FEUP
Departamento de Engenharia Electrotécnica e de Computadores

Multimedia systems - M.EEC0057

M.EEC 2023-2024

Multimedia streaming protocols Part3: Challenges for HAS

What will we learn?

- Existing challenges for HAS
- The use of client buffers and consequences on the media presentation
- Advances in low latency ABR approaches
- Standardised solutions to reduce latency
- Client implementations

HTTP Adaptive Streaming (HAS)

- existing HTTP adaptive streaming technologies (aka HAS) share the same approach of
  - generating multiple variations of the content
  - fragmenting it in segments
  - assigning to the client the freedom of independently download and decode such segments
- support dynamic adaptation of the multimedia presentation according to network conditions
- require the generation of additional metadata (manifest files)
  - with bit rates and URIs to access the content segments
  - using the manifest file, the client issues HTTP requests for consecutive segments with a target bitrate, matching the present network conditions as found by an Adaptive Bitrate (ABR) algorithm

Goals & challenges of HAS

- What and How
  - maximise the user Quality of Experience (QoE)
    - minimise/avoid the most annoying aspects of traditional streaming
      - frozen images or jumps in the playout due to buffer over and underflow
  - no need for dedicated server
  - dynamically adapt to network variability
    - given that network conditions will vary from client to client, empower the clients to decide by themselves
      - use additional metadata files to enable this
- Challenges
  - large latency, especially in live content services, as clients can only request segments after they become available on the server

What is Quality of Experience (QoE) in video streaming?

- refers to the overall satisfaction and perceived quality of the user's experience when consuming multimedia content over a network
- it is a **subjective** measure that encompasses various aspects of the streaming service, considering the end user's perception, expectations, and enjoyment
  - QoE reflects how users actually experience the audiovisual content
    - handled at the application layer
  - by comparison, Quality of Service (QoS), focuses on technical metrics like bitrate, latency, and packet loss
    - handled at the transport, network and physical layer

Key factors impacting QoE in video streaming (1)

- Video Quality
  - resolution (spatial, temporal, quality), clarity, and absence of visual artifacts contribute to video quality
- Audio Quality
  - clarity, fidelity, and consistency of audio playback are crucial
  - users expect high-quality audio that complements the video content
- Buffering and playback Smoothness
  - users prefer uninterrupted streaming with minimal buffering delays
  - smooth playback without stuttering or freezing is the great contributor to enhancing the viewing experience
- Start-up time/latency
  - time it takes for the video to start playing is a critical factor
  - fast start-up times contribute to a positive initial impression and overall satisfaction

Key factors impacting QoE in video streaming (2)

- Consistency
  - to ensure playout consistency/continuity, adaptive streaming technologies adjust video quality based on network conditions
    - but they ruin video quality consistency …
- User Interface and Interactivity
  - the design and usability of the streaming application impact user satisfaction
    - intuitive interfaces and interactive features contribute positively to QoE
- Content Relevance
  - relevance and quality of the content influence the overall enjoyment of the streaming experience
  - personalization and content recommendations can enhance QoE
- Consistency Across Devices
  - users expect a consistent experience across different devices in terms of offered functionality and content quality

Assessing QoE

- subjective methods are normally used
  - user surveys, interviews
  - quality ratings provided by viewers (polls)
  - dedicated viewing sessions implementing the Mean Opinion Score (MOS) method which is a “double stimuli” procedure
    - a heterogeneous audience visualises images and classifies them in a scale from 1 to 5
      - each person visualises the original image (with full quality, uncompressed) immediately followed by a compressed (and reconstructed) version
      - the score is given in a relative way by comparing the two images
        - 1 if the two images are very dissimilar (low quality / high degradation)
        - 5 if the two images are very similar (high quality / imperceptible degradation)

Challenges of HAS

- high end-to-end latency in live services
  - long delay between the instant content is captured and the instant the content is presented to the user
    - typically there is a 30-90s time gap in relation to a terrestrial or satellite transmission
- end-to-end latency depends on the entire delivery chain
  - content capture, encoding, packaging, encryption
  - CDN distribution
  - buffering and rendering at the client
- but client buffering is the main contributor
  - and the differentiating factor in relation to traditional broadcasting

Client buffering and ABR

- a client buffer is necessary to absorb changes in network conditions
  - to queue some received segments targeting continuous playback, thus providing a good viewing experience
- but, only an infinite buffer would be able to cope with any amount of congestion …
  - therefore, the client implements an Adaptive Bit Rate (ABR) that measures the network conditions periodically and adjusts the quality at which segments are requested
  - the size of the client buffer is defined according the the desired reaction time of the ABR algorithm and the maximum latency
    - how long does the algorithm have to evaluate the network conditions and react to deterioration before the client buffer underflows, freezing the presentation?

Client buffering and ABR (2)

- the maximum number of segments that are queued in the client buffer is determined based on the client's overall target latency
  - in live streaming, this latency will be the time-gap that will exist between broadcast and streaming services of the same live media program, so it should be as small as possible
    - because it is only possible to request segments in real-time …
    - in offline services (VoD for example) there is not such a big problem of time-gap
- the minimum number of segments that need to be queued in the client dictates the minimum achievable latency
- therefore, to aim low-latency in live streaming, it is necessary to have sophisticated ABR algorithms that use very small buffers and thus are able to react very fast

Client buffering and ABR (3)

- the main goal of an ABR algorithm is to maximise viewer QoE
  - translated into maximising the average video quality
    - ensuring playout continuity
      - preventing freezing and jumps
- however, changes in quality are also annoying so they should be minimised

$$changes\ in\ quality \Leftrightarrow rebuffering\ events$$

- the number of times quality is changed and the periods during which quality drops should be minimised
  - the target is thus to comply with network variability whilst minimising frequency and duration of rebuffering events

Solutions to reduce latency

- Specifications:
  - MPEG Common Media Application Format (CMAF).
    - defines the format to fragment DASH segments into chunks
  - Low Latency DASH (LL-DASH)
  - Low Latency HLS (LL-HLS)
    - Extension to HLS developed by Apple in 2020
  - HTTP/1.1 Chunked Transfer Encoding
    - An IETF RFC
- Player implementations:
  - DASH.js
  - HLS.js
  - Shaka player
  - Apple's AVPlayer

Low Latency solutions

- common design principles
  - each media segment is fragmented into smaller chunks
    - each chunk can be delivered as soon as it is available
    - a player can start rendering a segment right after its first chunk is received
  - the latency is thus reduced from the segment level to a chunk level
    - minimum latency was one complete segment period
      - which was 10 seconds for HLS
    - now it is only one chunk period
      - which can be a few hundred milliseconds

Low Latency solutions (2)

- reducing the segment size could be another possible approach, but
  - by definition in HAS, a segment is the unit that allows to switch between quality/bit rate
    - switching is possible to occur at the segment frontier given that each segment starts with an Intra frame
      - allowing a player to start the playback of a segment immediately, without the need to wait and download an earlier segment
      - and equally numbered segments of different versions correspond to the same point in time
  - to maintain such possibility, an I frame would have to be squeezed into a smaller number of bytes
    - quality deterioration

Low Latency solutions (2)

- Given that low-latency streams are delivered chunk by chunk
  - enabling to reduce the size of the buffer (and latency) at the client-side
- estimating the network bandwidth and making stream adaptation decisions becomes even more challenging

LL-HLS

- two approaches to reduce latency
  - enabling HTTP/1.1 chunked transport for segments
  - announcing segments before they are available
- LL-HLS provides backwards compatibility with regular/legacy HLS
  - players that do not understand the protocol extension will still be able to play the same streams, but with a higher latency. …
  - a single optimised server solution can be used both for optimised and non-optimised players
- it could also use HTTP/2 features such as HTTP Push and Ping
  - but there is not yet sufficient support of such features namely in CDNs

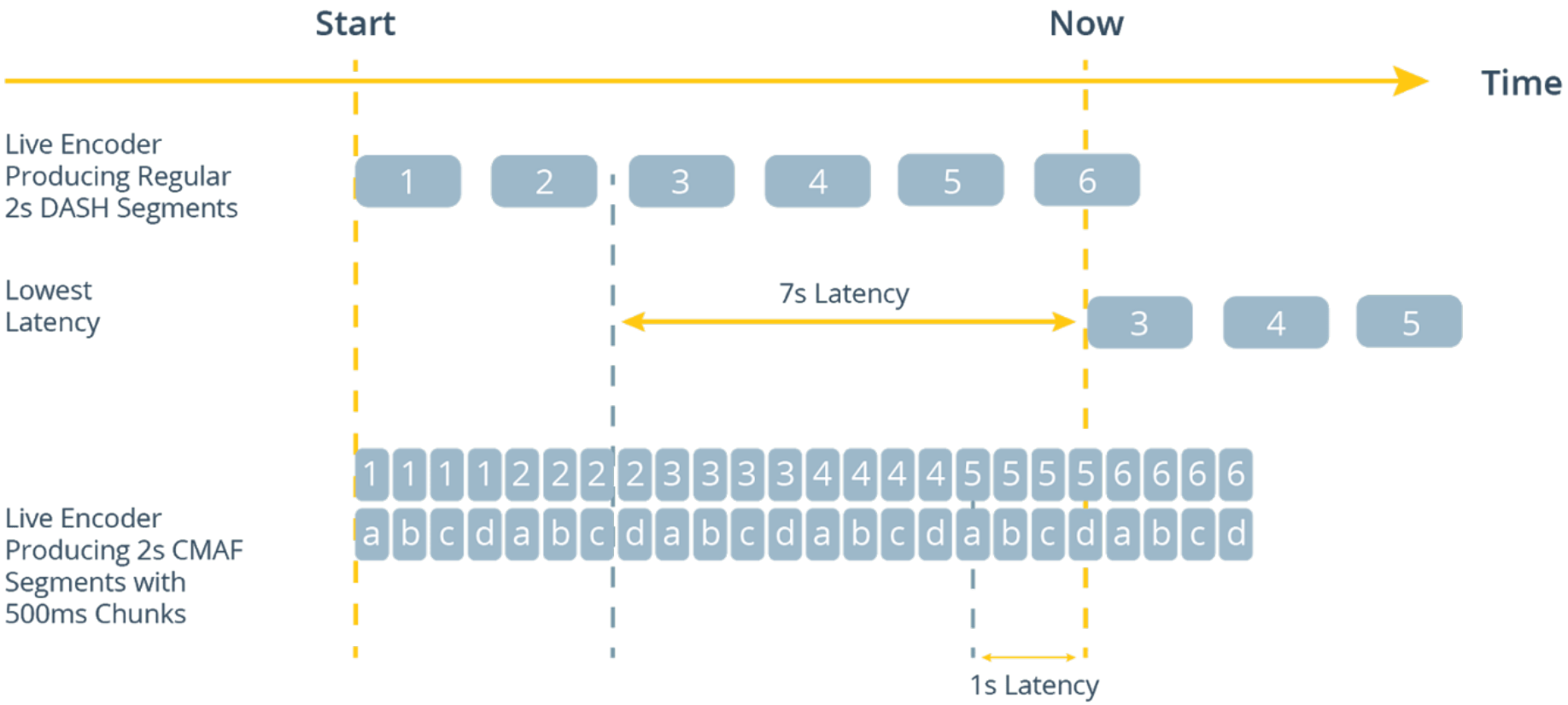
LL-HLS (2)

- Introduces a new tag
  - #EXT-X-PRELOAD-HINT
  - used by a server offering a low latency live HLS stream to announce the most likely location of the next chunk needed to continue playing the video at the client side
  - allows a player client to perform a request in anticipation, allowing the data to flow in as soon as it is available
    - this process allows to further reduced latency caused by the round-trip time when requesting new media data
      - the intended use for use HTTP/2 push

LL-DASH

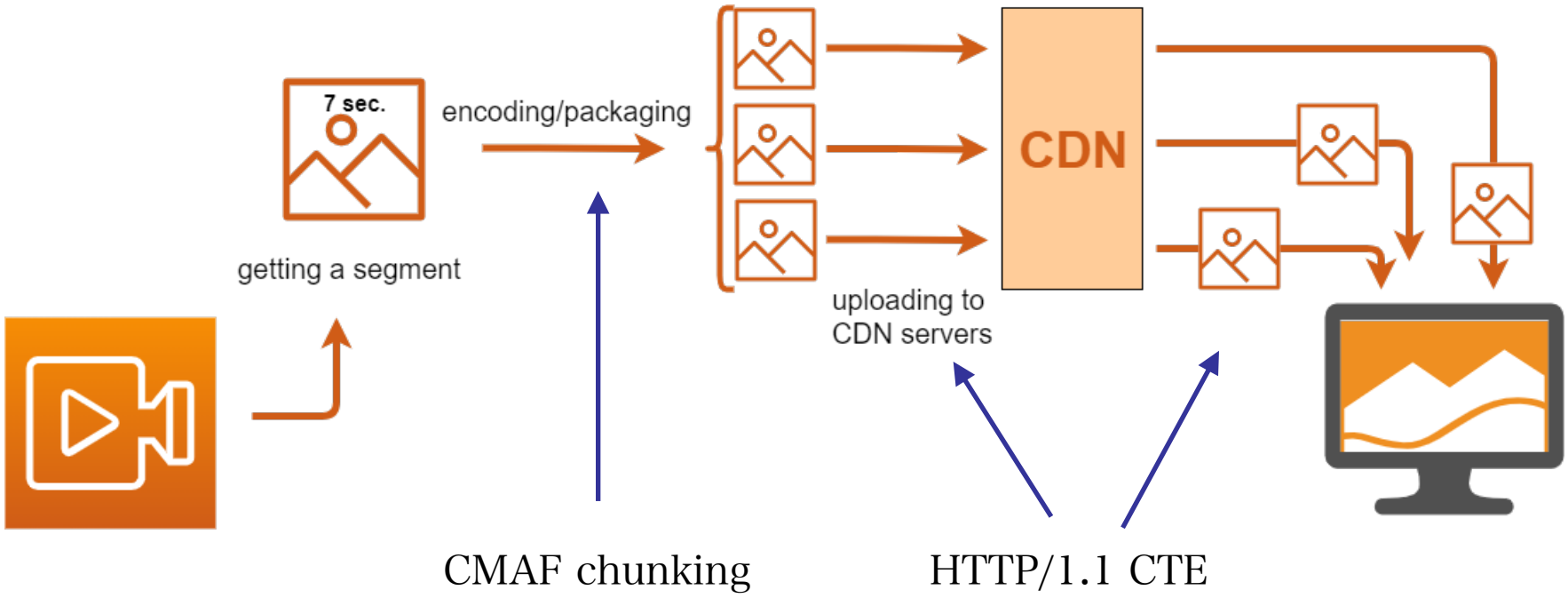
- LL-DASH uses CTE and is compatible with the Common Media Application Format (CMAF)
  - CMAF provides some tools that enable DASH to reduce latency
    - namely, it defines how to split up DASH media segments in smaller chunks and describe how such splitting was done
- sometimes LL-DASH is called DASH/CMAF/CTE …

LL-DASH (2)



- Assumes media segments are not smaller than 2 seconds and that the minimum buffer size to ensure smooth layout is 3-4 segments

LL-DASH (3)


7 sec.
encoding/packaging
CDN
getting a segment
uploading to
CDN servers
CMAF chunking
HTTP/1.1 CTE

HTTP\1.1 CTE

- CTE is signalled by using the transfer-encoding HTTP response header
  - instead of the usual content-length header of HTTP packets
- CTE provides the means for a client to verify that all the chunks, and then the complete file, have been received
- the main advantage of CTE is that it eliminates the preparatory work and delay required when sending large files or segments of data

Useful sources of information

- https://www.wowza.com/blog/low-latency-cmaf-chunked-transfer-encoding
- https://www.wowza.com/blog/what-is-cmaf
- https://mpeg.chiariglione.org/standards/mpeg-a/common-media-application-format/text-isoiec-cd-23000-19-common-media-application
- Abdelhak Bentaleb, Bayan Taani, Ali C. Begen, Christian Timmerer, and Roger Zimmermann, "A Survey on Bitrate Adaptation Schemes for Streaming Media Over HTTP". In IEEE Communications Surveys & Tutorials, Vol. 21, No. 1, first quarter 2019.
- Bo Zhang, Thiago Teixeira, and Yuriy Reznik. 2021. Performance of Low- Latency HTTP-based Streaming Players. In Proceedings of ACM Multimedia Systems conference (MMSys'21). ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/1234567890. Available at: https://dl.acm.org/doi/pdf/10.1145/3458305.3478442
- Video: https://www.facebook.com/watch/?v=616758079514649
- https://www.theoplayer.com/blog/evolution-of-ll-hls
- https://www.theoplayer.com/blog/low-latency-dash