DASH
Industry Forum

Server-Side Watermarking

DASH-IF End-to-End Architecture

DASH-IF Webinar – April 13, 2022

Presenters: Gwenaël Doërr (Synamedia), Nicolas Weil (AWS), Marcelo San Martin (Akamai)

With the support of Vladimir Zivkovic (Irdeto) and Laurent Piron (Nagravision)

Contents

Watermarking, forensic watermarking

- Recap and challenges in OTT deployments

DASH-IF end-to-end architecture

- Deep dive in the proposed specification
  - Content preparation
  - Content consumption

Conclusion

- Community Review
- Next steps

Forensic Watermarking

Forensic watermarking embeds an imperceptible unique identifier to each content copy for the purpose of tracking individual copies to deter unauthorized redistribution

- Forensic watermarking remains in the content after the other content protection features are removed (i.e., after DRM/CAS decryption is applied, after the content has left the Secure Video Path, etc.)

Various technologies and vendors

- Uncompressed (modify the baseband) or compressed (bitstream) video content
- May or may not require metadata / original content to retrieve the watermark payload

Has reached maturity and follows video evolution

- MPEG2-video to VVC or AV1, SDR to HDR

The main KPIs for a watermarking technology are

1. Fidelity
2. Robustness
3. Payload capacity
4. Computational complexity
5. Reliability

TV Use Cases

Content sharing

- Piracy transition from control word sharing to content sharing on pirate OTT platforms
- Not just the "casual" streaming on Internet
- More and more paid-for pirate services
  - OTT/IP STBs, Kodi add-ons, browser applications

MovieLabs security requirements for watermarking

- Hollywood Pre-release and UHD content

Live sport

- Required by content owners for tracking and taking-down streams

Integration and Deployment

For maximum efficiency, need to cover

- All distribution network
  - Broadcast, unicast, multicast, OTT
- All devices
  - Trusted, open, access to the software …

Diversity of non-functional requirements for watermarking

- Strong real time and latency constraints for Live
- Golden-eye level quality for premium VOD content

Already integrated in contribution (between networks), broadcast

Current “weak point” is OTT

- Huge diversity of (open) devices

Forensic Watermarking Deployments – ABR Content

Client-side watermarking

- On the rendering pipe / overlays
- Leverage legacy broadcast deployments
- Hard to maintain trust model with open devices

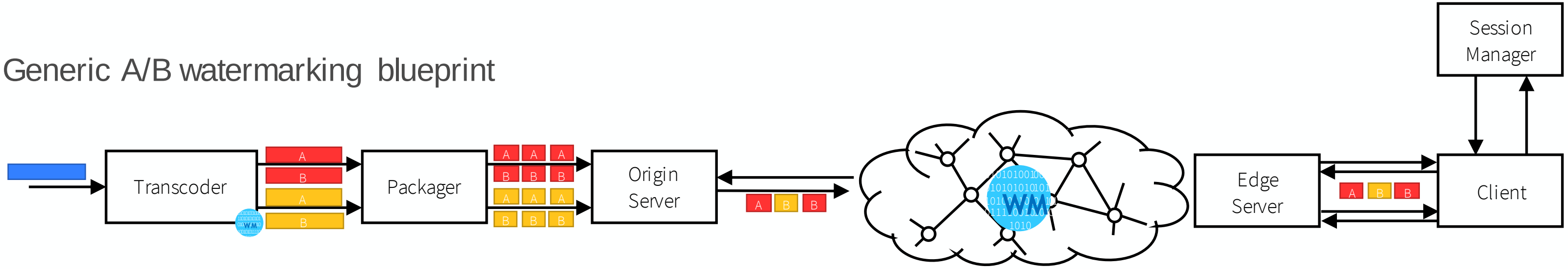
A/B watermarking
with playlist manipulation

- Unique playlist per device
- Limited scalability for Live content
- Requires explicit declaration of segment
  - No interoperability with DASH templates

A/B watermarking
with redirection at the edge

- Same playlist delivered to all devices
- Token-based A/B redirection logic implemented at the edge

Generic A/B watermarking blueprint



DASH-IF Architecture

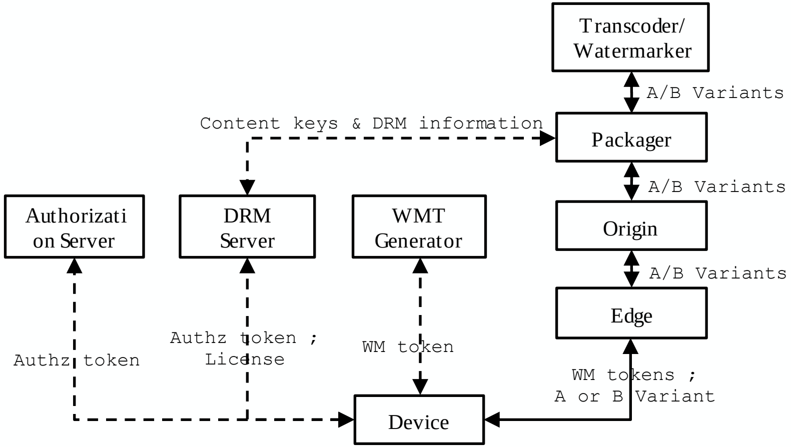
Address the pain points of A/B watermarking with redirection at the edge

- Embedding logic relying on ad-hoc naming conventions
- Brittle synchronization between watermarking components placed at encoder / edge
- Limited support for byte-range requests

End-to-end architecture that impacts encoder / packager / edge / device

Straightforward A/B routing logic at the Edge

- Watermark token containing a unique A/B pattern transmitted to the Edge by the device
- Explicit metadata about which index of the A/B pattern shall be considered for a given segment transported from the encoder to the Edge



WMPaceInfo Metadata

Alternative to naming conventions, resistant to time misalignment

Convey metadata from the encoder to the edge

- Defined per Variant for every segment in the encoder
- Almost the same for A and B Variant

Attribute	Producer	Consumers	Purpose
iswm	Transcoder	Packager, CDN	WM trigger
variant	Transcoder	CDN	Integration, debugging
pos	Transcoder	CDN	Bit position in the WM pattern
firstpart	Transcoder	Packager, Origin	Egress packaging
nbpart	Transcoder	Packager, Origin	Egress packaging

Delivery options

- From the encoder to the Origin
  - SEI, TS Adaptation Field, ISOBMFF box, HTTP header, JSON side car file
  - Rationale: diversity to account the many existing ingest protocols
- From the Origin to the edge
  - Side car file as part of the response to a specific HTTP request
  - Rationale: leverage on existing cache mechanisms at the edge and the side car file carries additional information

Content Preparation

The Packager/Origin manage A & B Variants and associated `WMPaceInfo`

- Needs to have ingest rules (proposal based on DASH MPD and HLS playlists)
- If required, the Packager shall extract `WMPaceInfo` from the ingest content (and zero-pad it)
  - `WMPaceInfo` shall not reach the end user devices
- The Packager may concatenate ingest segments if they have the same bit position in `WMPaceInfo`
- The Origin shall be able to reply to upstream A/B requests from the CDN

Video segments delivered as byte-range

- The encoder shall byte-align the A & B Variants
  - A single `sidx` box is delivered for both A & B Variants to end-user devices
  - No preferred option to achieve this byte alignment
- Strong preference for the side car file for `WMPaceInfo` delivery in VOD
  - Carries “dummy data” for the initialization segment

Dynamic ad insertion *[open topic]*

- Watermark or not the original ad?

Content Consumption

Several steps

1. Acquiring the WM token, the DASH manifest or the HLS playlists
2. Acquiring the initialization segment
3. Acquiring media segments

Only modification for the device

- Acquire a token and include the token in all requests to the Edge
- Error if there is no token

The Origin-Edge link has new interfaces for `WMPaceInfo` delivery

The Edge has new processing rules

Content Consumption

Acquiring the WM token

- Direct request: new API to integrate in the client application
- Indirect as a header response field from another server

Acquiring the DASH manifest or HLS playlists

- Unchanged; the device may include the WM token, but it is not mandatory

Acquiring the initialization and media segments

- Discrete segments
- Byte-range segments

Content Consumption

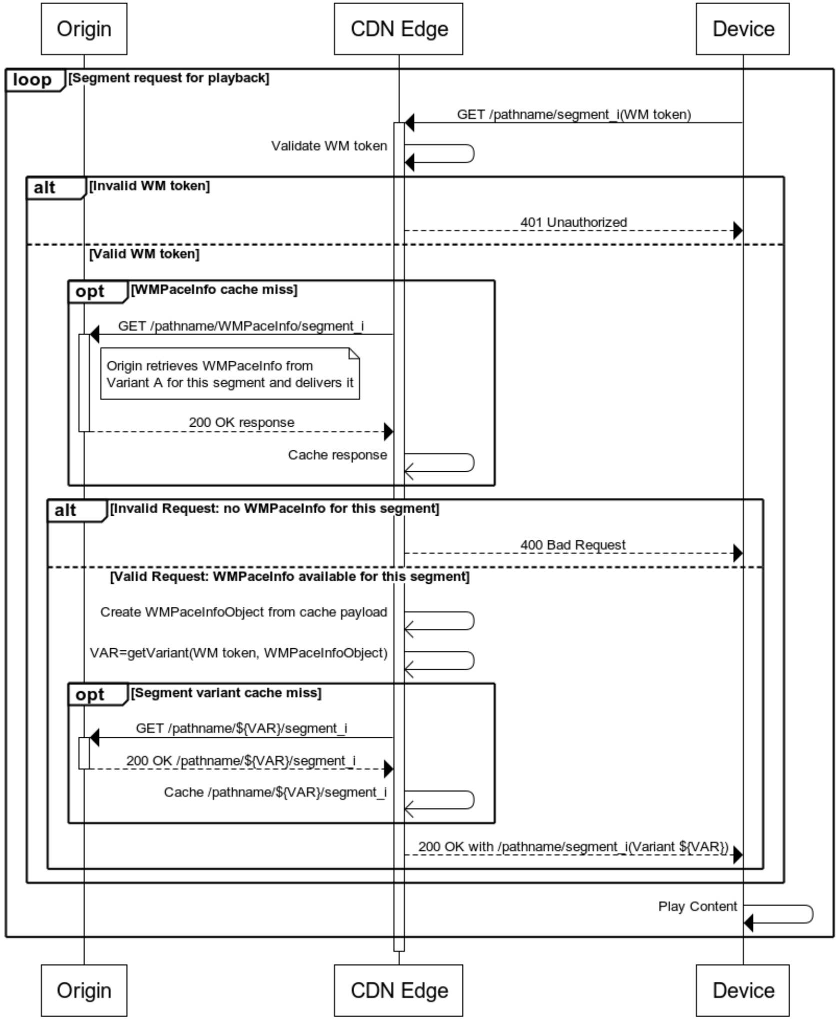
Discrete segments

For `/pathname/filename`, there are three entry points on the Origin server (A/B case)

- WMPaceInfo: `/pathname/`**`WMPaceInfo`**`/filename`
- Variant A: `/pathname/`**`A`**`/filename`
- Variant B: `/pathname/`**`B`**`/filename`

`WMPaceInfo` cached per segment

The edge implements the logic for selecting A or B using the pattern in the token and `WMPaceInfo`



Origin

CDN Edge

Device

loop [Segment request for playback]

GET /pathname/segment_i(WM token)

Validate WM token

alt [Invalid WM token]

401 Unauthorized

[Valid WM token]

opt [WMPaceInfo cache miss]

GET /pathname/WMPaceInfo/segment_i

Origin retrieves WMPaceInfo from Variant A for this segment and delivers it

200 OK response

Cache response

alt [Invalid Request: no WMPaceInfo for this segment]

400 Bad Request

[Valid Request: WMPaceInfo available for this segment]

Create WMPaceInfoObject from cache payload

VAR=getVariant(WM token, WMPaceInfoObject)

opt [Segment variant cache miss]

GET /pathname/${VAR}/segment_i

200 OK /pathname/${VAR}/segment_i

Cache /pathname/${VAR}/segment_i

200 OK with /pathname/segment_i(Variant ${VAR})

Play Content

Origin

CDN Edge

Device

Content Consumption

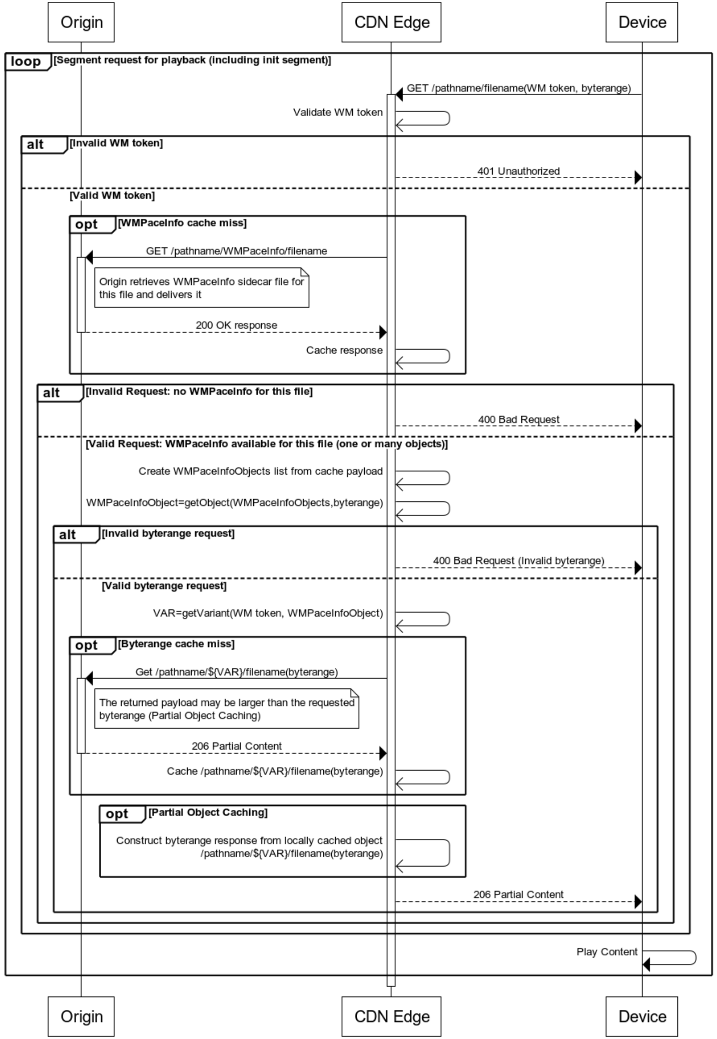
Byte-range segments

Same logic for entry points in the origin

`WMPaceInfo` (side-car file) per track cached at Edge

Byte-range requests are limited to those declared in the side car file

- Prevent byte-range requests overlapping several segments (with different WMPaceInfo metadata)
- Constraint relaxed for byte-range comprised within a byte range of a side car file
  - Concurrent requests per segment (Roku)
  - Low latency HLS
    - Byte-range request in discrete segments


Origin
CDN Edge
Device
loop [Segment request for playback (including init segment)]
GET /pathname/filename(WM token, byterange)
Validate WM token
alt [Invalid WM token]
401 Unauthorized
[Valid WM token]
opt [WMPaceInfo cache miss]
GET /pathname/WMPaceInfo/filename
Origin retrieves WMPaceInfo sidecar file for this file and delivers it
200 OK response
Cache response
alt [Invalid Request: no WMPaceInfo for this file]
400 Bad Request
[Valid Request: WMPaceInfo available for this file (one or many objects)]
Create WMPaceInfoObjects list from cache payload
WMPaceInfoObject=getObject(WMPaceInfoObjects,byterange)
alt [Invalid byterange request]
400 Bad Request (Invalid byterange)
[Valid byterange request]
VAR=getVariant(WM token, WMPaceInfoObject)
opt [Byterange cache miss]
Get /pathname/${VAR}/filename(byterange)
The returned payload may be larger than the requested byterange (Partial Object Caching)
206 Partial Content
Cache /pathname/${VAR}/filename(byterange)
opt [Partial Object Caching]
Construct byterange response from locally cached object /pathname/${VAR}/filename(byterange)
206 Partial Content
Play Content
Origin
CDN Edge
Device

Conclusion

Community Review until end of May 2022

- Document available at https://dash-industry-forum.github.io/docs/DASH-IF-IOP_OTT-Watermarking.pdf
- Anyone interested can propose corrections, changes
  - GitHub used for tracking https://github.com/Dash-Industry-Forum/Watermarking/issues

Next possible steps

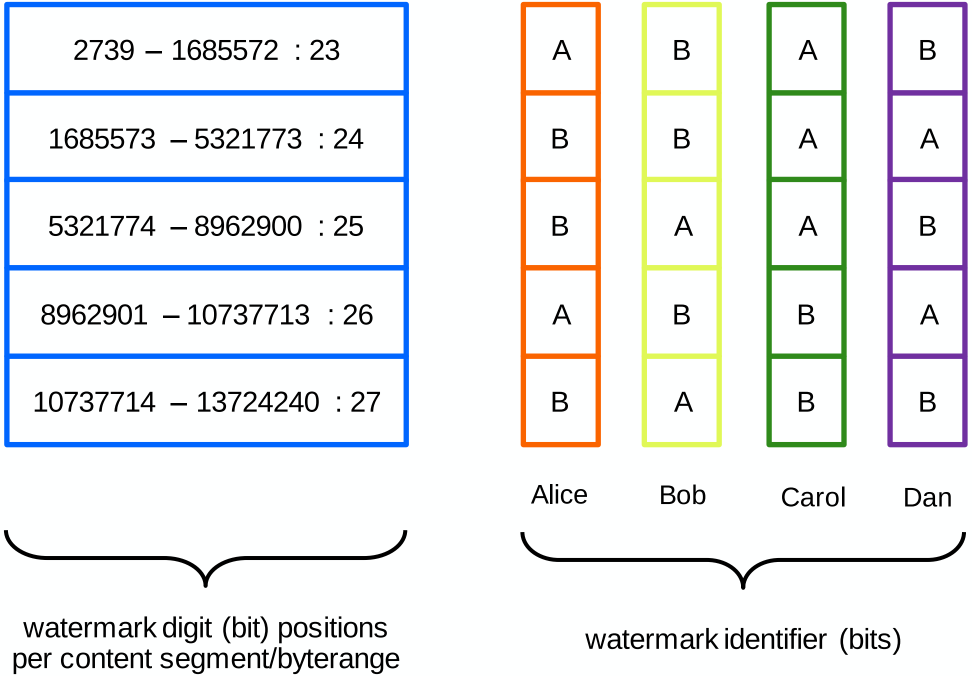
- Need feedback from all stakeholders (encoder, packager, CDN)
  - New approach for server-side watermarking
- Liaise with UHDF to request an update of their encoder specification
- Merge the watermarking and authentication tokens

Watermark Identifier

A unique sequence of A/B video segments can be created by matching:

- the digits of the unique identifier passed via the watermarking token, and
- watermark digits preprocessed in the content

Server-side A/B watermarking is dependent on position matching between user’s ID and preprocessed WM digit in a segment



	Alice	Bob	Carol	Dan
2739 – 1685572 : 23	A	B	A	B
1685573 – 5321773 : 24	B	B	A	A
5321774 – 8962900 : 25	B	A	A	B
8962901 – 10737713 : 26	A	B	B	A
10737714 – 13724240 : 27	B	A	B	B

watermark digit (bit) positions per content segment/byterange

watermark identifier (bits)

Content Consumption

Acquiring the WM token

- Direct request: new API to integrate in the client application
- Indirect as a header response field from another server

Acquiring the DASH manifest or HLS playlists

- Unchanged; the device may include the WM token, but it is not mandatory

Acquiring the initialization segments

- Discrete segment
  - No change
- Byte-range segment
  - Needs to provide a token
  - Rationale: the `init` segment is part of the track, the edge cannot infer whether the required byte-range is an `init` segment or a media segment
- Acquiring media segments