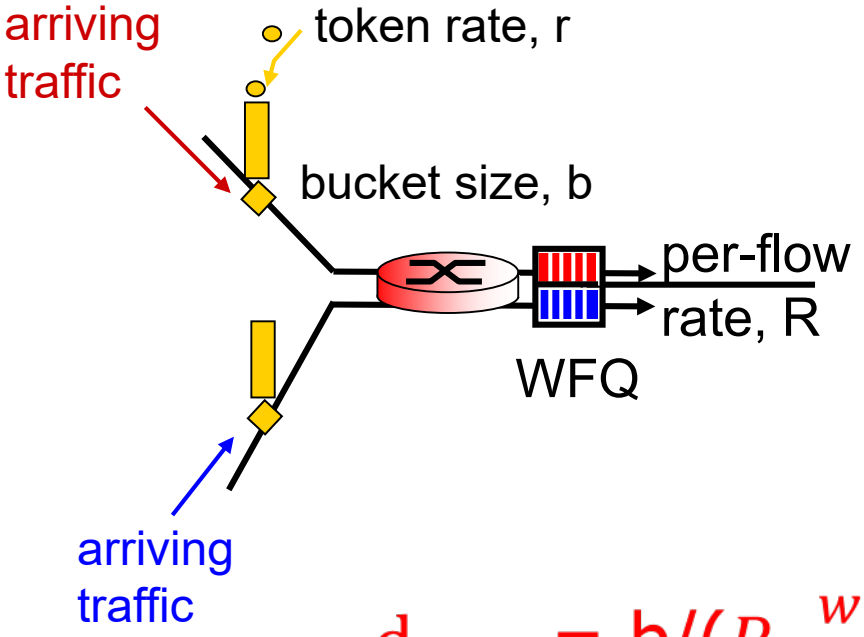
- *Weighted Fair Queuing (WFQ):*
  - Generalized Round Robin
  - Each class gets weighted amount of service in each cycle

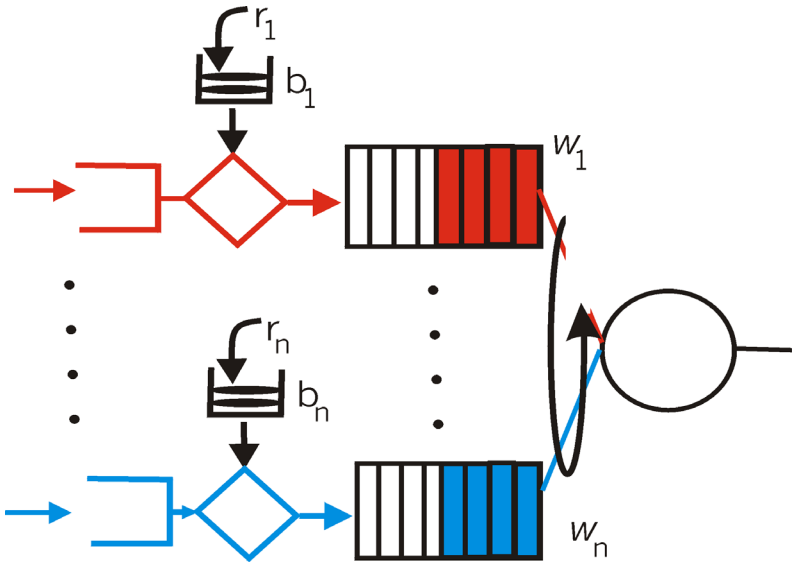
$$\text{bandwidth} \geq R \frac{w_i}{\sum w_j}$$



Policing and Scheduling for QoS guarantees

- Token bucket, WFQ combine to provide guaranteed upper bound on delay, i.e., *QoS guarantee!*



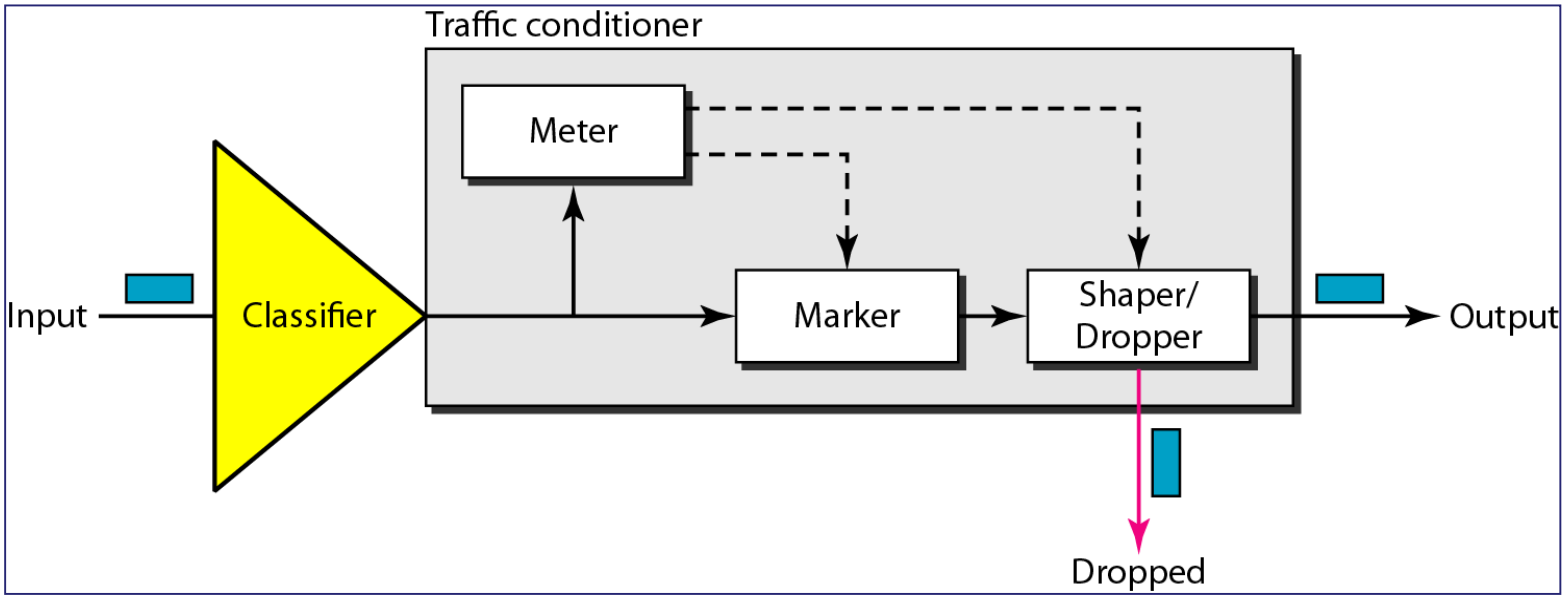


$$d_{max} = b/(R\frac{w_i}{\sum w_j})$$

Functions of Edge Routers

Traffic classification and conditioning per flow

- Classifier: separate packet flows into classes
- Meter: measure flow traffic for conformance to profile
- Marker: policing by remarking code-points if required
- Shaper: shaping packet flow using token bucket
- Dropper: drops packets if flow rate exceeds too much those specified in the class profile



Functions of Internal Routers

- Consistent interpretation of DS code-points within domain
  - Simple mechanisms to handle packets based on code-points (Class)
- Classifier
  - Differentiate packets based on DS code-point, src & dst addresses, high-level protocol, etc.
- Queue Management
  - Per Hop Behavior (PHB): queuing gives preferential treatment depending on code-point
  - Packet dropping rule dictates which to drop when buffer saturated

Comparison of QoS Frameworks

Approach	Granularity	Guarantee	Mechanisms	Complex	Deployed?
Making best of best effort service	All traffic treated equally	None or soft	No network support (all at application)	low	everywhere
Differentiated service	Traffic "class"	None of soft	Packet market, scheduling, policing.	med	some
Per-connection QoS	Per-connection flow	Soft or hard after flow admitted	Packet market, scheduling, policing, call admission	high	little to none

Summary

- Multimedia networking applications
  - Streaming stored media
    - DASH
  - VoIP
    - Skype
    - NAT
  - Live Streaming
- Real-time Transport Protocols
  - RTP, RTCP, RTSP, SIP
- Internet Quality of Service (QoS)
  - Integrated Services Architecture (ISA)
    - Architecture
    - Resource Allocation (RSVP)
  - Differentiated Services (DS)
    - Edge router
      - Per-flow traffic management
    - Internal router
      - Per class traffic management
    - Policing: Token Bucket
    - Scheduling: WFQ

Homework

- Textbook Charpter 7:

  R19, P1, P3, P5, P18, P20