Outline

- **Service landscape**
- **Example services**
  - Including location-based services, IoT, and increasing personalization ...
- **Service models and delivery architectures**
  - Client-server, p2p, peer-assisted, one-to-many, ...
  - E.g., middleboxes/proxies, CDNs, cloud-based, SDN, multicast/broadcast (including loss recovery at AL and eMBMS)
- **Example services**
  - E.g., mobile web and HAS/DASH streaming

Service/company landscape include

Applications (3)

- World Wide Web (WWW)
- Instant Messaging (Internet chat, text messaging on cellular phones)
- File sharing
- Internet phone/video (Voice/video-Over-IP)
- Video-on-demand
- Distributed Games
- …
- (endless list)

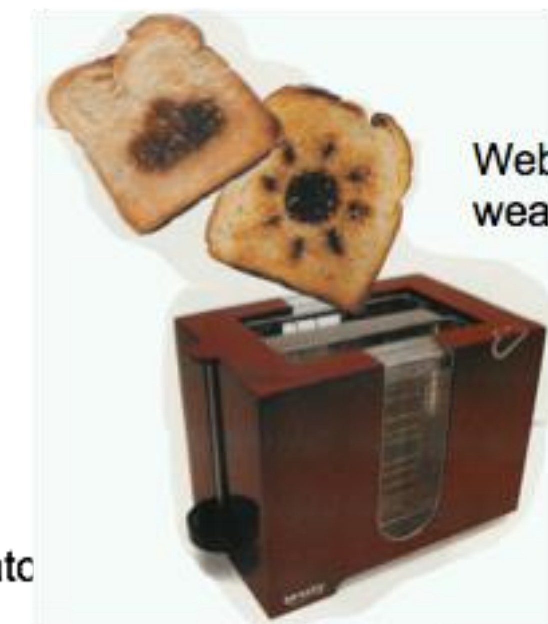
Today's end hosts …

Today's end hosts …

… well, already have …

Today's end hosts …

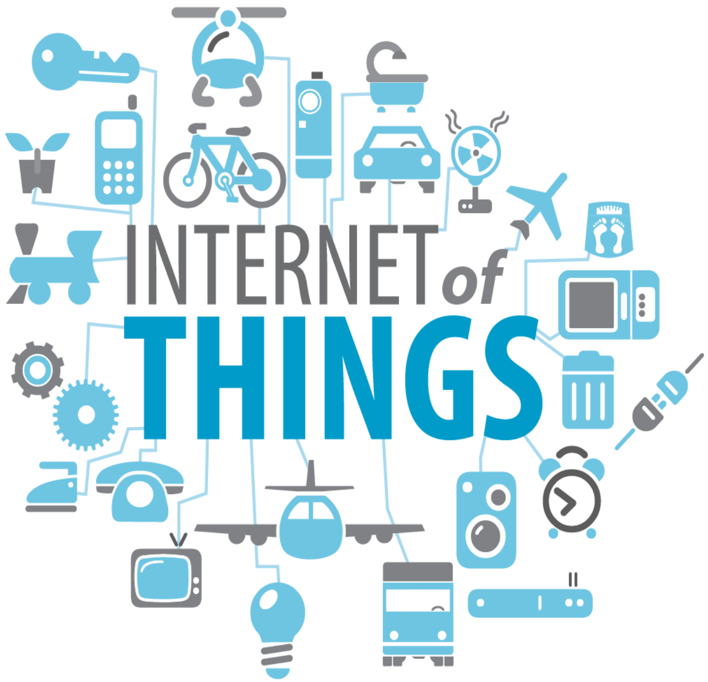
Internet
refrigerato

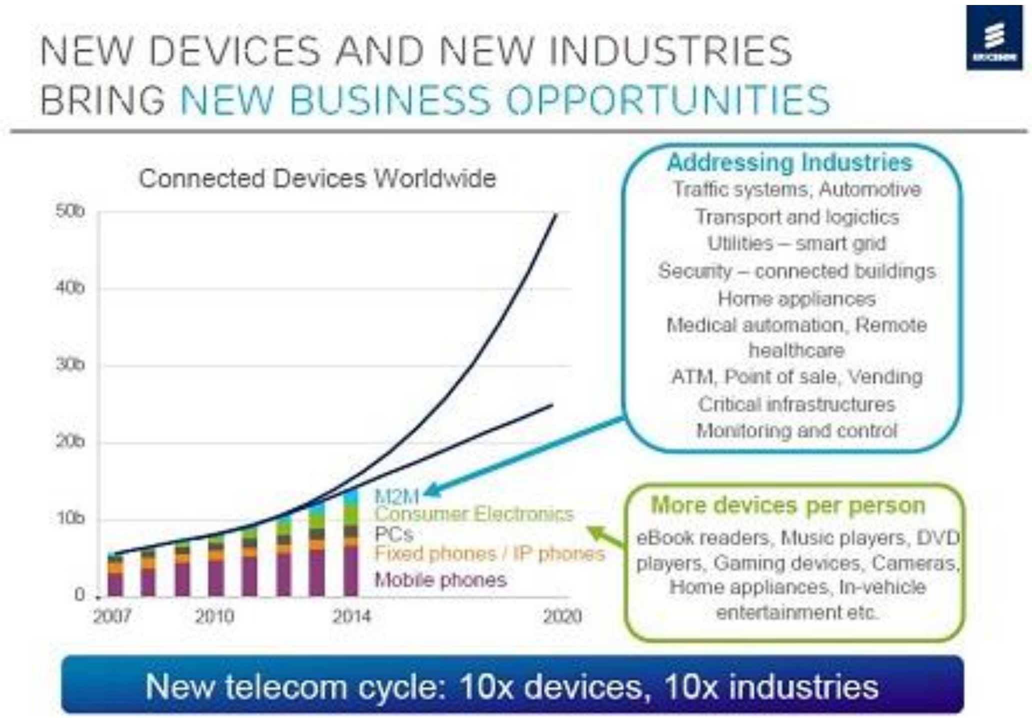


Web-enabled toaster +
weather forecaster

Tweet-a-watt:
monitor energy use

… looking towards the future …

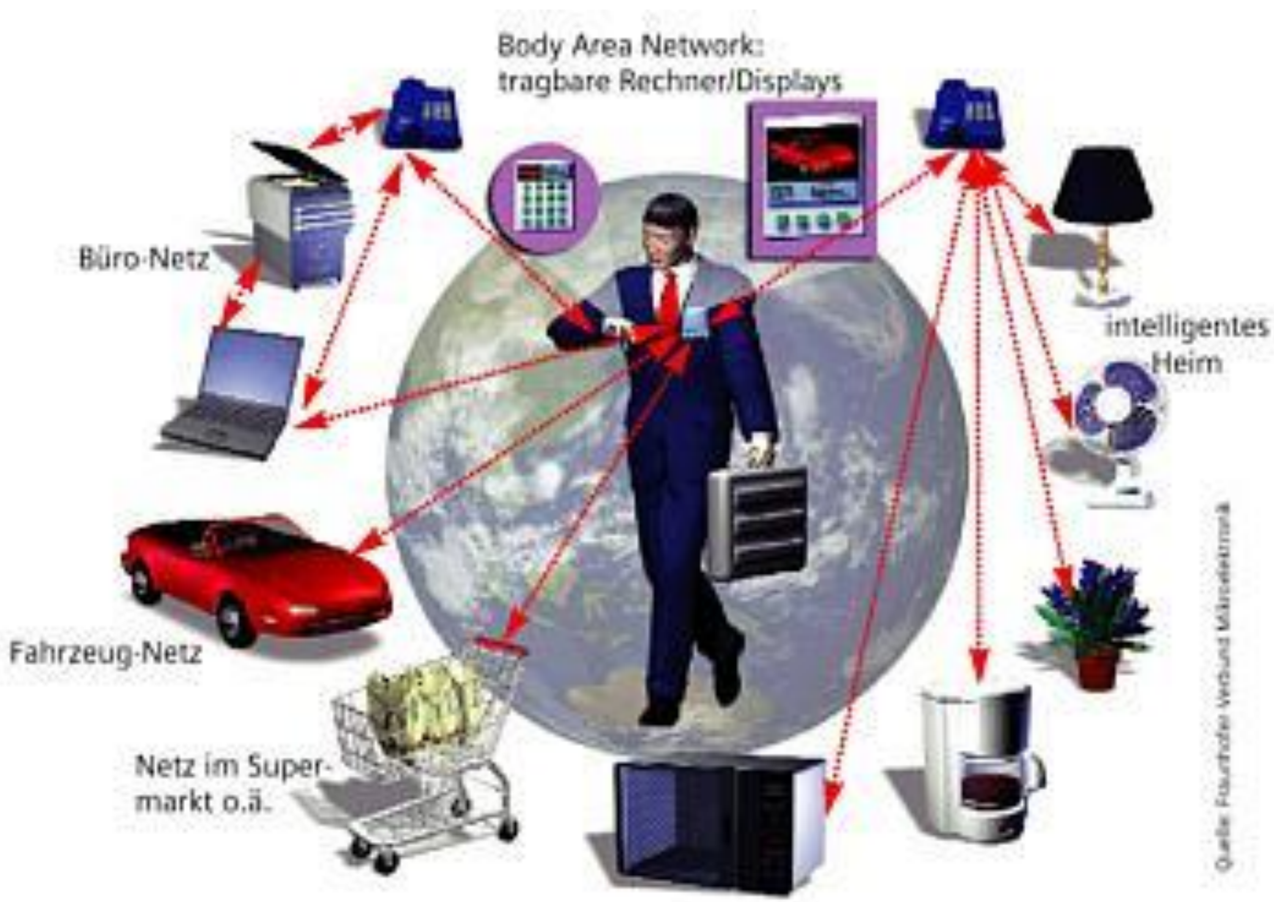




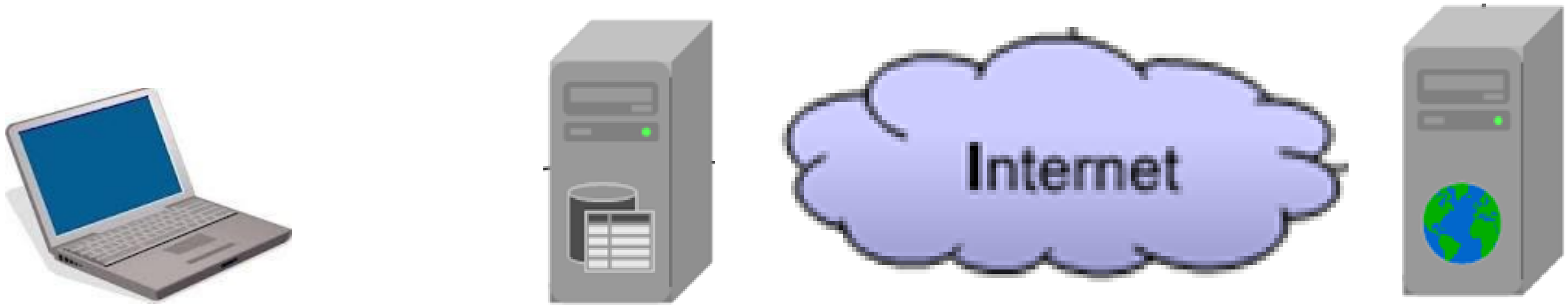
Examples where we are heading …

- Everything that can be connected will be connected
- IoT and smart cities
  - Machine-to-machine
- High-definition 3D streaming to heterogeneous client

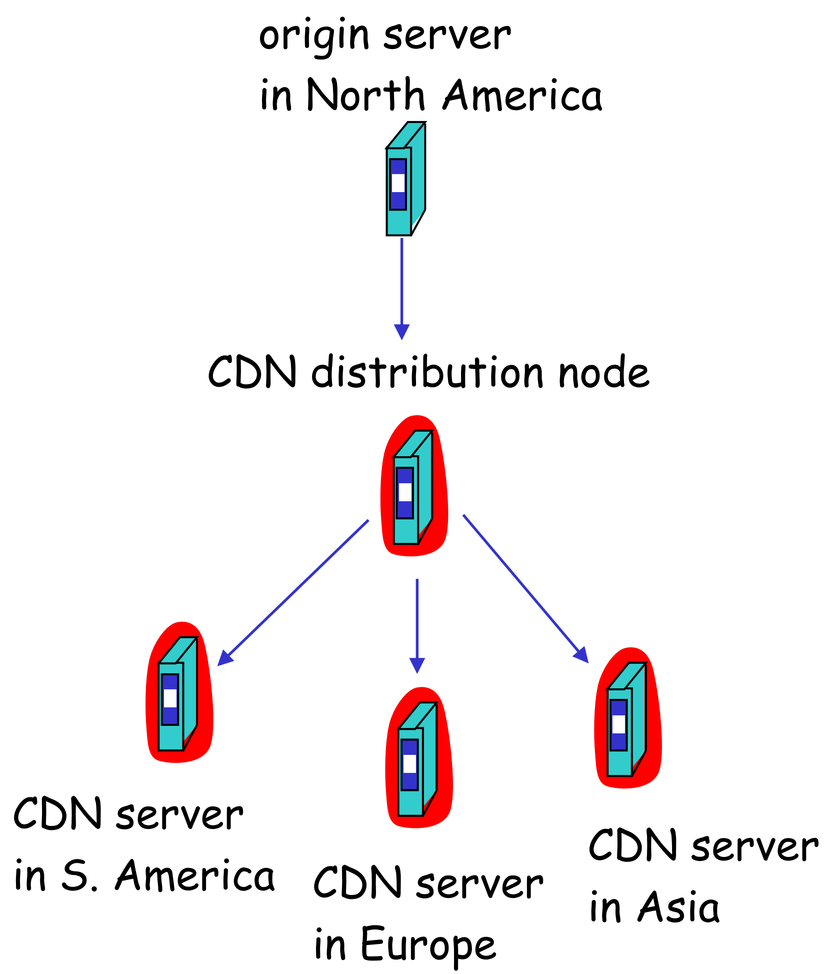
Personalized service and personal footprints in a connected world …



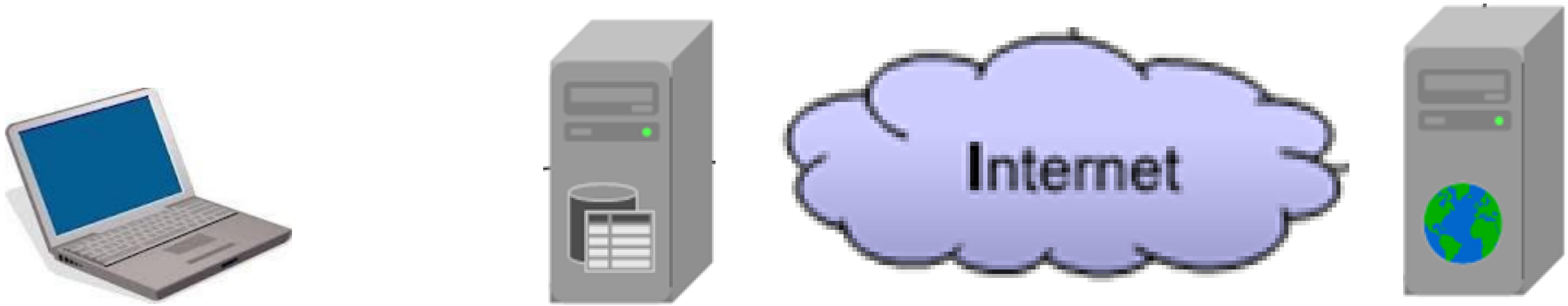
Proxies



Content Delivery Networks (CDNs)



Proxies



- ... and middleboxes

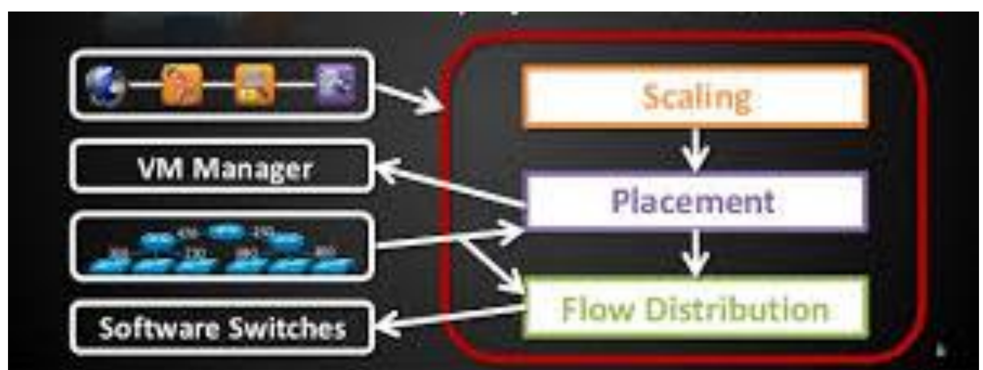
Gember-Jacobson et al., ACM SIGCOMM 2014
Sherry et al., ACM SIGCOMM 2012

Middleboxes + NF (and OpenNF)

Gember-Jacobson et al., ACM SIGCOMM 2014
Sherry et al., ACM SIGCOMM 2012

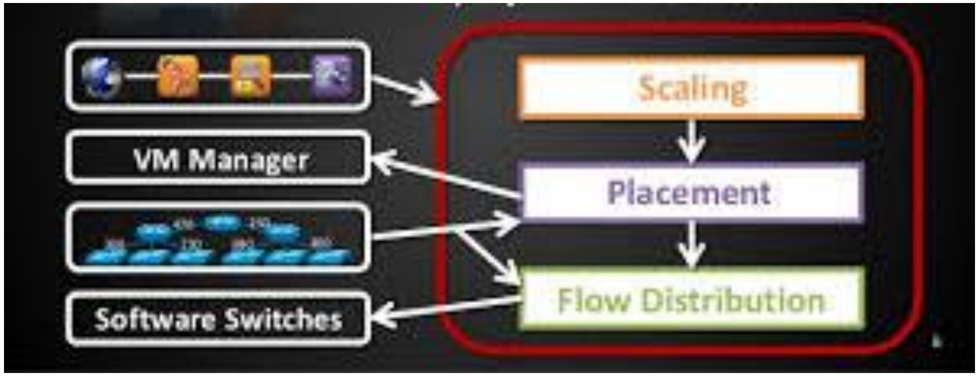
Middleboxes + NF (and OpenNF)

- "Network functions (NFs), or middleboxes, are systems that examine and modify packets and flows in sophisticated ways; e.g., intrusion detection systems (IDSs), load balancers, caching proxies, etc."
- "NFs play a critical role in ensuring security, improving performance, and providing other novel network functionality."



Middleboxes + NF (and OpenNF)

- Network functions virtualization (NFV) + software defined networking (SDN) has the potential to help operators (i) satisfy tight service level agreements, (ii) accurately monitor and manipulate network traffic, and (iii) minimize operating expenses.
- Example: OpenNF is a control plane designed to help redistribute packet processing across a collection of network function (NF) instances and coordinated control of both internal NF state and network forwarding state so to simultaneously achieving all three goals.



Cloud also used to offload the clients themselves ...

What is Mobile Cloud Computing?

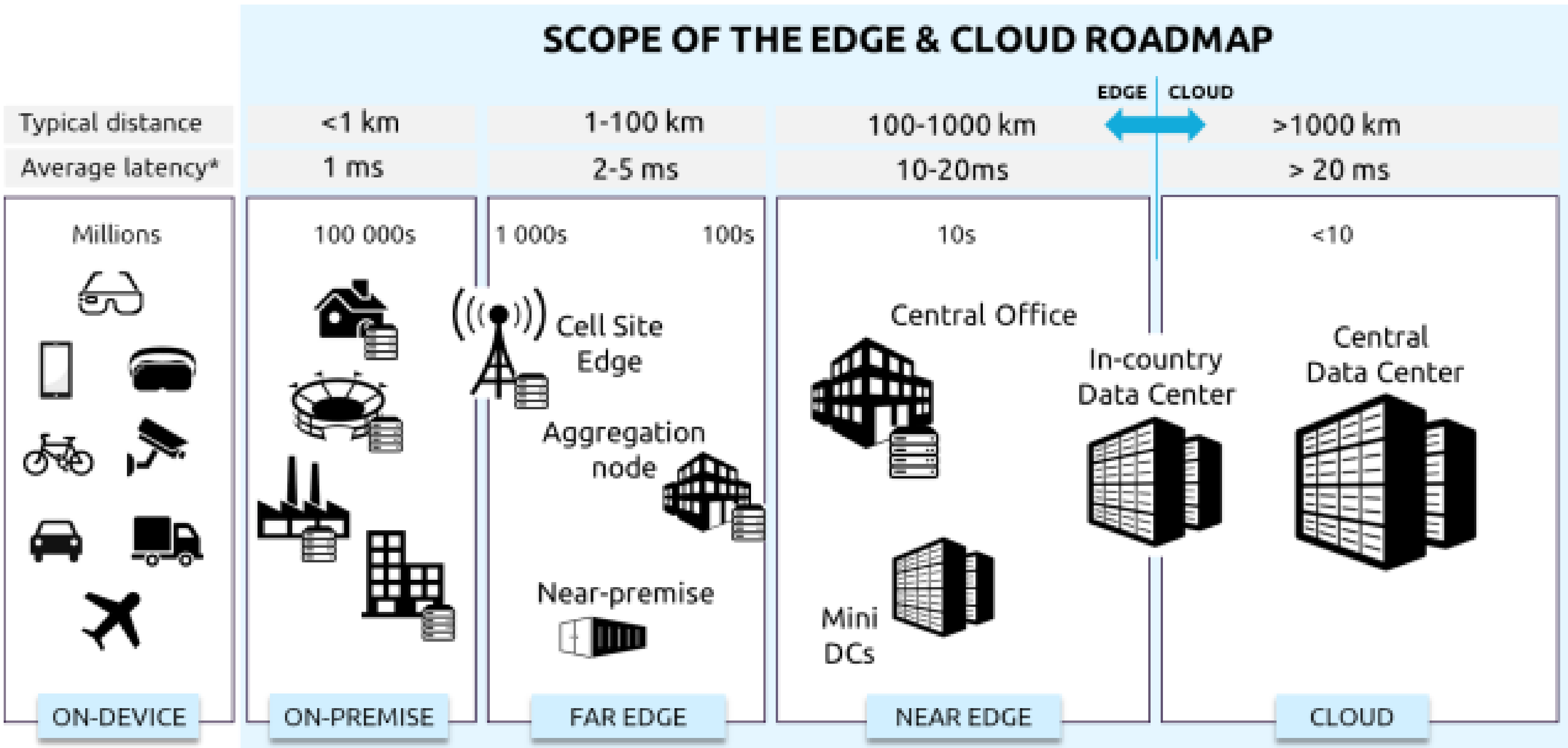
Mobile cloud computing (MCC) at its simplest, refers to an infrastructure where both the data storage and data processing happen outside of the mobile device.

Moves the computing power and data storage away from the mobile devices and into powerful and computing platforms located in clouds

Why Mobile Cloud Computing?

- Mobile devices face many resource challenges (battery life, storage, bandwidth etc.)
- Cloud computing allows users to use infrastructure, platforms and software elastically in an on-demand fashion
- Provides mobile users with data storage and processing services in clouds, removing the need to have a powerful device configuration (e.g. CPU speed, memory capacity etc)
- Resource-intensive computing can be performed in the cloud

Where to offload to?



* Latency does not depend only on distance. Other factors influencing latency are a) access technology (latency in 5G or FTTH much lower than in 4G), b) transport topology and technology, c) core network configuration (user plane location, breakout point), d) network optimization (traffic prioritization, bandwidth allocation, Edge node selection).

***Figure**: Scope of the Industrial Roadmap in the cloud-edge continuum*

European industrial technology roadmap for the next generation cloud-edge offering, May 2021

When to offload?

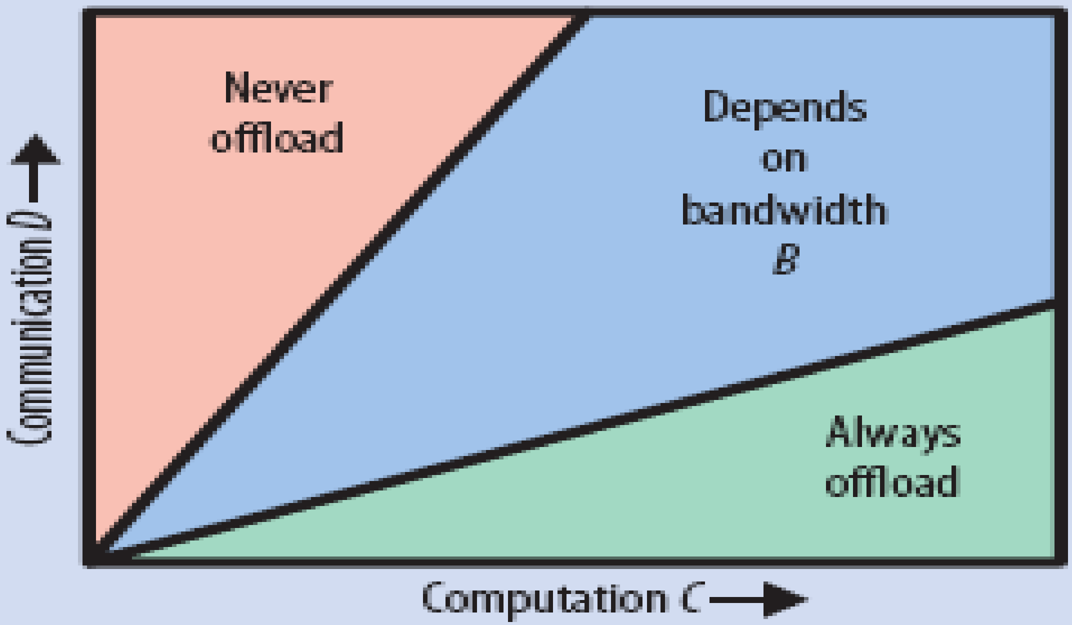


Figure 1. Offloading is beneficial when large amounts of computation C are needed with relatively small amounts of communication D.

… cost-efficient delivery …

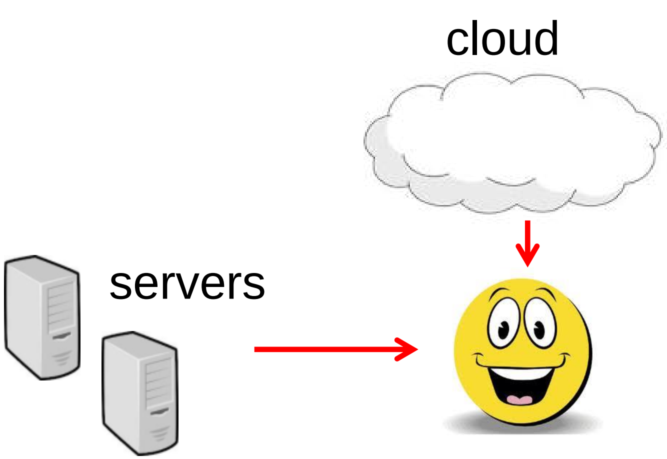
[Dan and Carlsson, IEEE INFOCOM 2014]

Example problem

- Minimize content delivery costs

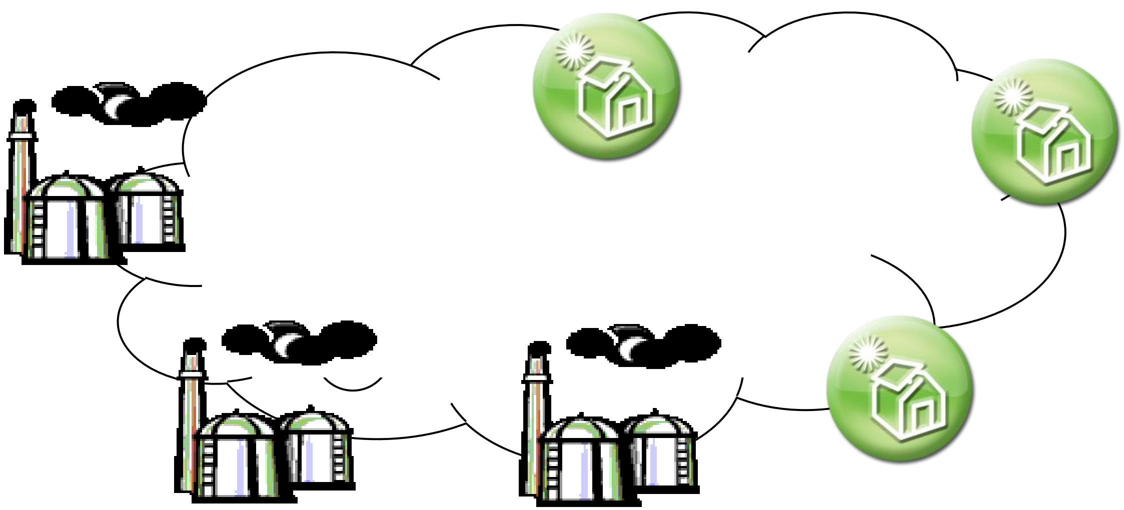
	Bandwidth	Cost
Cloud-based	Elastic/flexible	$$$
Dedicated servers	Capped	$

How to get the best of two worlds?



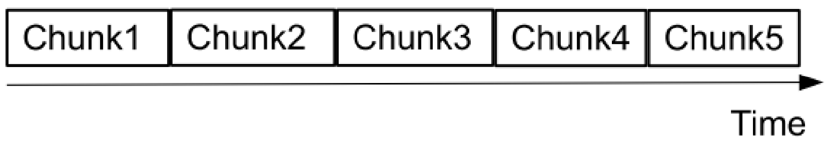
[Carlsson et al., IFIP Performance 2014]

… and from who?



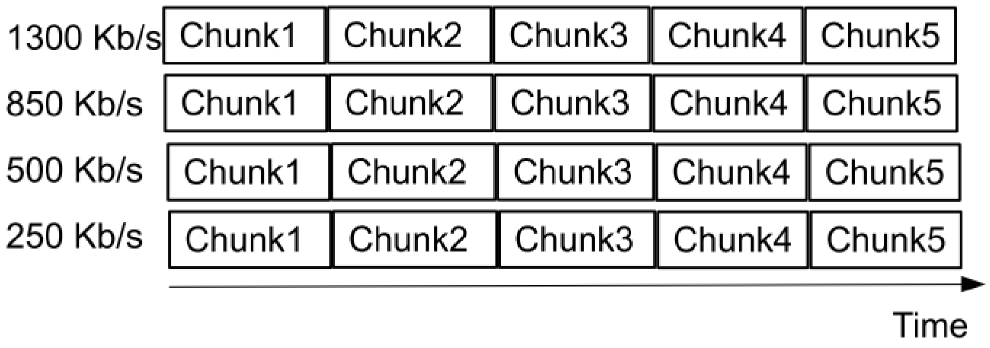
Mobile web (e.g., adaption and location-based services)

HTTP-based Adaptive Streaming (HAS)



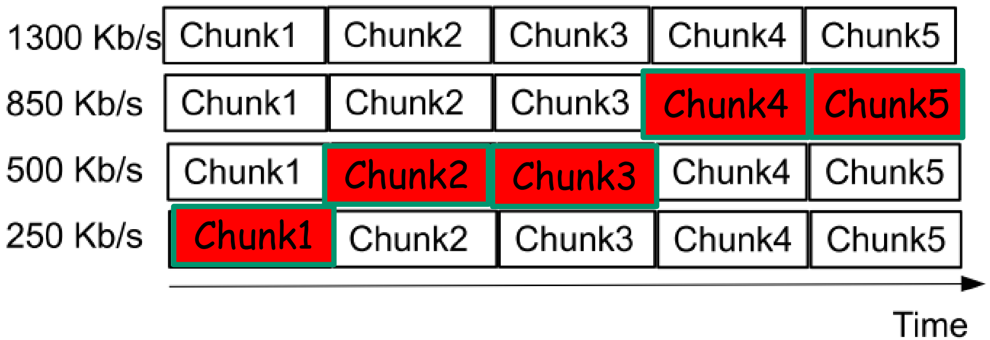
- HTTP-based ~~adaptive~~ streaming
  - Video is split into chunks

HTTP-based Adaptive Streaming (HAS)



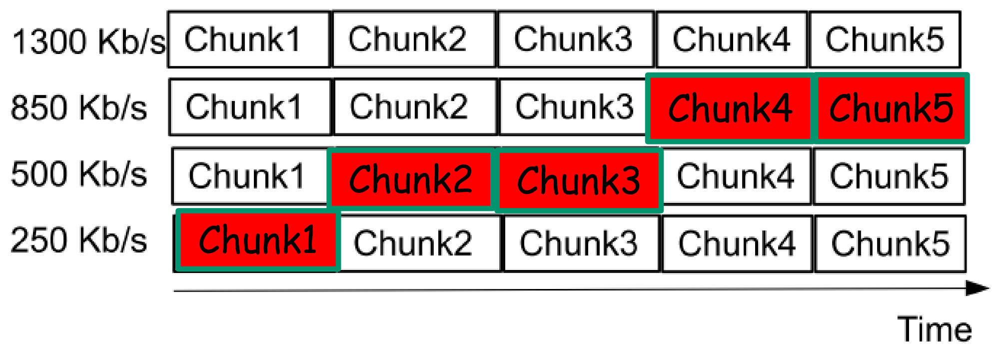
- HTTP-based **adaptive** streaming
  - Video is split into chunks
  - Each chunk in multiple bitrates (qualities), defined in manifest

HTTP-based Adaptive Streaming (HAS)



- HTTP-based **adaptive** streaming
  - Video is split into chunks
  - Each chunk in multiple bitrates (qualities)
  - Clients adapt quality encoding based on buffer/network conditions

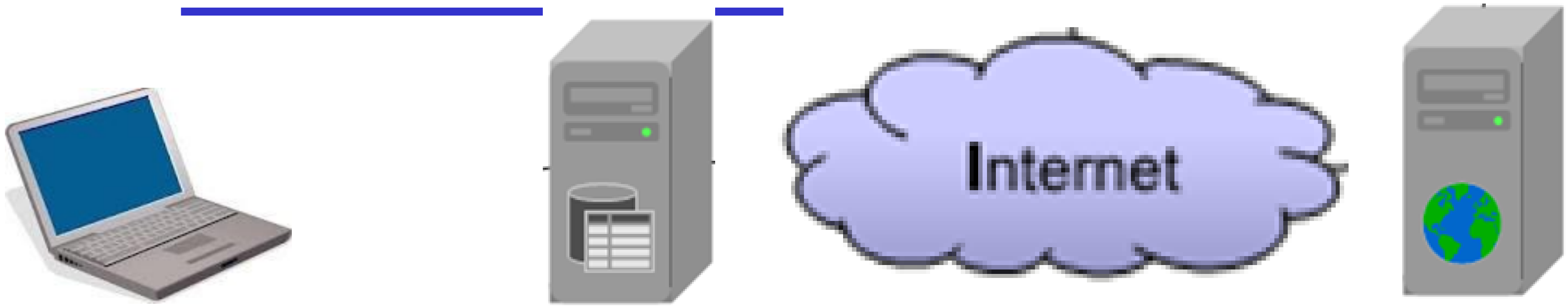
HTTP-based Adaptive Streaming (HAS)



- HTTP-based **adaptive** streaming
  - Video is split into chunks
  - Each chunk in multiple bitrates (qualities)
  - Clients adapt quality encoding based on buffer/network conditions
  - Use of TCP provides natural bandwidth adaptation
  - Some support for interactive VoD
  - Allows easy caching, NAT/firewall traversal, etc.

Problem: Proxy-assisted HAS

Slides from: V. Krishnamoorthi et al. "Helping Hand or Hidden Hurdle: Proxy-assisted HTTP-based Adaptive Streaming Performance", Proc. IEEE MASCOTS, 2013



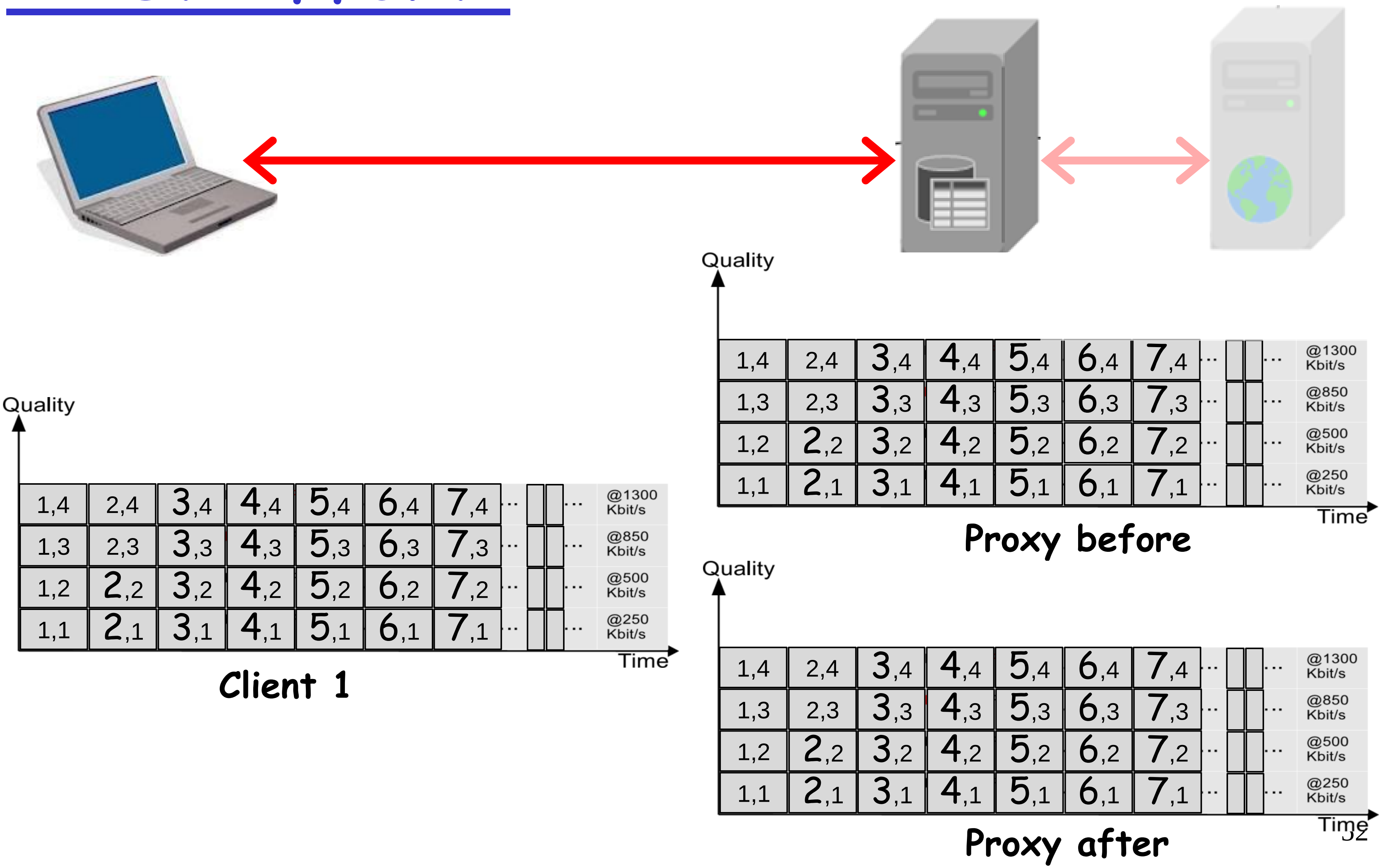
Clients' want

- High playback quality
- Small stall times
- Few buffer interruptions
- Few quality switches

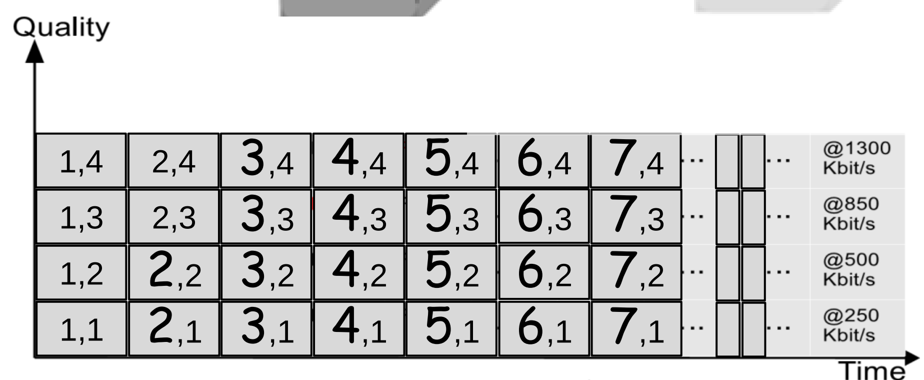
Network providers' want

- High QoE of customers/clients
- Low bandwidth usage
- High hit rate

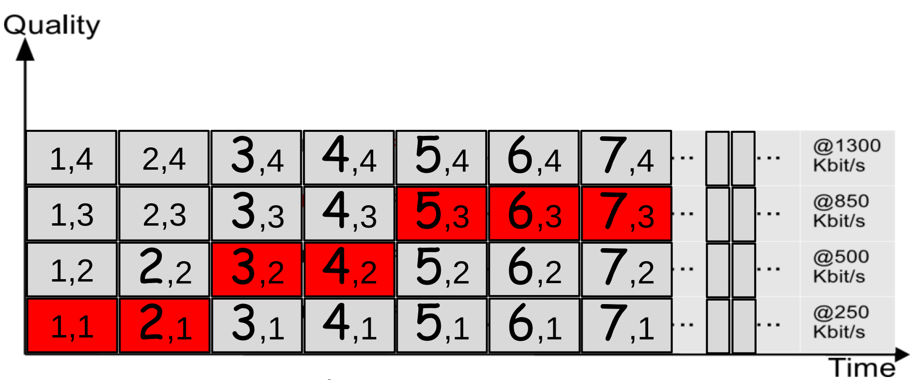
Example: Default "best-effort"



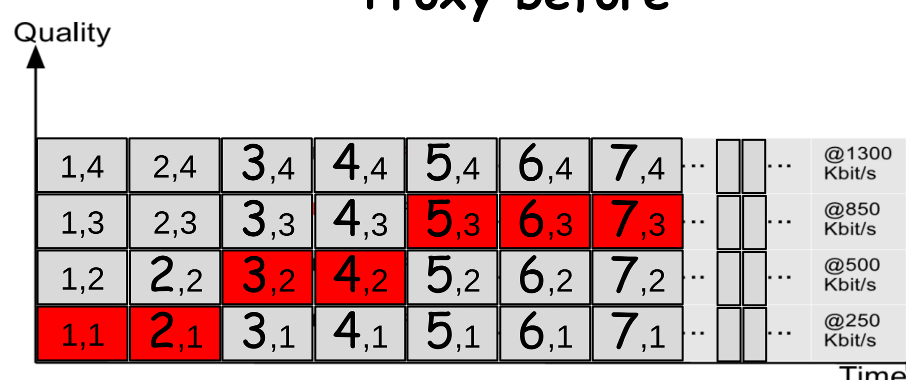
Example: Default "best-effort"



Proxy before

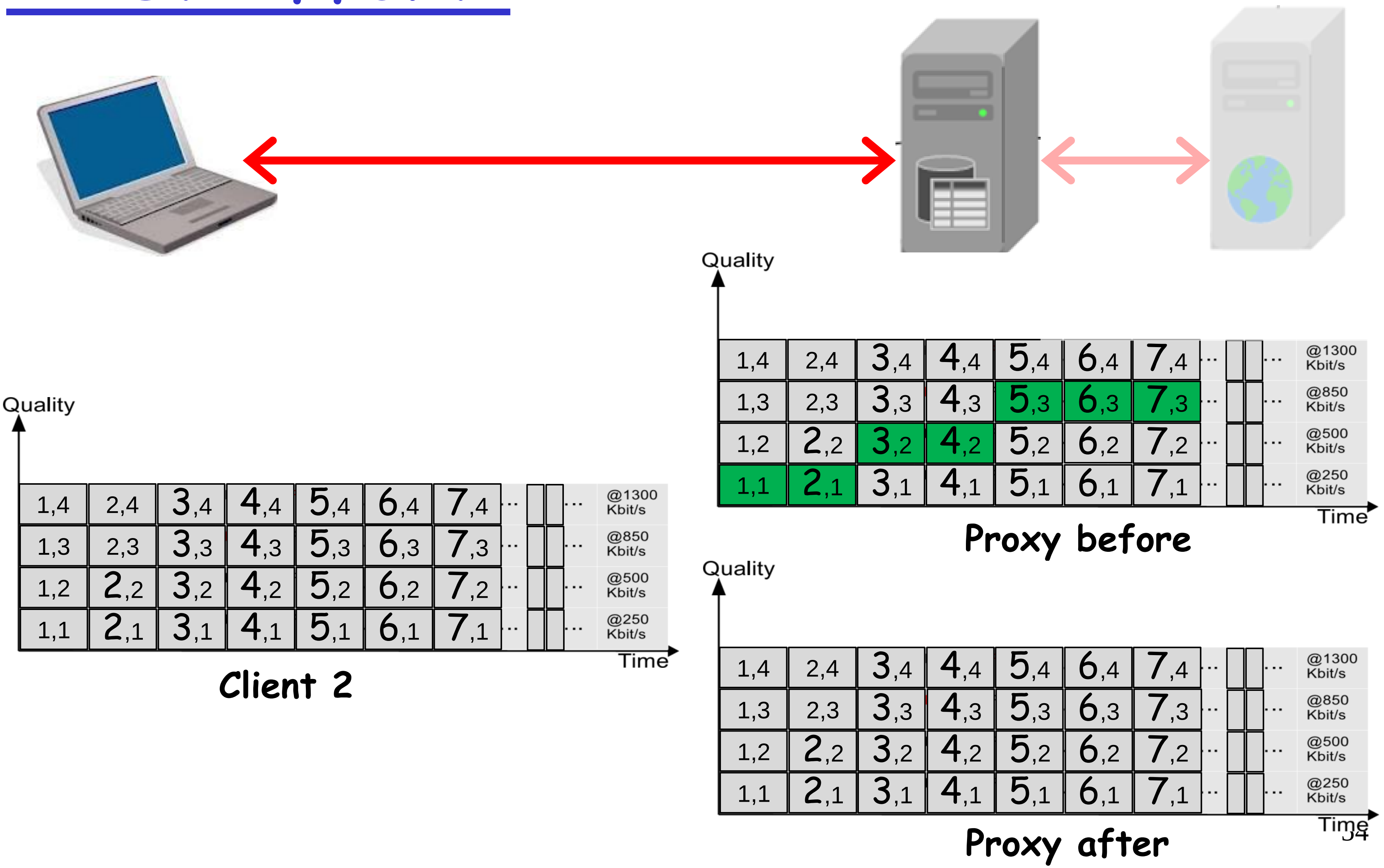


Client 1

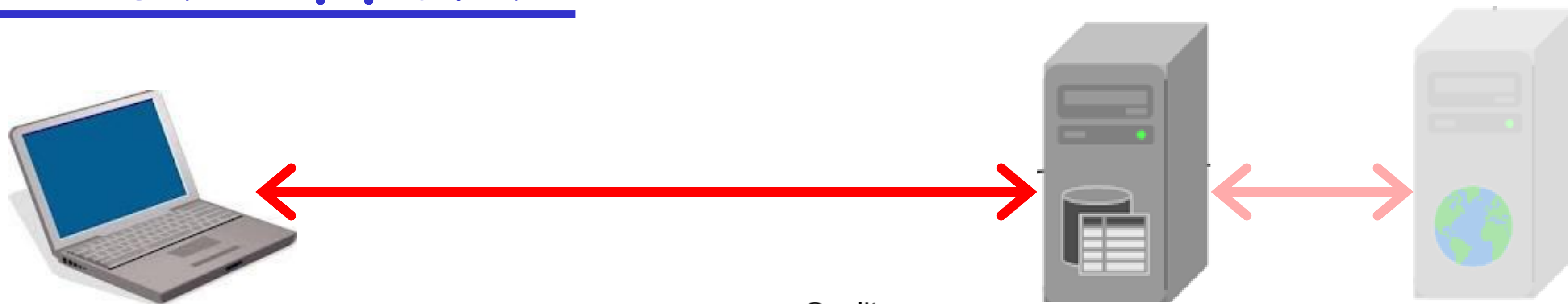


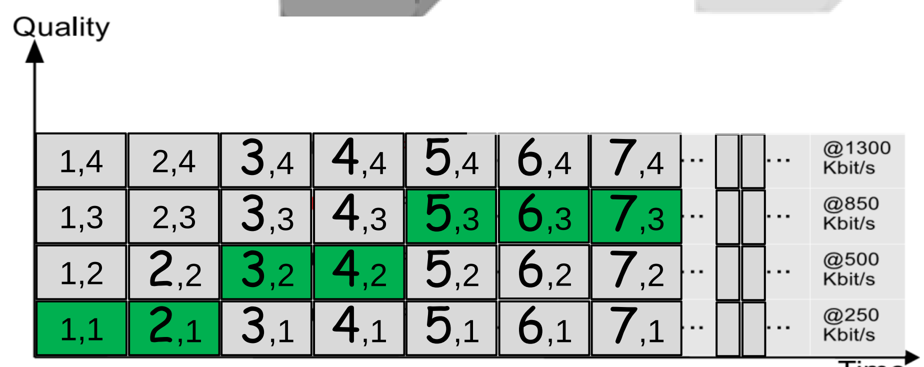
Proxy after

Example: Default "best-effort"

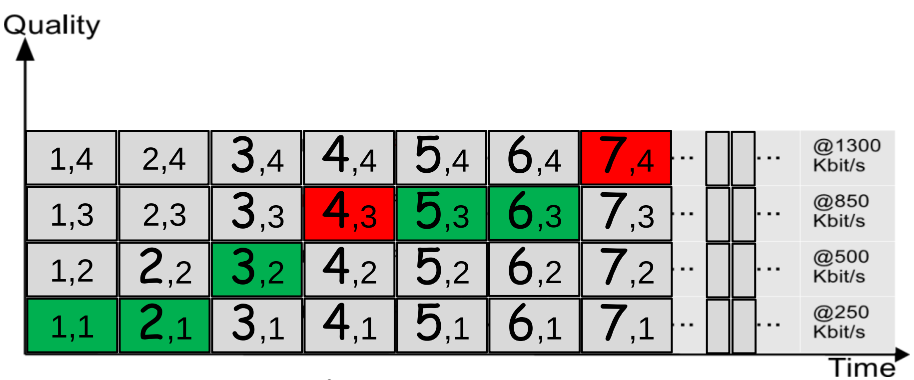


Example: Default "best-effort"

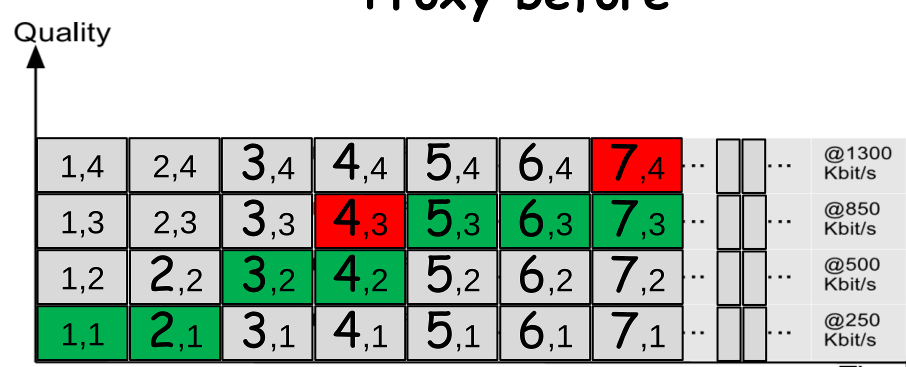




Proxy before



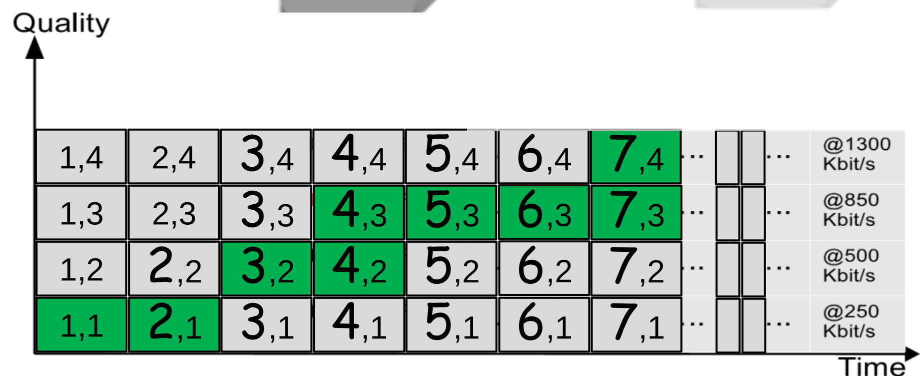
Client 2



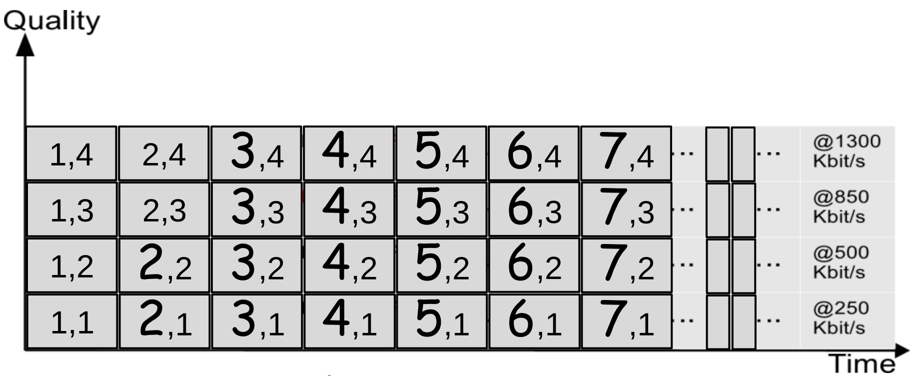
Proxy after

Good ! ☺

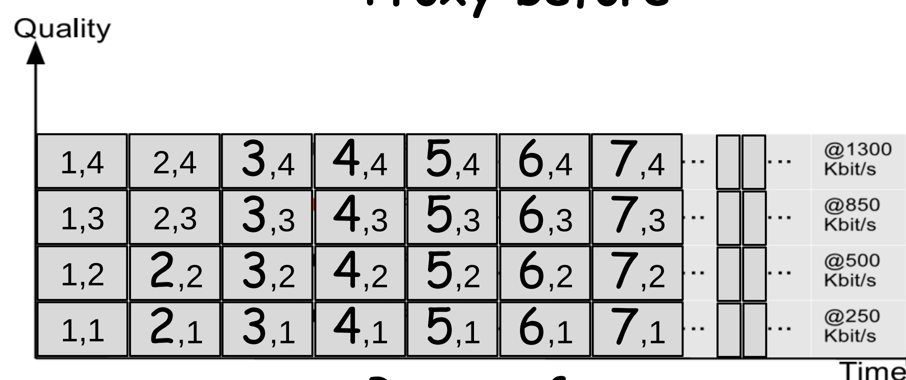
Example: Default "best-effort"



Proxy before



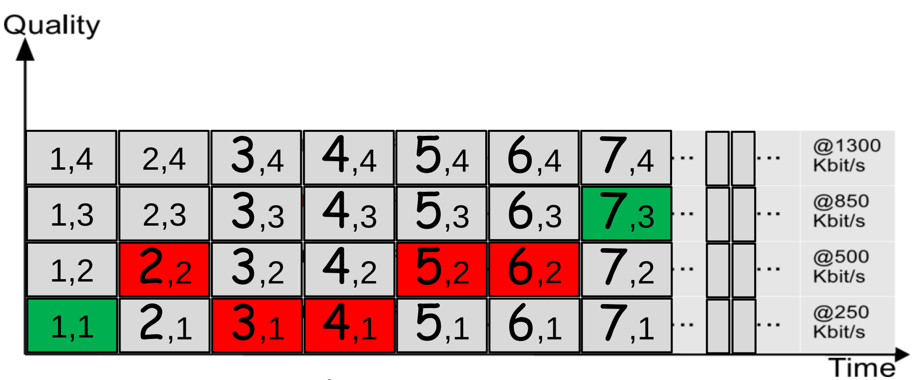
Client 3



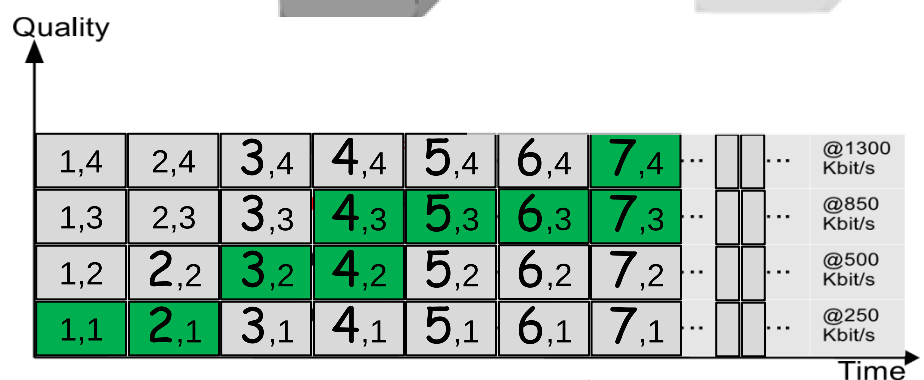
Proxy after

… but …

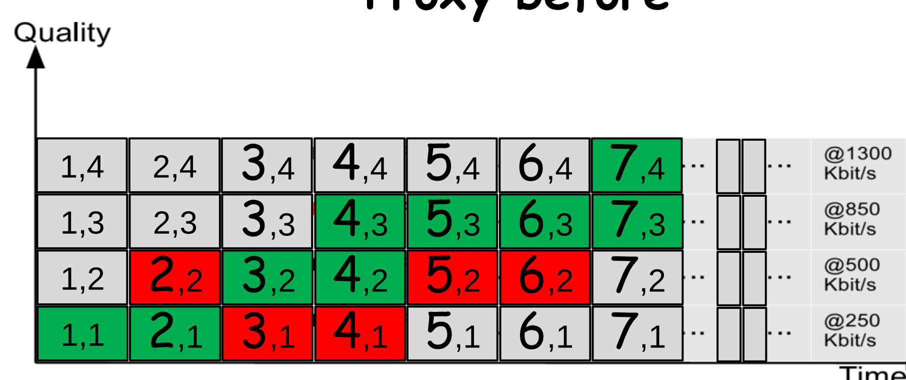
Example: Default "best-effort"



Client 3



Proxy before

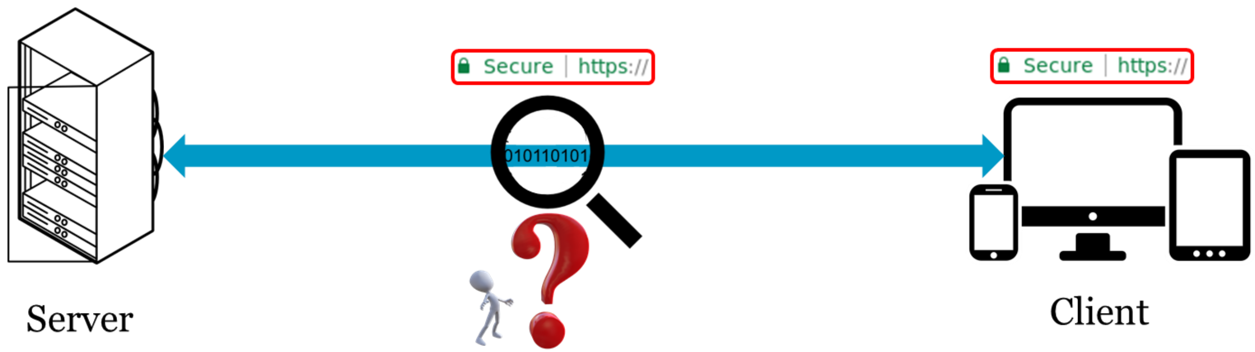


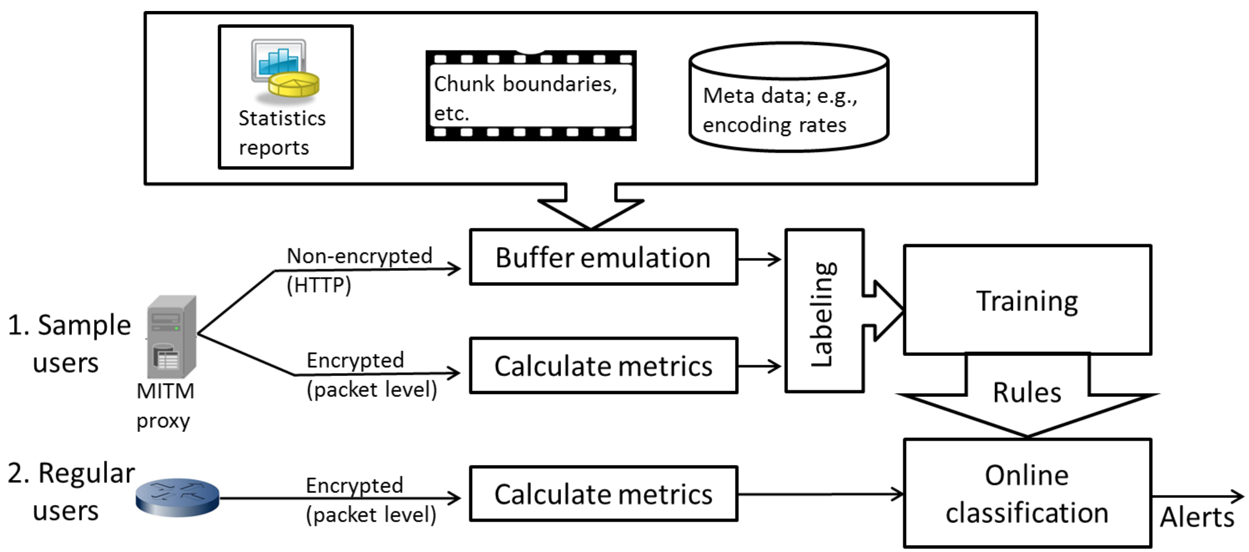
Proxy after

… sometimes bad ! ☹

Krishnamoorthi et al. ACM MMSys 2017
(+ IEEE/ACM IWQoS 2018)

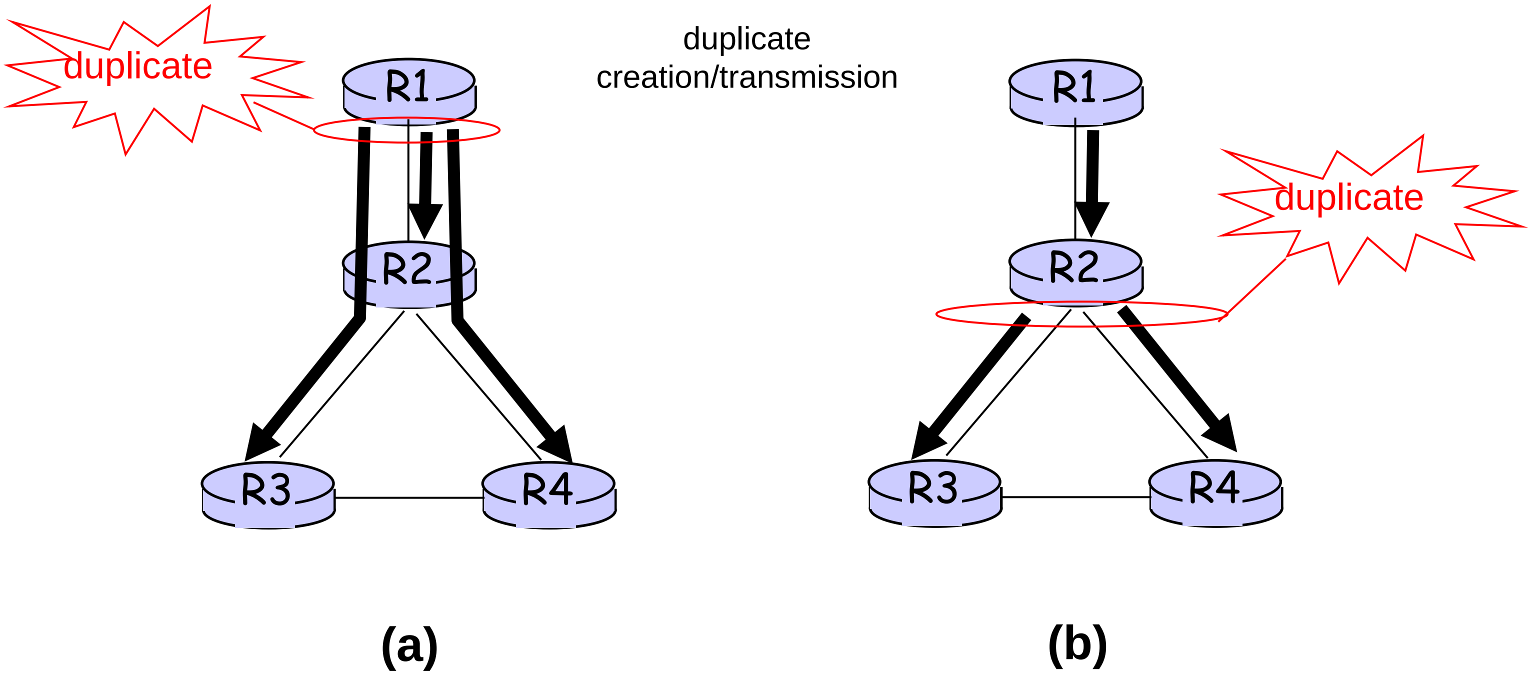
BUFFEST (+ rates @middlebox)





Additional Multimedia Support

Multicast/Broadcast



Source-duplication versus in-network duplication.
(a) source duplication, (b) in-network duplication

Also, application-layer multicast …

Evolved Multimedia Broadcast/Multicast Service (eMBMS) in LTE-advanced

Evolved Multimedia Broadcast/Multicast Service (eMBMS) in LTE-advanced

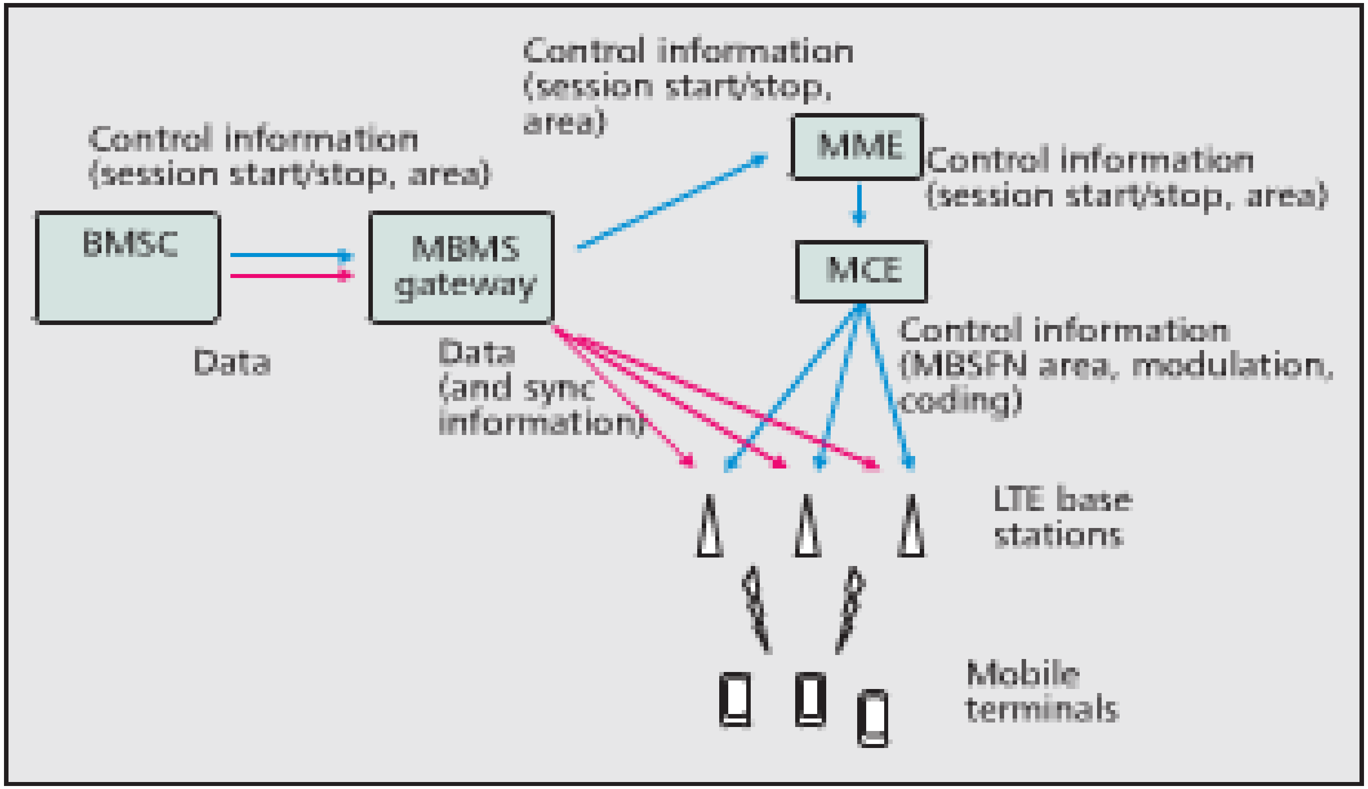


Figure 4. *RAN architecture for SFN across LTE base stations.*

- Separation of control plane and data plane

Image from: Lecompte and Gabin, Evolved Multimedia Broadcast/Multicast Service (eMBMS) in LTE-Advanced: Overview and Rel-11 Enhancements, IEEE Communications Magazine, Nov. 2012.

Evolved Multimedia Broadcast/Multicast Service (eMBMS) in LTE-advanced

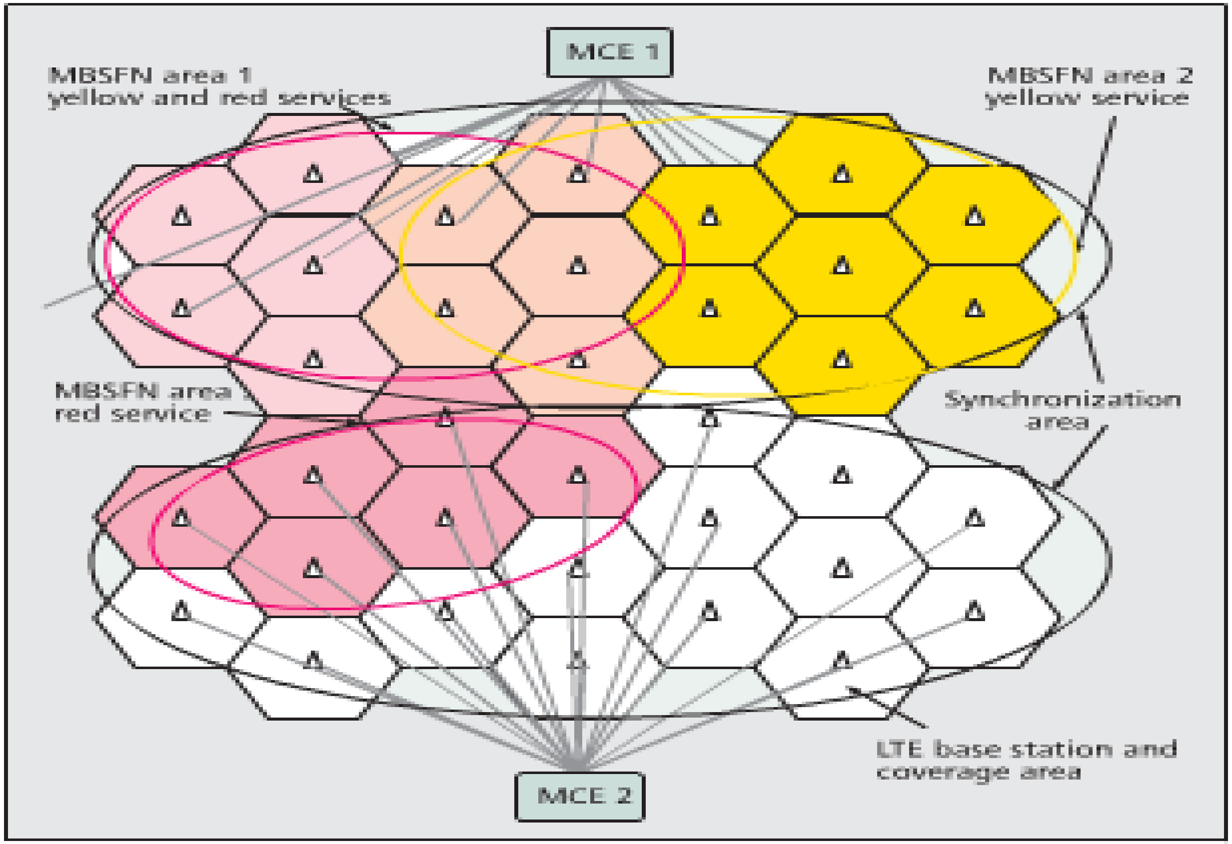


Figure 5. *Example with two MBMS services with different services areas.*

- MBMSFN and use of services areas

Image from: Lecompte and Gabin, Evolved Multimedia Broadcast/Multicast Service (eMBMS) in LTE-Advanced: Overview and Rel-11 Enhancements, IEEE Communications Magazine, Nov. 2012.

Packet Loss

- network loss: IP datagram lost due to network congestion (router buffer overflow) or losses at wireless link(s)
- delay loss: IP datagram arrives too late for playout at receiver (effectively the same as if it was lost)
  - delays: processing, queueing in network; end-system (sender, receiver) delays
  - Tolerable delay depends on the application

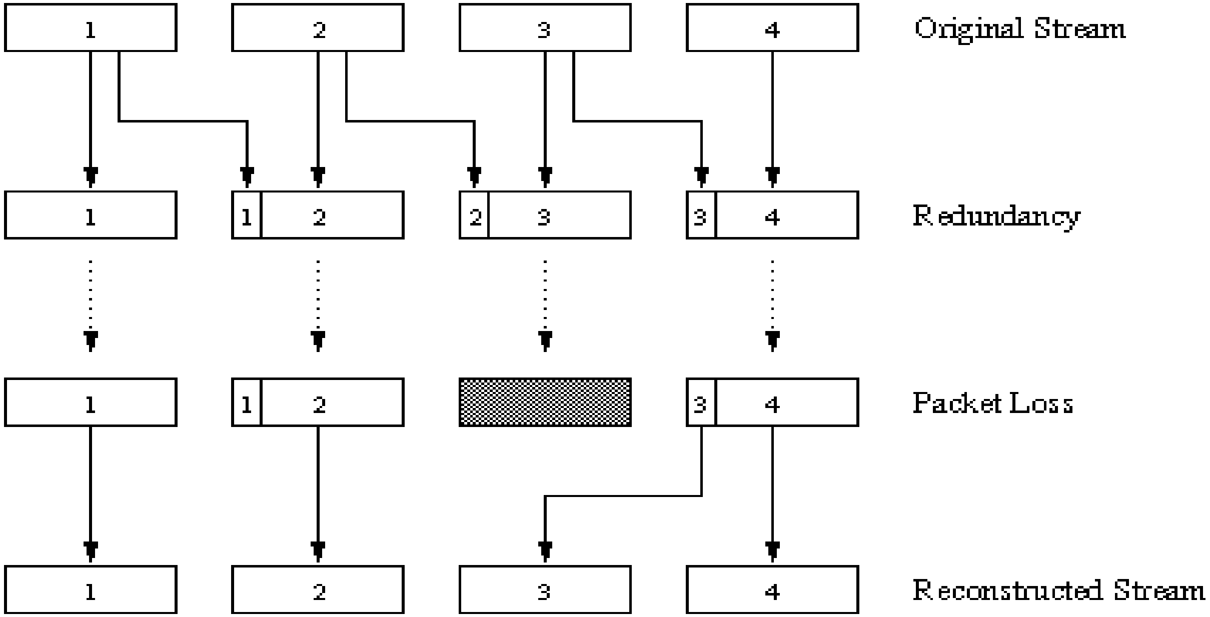
Receiver-based Packet Loss Recovery

- Generate replacement packet
  - Packet repetition
  - Interpolation
  - Other sophisticated schemes
- Works for relatively low loss rates (e.g., < 5%)
- Typically, breaks down on "bursty" losses

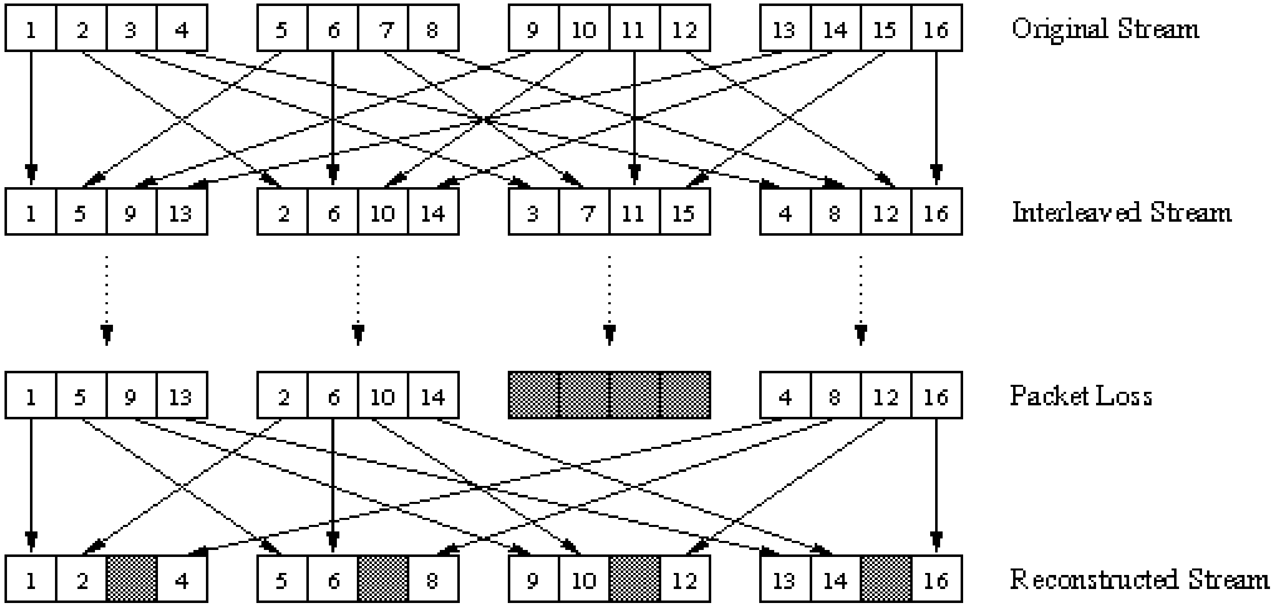
Forward Error Correction (FEC)

- For every group of n actual media packets, generate k additional redundant packets
- Send out n+k packets, which increases the bandwidth consumption by factor k/n.
- Receiver can reconstruct the original n media packets provided at most k packets are lost from the group
- Works well at high loss rates (for a proper choice of k)
- Handles "bursty" packet losses
- Cost: increase in transmission cost (bandwidth)

Another FEC Example



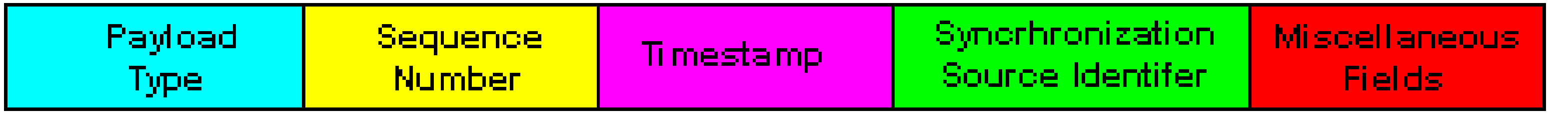
Interleaving: Recovery from packet loss



Real-Time Protocol (RTP)

- RTP specifies packet structure for packets carrying audio, video data
- RFC 3550
- RTP runs in end systems
- RTP packets encapsulated in UDP segments

RTP Header



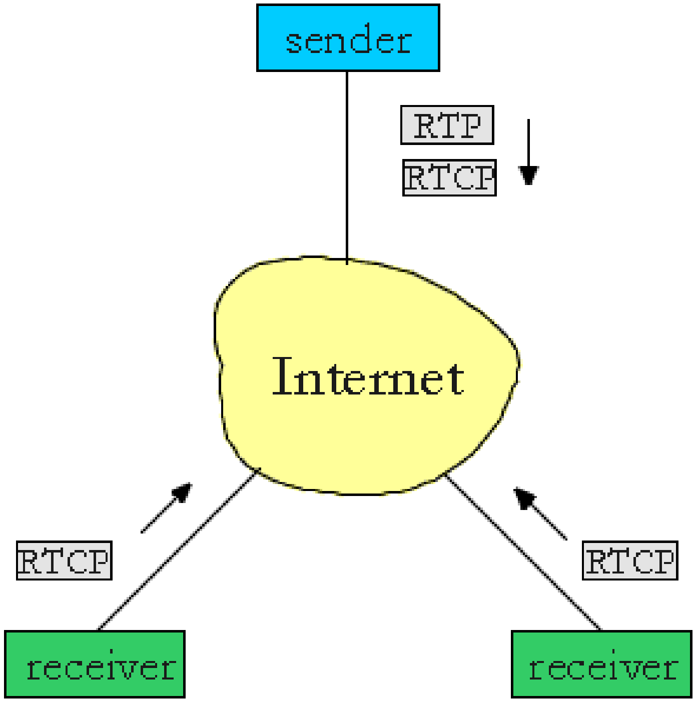
RTP Header

Payload Type (7 bits): Indicates type of encoding currently being used. If sender changes encoding in middle of conference, sender informs receiver via payload type field.

- 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

Sequence Number (16 bits): Increments by one for each RTP packet sent, and may be used to detect packet loss and to restore packet sequence.

Real-time Control Protocol (RTCP)



Receiver report packets:

- fraction of packets lost, last sequence number, average interarrival jitter

Sender report packets:

- SSRC of RTP stream, current time, number of packets sent, number of bytes sent

- RTCP attempts to limit its traffic to 5% of session bandwidth

More slides ...

A protocol family for streaming

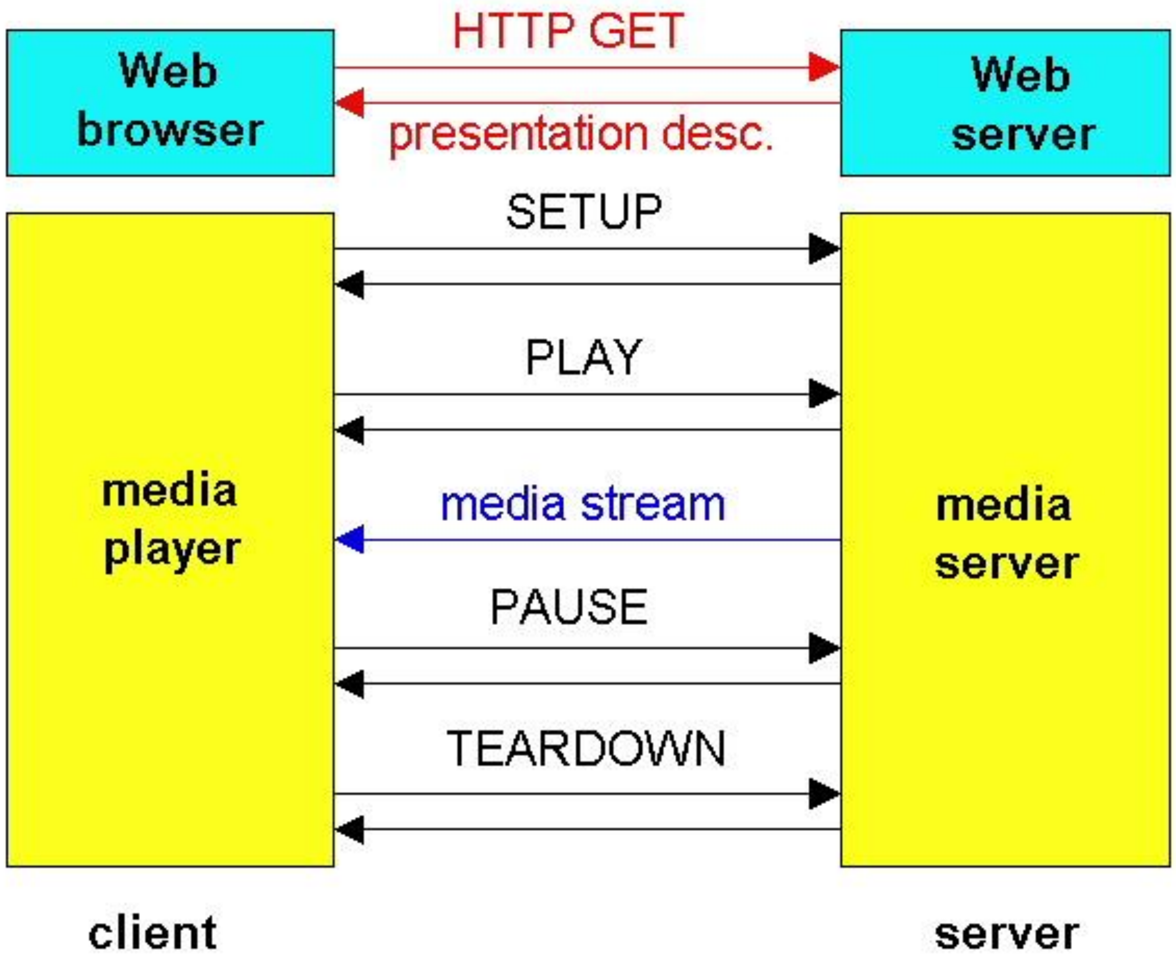
- RTSP
- RTP
- RTCP

RTSP Example

Scenario:

- metafile communicated to web browser
- browser launches player
- player sets up an RTSP control connection, data connection to streaming server

RTSP Operation



Real-Time Protocol (RTP)

- RTP specifies packet structure for packets carrying audio, video data
- RFC 3550
- RTP runs in end systems
- RTP packets encapsulated in UDP segments

RTP Header

Payload Type	Sequence Number	Timestamp	Syncrhronization Source Identifer	Miscellaneous Fields

RTP Header

Payload Type (7 bits): Indicates type of encoding currently being used. If sender changes encoding in middle of conference, sender informs receiver via payload type field.

- 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

Sequence Number (16 bits): Increments by one for each RTP packet sent, and may be used to detect packet loss and to restore packet sequence.

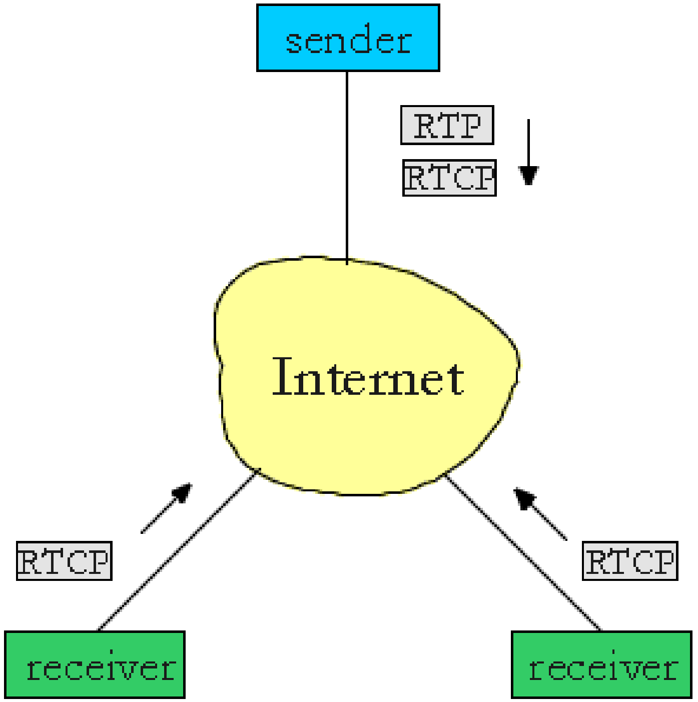
Real-time Control Protocol (RTCP)

Receiver report packets:

- fraction of packets lost, last sequence number, average interarrival jitter

Sender report packets:

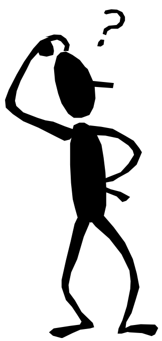
- SSRC of RTP stream, current time, number of packets sent, number of bytes sent



- RTCP attempts to limit its traffic to 5% of session bandwidth

Multimedia Over "Best Effort" Internet

- TCP/UDP/IP: *no guarantees on delay, loss*



But you said multimedia apps requires QoS and level of performance to be effective!

Today's multimedia applications implement functionality at the app. layer to mitigate (as best possible) effects of delay, loss

Summary:
Internet MM "tricks of the trade"

- UDP vs TCP
- client-side adaptive playout delay: to compensate for delay
- server side matches stream bandwidth to available client-to-server path bandwidth
  - chose among pre-encoded stream rates
  - dynamic server encoding rate
- error recovery (on top of UDP) at the app layer
  - FEC, interleaving
  - retransmissions, time permitting
  - conceal errors: repeat nearby data

More slides …

Some more on QoS: Real-time traffic support

- Hard real-time
- Soft real-time

- Guarantee bounded delay
- Guarantee delay jitter

- End-to-end delay = queuing delays + transmission delays + processing times + propagation delay (and any potential re-transmission delays at lower layers)

How to provide better support for Multimedia? (1/4)

Integrated Services (IntServ) philosophy:

- architecture for providing QoS guarantees in IP networks for individual flows
- requires fundamental changes in Internet design so that apps can reserve end-to-end bandwidth
- Components of this architecture are
  - Reservation protocol (e.g., RSVP)
  - Admission control
  - Routing protocol (e.g., QoS-aware)
  - Packet classifier and route selection
  - Packet scheduler (e.g., priority, deadline-based)

How to provide better support for Multimedia? (2/4)

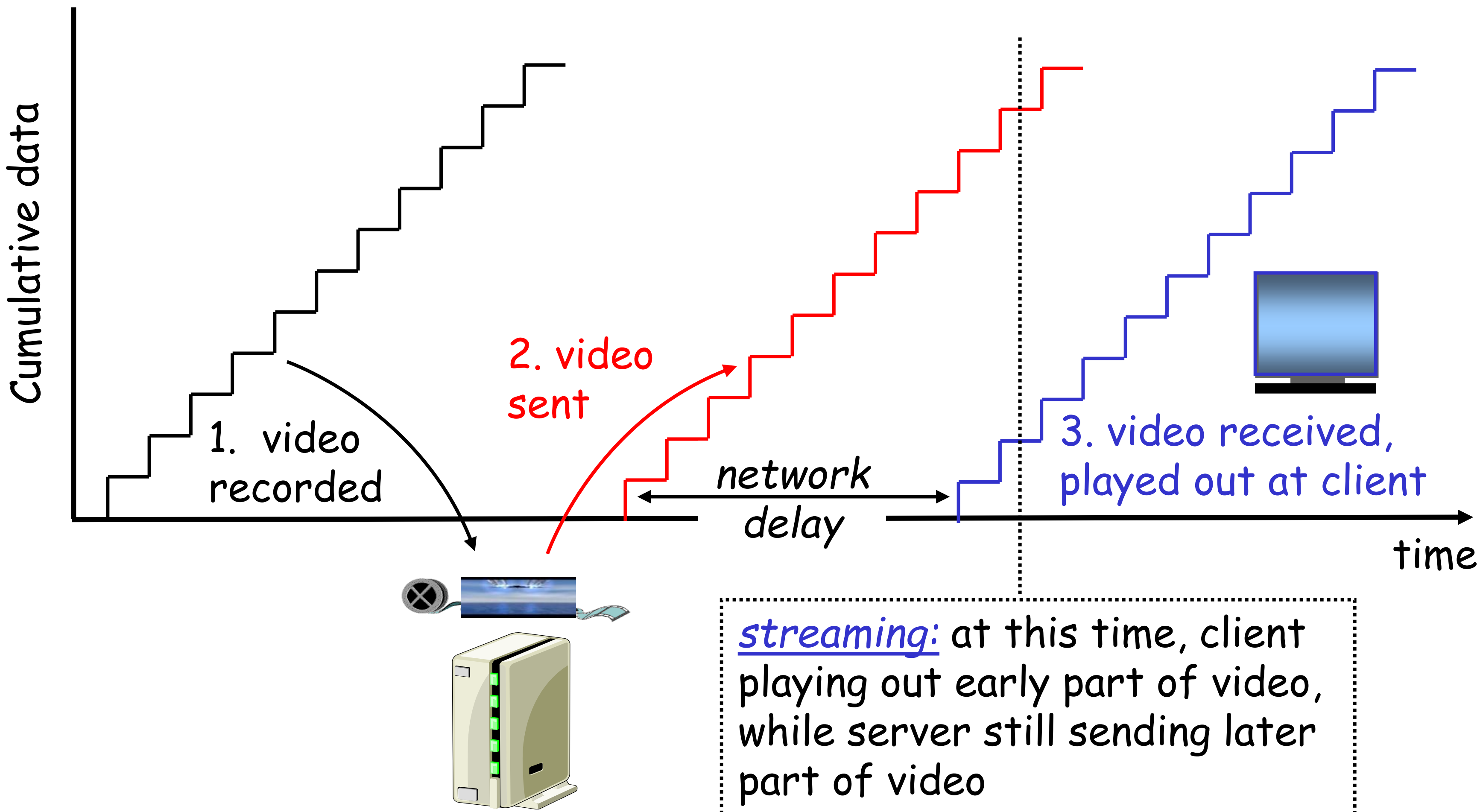
Concerns with IntServ:

- Scalability: signaling, maintaining per-flow router state difficult with thousands/millions of flows
- Flexible Service Models: IntServ has only two classes. Desire "qualitative" service classes
  - E.g., Courier, ExpressPost, and normal mail
  - E.g., First, business, and economy class

Differentiated Services (DiffServ) approach:

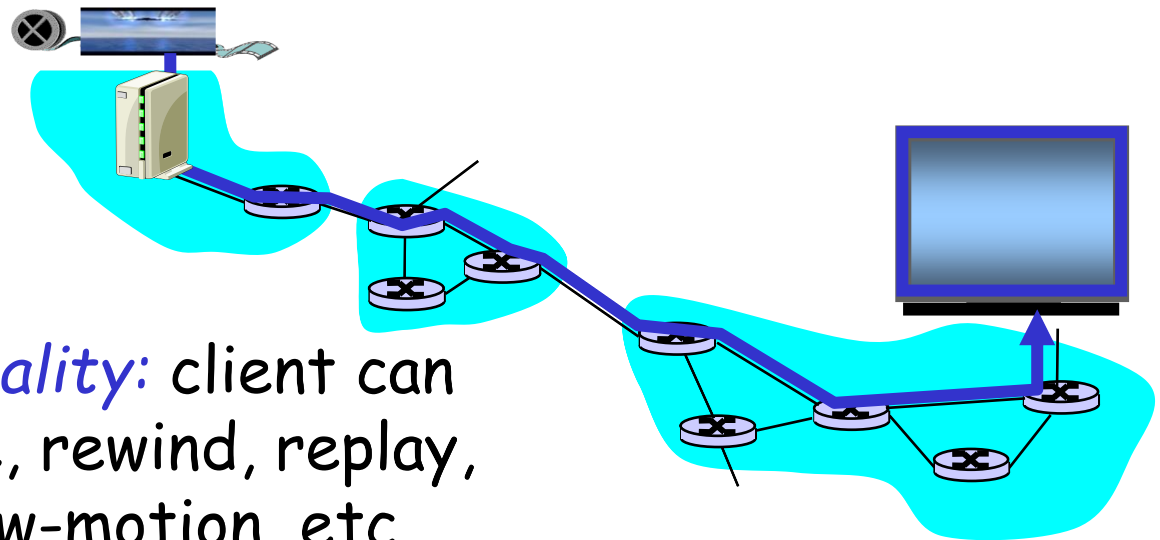
- simple functions in network core, relatively complex functions at edge routers (or hosts)
- Don't define the service classes, just provide functional components to build service classes

Streaming Stored Multimedia (1/2)



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

Streaming Stored Multimedia (2/2)



- *VCR-like functionality:* client can start, stop, pause, rewind, replay, fast-forward, slow-motion, etc.
  - 10 sec initial delay OK
  - 1-2 sec until command effect OK
  - need a separate control protocol?
- timing constraint for data that is yet to be transmitted: must arrive in time for playback

Streaming Live Multimedia

Examples:

- Internet radio talk show
- Live sporting event

Streaming

- playback buffer
- playback can lag tens of seconds after transmission
- still have timing constraint

Interactivity

- fast-forward is not possible
- rewind and pause possible!

Interactive, Real-time Multimedia

- applications: IP telephony, video conference, distributed interactive worlds
- end-end delay requirements:
  - audio: < 150 msec good, < 400 msec OK
    - includes application-layer (packetization) and network delays
    - higher delays noticeable, impair interactivity
- session initialization
  - callee must advertise its IP address, port number, frame rate, encoding algorithms

Why Study Multimedia Networking?

- Majority of traffic …
- Industry-relevant research topic
- Multimedia is everywhere
- Lots of open research problems
- Exciting and fun!

Scalable Content Delivery

Motivation

- Use of Internet for content delivery is massive … and becoming more so (e.g., majority of all IP traffic is video content)
- Variety of approaches: HTTP-based Adaptive Streaming (HAS), broadcast/multicast, batching, replication/caching (e.g. CDNs), P2P, peer-assisted, …
- In these slides, we only provide a few high-level examples

Service models

Multimedia Networking Applications

Classes of MM applications:

Multimedia Networking Applications

Classes of MM applications:

1) Streaming stored audio and video

2) Streaming live audio and video

3) Real-time interactive audio and video

Multimedia Networking Applications

Fundamental characteristics:

Multimedia Networking Applications

Fundamental characteristics:

- Inherent frame rate
- Typically **delay-sensitive**
  - end-to-end delay
  - delay jitter
- But **loss-tolerant**: infrequent losses cause minor transient glitches
- Unlike data apps, which are often delay-tolerant but loss-sensitive.

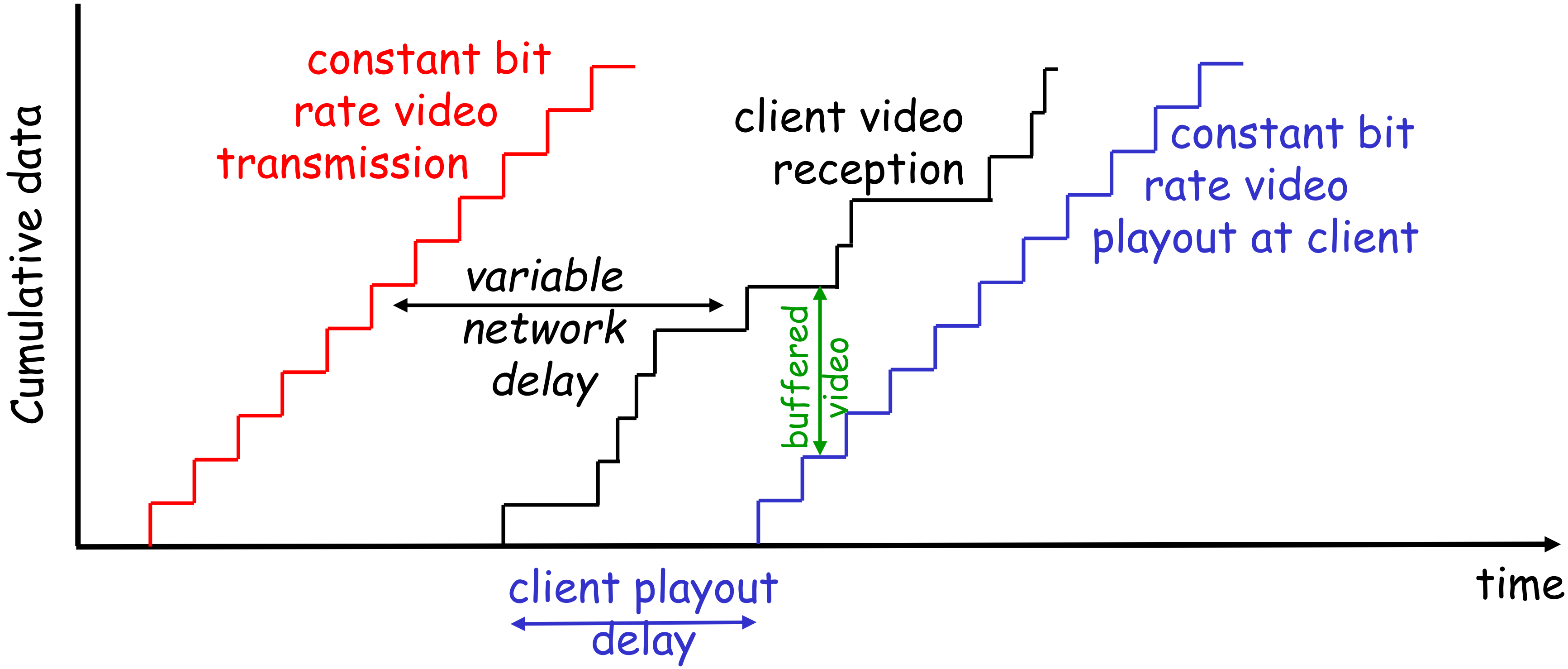
Multimedia Networking Applications

Fundamental characteristics:

- Inherent frame rate
- Typically **delay-sensitive**
  - end-to-end delay
  - delay jitter
- But **loss-tolerant**: infrequent losses cause minor transient glitches
- Unlike data apps, which are often delay-tolerant but loss-sensitive.

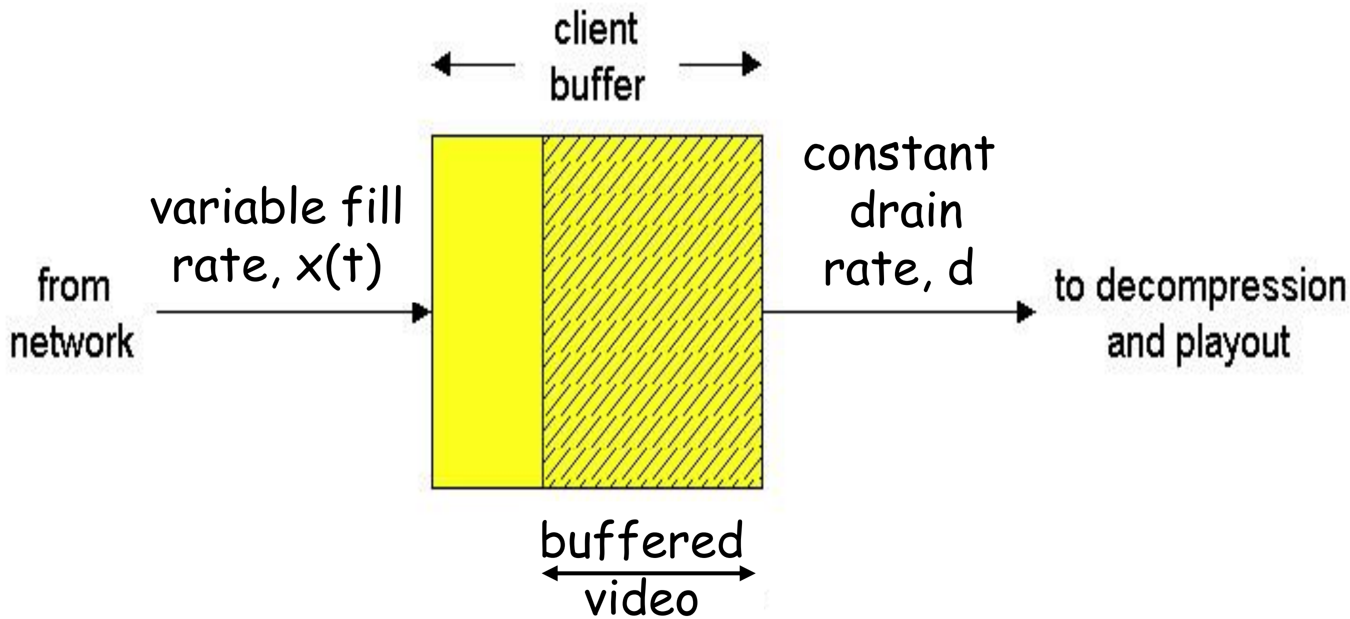
Jitter is the variability of packet delays within the same packet stream

Streaming Multimedia: Client Buffering



- Client-side buffering, playout delay compensate for network-added delay, delay jitter

Streaming Multimedia: Client Buffering



- Client-side buffering, playout delay compensate for network-added delay, delay jitter

Streaming Multimedia: UDP or TCP?

UDP

- server sends at rate appropriate for client (oblivious to network congestion !)
  - often send rate = encoding rate = constant rate
  - then, fill rate = constant rate - packet loss
- short playout delay (2-5 seconds) to compensate for network delay jitter
- error recover: time permitting

TCP

- send at maximum possible rate under TCP
- fill rate fluctuates due to TCP congestion control
- larger playout delay: smooth TCP delivery rate
- HTTP/TCP passes more easily through firewalls