Common Catalog Format

MoQ IETF #118, Prague, Nov 2023

Suhas Nandakumar, Will Law, Mo Zanaty

https://github.com/moq-wg/catalog-format

Agenda for this session

Following presentation at the interim, repo moved from private repo to the moq-wg repository https://github.com/moq-wg/catalog-format .

Today I would like to discuss:

- PRs
  - Adding JSON Patch PR #32 (PR #6)
  - Allowing relative track names and inherited namespace (PR #34)
- Proposals
  - Content protection parameters (issue #28)
  - Bitrate definition (issue #25)
  - How does the client learn about group numbers and their relationship to media time and wallclock time? (issue #22)

Issue #6: JSON patch/merge as an alternate to the proprietary delta updates

Current Catalog provides a delta-update definition, which is a mechanism to apply a small update to the JSON object.

Mike English points out in issue #21 that there are existing standards for incremental updates to JSON files: RFC 6902 (JSON Patch) and RFC 7396 (JSON Merge Patch).

- RFC 6902 - JSON Patch
  - an array of atomically executed, mutating operations on a JSON document.
  - Removes array items by index, so order of items in arrays is important.
  - Many libraries are available.

- RFC 7396 JSON Merge Patch
  - a diff file, containing the nodes of the document which should be different after execution.
  - not possible to change a key’s value to null,
  - array elements cannot be manipulated by merge patches. You have to include entire array in patch even it only changing one of the elements

What would a JSON patch update look like?

Adding a slide track to an established video conference

Note: no sequence or parent sequence number. We rely upon object header sequence.

```
{
 "sequence": 1,
 "parentSequence":0,
 "tracks": [
  {
   "name": "slides",
   "selectionParams":{
       "codec":"av01.0.08M.10.0.110.09",
       "width":1920,
       "height":1080,
       "framerate":15,
       "Bitrate":750000
       },
       "renderGroup":1
   }
  ]
}
```

CURRENT

```
[
  { "op": "add", "path": "/tracks/-", "value": {
   "name": "slides",
   "selectionParams": {
         "codec":"av01.0.08M.10.0.110.09",
         "width":1920,
         "height":1080,
         "framerate":15,
         "Bitrate":750000
         },
         "renderGroup":1
       }
  }
]
```

PROPOSED

What would a JSON patch update look like?

Removing 3 tracks from a conference

```
{
  "sequence": 3,
  "parentSequence":2,
  "operation": "delete",
  "tracks": [
          {"name": "audio"},
          {"name": "video"},
          {"name": "slides" ]
}
```

CURRENT

Note: removal occurs via index number

```
[
  { "op": "remove", "path": "/tracks/2"},
  { "op": "remove", "path": "/tracks/1"},
  { "op": "remove", "path": "/tracks/0"},
]
```

or

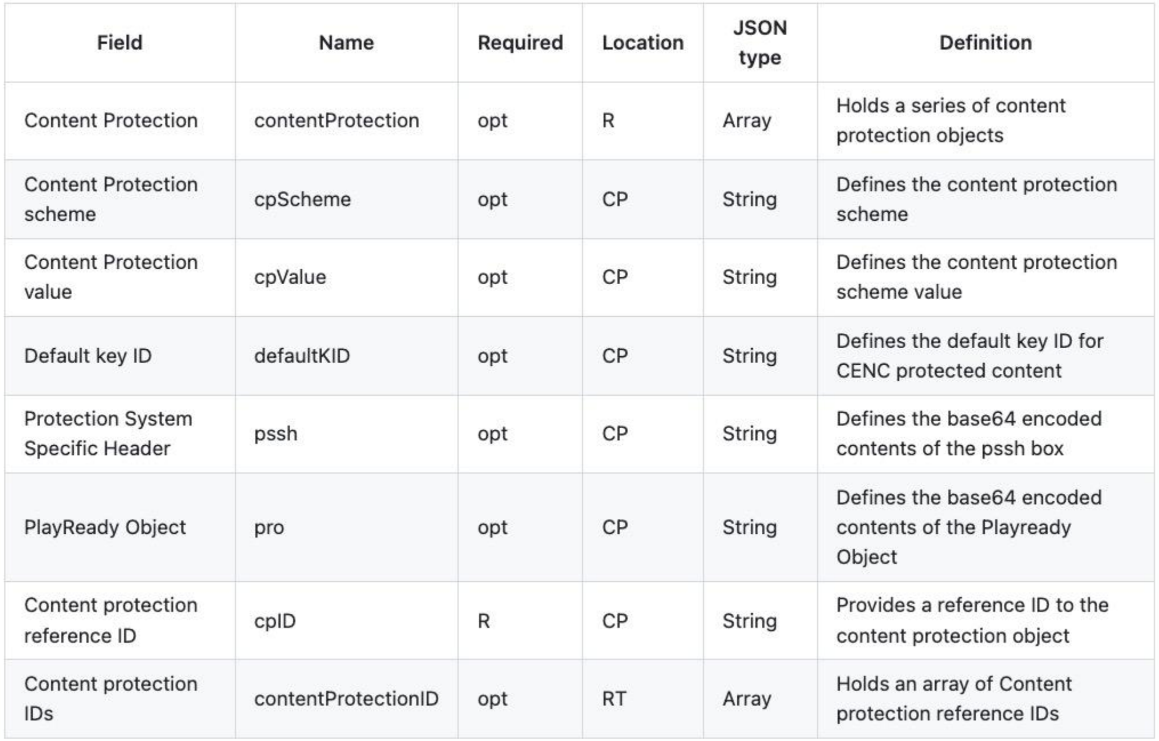
```
[
  { "op": "remove", "path": "/tracks/0"},
  { "op": "remove", "path": "/tracks/0"},
  { "op": "remove", "path": "/tracks/0"},
]
```

PROPOSED

Issue #28: Encryption/DRM info for CMAF

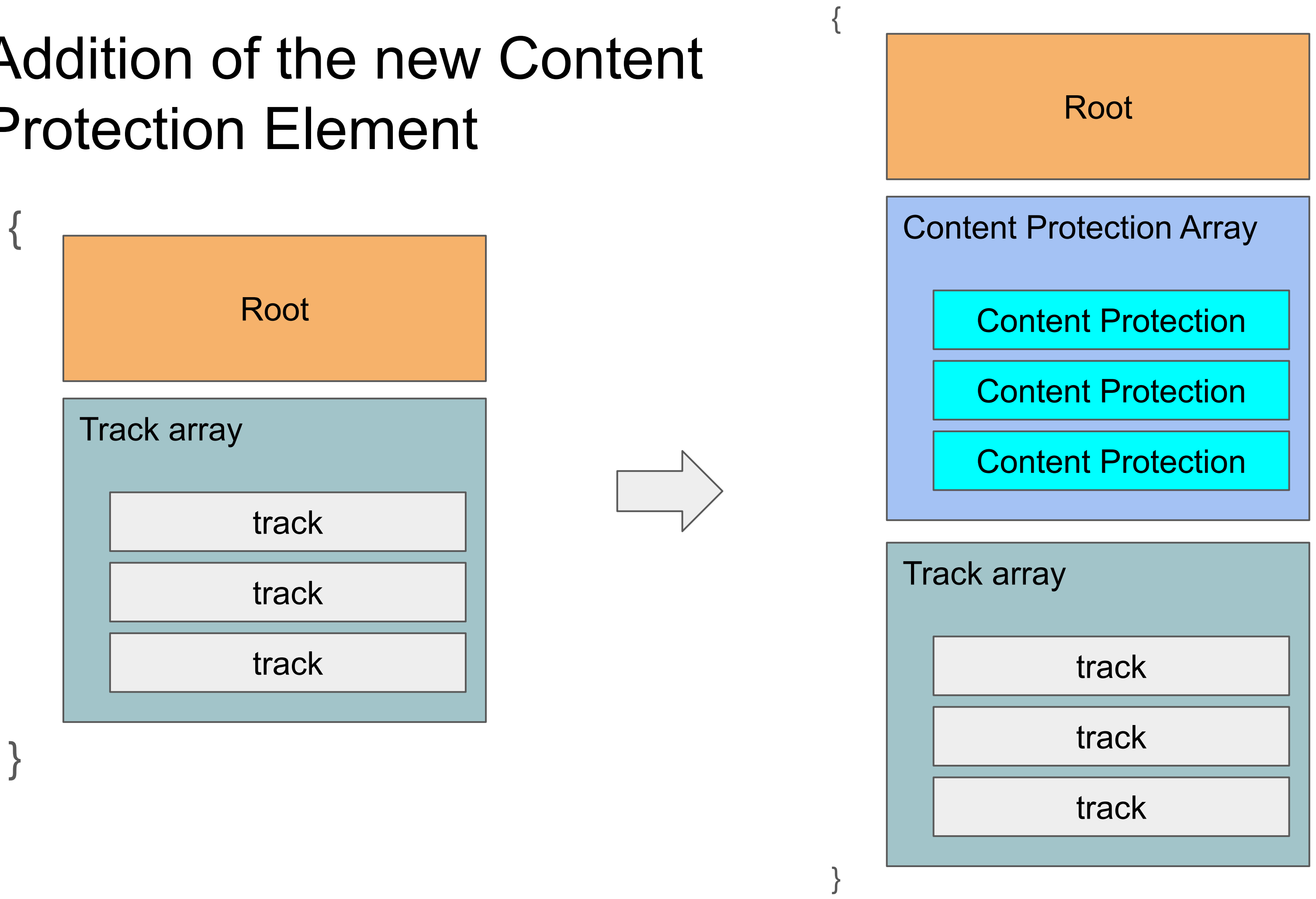
OP called for plans to provide encryption/DRM info for CMAF selectionParams.

To address this, I propose the following new catalog elements:



Field	Name	Required	Location	JSON type	Definition
Content Protection	contentProtection	opt	R	Array	Holds a series of content protection objects
Content Protection scheme	cpScheme	opt	CP	String	Defines the content protection scheme
Content Protection value	cpValue	opt	CP	String	Defines the content protection scheme value
Default key ID	defaultKID	opt	CP	String	Defines the default key ID for CENC protected content
Protection System Specific Header	pssh	opt	CP	String	Defines the base64 encoded contents of the pssh box
PlayReady Object	pro	opt	CP	String	Defines the base64 encoded contents of the Playready Object
Content protection reference ID	cpID	R	CP	String	Provides a reference ID to the content protection object
Content protection IDs	contentProtectionID	opt	RT	Array	Holds an array of Content protection reference IDs

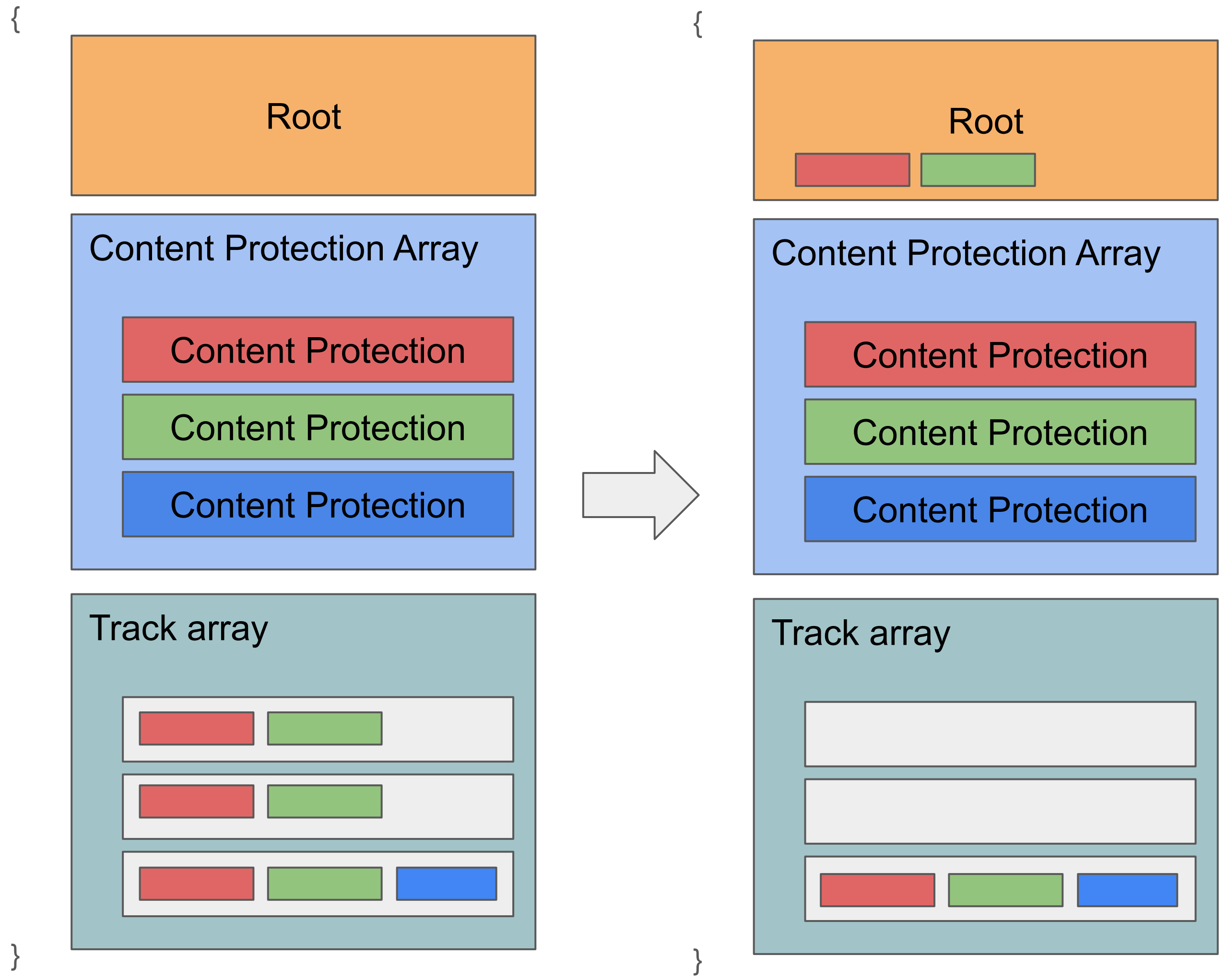
Addition of the new Content Protection Element



Inheritance of Content Protection

Content Protection can be declared at the root level and then inherited by all tracks.

Content Protection declarations at the track level override any inherited value.



Example CMAF catalog with CENC DRM info

```
{
  "version": 1, "sequence": 0,
  "streamingFormat": 1, "streamingFormatVersion": "0.2",
  "namespace": "sports.example.com/games/08-08-23/12345",
  "packaging": "cmaf", "renderGroup":1,
  "contentProtection": [
     { "cpID":"1",
       "value":"cenc",
       "schemeID":"urn:mpeg:dash:mp4protection:2011",
       "defaultKID":"80399bf5-8a21-4014-8053-e27e748e98c0"
     },
     { "cpID":"2",
       "value":"MSPR 2.0",
       "schemeID":"urn:uuid:9a04f079-9840-4286-ab92-e65be0885f95",
       "pssh":"AAAB..",
       "pro":"xAEA..."
     },
     { "cpID":"3",
       "value":"Widevine",
       "schemeID":"urn:uuid:edef8ba9-79d6-4ace-a3c8-27dcd51d21ed",
       "pssh":"AAAA.."
     }
  ],
```

```
"contentProtectionID": ["1","2","3"],
"tracks": [
  {
    "name": "video_4k", "selectionParams":{"codec":"avc1.640033","mim…},
    "initTrack":"init_video_4k", "altGroup": 1,
    "contentProtectionID": ["1","2"]
  },
  {
    "altGroup": "video_1080", "selectionParams":{"codec":"avc1.640…},
    "initTrack":"init_video_1080", "altGroup": 1
  },
  {
    "altGroup": "video_720", "selectionParams":{"codec":"avc1.640…}},
    "initTrack":"init_video_720", "altGroup": 1
  },
  {
    "altGroup": "audio_aac", "selectionParams":{"codec":"mp4a.40.5"...},
    "initTrack":"init_audio_aac", "altGroup": 2
  },
  {
    "name": "audio_ec3", "selectionParms":{"codec":"ec-3",....},
    "initTrack":"init_audio_ec3", "altGroup": 2
  }
```

Questions on DRM

1. Are these DRM keys sufficient do describe current protected media?
2. Should these fields be added to the Common Catalog definition, or defined separately as an Extension by the WARP Spec?
3. What Content Protection elements will LOC require?
4. Is there a cleaner way of specifying this info?

PR #34: Allowing relative track names and inherited namespace

This PR makes the namespace field optional in a catalog.

If namespace is absent, then the namespace is inherited from the catalog namespace.

If the catalog full track name is **`premiergaming/gamer34/catalog`** and it looks like this:

```
{
    tracks: [{“name”:”video”},{“name”:”audio”}]
}
```

Client would subscribe to

`prepremiergaming/gamer34/video`

`premiergaming/gamer34/audio`

Issue #25: Bitrate definition

Sect 3.2.25 references “bitrate” without defining how it is calculated or how VBR is accommodated.

HLS defines variant bitrate using two parameters

> BANDWIDTH : It represents the peak segment bit rate of the Variant Stream.
> AVERAGE-BANDWIDTH: the value is a decimal-integer of bits per second. It represents the average segment bit rate of the Variant Stream.

DASH defines bitrate using there interplay between two parameters

> @Bandwidth @MinBufferTime - consider a hypothetical constant bitrate channel of bandwidth with the value of this attribute in bits per second (bps). Then, if the Representation is continuously delivered at this bitrate, starting at any SAP that is indicated either by @startWithSAP or by any Segment Index box, a client can be assured of having enough data for continuous playout providing playout begins after @minBufferTime * @Bandwidth bits have been received.

WebCodecs defines bitrate quite ambiguously

> bitrate - The average bitrate of the encoded video given in units of bits per second.

How should we define bitrate when used with Moq-transport streaming formats?

Should we provide two placeholders: **averageBitrate** and **peakBitrate** and let the streaming format define the timebase?

Issue #26: Registry for catalog fields

Proposal that all CATALOG fields are defined in an IANA registry

Arguments for:

- If we end up needing new fields (which we certainly will), we don't need a revised RFC. The fields table can be amended/expanded in a number of ways.

Arguments against:

- IANA is useful for ensuring global uniqueness. If a single specification defines the catalog format, then that spec is a much more efficient and natural place to define the fields used by that format.
- How would you version the registry to know if your parser can understand all the fields?
- Fields have relationships between them (i.e CENC fields only apply to CMAF packaged content). How would express these relationships in an IANA table?

WARP: issue #22 - How does the client learn about group numbers and their relationship to media time and wallclock time?

A client playing a live stream only at the live edge does not need to know the history of group numbers and their relationship to media time and wallclock time.

However a client implementing any of the following actions does need to understand this relationship

- Seeking back (DVR)
- Sub-clipping
- Building a scrub bar UI
- Syncing with externally transmitted timeline data, such as sports scores.
- Avoiding the ambiguity of relative starting subscriptions

Number of ways to achieve this:

1. Have the catalog update every group to show group number history
2. Use a template, communicated in the catalog, so that group/time relationship is predictable
3. Add a timeline track that clients subscribe to when and if they need it.

Timeline track proposal

A track, declared in the catalog, which supplies an array of offsets of group number to media time and wallclock time. Group boundaries could supply the complete history and delta updates would supply relative changes. Payload would be binary. This schema allows for variable group durations.

```
// Group number, wall-clock time, media PTS

[

{0,1698351160362,0}

{1,1698353162,2002}

{2,1698355164,4004}

...

{3745,1705848650,7497490}

]
```

A player could start playback by subscribing to the last object of this "live edge" timeline and only subscribe to the complete timeline if it needed to build a scrub bar or seek. Note that this live edge discovery is an alternate to the TRACK _STATUS message proposed in #332