The Web (part 3)

Lecture 8

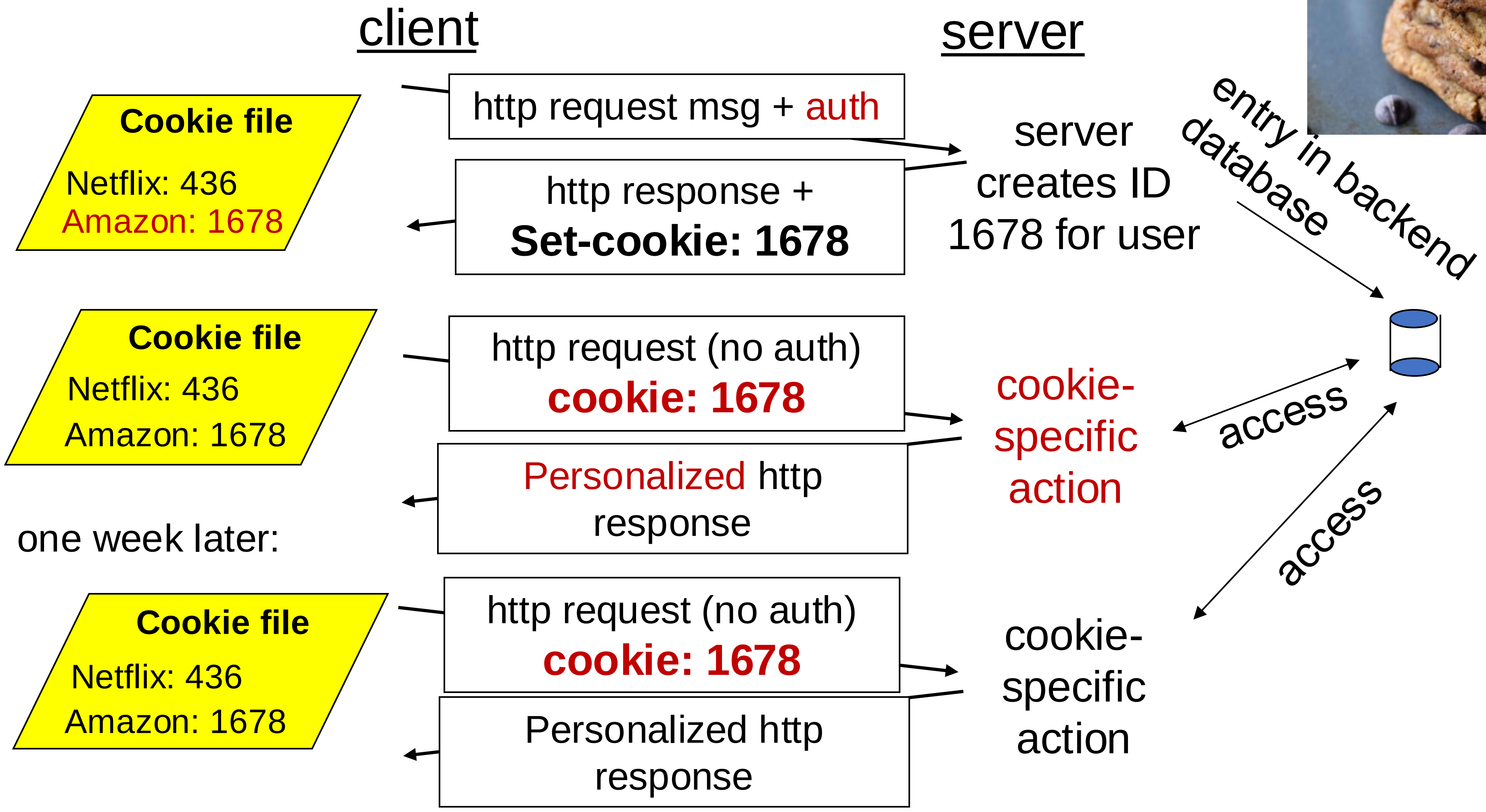
http://www.cs.rutgers.edu/~sn624/352-F24

Srinivas Narayana

RUTGERS
UNIVERSITY | NEW BRUNSWICK

Cookies: Keeping user memory

Cookie is typically opaque to client.



How cookies work

Collaboration between client and server to track user state.

Four components:

1. cookie header line of HTTP response message
2. cookie header line in HTTP request message
3. cookie file kept on user endpoint, managed by user’s browser
4. back-end database maps cookie to user data at Web endpoint

Cookies come with an expiration date (yet another HTTP header)

Cookies have many uses

- The good: Awesome user-facing functionality
  - Shopping carts, auth, … very challenging or impossible without it
- The bad: Unnecessary recording of your activities on the site
  - First-party cookies: performance statistics, user engagement, …
- The ugly: Tracking your activities across the Internet
  - Third-party cookies (played by ad and tracking networks) to track your activities across the Internet
  - personally identifiable information (PII)
  - Ad networks target users with ads; may sell this info
  - Scammers can target you too

PSA: Cookies and Privacy

- Disable and delete unnecessary cookies by default

- Suggested privacy-conscious browsers, websites, tools:

- DuckDuckGo (search)
- Brave (browser)
- AdBlock Plus (extension)
- ToR (distract targeting)
- … assuming it doesn't break the functions of the site

https://gdpr.eu/cookies/

Web Caching

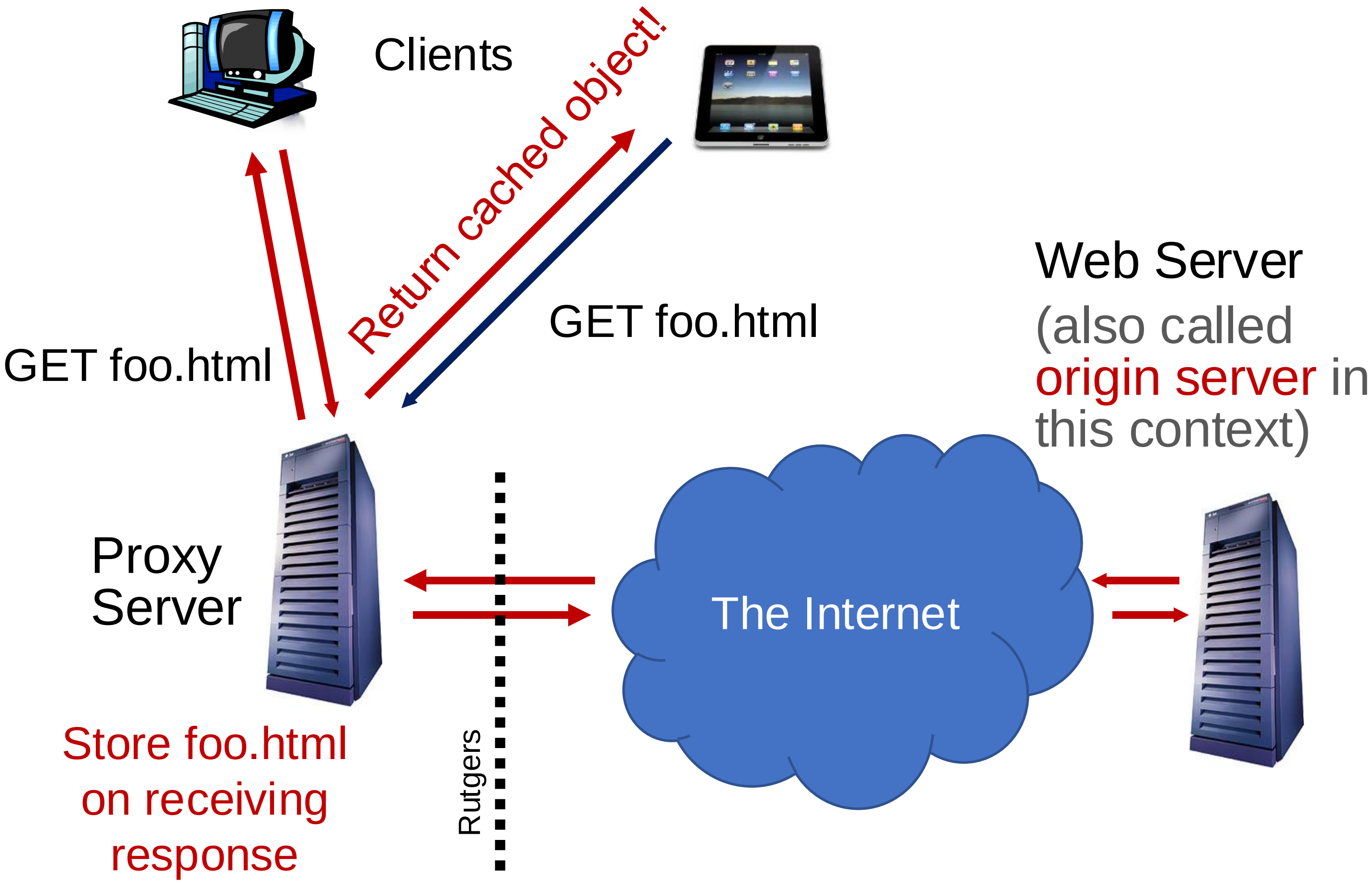
Web caches

Web caches: Machines that remember web responses for a network

Why cache web responses?

- Reduce response time for client requests
- Reduce traffic on an organization’s access link

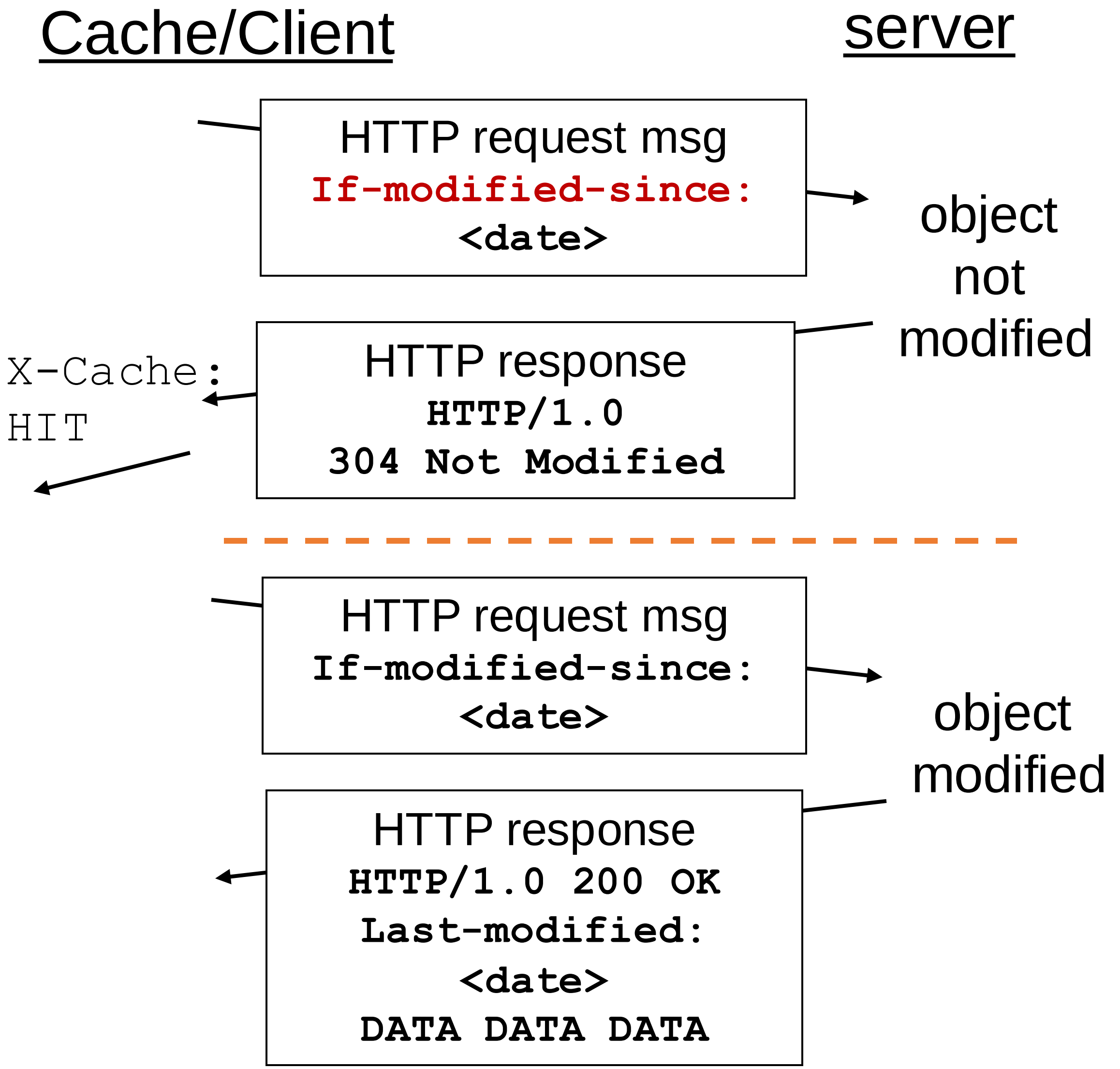
Web caching using a proxy server



- You can configure a HTTP proxy on your laptop's network settings.
- If you do, your browser sends all HTTP requests to the proxy (cache).
- Hit: cache returns object
- Miss: obtain object from originating web server (origin server) and return to client
  - Also cache the object locally

Caching in the HTTP protocol

- Conditional GET guarantees cache content is up-to-date while still saves traffic and response time whenever possible
- Date in the cache's request is the last time the server provided in its response header Last-Modified



Content Distribution Networks (CDNs)

A global network of web caches

- Provisioned by ISPs and network operators
- Or content providers, like Netflix, Google, etc.

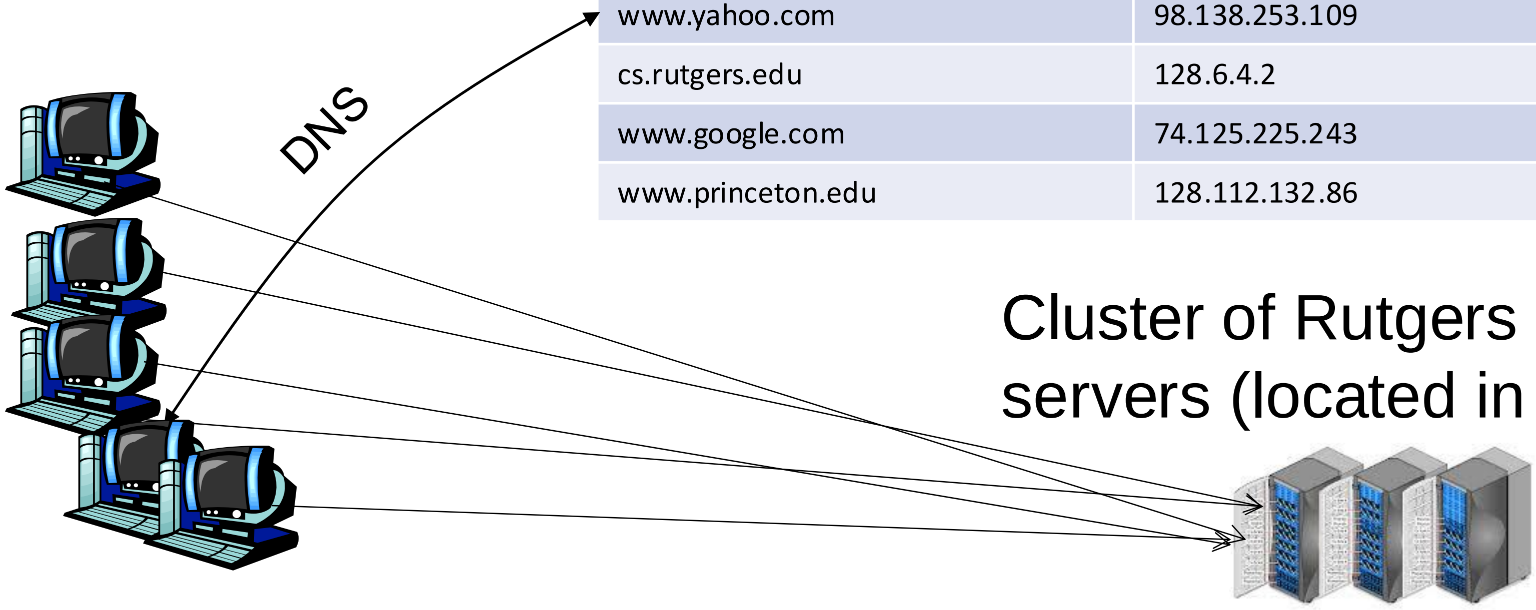
Uses (overlaps with uses of web caching in general)

- Reduce traffic on a network's Internet connection, e.g., Rutgers
- Improve response time for users: CDN nodes are closer to users than origin servers (servers holding original content)
- Reduce bandwidth requirements on the content provider
- Reduce cost to maintain origin servers

Without CDN

DOMAIN NAME	IP ADDRESS
www.yahoo.com	98.138.253.109
cs.rutgers.edu	128.6.4.2
www.google.com	74.125.225.243
www.princeton.edu	128.112.132.86

Clients distributed all over the world



Cluster of Rutgers CS origin servers (located in NJ, USA)

128.6.4.2

- Problems:
- Huge bandwidth requirements for Rutgers
- Large propagation delays to reach users

Where the CDN comes in

- Distribute content of the origin server over geographically distributed CDN servers

- But how will users get to these CDN servers?

- Use DNS!
  - DNS provides an additional layer of indirection
  - Instead of returning an IP address, return another DNS server (NS record)
    - Much like a response to any other iterative query
  - The second DNS server (run by the CDN) returns the IP address of the client

- The CDN runs its own DNS servers (CDN name servers)
  - Custom logic to send users to the “closest” CDN web server

With CDN

DOMAIN NAME	IP ADDRESS
www.yahoo.com	98.138.253.109
cs.rutgers.edu	124.8.9.8 (NS record pointing to CDN name server)
www.google.com	74.125.225.243

DNS reply

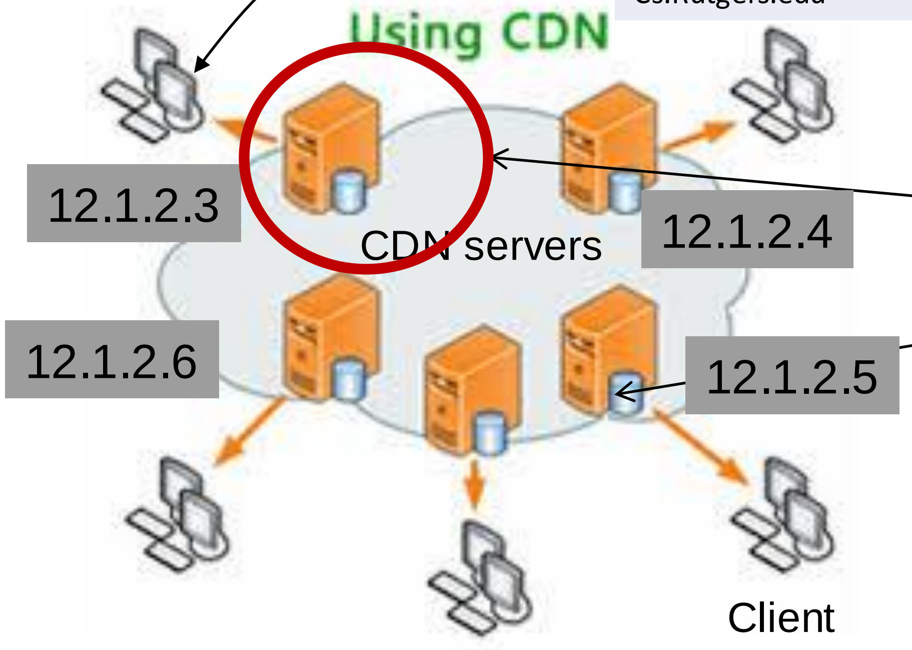
NS record delegates the choice of IP address to the CDN name server.

CDN Name Server (124.8.9.8)

DOMAIN NAME	IP ADDRESS
Cs.Rutgers.edu	12.1.2.3
Cs.Rutgers.edu	12.1.2.4
Cs.Rutgers.edu	12.1.2.5
Cs.Rutgers.edu	12.1.2.6

Custom logic to map ONE domain name to one of many IP addresses!

Popular CDNs:
CloudFlare
Akamai
Level3
…



128.6.4.2
Origin server

Most requests go to CDN servers (caches).
CDN servers may request object from origin
Few client requests go directly to origin server

Seeing a CDN in action

- `dig +trace freshtohome.com`

- `dig web.mit.edu` (or) `dig +trace web.mit.edu`

Summary of HTTP

- Request/response protocol
- ASCII-based human-readable message structures
- Enhanced stateful functionality using cookies
- Improve performance using caching and CDN
- Persistence and pipelining to improve performance
- Simple, highly-customizable protocol
    - Just add headers
- The protocol that is the basis of the web we enjoy today

Multimedia over the Internet

RUTGERS
UNIVERSITY | NEW BRUNSWICK

Internet Multimedia

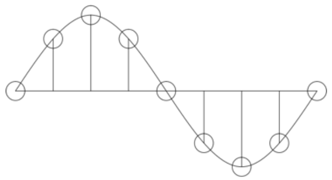
- Many applications on the Internet use audio or video

- Comparison with traditional web/HTTP:
    - Cannot tolerate loss, but a little delay may be ok
    - Data used after the transfer is complete

- Multimedia is more real-time
    - Performance *during* the data transfer matters

Digital representation of audio and video

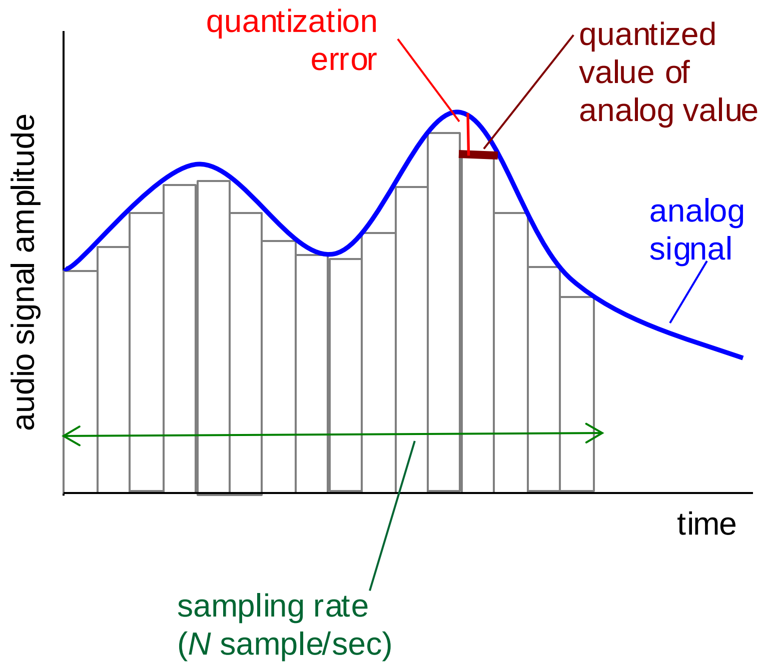
Digital representation of audio



- Must convert analog signal to digital representation
- Sample
  - How many times (twice the max frequency in the signal)
- Quantize
  - How many levels or bits to represent each sample
  - More levels ➔ more accurate representation of signal
  - More levels ➔ more bits to store & need more bandwidth to transmit
- Compress
  - Compact representation of quantized values

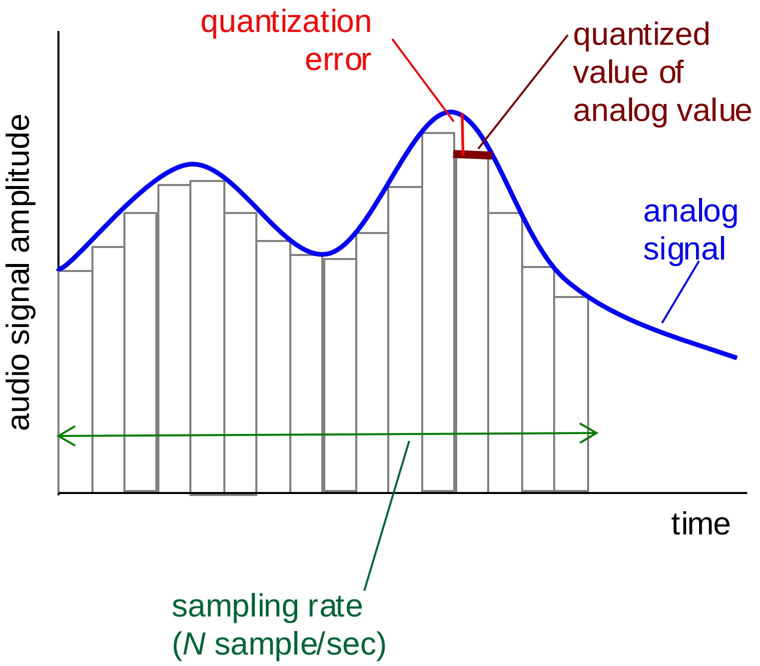
Audio representation

- 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
  - each quantized value represented by bits, e.g., 8 bits for 256 values



Audio representation

- example: 8,000 samples/sec, 256 quantized values
- Bandwidth needed: 64,000 bps
- receiver converts bits back to analog signal:
  - some quality reduction

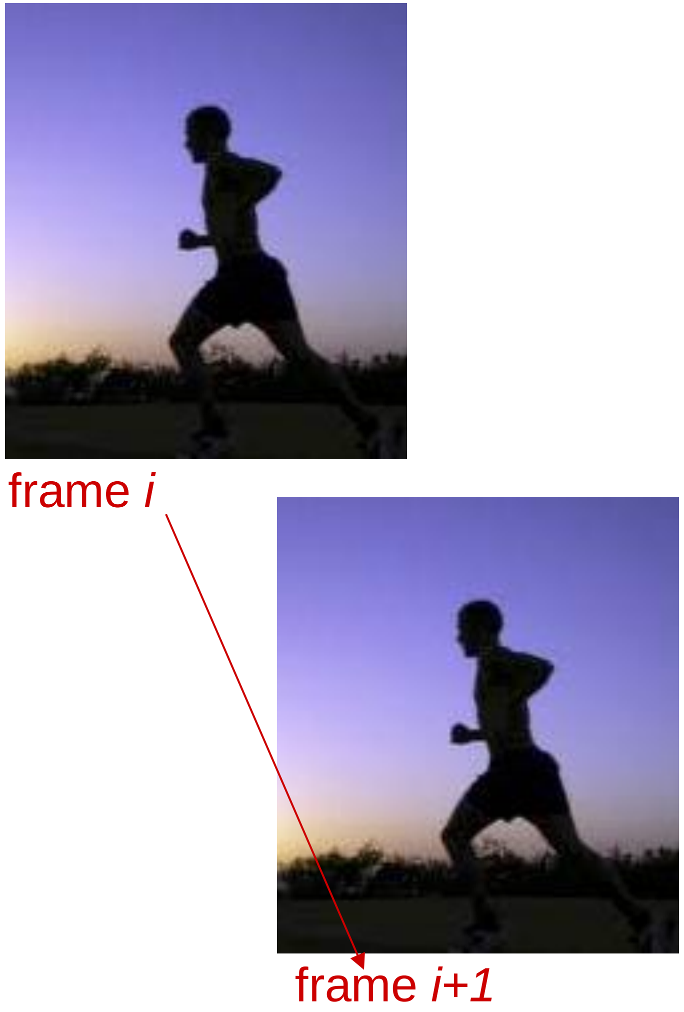


Example rates

- CD: 1.411 Mbps
- MP3: 96, 128, 160 Kbps
- Internet telephony: 5.3 Kbps and up

Video representation

- Video: sequence of images displayed at constant rate
  - e.g., 30 images/sec
  - Appear continuous due to the stroboscopic effect



Video representation

- Digital image: array of pixels
  - each pixel represented by bits
  - Encode luminance and color
  - Number of pixels: resolution
- Coding: use redundancy *within* and *between* images to decrease # bits used to encode image
  - spatial (within image)
  - temporal (from one image to next)
- Encoding/decoding algorithm often called a codec

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 (motion vectors)

frame *i+1*

Video codecs: terminology

- Video *bit rate*: effective number of bits per second of the video after encoding
- It depends on many factors
  - Resolution of each image: more pixels = more bits
  - Detail per pixel: more luminance & color detail = more bits
  - Amount of movement in the video. More movement = more bits
  - Quality of overall compression in the codec
- Video bit rate is typically correlated with quality of perception
  - Higher bit rate == better to perceive

Bit-rates: terminology

- Bit-rate of a video changes over the duration of the video
- CBR: (constant bit rate): fixed bit-rate video
- VBR: (variable bit rate): different parts of the video have different bit rates, e.g., changes in color, motion, etc.
  - For VBR, we talk about average bit-rate over video's duration
- Examples of average video bit-rates
  - MPEG 1 (CD-ROM) 1.5 Mbps. MPEG2 (DVD) 3-6 Mbps
  - MPEG4 (often used in Internet, < 1 Mbps)
  - In general, one Internet video stream takes up a few Mbit/s (more for HD)

https://blog.video.ibm.com/streaming-video-tips/what-is-video-encoding-codecs-compression-techniques/

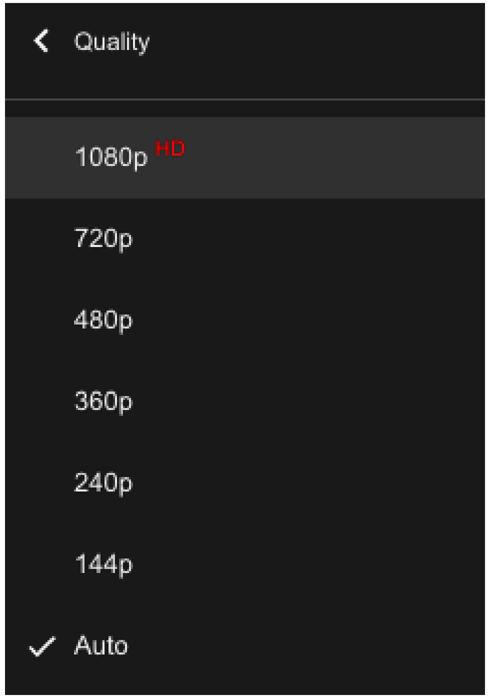
Networking multimedia: 3 types

- On-demand streamed video/audio
  - Can begin playout before downloading the entire file
  - Ful video/audio stored at the server: able to transmit faster than audio/video will be rendered (with storing/buffering at client)
  - e.g., Spotify, YouTube, Netflix
- Conversational voice or video over IP
  - interactive human-to-human communication limits delay tolerance
  - e.g., Zoom, Google Stadia
- Live streamed audio, video
  - e.g, sporting event on sky sports
  - Can delay a little, but must be close to the “live edge” of content

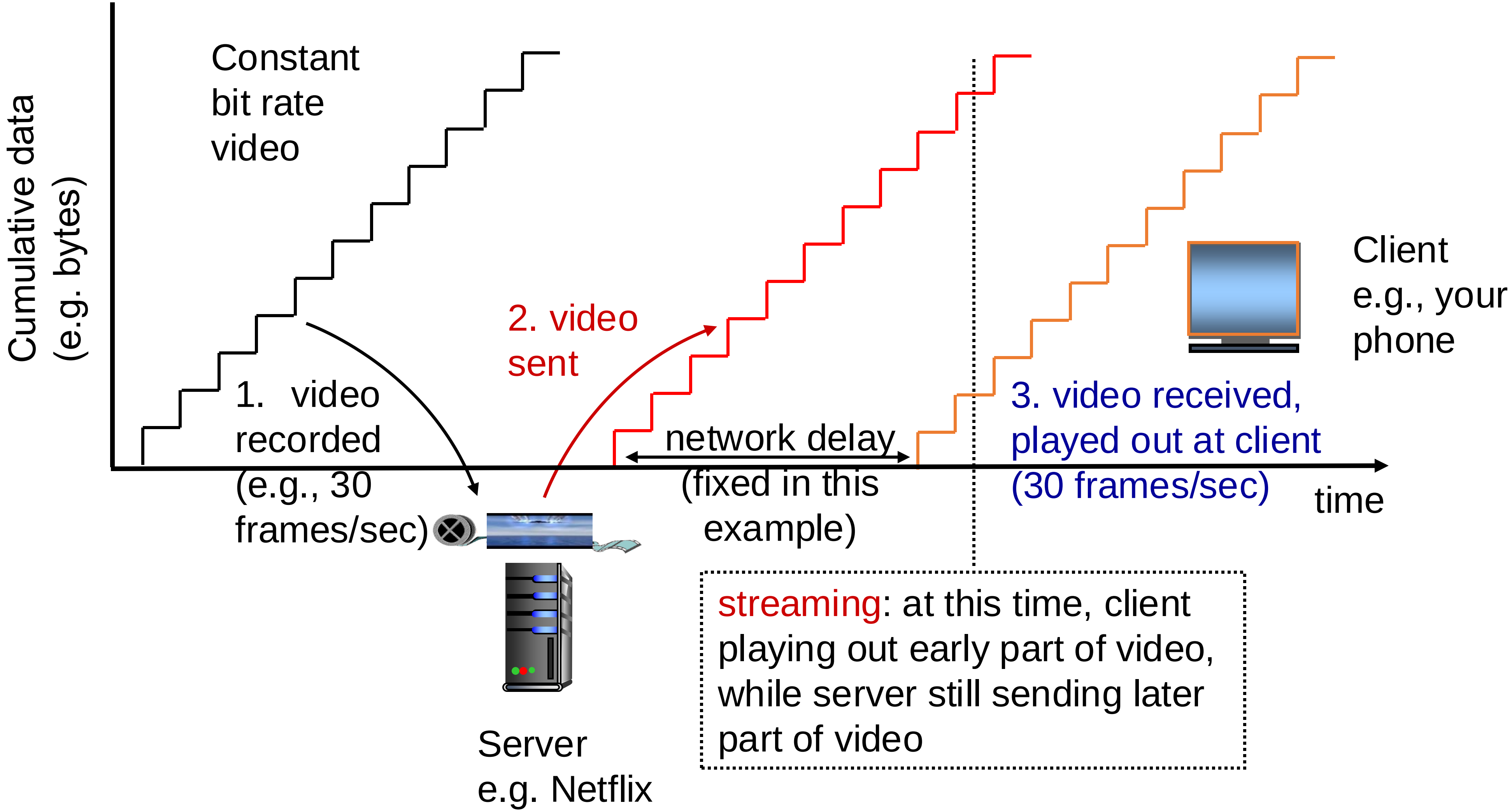
On-demand Video Streaming

Streaming (stored) video

- Media is prerecorded at different qualities
  - Available in storage at the server
- Client downloads an initial portion and starts viewing
  - The rest is downloaded as time progresses
  - No need for user to wait for entire content to be downloaded!
- Can change the quality of the content and where it’s fetched mid-stream
  - More on this soon



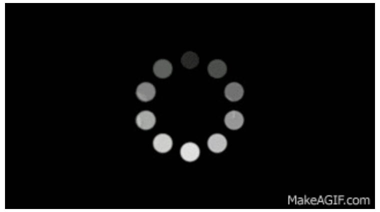
Streaming stored video



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

Streaming stored video: challenges

- Continuous playout constraint: once video playout begins at client, time gap between frames must match the original time gap in the video (why?)

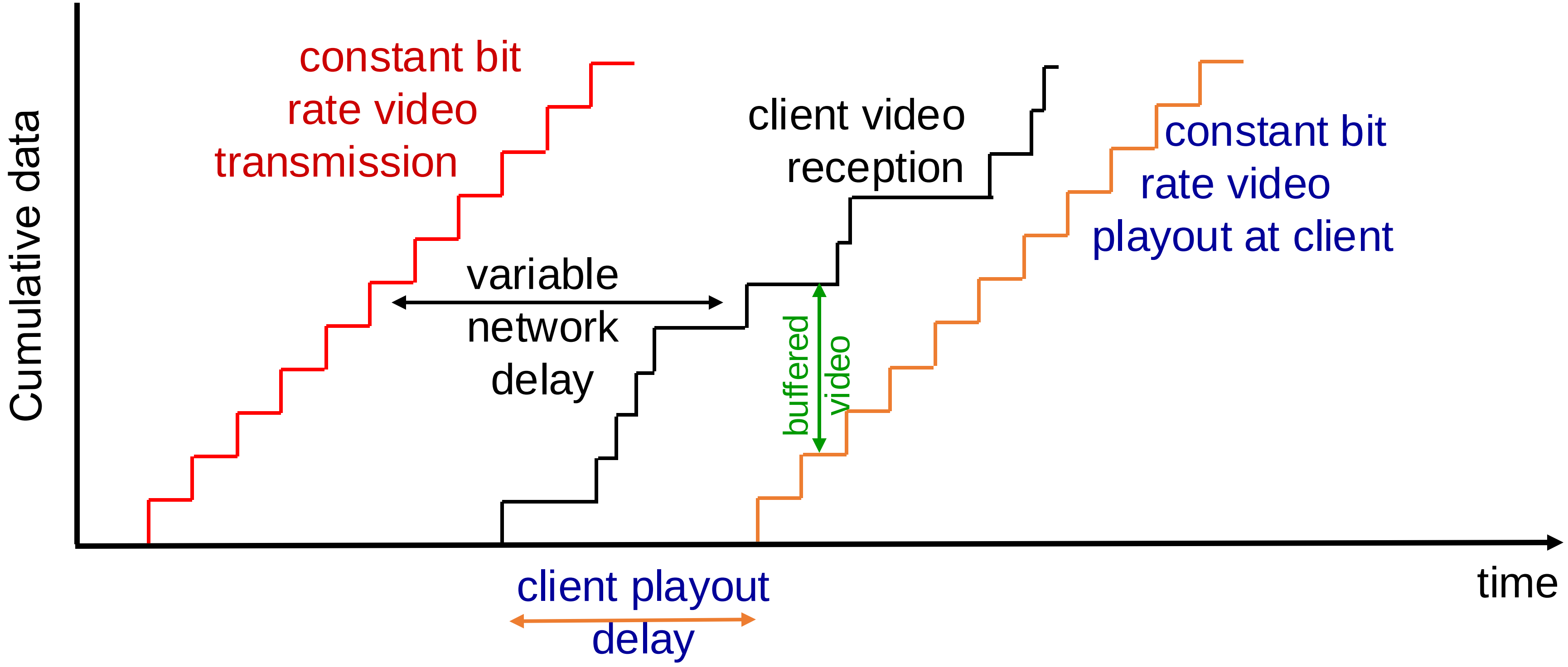


- But network delays are variable!

- Clients have a client-side buffer of downloaded video to absorb variation in network conditions

- Buffer also helps with user interactions: pause, fast-forward, rewind, jump through video

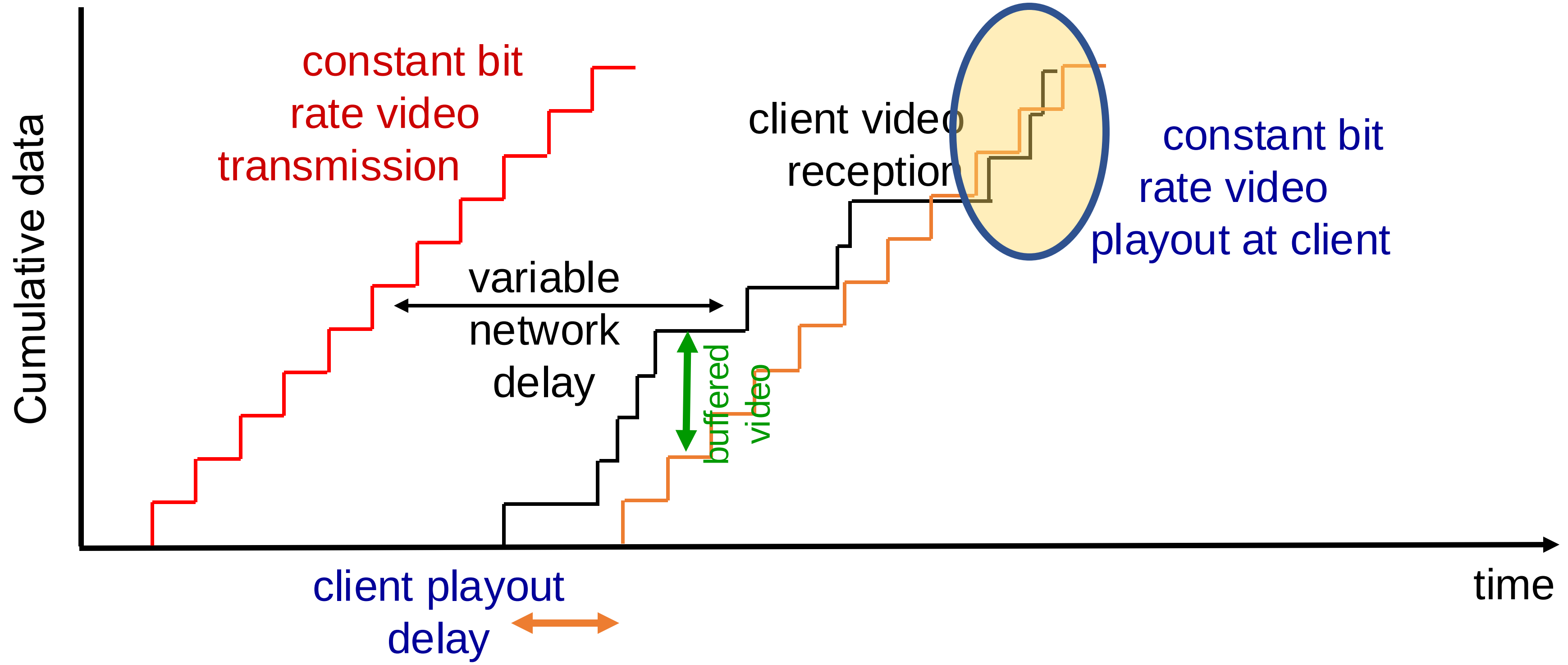
Scenario 1: Constant bit-rate video



Client-side buffering with playout delay:

compensate for network-added delays and variations in the delay

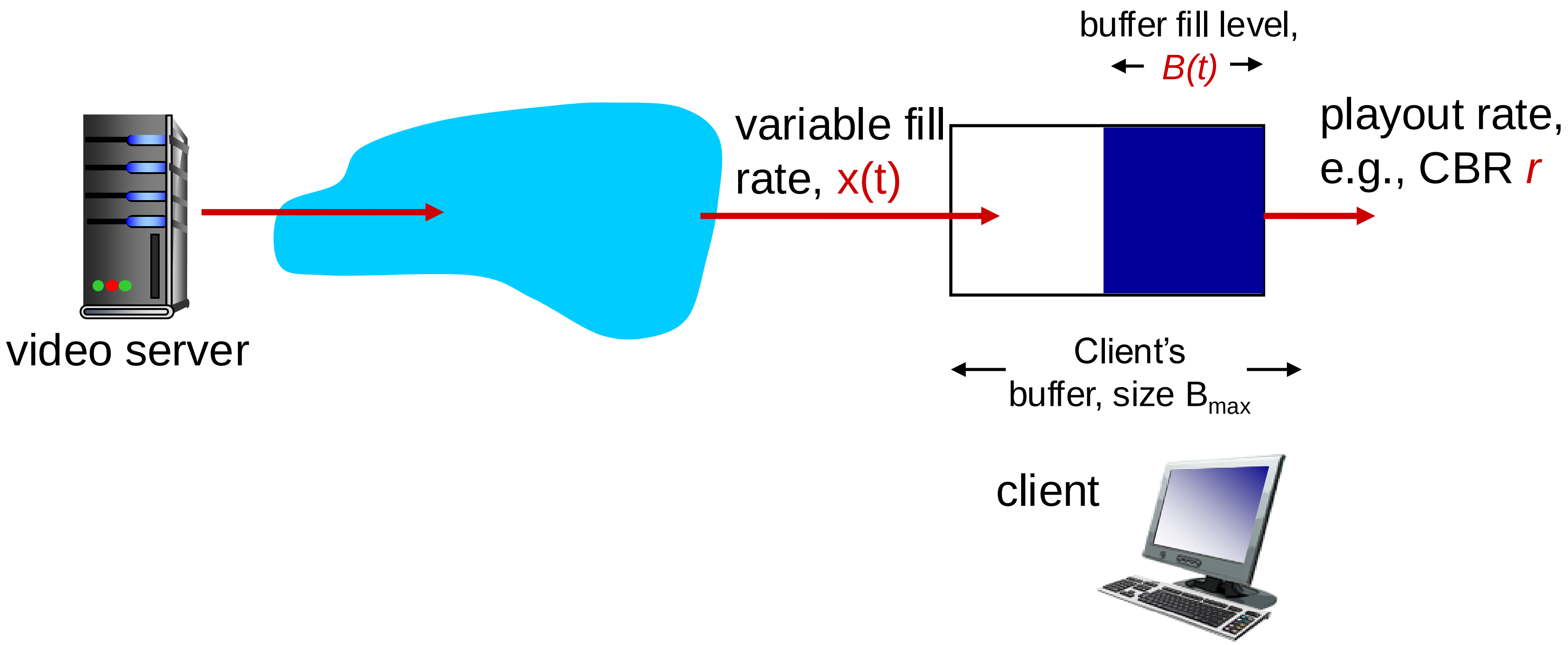
Scenario 2: Small playout delay



Playout delay that's too small can cause stalls

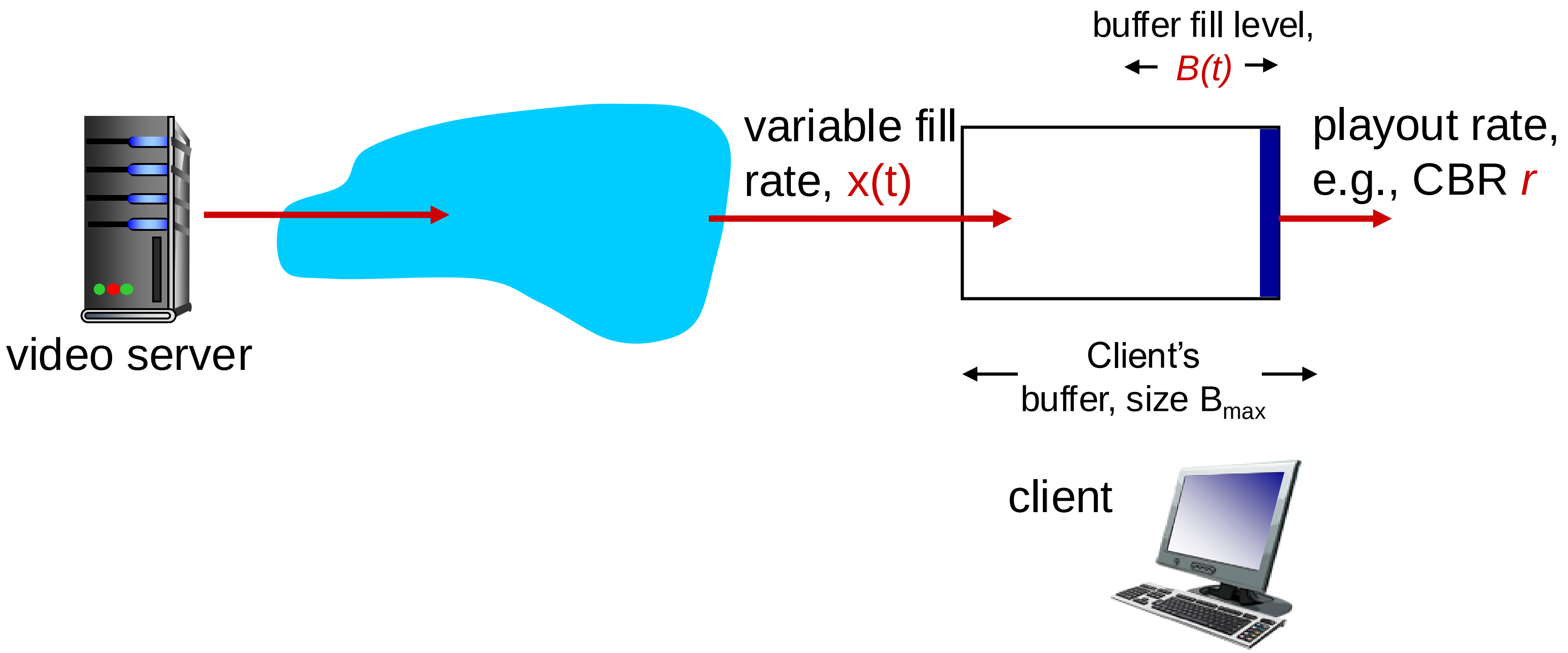
There's nothing in the buffer to show to the user

Client-side buffering, playout



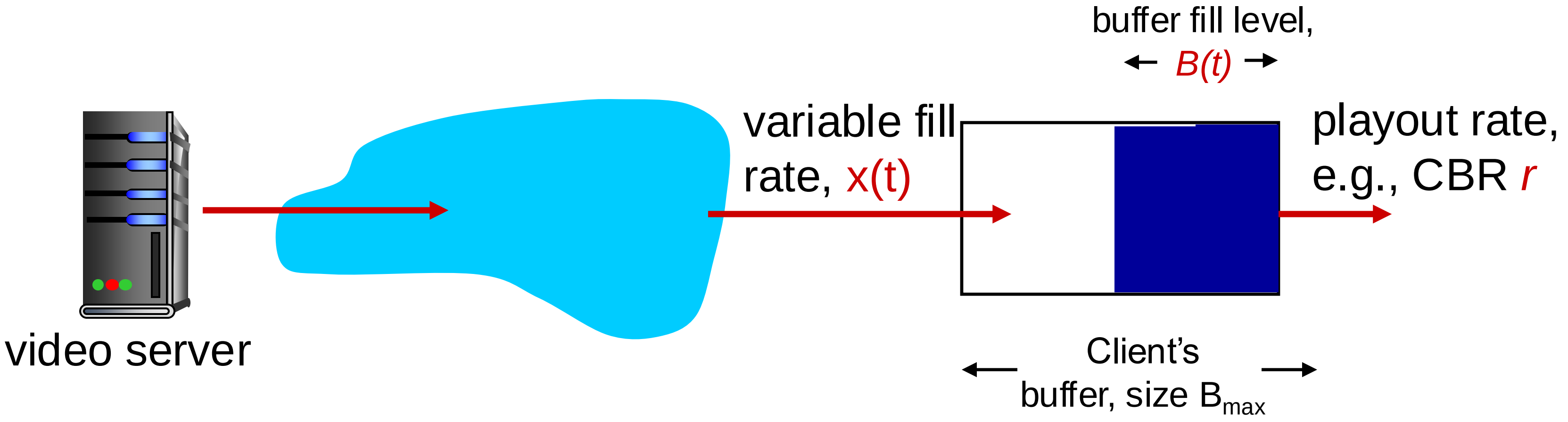
Most video is broken up in time into multiple segments
Client downloads video segment by segment
For example: a segment might be 4 seconds worth of video.

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 (assume playout rate r is constant for now)

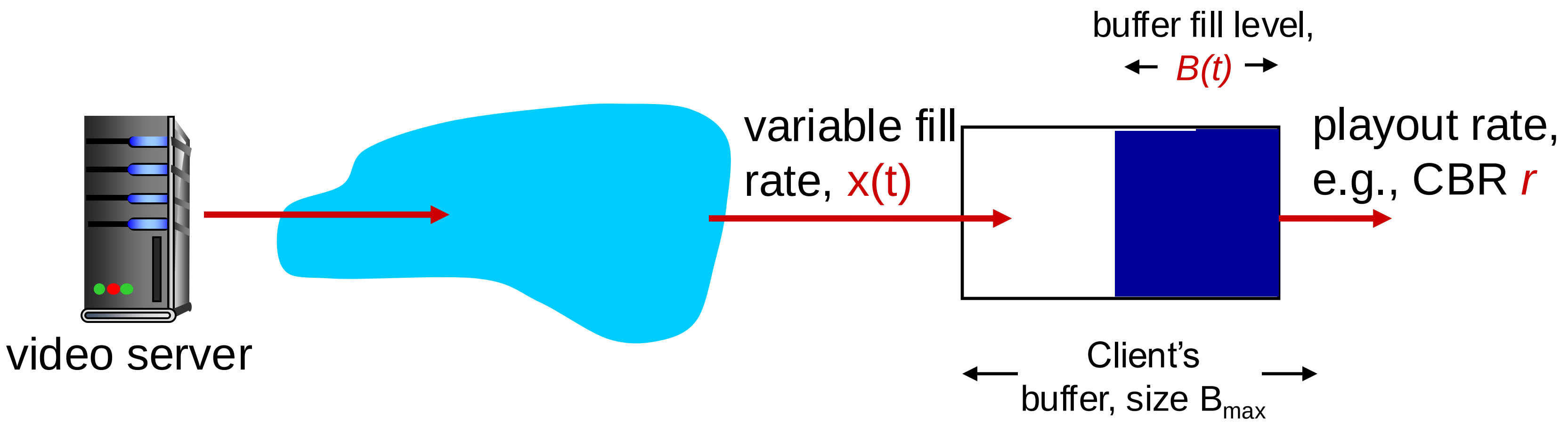
Client-side buffering, playout



playout buffering: average fill rate ($\bar{x}$), playout rate (r):

- $\bar{x} < r$: buffer eventually empties for a sufficiently long video. Stall and rebuffering
- $\bar{x} > r$: buffer will not empty, provided the initial playout delay is large enough to absorb variability in x(t)
  - *initial playout delay tradeoff:* buffer starvation less likely with larger delay, but also incur a larger delay until the user begins watching

Client-side buffering, playout

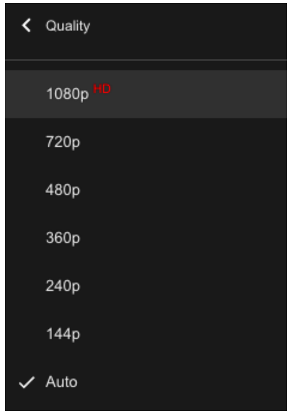


playout buffering: average fill rate ($\bar{x}$), playout rate (r):

- is $\bar{x}$ < r or $\bar{x}$ > r for a given network connection?
- It is hard to predict this in general!
  - Best effort network suffers long queues, paths with low bandwidth, …
- How to set playout rate r?
  - Too low a bit-rate r: video has poorer quality than needed
  - Too high a bit-rate r: buffer might empty out. Stall/rebuffering!

Adaptive bit–rate video

- Motivation: Want to provide high quality video experience, without stalls
- Observations:
  - Videos come in different qualities (average bit rates)
  - Versions of the video for different quality levels readily available
  - Different segments of video can be downloaded separately



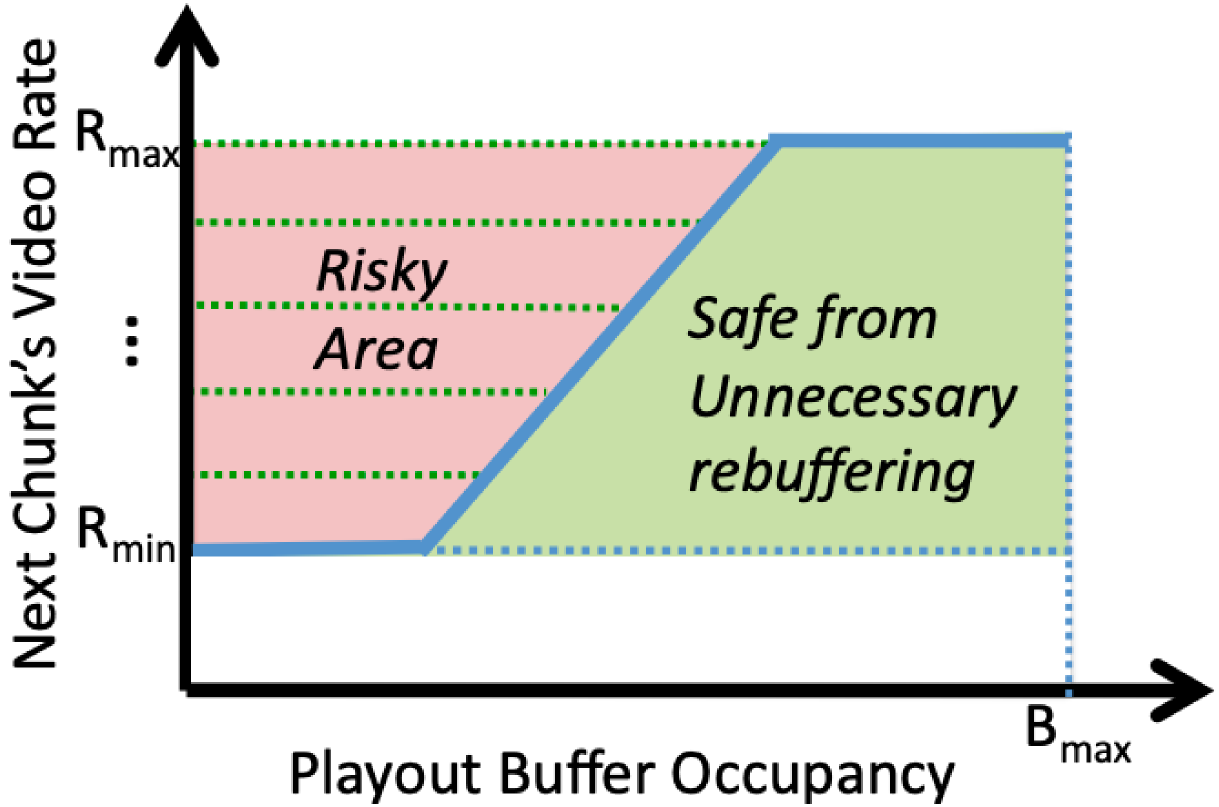
- Adapt bit rate per segment through collaboration between the video client (e.g., your browser) and the server (e.g., @ Netflix)
- Adaptive bit-rate (ABR) video: change the bit-rate (quality) of next video segment based on network and client conditions
- A typical strategy: Buffer-based rate adaptation

Buffer-based bit-rate adaptation

- Key idea: If there is a large stored buffer of video, optimize aggressively for video quality, i.e., high bit rates

- Else (i.e., buffer has low occupancy), avoid stalls by being conservative and ask for a lower quality (bit-rate)
  - Hope: lower bandwidth requirement of a lower quality stream is satisfiable more easily

Buffer-based bit-rate adaptation



A highly effective method to provide high video quality despite variable and intermittently poor network conditions.

Used by Netflix.

http://yuba.stanford.edu/~nickm/papers/sigcomm2014-video.pdf

A Buffer-Based Approach to Rate Adaptation