Multimedia Networking

School of Informatics
Institute for Computing Systems Architecture

(Networked) Multimedia Applications

- Definition: Networked applications that employ audio or video

- Commonly used nowadays (e.g., YouTube, BBC iPlayer, Netflix, Skype)

- By 2017, video in one form or another will make up 80 – 90% of global consumer Internet traffic

  Source: Cisco Visual Networking Index: Forecast and Methodology, 2012–2017

(Networked) Multimedia Applications

- Delay sensitive but loss tolerant in contrast with traditional elastic Internet data apps like email, file transfer and web browsing
  - Variation in delay (jitter) needs to be kept low
  - Absolute delay also matters for interactive audio/video communications and live streaming
    - For conversational voice, delays ≤ 150ms for best user experience while > 400ms unacceptable

Outline

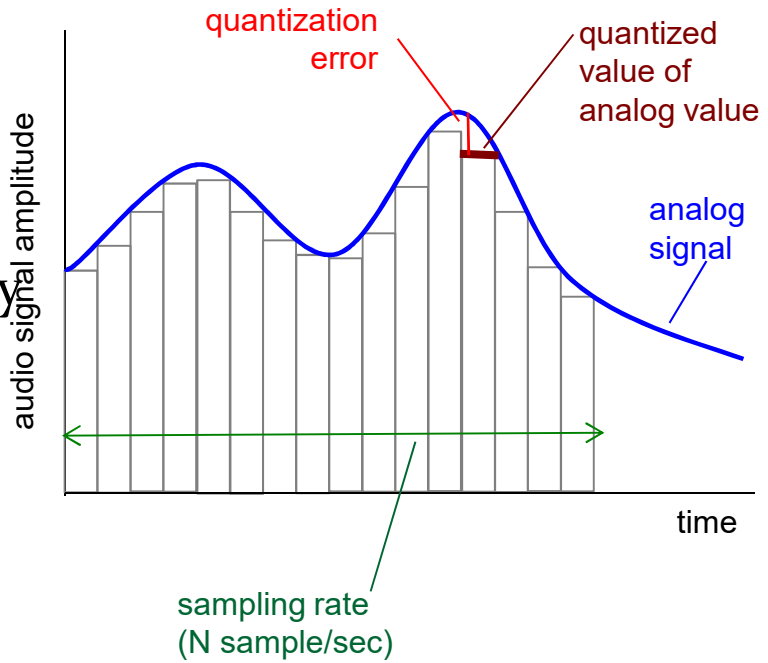
- Multimedia data
  - Overview of audio and video compression
- Different types of multimedia applications and their characteristics / service requirements, design issues and protocols
- Network support for multimedia apps

Multimedia Data

School of Informatics
Institute for Computing Systems Architecture

Digital Audio

- Digitization of human speech, music, etc.
  1. **Sampling**: original analog signal sampled at some fixed rate; the value of each sample is an arbitrary real number
  2. **Quantization**: each sample value is rounded to a finite number of values or quantization levels and represented by a binary representation of the level



- E.g., Pulse code modulation (PCM)
  - Speech encoding with PCM: sampling rate of 8000 samples per second and 8 bits per sample (256 quantization levels on a nonlinear scale) ➔ 64 Kbps
  - Audio CD also uses PCM: sampling rate of 44100 samples per second and 16 bits per sample on a linear scale ➔ 705.6 Kbps (mono) and 1.411 Mbps (stereo)

Audio Compression Overview

- Compression/Encoding: audio transmitted on the Internet is typically compressed with a reduced bit rate
  - Human speech offers a great potential for compression to as low as 2.4 Kbps
  - MPEG 1 audio layer 3 (MP3): popular audio compression technique for music streaming encoded at 128 Kbps rate
  - Advanced Audio Coding (AAC): successor to MP3 and default audio encoding used in MPEG 4
- Decompression/Decoding
  - Approximately recover the original signal
  - Higher sampling rate and more number of quantization levels (higher bit rate) better is the approximation
- Ears more sensitive than eyes ➔ audio glitches need to be kept minimal to ensure good user experience

Digital Video

- Video is a sequence of images that are typically displayed at a constant rate, typically referred to as frames per second (fps), e.g., 24 or 30 fps
  - Image is an array of pixels, each encoded into a number of bits to represent luminance and color
- High bit rate compared to music streaming and image transfers
  - from 100 Kbps for low quality video conferencing to over 3 Mbps for streaming HD movies
  - Cisco report from 2011 predicts that 90% of global Internet traffic from streaming stored video by 2015 ← popularity of video content plus high bit rate nature

Video Compression Overview

- Compression: as with audio, can be compressed to reduce bit rate (and video quality)
  - By exploiting spatial and/or temporal redundancy
  - To create multiple versions of same video for adaptive delivery depending on network conditions (end-to-end available bandwidth) and host characteristics

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

School of Informatics
Institute for Computing Systems Architecture

Multimedia Applications

School of Informatics
Institute for Computing Systems Architecture

Networked Multimedia Applications: 3 Types

1. Streaming stored audio/video (e.g., Netflix, Hulu, Kankan, BBC iPlayer, YouTube, YouKu, Dailymotion)

2. Streaming live audio/video (e.g., Internet radio, IPTV)

3. Conversational voice and video over Internet (e.g., Skype, Google Talk)

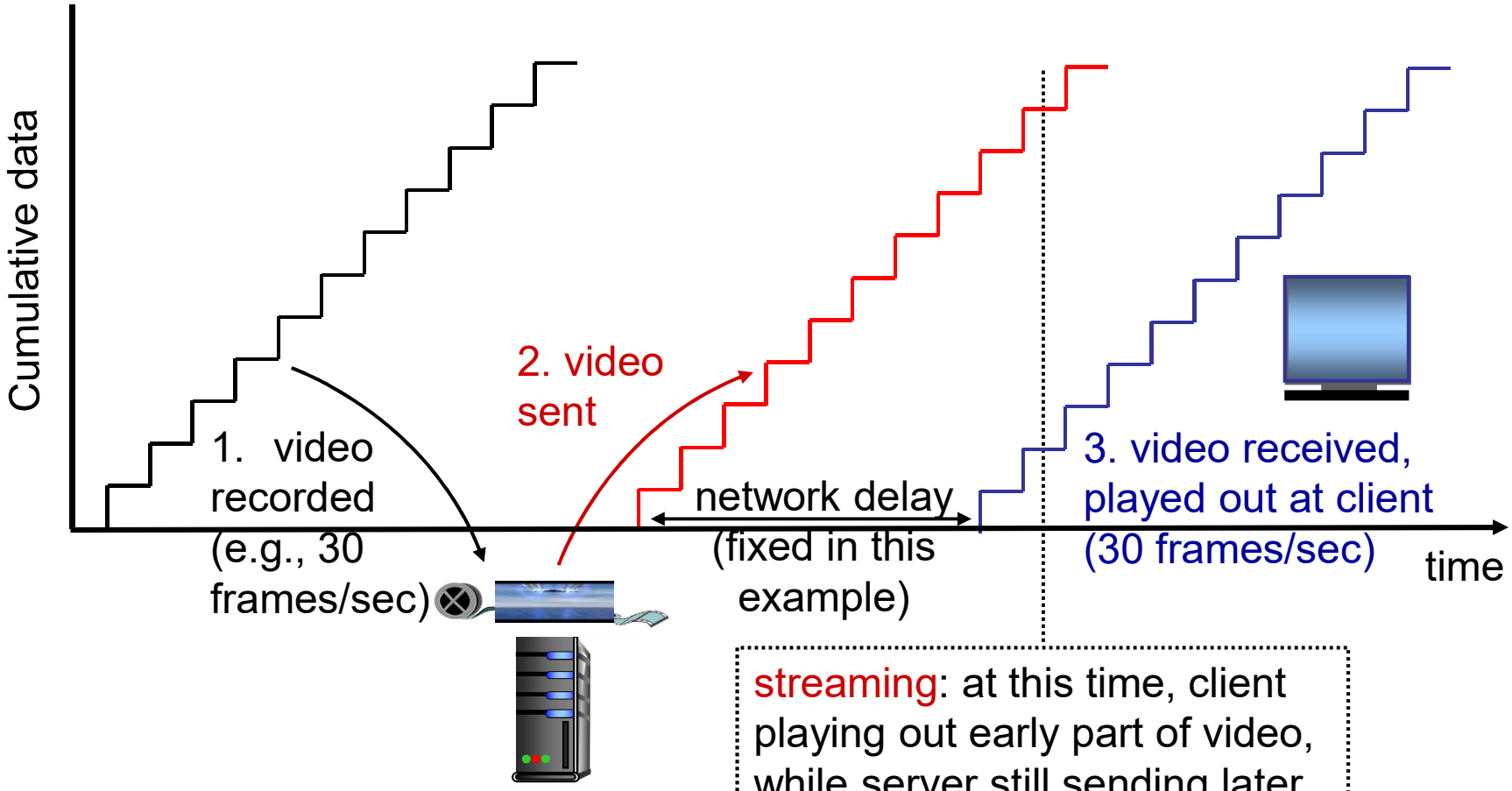
Streaming Stored Audio and Video (1)

- Multimedia content stored on servers, possibly in different encodings
  - Content is often placed on a content distribution network (CDN) for faster access
  - With P2P streaming applications, peers hold different chunks of content and they collectively form the "server"
- Clients make *on-demand* requests for stored audio/video content from servers
- Examples
  - Online movie watching sites (Netflix, Hulu, Lovefilm, Kankan, etc.)
  - Catch up TV (BBC iPlayer, etc.)
  - Online video sharing sites (YouTube, Dailymotion, YouKu, etc.)

Streaming Stored Audio and Video (2)

- Three key characteristics/requirements
  - *Streaming* (as opposed to download-and-play)
  - *Interactivity* (e.g., pause, forward)
  - *Continuous playout* as per the timing of the reference stored video; not receiving frames in time causes stalls on the client side and degrades user experience
- Sufficient to focus on streaming stored video as it usually contains both video and audio components
  - Streaming stored audio alone ➔ much lower bit rates, hence less challenging
  - Streaming stored video sometimes also referred to as video-on-demand (VoD)
  - Makes up over 50% of downstream traffic in Internet access networks today [Cisco 2011]

Streaming Stored Video Illustrated



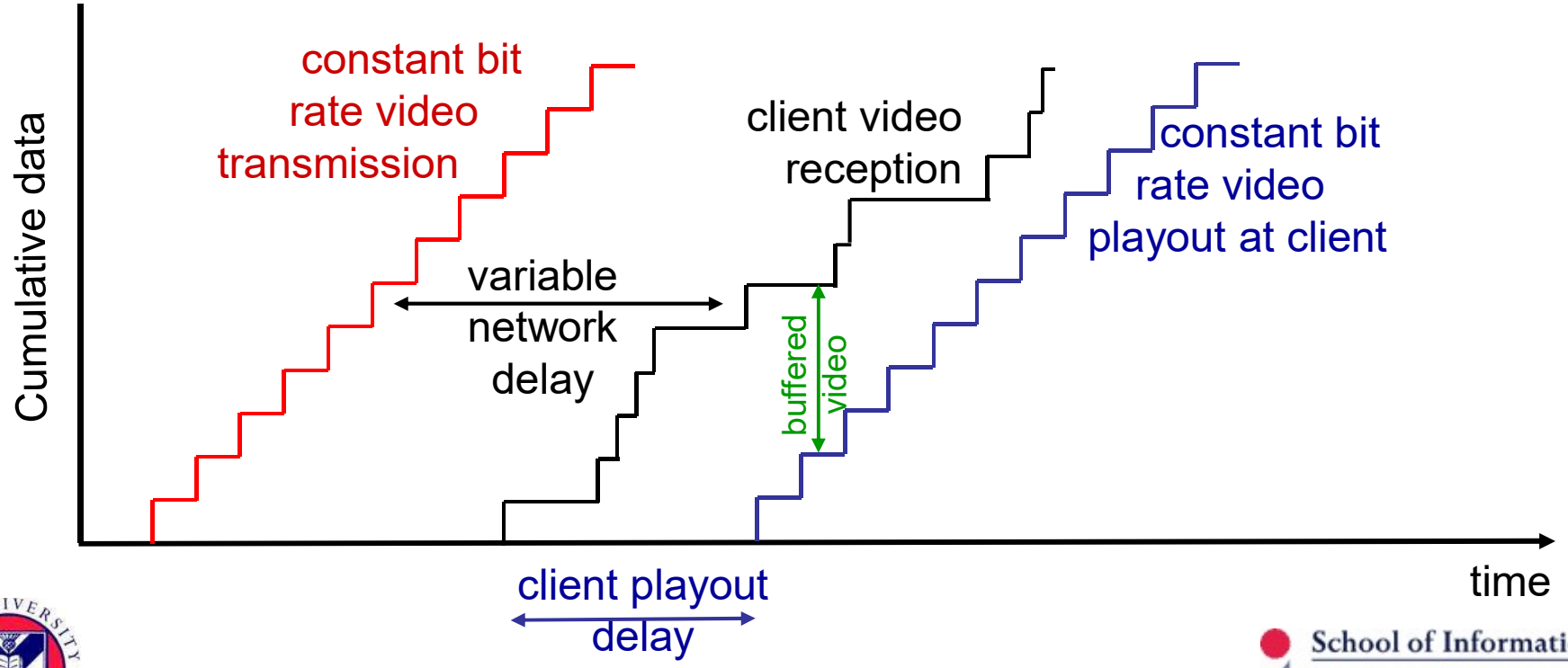
streaming: at this time, client playing out early part of video, while server still sending later part of video

Streaming Stored Video: Types and Optimisations

- Three types, depending on mode of transport:
  1. UDP streaming
  2. HTTP streaming
  3. Adaptive HTTP streaming
- Majority of current systems use (adaptive) HTTP streaming
- ***Average throughput*** is a key performance measure for streaming stored video, higher the better
  - Should be above the video rate to provide continuous playout
- Optimisation strategies
  - Client buffering and prefetching
  - Adapting video quality to available bandwidth
  - CDN based distribution

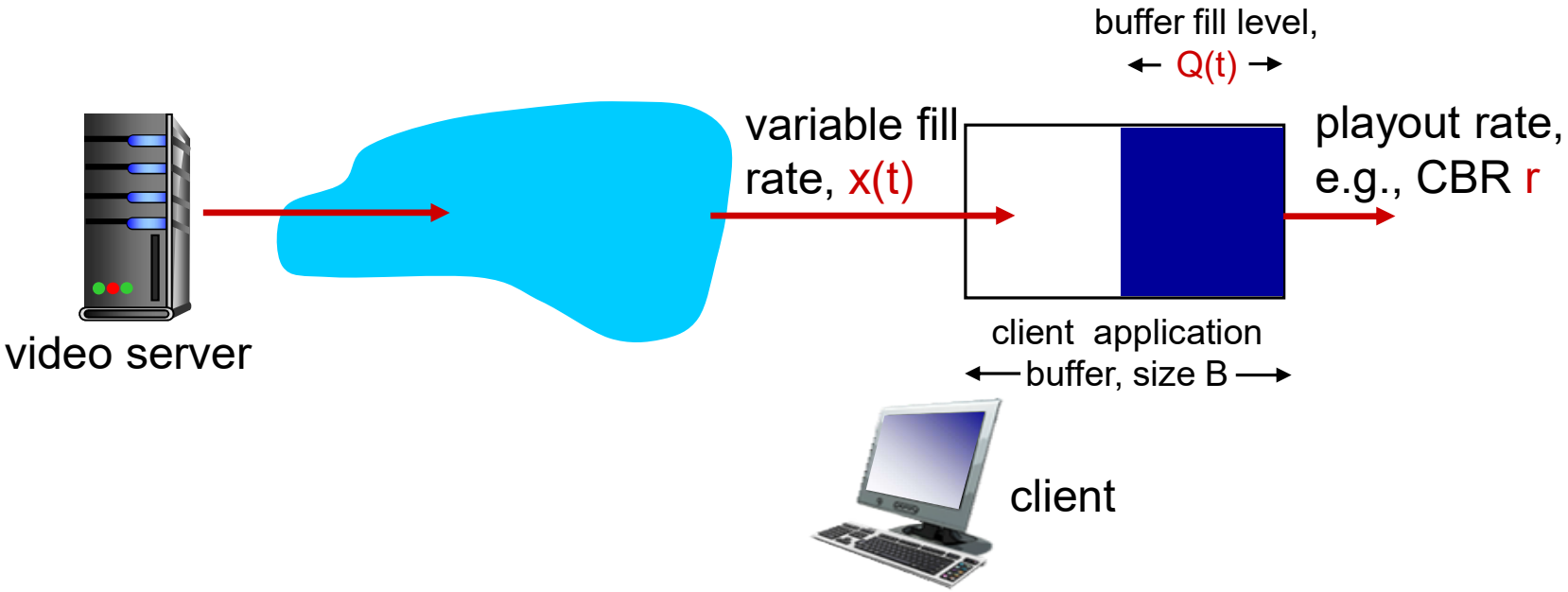
Client-Side Buffering for Streaming Video

- Common technique employed in all forms of streaming video systems to absorb network delay variability and enable continuous playout
  - Higher the delay variability, longer the playout delay needed

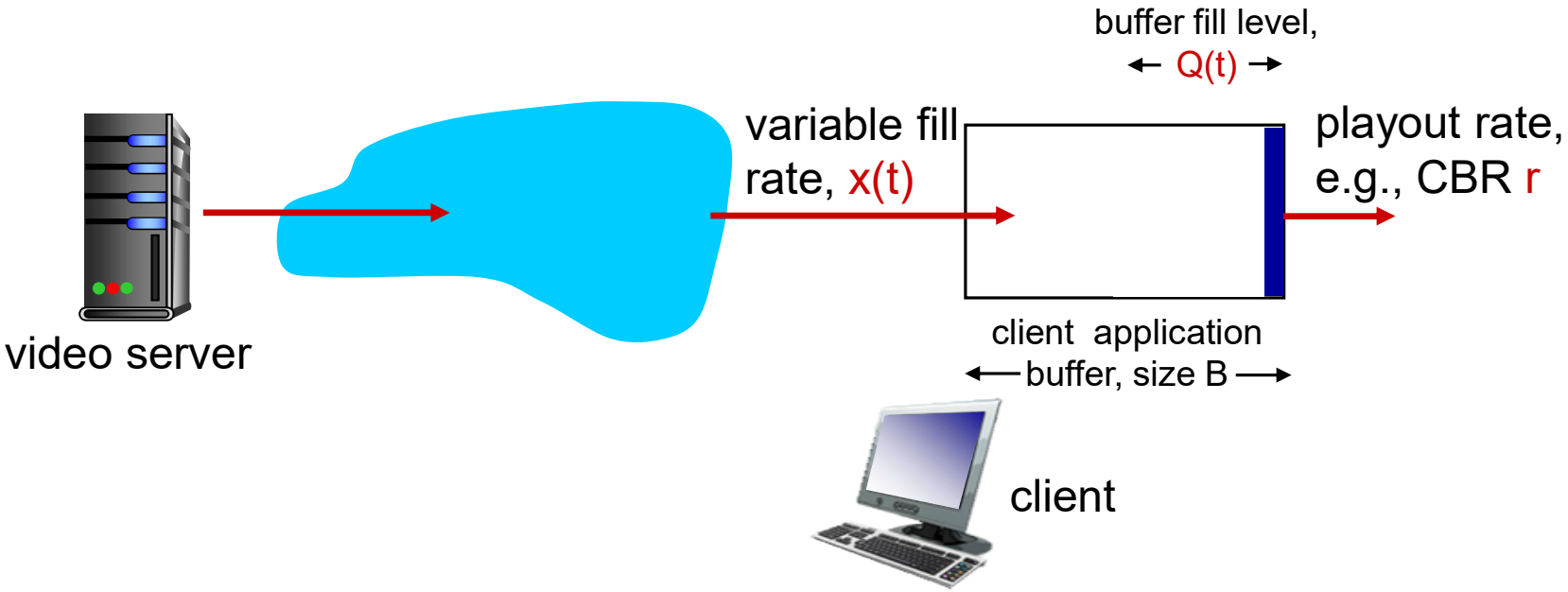


School of Informatics
Institute for Computing Systems Architecture

A Different View of Client-Side Buffering

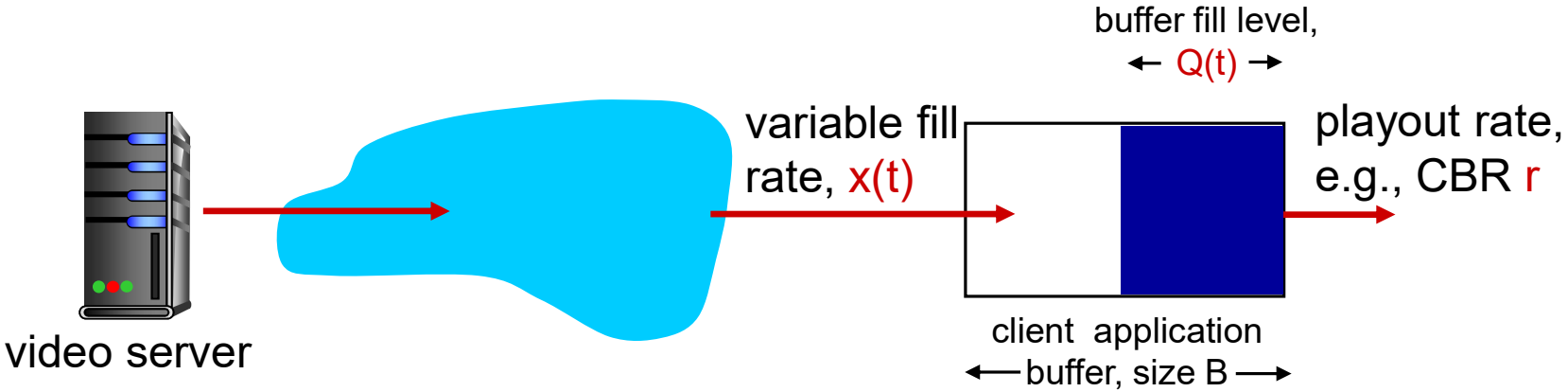


A Different View of Client-Side Buffering



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

Analysis of Client-Side Buffering for Streaming Multimedia



Key parameters: average fill rate (x_{avg}), playout rate (r)

- $x_{avg} < r$: buffer eventually empties (causing freezing of video playout until buffer again fills)
- $x_{avg} > r$: 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

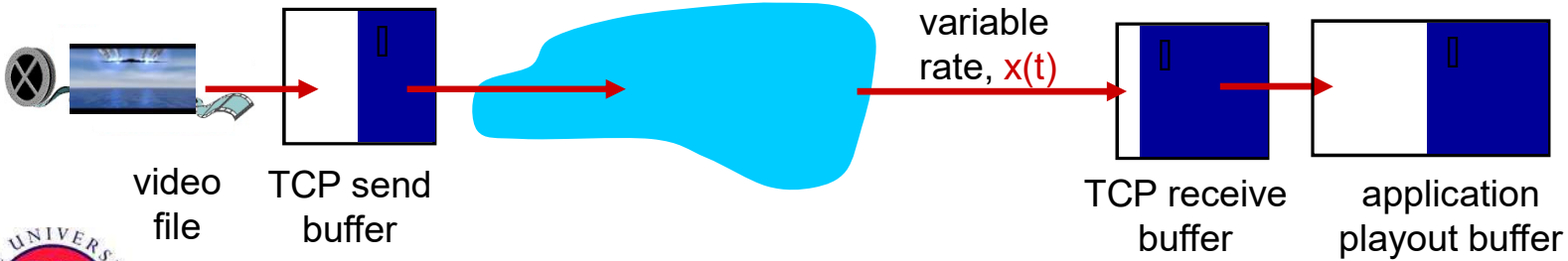
School of Informatics
Institute for Computing Systems Architecture

UDP Streaming

- Audio/video chunks are placed typically inside Real-Time Transport Protocol (RTP) or other similar protocol packets, which become the payload in UDP packets
- 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) on client-side to remove network jitter
- Error recovery: application-level, time permitting
- For interactivity, a separate parallel "control" connection to the server via Real-Time Streaming Protocol (RTSP) is used
- Limitations:
  - Constant rate streaming may not provide continuous playout
  - Requirement for media control server (e.g., RTSP server) to support interactivity
  - UDP traffic maybe blocked by many firewalls

HTTP Streaming

- Multimedia file stored at a specific URL retrieved via HTTP GET
- HTTP/TCP passes more easily through firewalls
- Can rely on HTTP byte-range header and manage without a media control server
  - Also helps in reducing inefficiency from early termination and repositioning of video
- Fill rate fluctuates due to TCP congestion control, retransmissions (in-order delivery)
- However can throughout keep sending at maximum possible rate that TCP allows
  - ➢ A form of prefetching from client perspective
- Additionally using a larger playout delay ➔ smooth TCP delivery rate



School of Informatics
Institute for Computing Systems Architecture

Adaptive HTTP Streaming via 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 via a *rate determination algorithm*
    - can choose different coding rates at different points in time (depending on available bandwidth at any given point in time)

School of Informatics
Institute for Computing Systems Architecture

Adaptive HTTP Streaming via DASH

- *"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)
- A byproduct: improved server-side scalability

Content Distribution Challenge

- Challenge: how to stream content (selected from millions of videos) to hundreds of thousands of simultaneous users?

- Option 1: single, large "mega-server"
  - single point of failure
  - point of network congestion
  - long path to distant clients
  - multiple copies of same videos sent over outgoing links

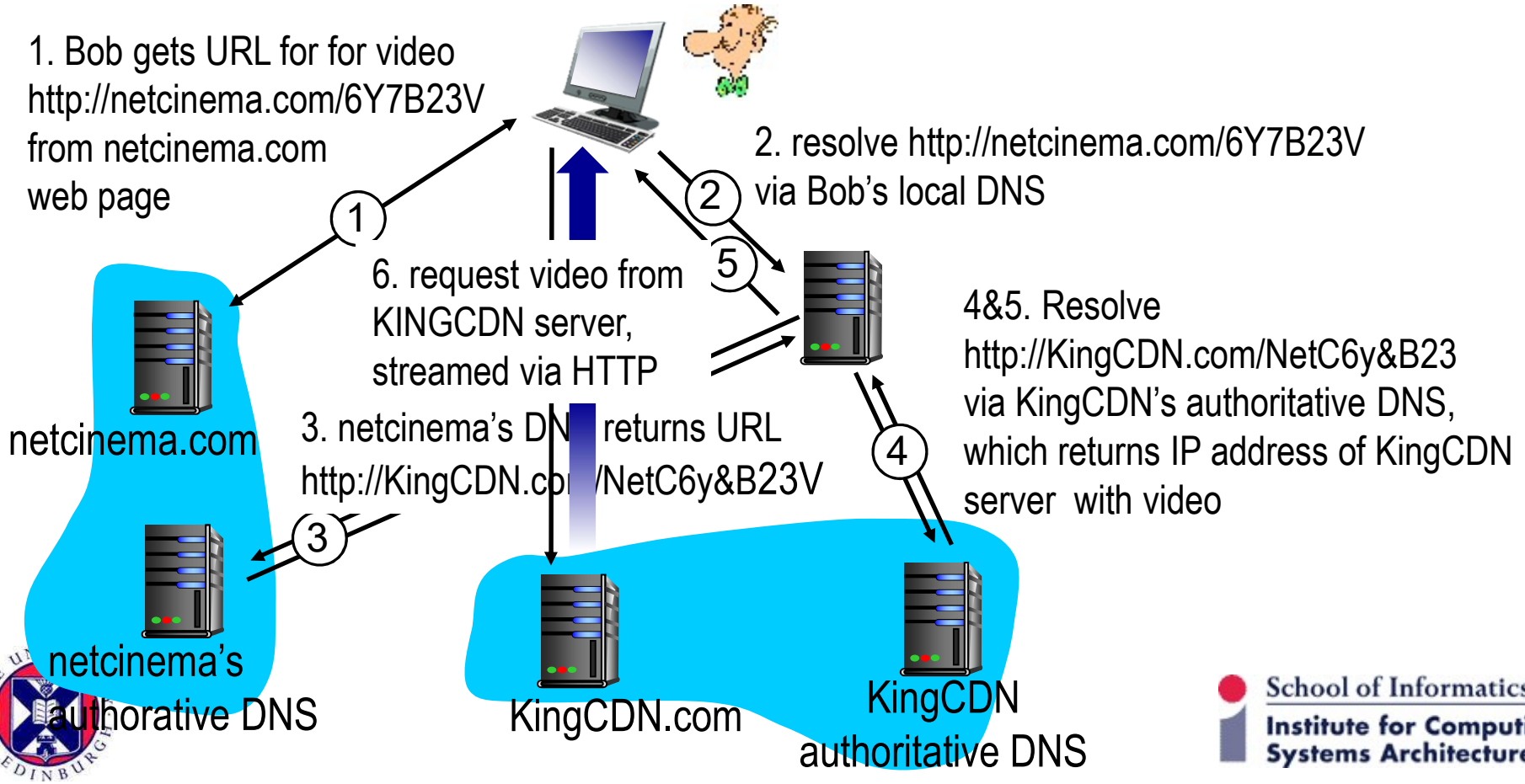
  .. quite simply: this solution *doesn't scale*

.. a better solution approach: Content Distribution Networks (CDNs)

- A CDN consists of multiple server clusters distributed geographically around the world
- Store copies of multimedia content at multiple CDN locations and serve a user request for content from the "closest" CDN location with the requested content
- Two approaches for placement of CDN server clusters
  1. *enter deep:* push CDN servers deep into many access networks
     - close to users
     - used by Akamai, 1700 locations
  2. *bring home:* smaller number (10's) of larger clusters near PoPs of many tier-1 ISPs
     - used by Limelight

Illustration of CDN Operation

- Bob (client) requests video at http://netcinema.com/6Y7B23V
  - Video stored in CDN at http://KingCDN.com/NetC6y&B23V



School of Informatics
Institute for Computing Systems Architecture

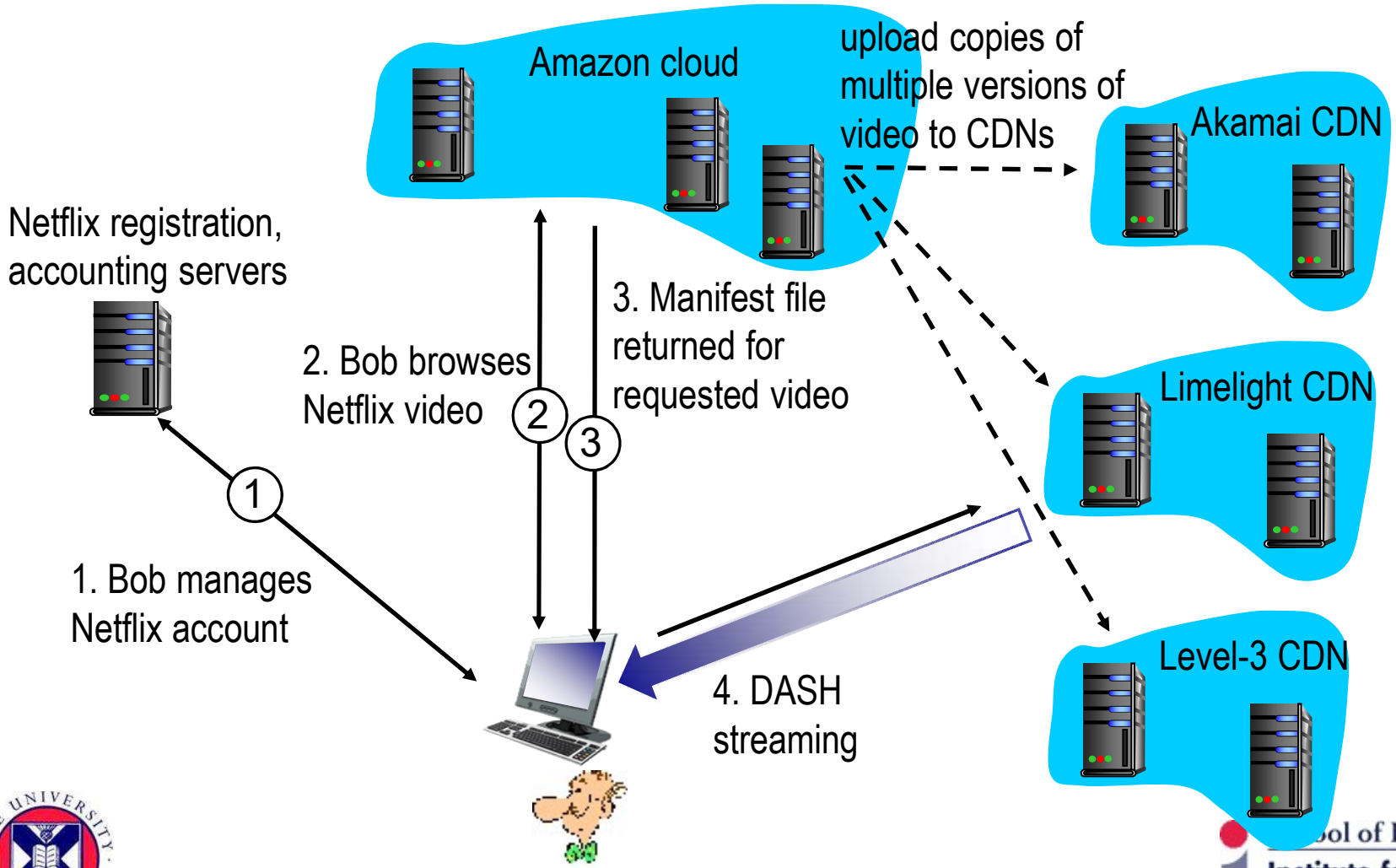
CDN Cluster Selection Strategies

- Problem: how does CDN DNS select "good" CDN cluster to stream requested content to client?
- Some possible solution strategies:
  1. Pick the CDN cluster geographically closest to client's LDNS server, several drawbacks:
     - Closest geographically does not mean closest in network terms
     - May not be a good choice with remote local DNS servers
     - Network dynamics not taken into account
  2. Measurement based selection of best CDN cluster to serve content
     - Active measurement of delay and loss through dedicated probes or by exploiting DNS query traffic
     - Passive monitoring of recent/ongoing traffic between clients and CDN servers
  3. IP anycast: does not account for network dynamics
  4. Let the client decide: give client a list of several CDN servers, client pings servers and picks "best"
     - This is the approach taken by Netflix on top of default ranking of CDNs
- Other factors to consider: load on clusters, ISP delivery cost

Case Study: Netflix

- According to a study, 30% downstream US traffic in 2011
- Owns very little infrastructure, uses 3rd party services:
  - Own registration and payment servers
  - Amazon (3rd party) cloud services:
    - Cloud hosts Netflix web pages for user browsing
    - Netflix uploads studio master to Amazon cloud
    - Create multiple versions of movie (different encodings) in cloud
    - Upload versions from cloud to CDNs
  - *Three* 3rd party CDNs host/stream Netflix content: Akamai, Limelight, Level-3

Case Study: Netflix



Streaming Live Audio and Video

- E.g., Internet radio, IPTV
- Typically live content is served to many users as with conventional radio and television broadcasting
- Distribution of live audio/video to many receivers can be done via:
  - Multiple unicast streams
  - IP multicasting
  - Application layer multicast (using P2P networks or CDNs)
- Techniques used to optimise streaming stored multimedia also help with live streaming
- As with streaming stored multimedia, average throughput greater than video rate desired
- Delay constraints more stringent than streaming stored video but less than conversational voice/video
  - Initial playout delay around 10 seconds tolerable
- Some form of forward error correction (FEC) more effective than reactive loss recovery

Conversational Voice and Video over Internet

- E.g., Skype, Google Talk
- Includes group meetings involving more than two participants
- Conversational multimedia applications are *highly delay sensitive*
- Strategies to mitigate network-induced delay, jitter and packet loss
  - Playout strategies: fixed or adaptive
  - Loss recovery schemes and error concealment: FEC, interleaving
- We will focus on **conversational voice over Internet / Internet telephony / Voice-over-IP (VoIP)**
  - Protocols: RTP, SIP

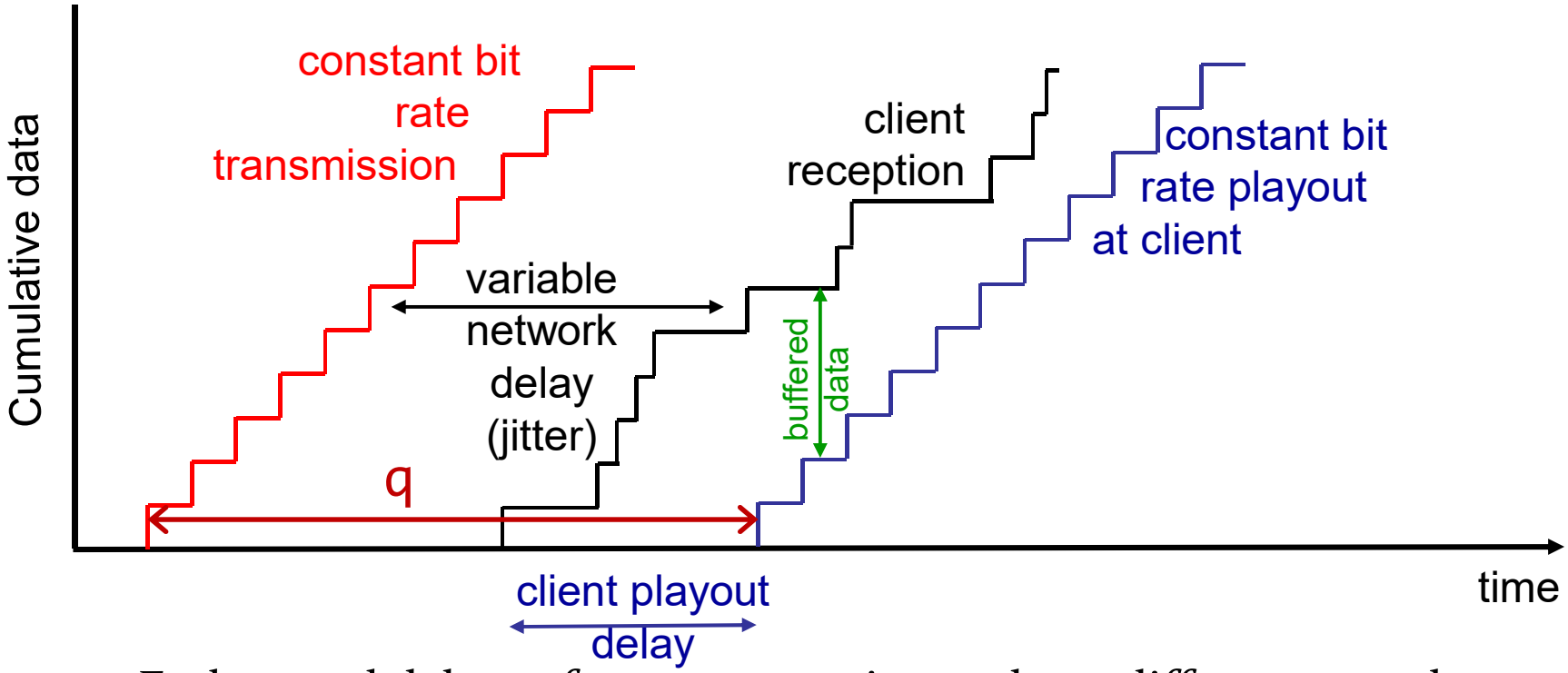
Voice-over-IP (VoIP) Applications: Characteristics / Service Requirements

- Digitised voice encapsulated in packets and transported between 2 or more VoIP call participants
- Strict "end-to-end" delay constraints:
  - End-to-end delay is the sum of:
    - propagation delay of each link on the path
    - transmission, processing and queuing delay at each router along the path (incl. the source)
    - end-system processing delays
  - ≤ 150ms ideal
  - > 400ms unacceptable

Voice-over-IP (VoIP) Applications: Characteristics / Service Requirements (contd.)

- High packet delay jitter undesirable and needs to be smoothed out
  - Delay jitter = delay variations across packets, mainly due to time-varying queuing delays
- Packet losses up to a certain extent (1-10%) tolerable
  - Overly delayed packets effectively lost from application viewpoint
  - Packet losses can also happen due to:
    - buffer overflows at congested routers
    - link transmission errors (e.g., over wireless links)
- VoIP apps use UDP by default
  - TCP can be too slow in terms of loss recovery and due to its congestion control

Delay Jitter



- End-to-end delays of two consecutive packets: difference can be more or less than difference between their transmission times (e.g., 20ms)
- Jitter effect can be mitigated via packet sequence numbers, timestamps and playout delay

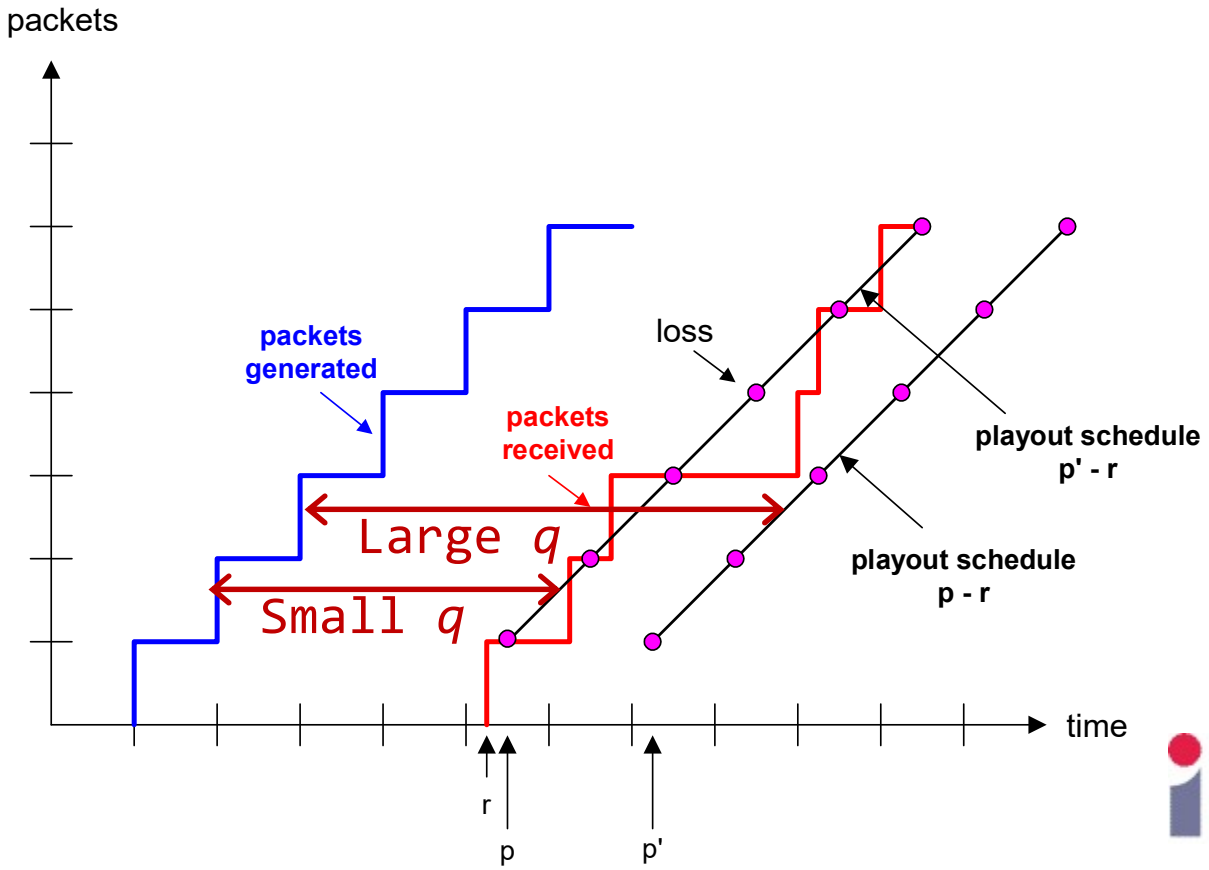
VoIP Jitter Reduction via Fixed Playout Delay

- Receiver attempts to playout each chunk exactly q time units, say ms, after its transmission
  - A chunk has a timestamp t ➔ it is played at time $t + q$
  - If the chunk arrives at receiver after $t + q$, then it is too late and treated as loss
- How to choose q?
  - Small value ➔ better interactive experience for user
  - Large value ➔ less delay-induced loss

School of Informatics
Institute for Computing Systems Architecture

Illustration of Playout Delay – Loss Tradeoff with Fixed Playout Delay Strategy

- First packet received at *r*
- Small playout delay (*p* – *r*) case: begins at *p*
- Larger playout delay (*p'* – *r*) case: begins at *p'*



Adaptive Playout Delay

- Goal: low playout delay, low delay-induced loss rate
- Approach: adaptive playout delay selection during the call via estimation of average packet delay and delay deviation from average
- Estimation of average packet delay at reception of i^{th} packet (similar to TCP RTT estimation), d_i:

$$d_i = (1-\alpha)d_{i-1} + \alpha\,(r_i - t_i)$$

- d_i: delay estimate after ith packet
- α: small constant, e.g. 0.1
- $r_i - t_i$: time received - time sent (timestamp) = measured delay of ith packet

School of Informatics
Institute for Computing Systems Architecture

Adaptive Playout Delay (contd.)

- Estimation of average deviation of delay, v_i:

$$v_i = (1-\beta)v_{i-1} + \beta\,|r_i - t_i - d_i|$$

- Estimates d_i and v_i calculated for every received packet but used only at start of talk spurt
- For first packet in a talk spurt, playout time is:

$$playout\_time_i = t_i + d_i + Kv_i$$

- Remaining packets in the talk spurt are played out at a constant rate (e.g., every 20ms)

Adaptive Playout Delay: Talk Spurt Detection

- How does the receiver determine whether packet is first in a talk spurt?
  - If no loss, receiver looks at successive timestamps
    - difference > 20ms ➔ beginning of talk spurt
  - With losses, it must look at both timestamps and sequence numbers
    - difference > 20ms plus sequence numbers without gaps ➔ beginning of talk spurt

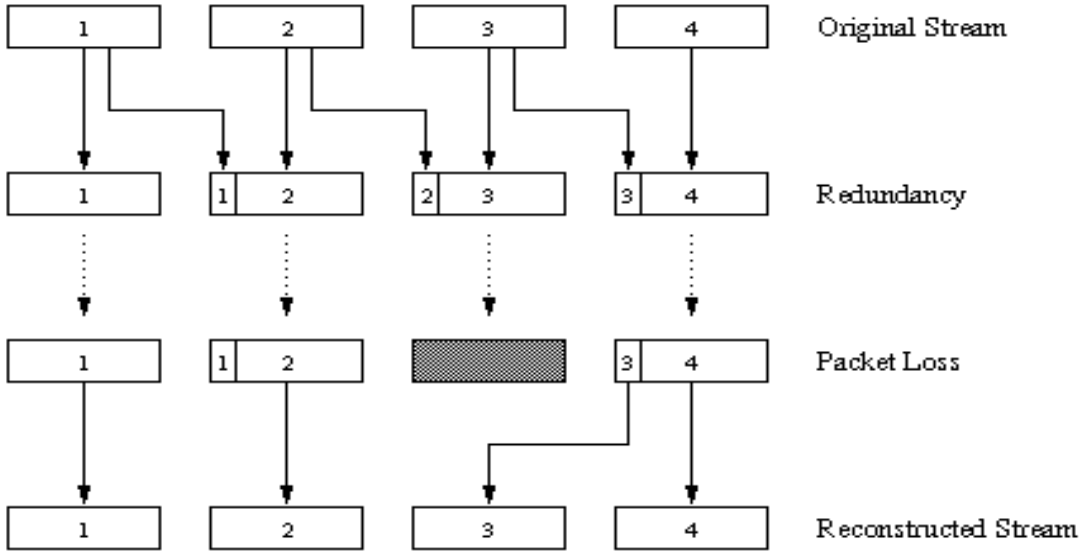
School of Informatics
Institute for Computing Systems Architecture

Proactive Packet Loss Recovery via FEC (1)

- Reactive packet loss recovery via ARQ (ACK/NACK + retransmission) takes too long to be useful
- Forward Error Correction (FEC), a proactive approach:
  - Basic idea behind FEC is to send enough additional/redundant data to allow loss recovery at receiver without requiring the sender to retransmit
- A simple FEC mechanism:
  - for every group of *n* chunks, create a redundant chunk by exclusive OR-ing *n* original chunks
  - send *n+1* chunks, increasing bandwidth by factor *1/n*
  - can reconstruct original *n* chunks if at most one chunk lost from *n+1* chunks
  - Increases the playout delay
  - Effective loss recovery with small *n* but with more overhead

Proactive Packet Loss Recovery via FEC (2)

- Another FEC mechanism: piggyback lower quality audio stream as the redundant data
  - E.g., nominal PCM encoded stream at 64kbps plus redundant GSM encoded stream at 13kbps

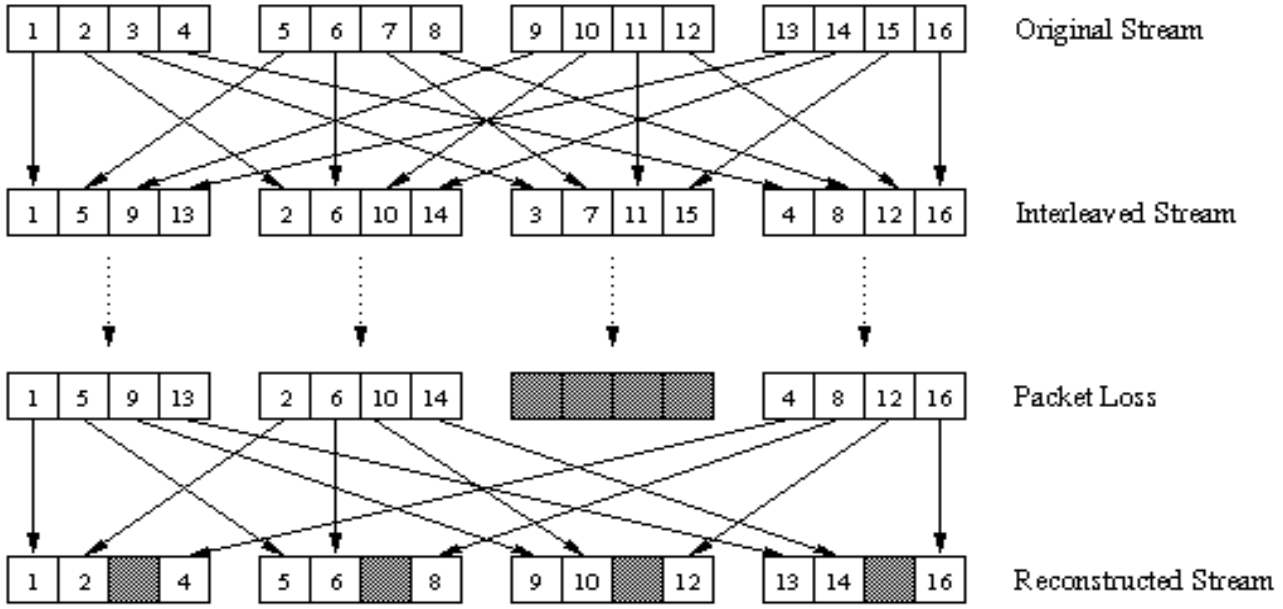


- Receiver can conceal losses if they are non-consecutive
- To cope with consecutive losses, can append multiple preceding low bit-rate chunks, e.g., $(n\text{-}1)^{st}$ and $(n\text{-}2)^{nd}$ low bit-rate chunks

School of Informatics
Institute for Computing Systems Architecture

Interleaving to Conceal Loss

- Audio chunks divided into smaller chunks, e.g., four 5ms units per 20ms audio chunk
- Each packet contains small units from different chunks



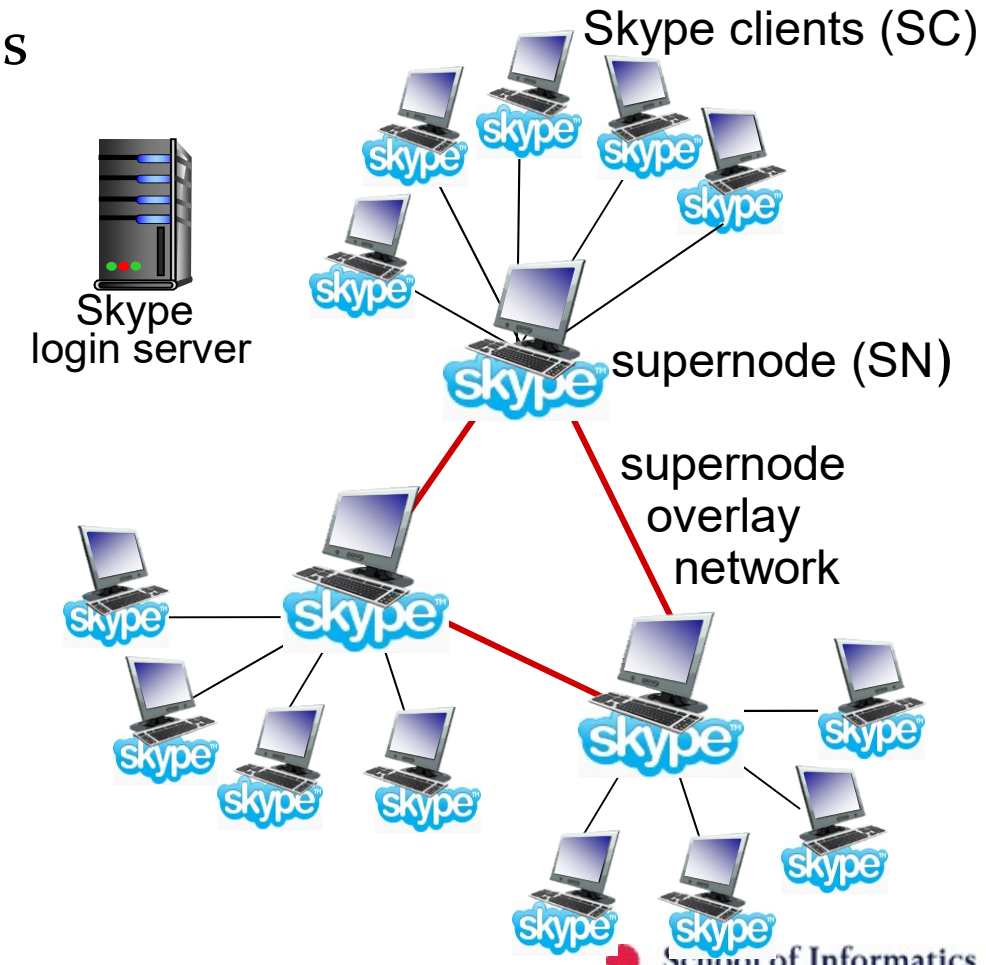
- If a packet loss, still would have most of every original chunk
- No redundancy overhead but increase in playout delay
- Other error concealment techniques: packet repetition, interpolation

Case Study: Skype

- Supports a variety of conversational voice/video over IP services: host-to-host VoIP, host-to-phone, phone-to-host, multi-party host-to-host video conferencing
- Proprietary system: what we know about it is mainly via reverse engineering by researchers using measurements
- Uses wide variety of audio and video codecs
  - Audio codecs use a higher sampling rate 16000 samples/second for better audio quality
- Control packets over TCP
- By default, audio and video packets sent over UDP
  - Uses FEC for loss recovery
- Adapts audio and video streams to current network conditions by varying audio/video bit-rate and FEC overhead

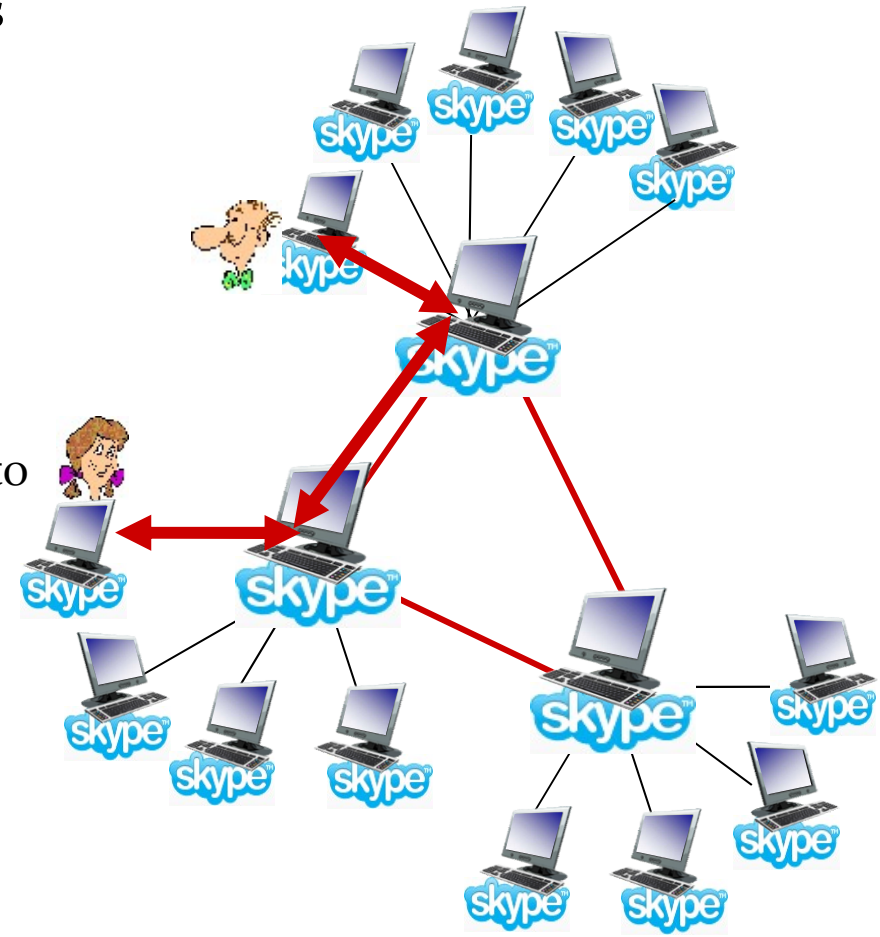
Skype is a P2P System

- A hierarchical overlay network with super nodes (SNs) and clients
- Index of mappings between usernames and current IP addresses distributed across SNs
- Uses P2P techniques for several purposes:
  - Host-to-host VoIP service, inherently P2P
  - User location by querying the distributed index
  - NAT traversal



NAT Traversal Using SNs as Relays in Skype

- Problem: both parties behind NATs
  - NAT prevents outside peer from initiating connection to a peer behind it
  - Peer behind NAT *can* however initiate connection to outside
- Solution: use SN as a relay
  - Client maintain open connections to their respective SNs
  - Caller client notifies its SN to connect to callee
  - SNs for caller and callee connect with each other
  - Another SN is identified as a relay between the two parties and both connect to that SN and engage in the call



Multi-Party Audio/Video Conference Calls with Skype

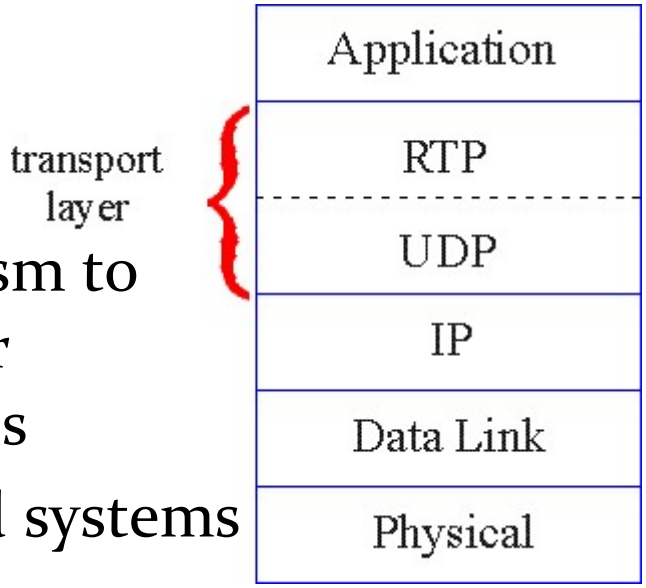
- Multi-Party = number of participants, $N > 2$
- With audio streams, the conference initiator collects streams from all parties, combines them and distributes back the combined stream
  - reduce communication overhead from sending $N(N-1)$ to $2(N-1)$ streams
- With video streams, each participant's video stream sent to a server cluster which in turn sends to the rest of the participants
  - $N(N-1)$ streams sent in total but does not stress the upstream access links which typically have much lower bandwidth than downstream links

Real-Time Protocol (RTP)

- RTP specifies packet structure for packets carrying audio, video data
- RFC 3550
- RTP packet provides
  - payload type identification
  - packet sequence numbering
  - time stamping
- RTP runs in end systems
- RTP packets encapsulated in UDP segments
- Interoperability: if two VoIP applications run RTP, they may be able to work together

RTP (contd.)

- Typically runs on top of UDP
- RTP libraries provide transport layer interface that extends UDP
  - Port numbers, IP addresses
  - **Payload type identification**
  - **Packet sequence numbering**
  - **Packet timestamps**
- RTP does *not* provide any mechanism to ensure timely data delivery or other Quality of Service (QoS) guarantees
- RTP encapsulation only seen at end systems (*not* by intermediate routers)
  - routers provide best-effort service, making no special effort to ensure that RTP packets arrive at destination in timely manner



RTP Example

Consider sending 64 kbps PCM-encoded voice over RTP

- application collects encoded data in chunks, e.g., every 20 msec = 160 bytes in a chunk
- audio chunk + RTP header form RTP packet, which is encapsulated in UDP segment

- RTP header indicates type of audio encoding in each packet
  - sender can change encoding during conference
- RTP header also contains sequence numbers, timestamps

RTP Header

- Payload type (7 bits): indicates type of encoding currently being used
  - Payload type 0: PCM mu-law, 64 kbps
  - Payload type 3: GSM, 13 kbps
  - Payload type 7: LPC, 2.4 kbps
  - Payload type 26: Motion JPEG
  - Payload type 31: H.261
  - Payload type 33: MPEG2 video
- If sender changes encoding during call, sender informs receiver via payload type field
- Sequence number (16 bits): increment by one for each RTP packet sent
  - detect packet loss, restore packet sequence

School of Informatics
Institute for Computing Systems Architecture

RTP Header

- Timestamp field (32 bits long): sampling instant of first byte in this RTP data packet
  - for audio, timestamp clock increments by one for each sampling period (e.g., each 125 µsecs for 8 KHz sampling clock)
  - if application generates chunks of 160 encoded samples, timestamp increases by 160 for each RTP packet when source is active. Timestamp clock continues to increase at constant rate even when source is inactive.

- Synchronization Source or SSRC field (32 bits long): identifies source of RTP stream. Each stream in **RTP session** has distinct SSRC

School of Informatics
Institute for Computing Systems Architecture

SIP: Session Initiation Protocol [RFC 3261]

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

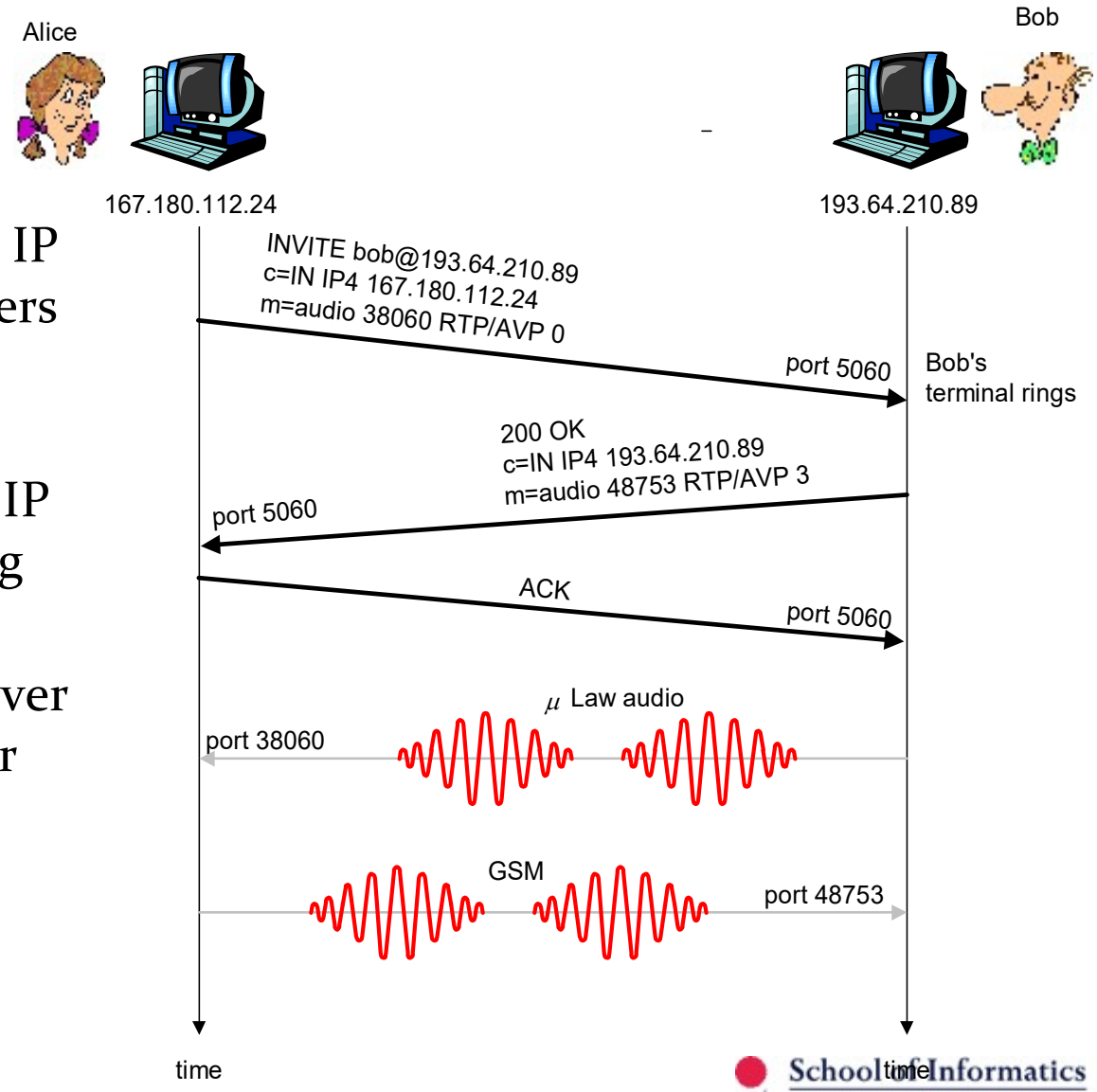
School of Informatics
Institute for Computing Systems Architecture

SIP Services

- SIP provides mechanisms for call setup:
  - for caller to let callee know he/she wants to establish a call
  - so caller, callee can agree on media type, encoding
  - to end call
- Determine current IP address of callee:
  - maps mnemonic identifier to current IP address
- Call management:
  - add new media streams during call
  - change encoding during call
  - invite others
  - transfer, hold calls

SIP Example: Setting up call to a known IP address

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



More on setting up a call with SIP

- codec negotiation:
  - suppose Bob doesn't have PCM μlaw encoder
  - Bob will instead reply with 606 Not Acceptable Reply, listing his encoders. Alice can then send new INVITE message, advertising different encoder
- rejecting a call
  - Bob can reject with replies "busy," "gone," "payment required," "forbidden"
- media can be sent over RTP or some other protocol

Example of SIP Message

```
INVITE sip:bob@domain.com SIP/2.0
Via: SIP/2.0/UDP 167.180.112.24
From: sip:alice@hereway.com
To: sip:bob@domain.com
Call-ID: a2e3a@pigeon.hereway.com
Content-Type: application/sdp
Content-Length: 885

c=IN IP4 167.180.112.24
m=audio 38060 RTP/AVP 0
```

Notes:

- HTTP message syntax
- sdp = session description protocol
- Call-ID is unique for every call

- Here we don't know Bob's IP address
  - Intermediate SIP servers needed
- Alice sends, receives SIP messages using SIP default port 5060
- Alice specifies in header that SIP client sends, receives SIP messages over UDP

Name Translation and User Location

- Caller wants to call callee, but only has callee's name or e-mail address.
- Need to get IP address of callee's current host:
  - user moves around
  - DHCP protocol
  - user has different IP devices (PC, smartphone, car device)
- Result can be based on:
  - time of day (work, home)
  - caller (don't want boss to call you at home)
  - status of callee (calls sent to voicemail when callee is already talking to someone)

SIP Registrar

- One function of SIP server: registrar
- When Bob starts SIP client, client sends SIP REGISTER message to Bob's registrar server

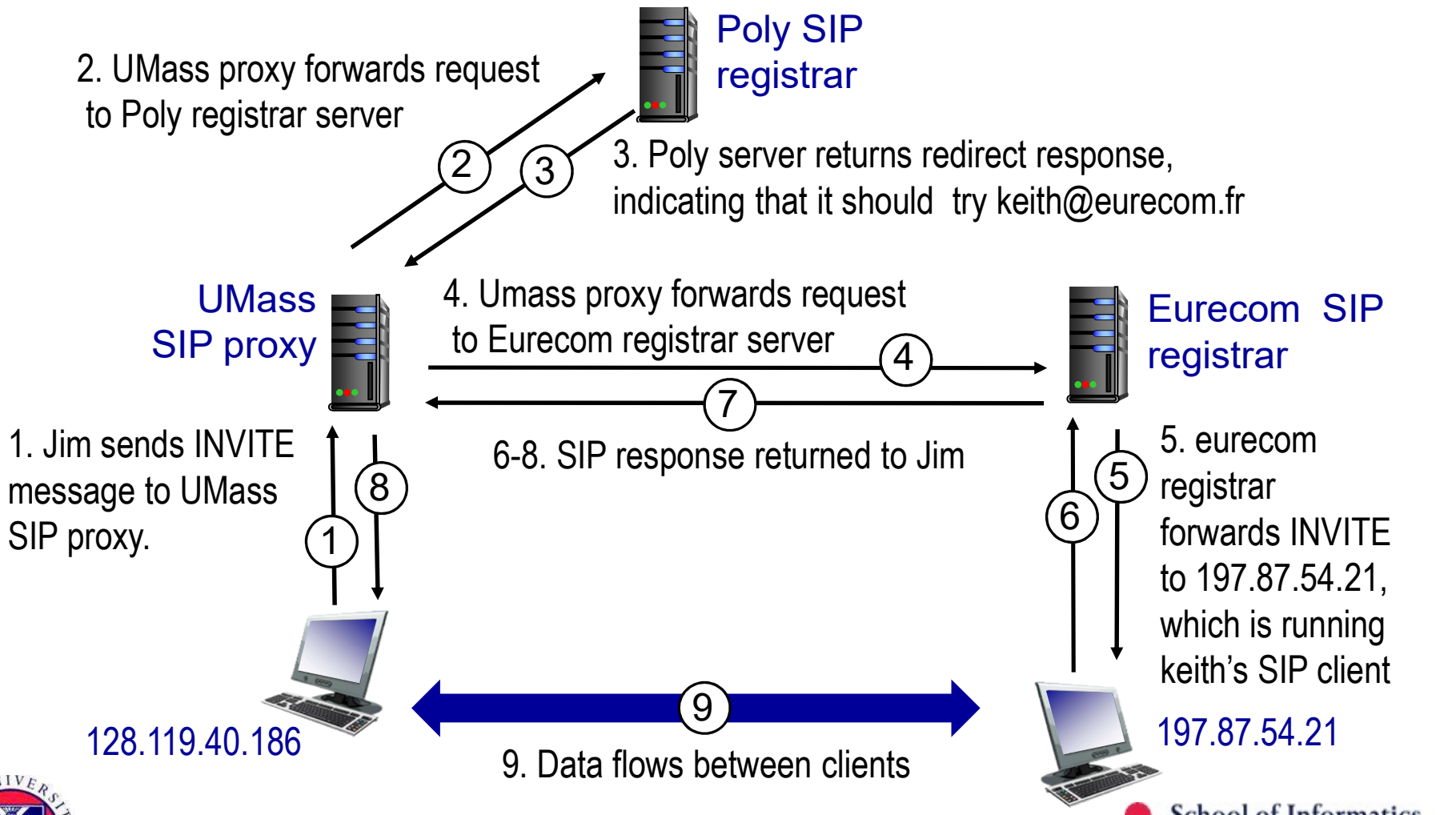
SIP REGISTER message:

```
REGISTER sip:domain.com SIP/2.0
Via: SIP/2.0/UDP 193.64.210.89
From: sip:bob@domain.com
To: sip:bob@domain.com
Expires: 3600
```

SIP Proxy

- Another function of SIP server: proxy
- Alice sends invite message to her proxy server
  - contains address sip:bob@domain.com
  - proxy responsible for routing SIP messages to callee, possibly through multiple proxies
- Bob sends response back through same set of SIP proxies
- Proxy returns Bob's SIP response message to Alice
  - Contains Bob's IP address

SIP Example: jim@umass.edu calls keith@poly.edu



Comparison with H.323

H.323: another signaling protocol for real-time, interactive multimedia

- Complete, vertically integrated suite of protocols for multimedia conferencing: signaling, registration, admission control, transport, codecs
- Comes from the ITU (telephony)

SIP: embraces simplicity

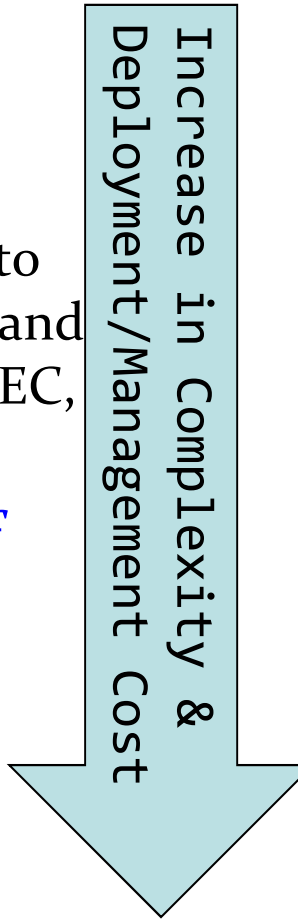
- Single component. Works with RTP, but does not mandate it. Can be combined with other protocols, services
- Comes from IETF: borrows much of its concepts from HTTP

Network Support for Multimedia

School of Informatics
Institute for Computing Systems Architecture

Three Broad Approaches

1. The default Internet best effort service, ideally over a suitably dimensioned network, with application layer support overlaid on top
   - Application layer mechanisms: client buffering and fixed/adaptive playout, prefetching, adapting media quality to available bandwidth, content distribution networks (CDNs) and P2P overlays, loss recovery and concealment mechanisms (FEC, interleaving)
2. Differentiated service for soft/probabilistic Quality of Service (QoS) guarantees at the traffic class level granularity
   - QoS mechanisms needed: packet marking, traffic policing, efficient and fair scheduling
3. Per-connection QoS guarantees for hard guarantees
   - QoS mechanisms needed: call admission, resource reservation



Dimensioning Best-Effort Networks

- Resource contention is a key source of the problem
  - Queuing at routers causes high/variable end-to-end delays and packet losses, and these in turn can reduce a multimedia flow's throughput
- So throw more resources at the problem (i.e., redundant topology and abundant link capacities) such that network congestion rarely happens
- Since this has a monetary cost and requires cooperation of multiple ISPs, this approach is hindered by economic and organizational concerns in practice
- Let's set them aside and look (briefly) at bandwidth provisioning and network dimensioning

Dimensioning Best-Effort Networks (2)

- **Bandwidth provisioning**: Given a topology, what should be the capacity of network links to achieve a desirable level of performance?
- **Network dimensioning** (subsumes the bandwidth provisioning problem): how to design a network topology to achieve a desirable level of end-to-end performance?
- Need the following to address bandwidth provisioning and network dimensioning
  - Models of traffic demand between network end points
  - Well-defined performance requirements
  - Models to predict end-to-end performance for a given workload model, and techniques to find a minimal cost bandwidth allocation that will result in all user requirements being met

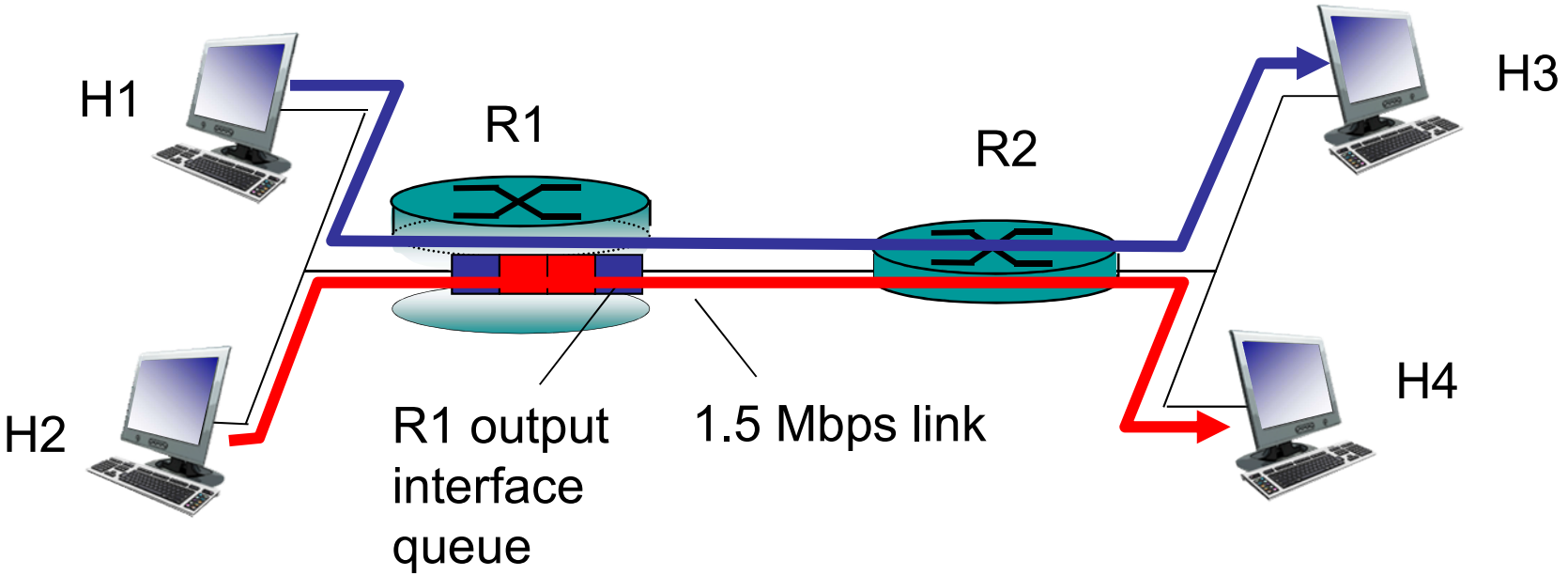
Beyond Best-Effort: Providing Soft/Hard Quality of Service (QoS) Guarantees

School of Informatics
Institute for Computing Systems Architecture

Soft QoS Guarantees via Differentiated Service: Providing Multiple Classes of Service

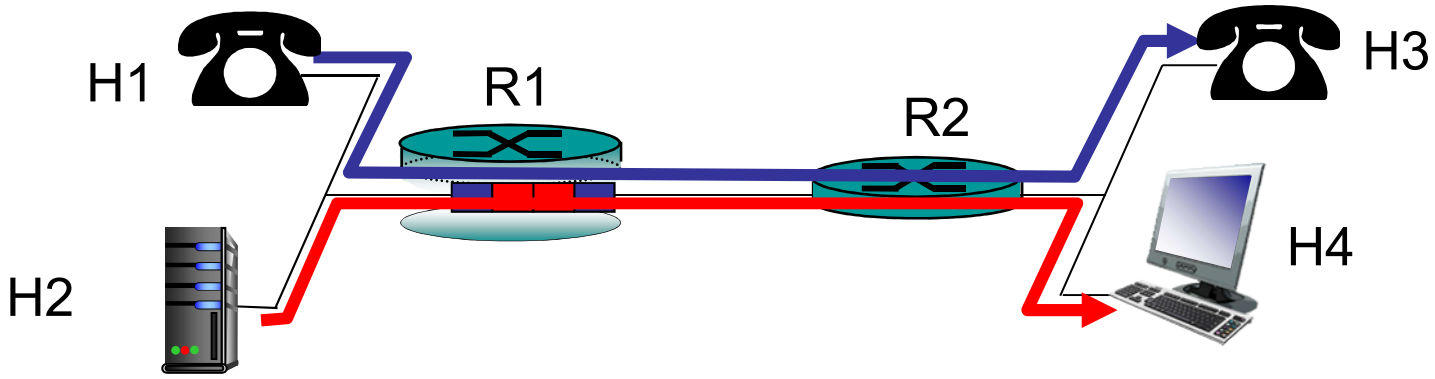
- Divide traffic into classes and provide different levels of service to different traffic classes
- Applied over aggregated traffic of each class
  - Not at individual packet or flow level
  - All traffic from the same class treated equally
- Can be done to differentiate:
  - Traffic from different applications with different service requirements (e.g., VoIP vs. HTTP)
  - High paying subscribers from the rest
- Real world analogies: airline and train fares (first class vs. economy)
- Old idea dating back to the late seventies

Example Scenario to Illustrate QoS Mechanisms Needed for Differentiated Service



Packet Marking

- Suppose: 1Mbps VoIP (between H1 and H3), HTTP download from H2 to H4, both flows share 1.5 Mbps link at R1
  - HTTP bursts can congest router, cause audio loss
  - Need to give priority to audio over HTTP

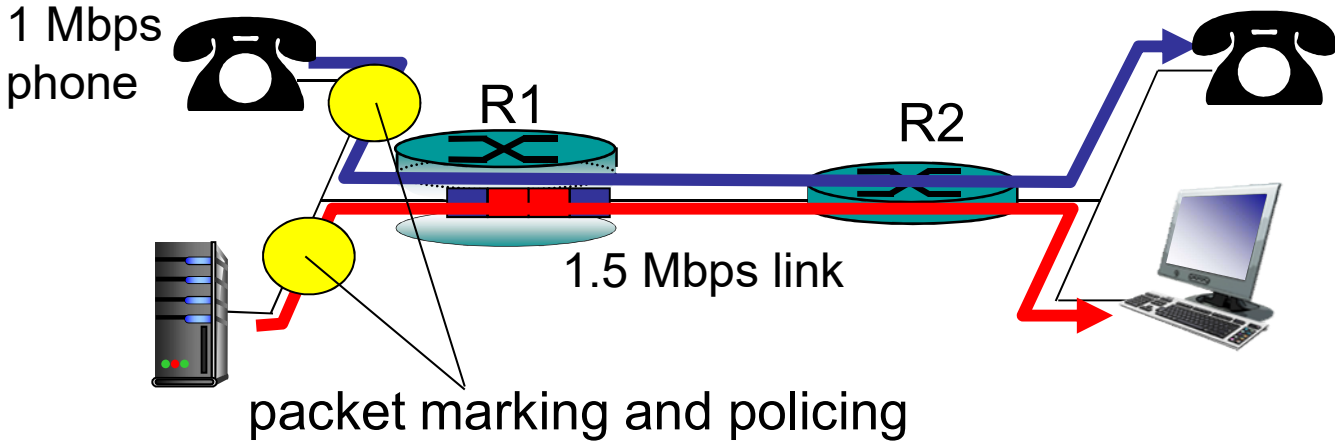


Packet Marking

Packet marking needed for router to distinguish between different classes; and new router policy to treat packets accordingly

Traffic Isolation

- What if applications misbehave (VoIP sends higher than declared rate)
  - Policing to force source adherence to bandwidth allocations
- Marking *and* policing at host or network edge router

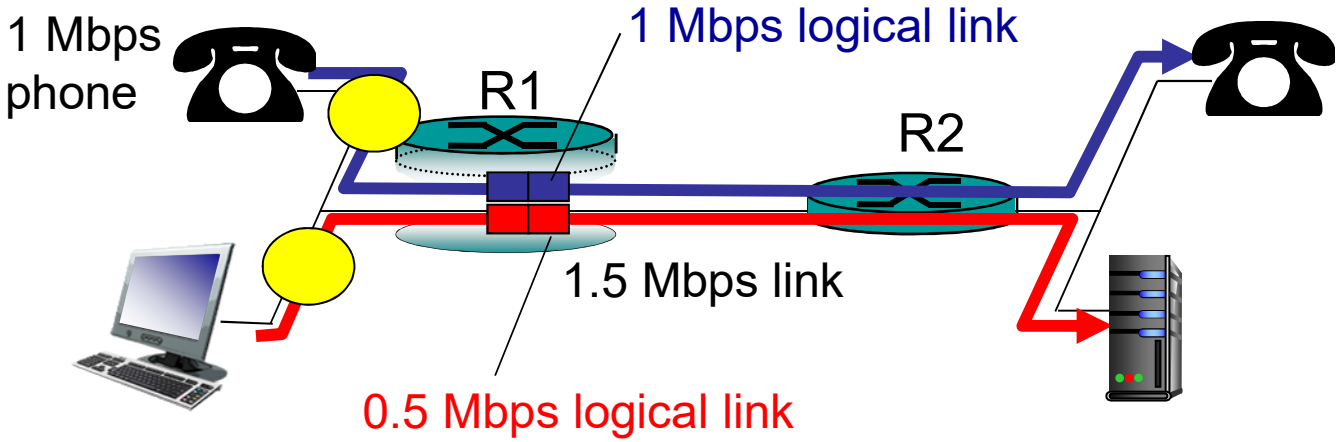


Traffic Isolation

Provide protection (isolation) for one class from others

Efficient and Fair Link Scheduling

- Allocating *fixed* (non-sharable) bandwidth to a flow leads to *inefficient* use of bandwidth if the flow does not use its allocation



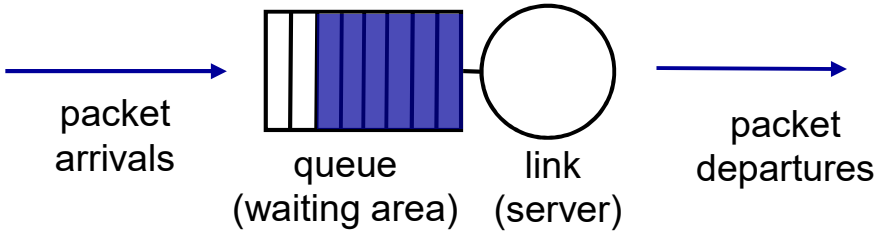
Efficient and Fair Scheduling

While providing isolation, it is desirable to use resources as efficiently as possible

School of Informatics
Institute for Computing Systems Architecture

Link Scheduling Schemes

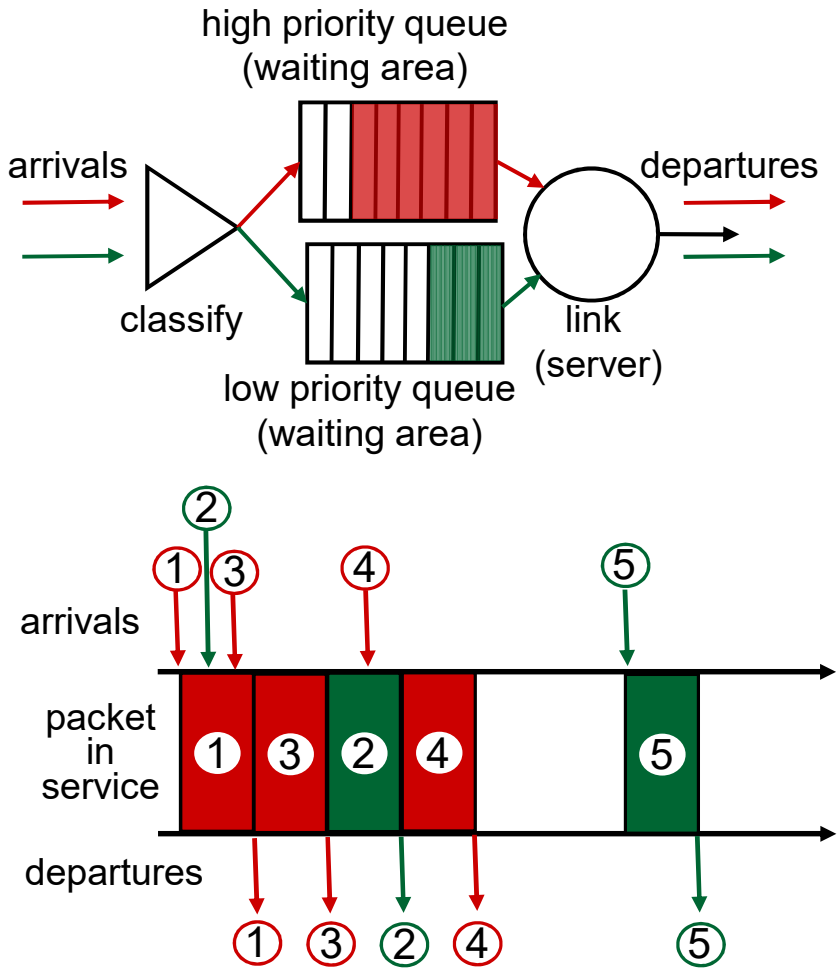
- *Scheduling:* choose next packet to send on link
- *FIFO (first in first out) or FCFS (first come first serve) Scheduling:* send in order of arrival to queue
  - *Discard Policy:* if packet arrives to full queue: who to discard?
    - *tail drop:* drop arriving packet
    - *oldest:* drop packet that has been waiting in the queue the longest
    - *priority:* drop/remove on priority basis
    - *random:* drop/remove randomly



School of Informatics
Institute for Computing Systems Architecture

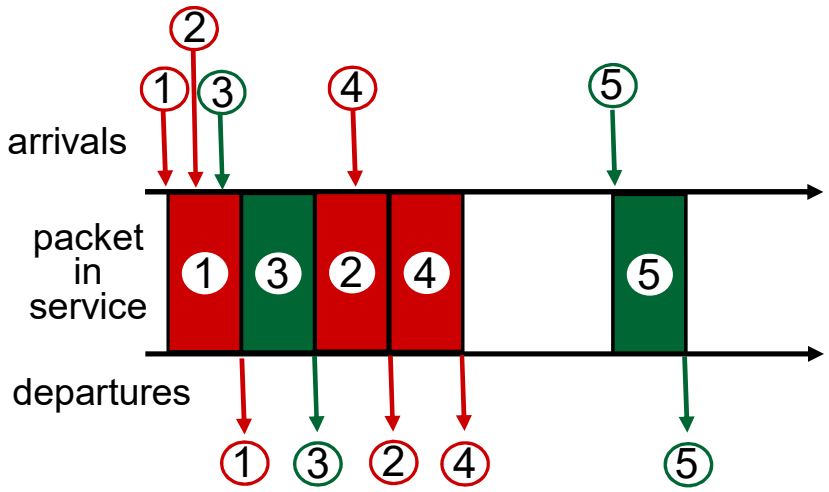
Link Scheduling Schemes (2)

- *Priority Queuing:* send highest priority queued packet
- Multiple *classes* with different priorities
  - Class may depend on marking or other header info, e.g. IP source/dest, port numbers, etc.



Link Scheduling Schemes (3)

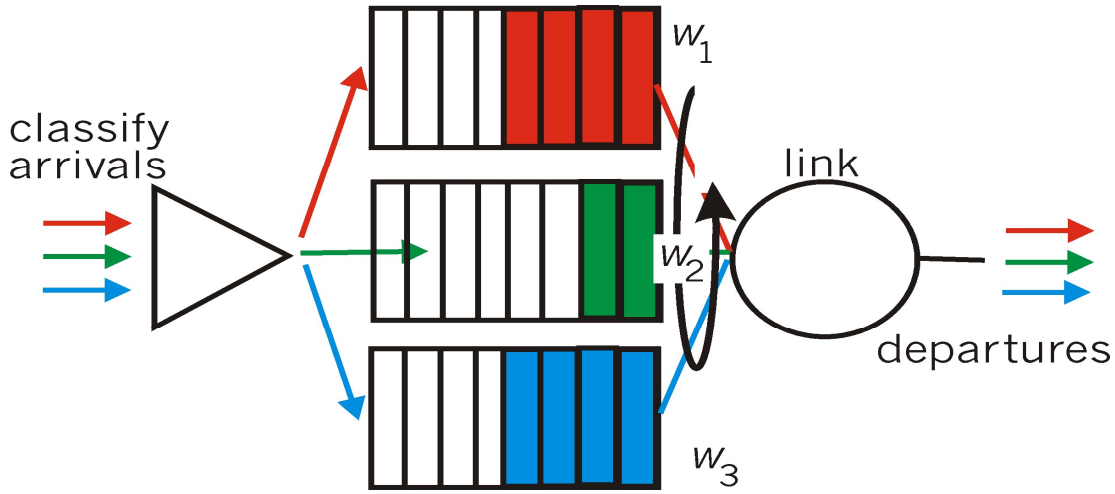
- *Round Robin (RR) Scheduling:*
  - multiple classes
  - cyclically scan class queues, sending one complete packet from each class (if available)



Link Scheduling Schemes (4)

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

    $R \cdot w_i / (\sum w_j)$, where R is the link capacity



Traffic Policing

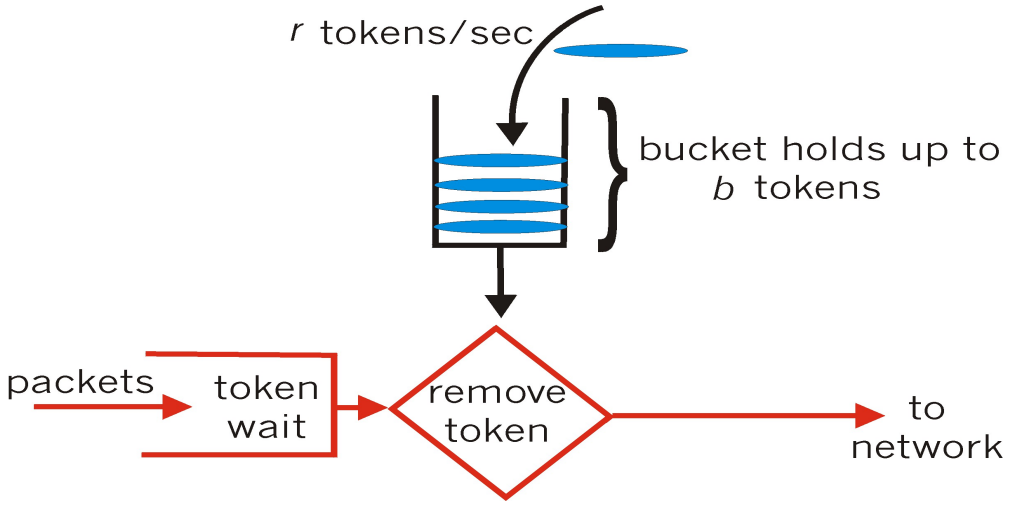
Goal: limit traffic to not exceed declared parameters

Three commonly used parameters:

- *(long term) average rate:* how many packets can be sent per unit time (in the long run)
  - crucial question: what is the interval length?
    - 100 packets per sec or 6000 packets per min have same average!
- *peak rate:* e.g., 6000 packets per min (ppm) average and 1500 pps peak rate
- *(max.) burst size:* max number of packets sent consecutively (with no intervening idle)

The Leaky Bucket Mechanism

Token bucket: limits input to specified *average rate* and *burst size*

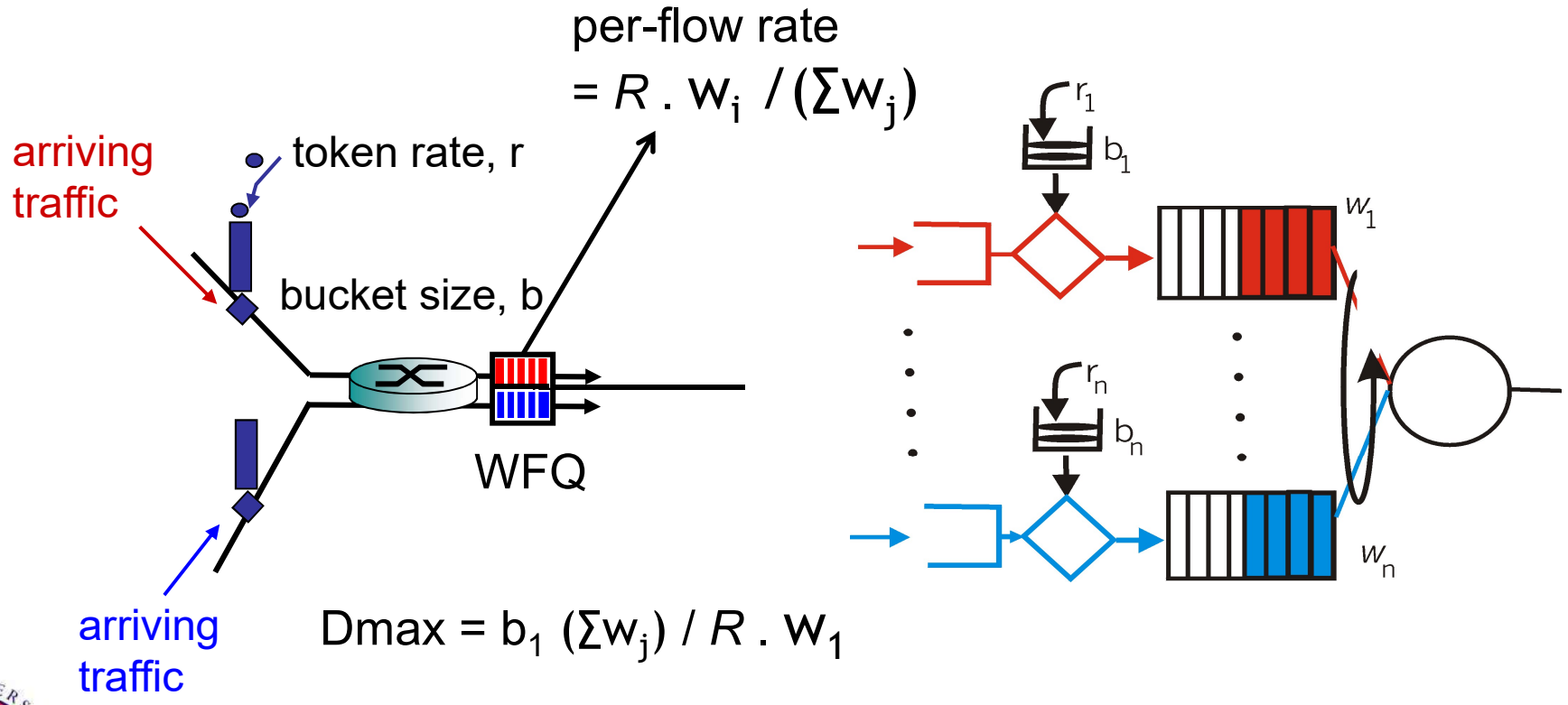


- Bucket can hold b tokens
- Tokens generated at rate r tokens/sec unless bucket full
- Over an interval of length t, the number of packets admitted $\leq r\,t + b$

School of Informatics
Institute for Computing Systems Architecture

Use of Link Scheduling and Policing Mechanisms to achieve QoS Guarantees

- Example: combination of leaky bucket and WFQ to place an upper bound on delay, a QoS guarantee



Internet Diffserv Architecture

- Scalability a key design goal since routers in the network core may need to handle millions of flows

- This goal met by:
  - placing only simple functionality in the network core
  - more complex control operations implemented at the network edge

- Does not define service classes, just provides functional components to build service classes

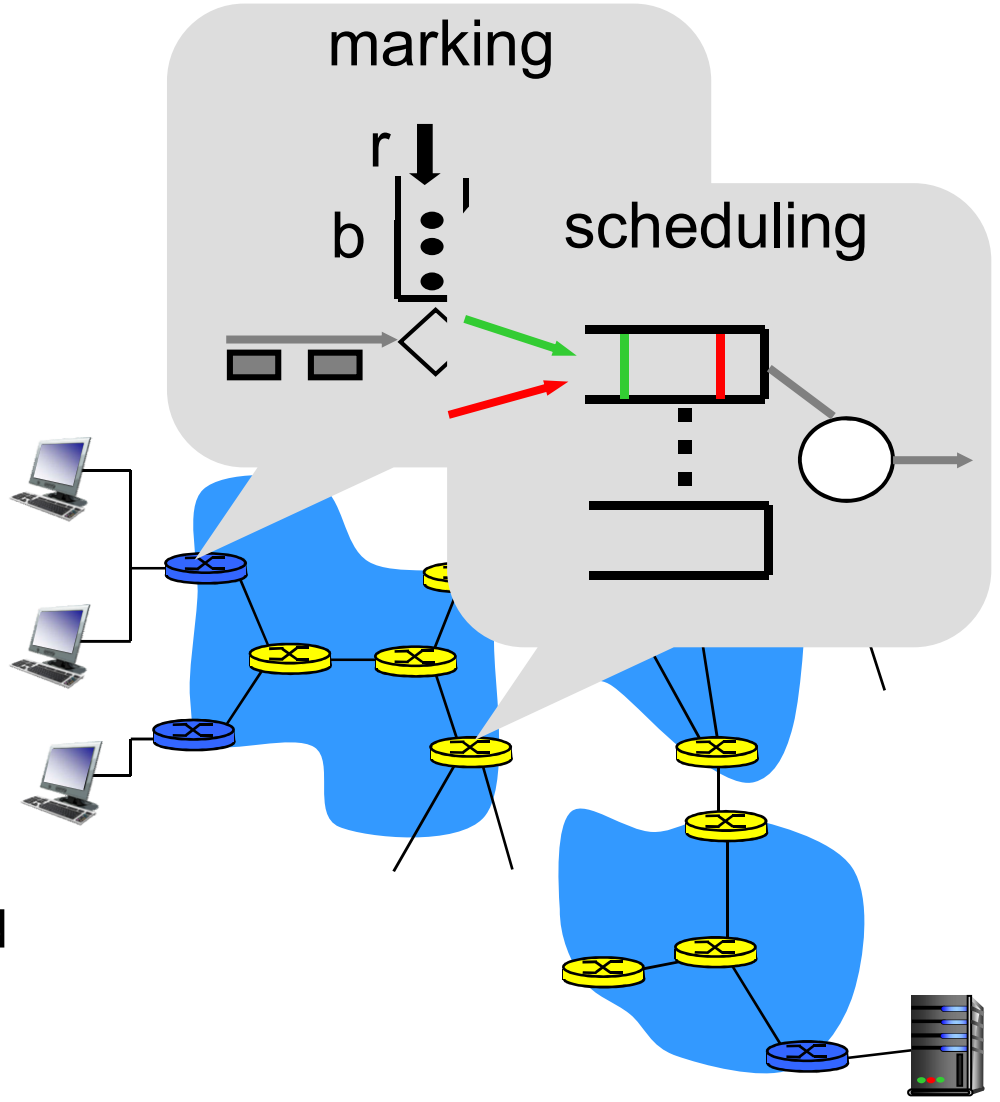
Internet Diffserv Architecture (2)

edge router:

- per-flow traffic management
- marks packets as in-profile and out-profile

core router:

- per class traffic management
- buffering and scheduling based on marking at edge
- preference given to in-profile packets over out-of-profile packets



Diffserv Packet Marking: Details

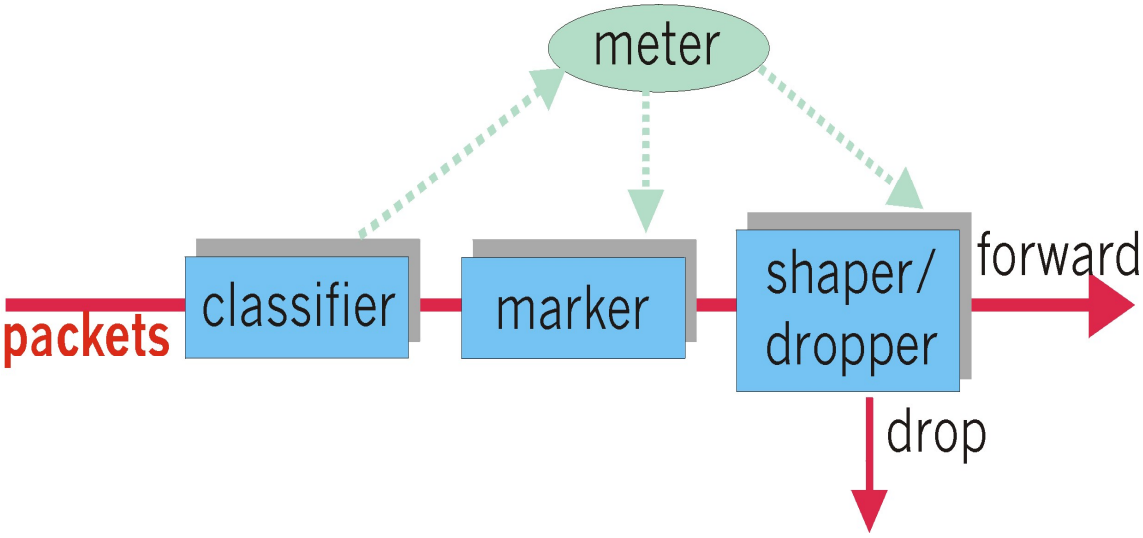
- Packet is marked using the modified Type of Service (TOS) field in IPv4, and Traffic Class in IPv6
- 6 bits used for Differentiated Service Code Point (DSCP)
  - determine Per-Hop Behavior (PHB) that the packet will receive
- 2 bits used for Explicit Congestion Notification (ECN)

School of Informatics
Institute for Computing Systems Architecture

Diffserv Edge Functions Overview

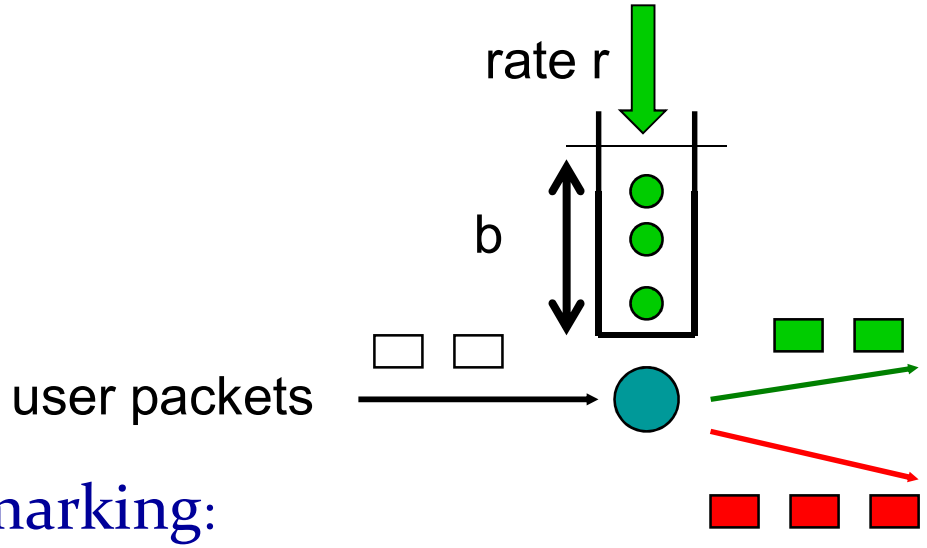
May be desirable to limit traffic injection rate of some class:

- user declares traffic profile (e.g., rate, burst size)
- traffic metered, shaped if non-conforming



Edge-Router Packet 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

School of Informatics
Institute for Computing Systems Architecture

Forwarding Per-Hop Behavior (PHB)

- PHB result in a different *observable (measurable)* forwarding performance behavior
- PHB does *not* specify what mechanisms to use to ensure required PHB performance behavior
- Examples:
  - class A gets x% of outgoing link bandwidth over time intervals of a specified length
  - class A packets leave first before packets from class B

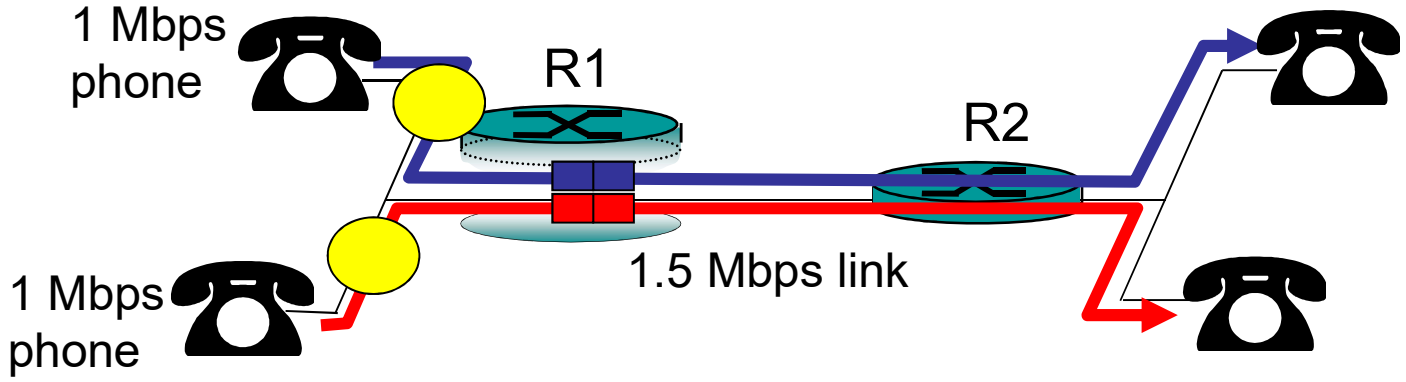
Forwarding PHB

Two main types of PHBs proposed:

- *expedited forwarding (EF):* given strict priority over other traffic classes
  - suitable for voice, video and other real-time services
  - EF traffic limited to no more than 30% of link capacity
- *assured forwarding (AF):* 4 classes of traffic
  - each guaranteed minimum amount of bandwidth
  - each with three drop preference partitions

Per-Connection and Hard QoS Guarantees

- Hard real constraint: cannot support traffic demands beyond link capacity
  - not respecting this constraint leads to violation of QoS guarantees

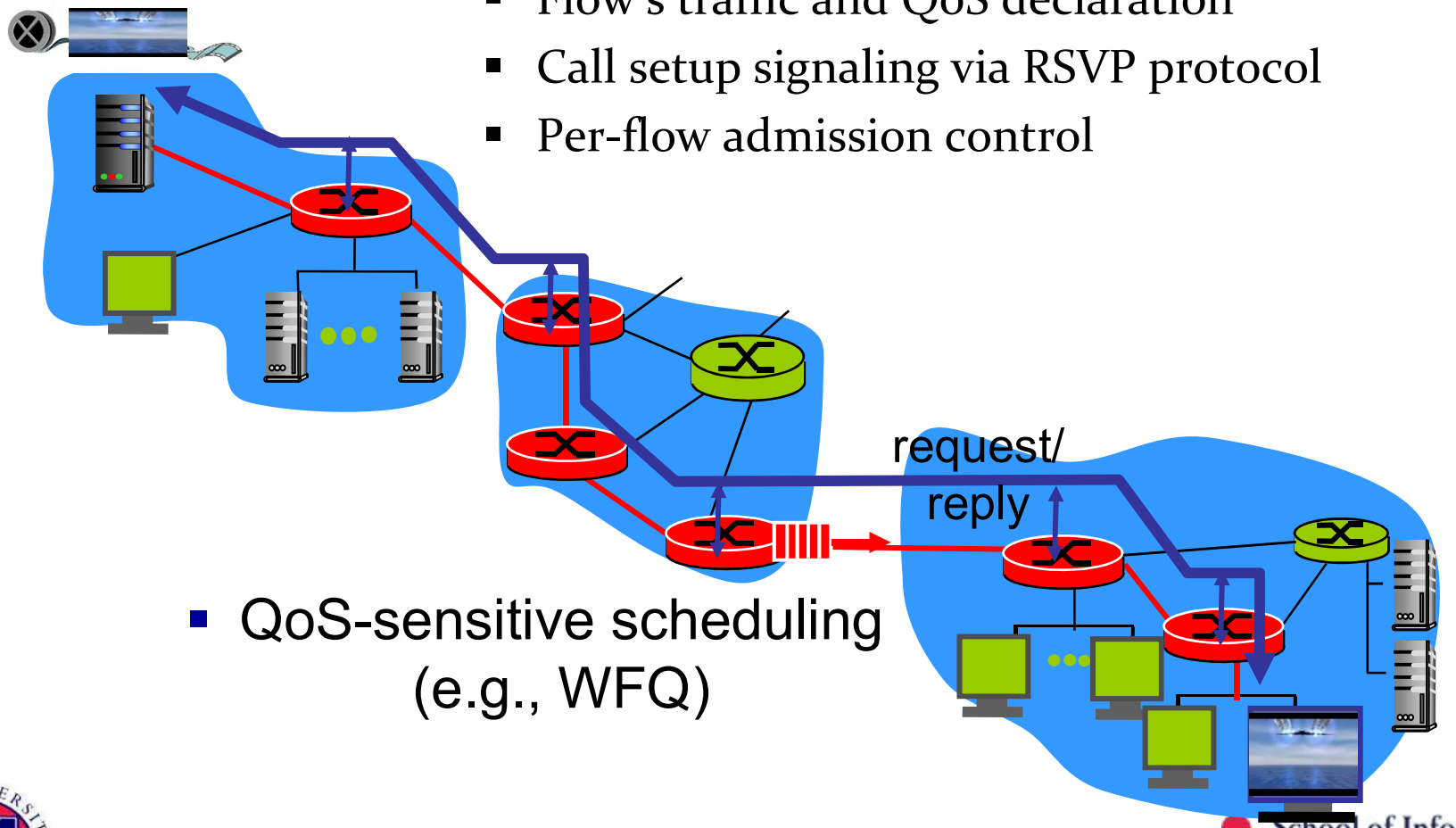


Call Admission

With call admission, flow declares its needs; network may block the flow (call) if it cannot meet flow's needs

Resource Reservation

- Key to providing hard QoS guarantees and to decide whether to admit/block a call
  - Flow's traffic and QoS declaration
  - Call setup signaling via RSVP protocol
  - Per-flow admission control



- QoS-sensitive scheduling (e.g., WFQ)