

MSF & CMSF Updates & Issues

<https://github.com/moq-wg/msf>

Will Law

IETF moq interim, Boulder, February 2026

Brain-storming side meeting - Feb 5, 2026

We held a brain-storming meeting to gather input and comments on MSF & CMSF.

32 attendees, 17 issues filed.

Good discussions with representatives of companies across the OTT landscape.

Agenda [available](#).

Thanks to the room sponsor



Rename complete (PR#71)

At IETF #124 we decided to rename the the two adopted streaming formats:

WARP -> MOQT Streaming Format (**MSF**)

CARP -> CMAF-Compliant MOQT Streaming Format (**CMSF**)

I-D drafts are updated:

<https://datatracker.ietf.org/doc/draft-ietf-moq-msf/>

<https://datatracker.ietf.org/doc/draft-ietf-moq-cmsf/>

URL syntax

We need a mechanism to point a player at a catalog resource (Issue #60)

Two high-level questions:

1. Is MOQT Spec going to define a generic method of pointing a client at a connection + track? If it will do this, then MSF can inherit that
 - a. Currently Sect 3.1 defines a syntax for connecting to Webtransport or QUIC server. But there is no definition for how to indicate the namespace + name of a track during this connection.
2. Should the URL
 - a. Allow the player to choose whether it connects via QUIC or Webtransport, such that we need a generic URL syntax, \
 - b. Deterministically indicate which connection method is to be used? (The preference in the brain-storming meeting was strongly for this option).

To URI formats have been proposed (PR #72 and #87)

#72



Note:

- 1. With this scheme, query-args are stripped and never sent to the relay. They are only used for the player.*
- 2. We could use the MOQT convention of namespace+name logging to collapse the track names down to a single argument.*

#72 Reserved query keys

Certain query keys are reserved

| Name | Description |
|-----------------|---|
| ns | The Namespace of the track |
| t | The Name of a non-catalog track |
| wallclock-range | A subclip defined by a wallclock time range |
| mediatime-range | A subclip defined by a media time range |
| location-range | A subclip defined by a MOQT Location range |
| c4m | A base64 encoded C4M token |

- **ns** - the Namespace of the track. The '/' character in the String defines tuple field boundaries and is not included in the tuple fields. The '/' character MUST NOT be used other than to define tuple boundaries. A closing '/' for the last tuple field MUST NOT be included. This key MUST be included if 't' is present.
- **t** - the track name of a non-catalog track. This field MUST NOT be included if the URL is referencing a catalog track.
- **wallclock-range** - a range defined by start and end wallclock times, each expressed as milliseconds since Unix Epoch and separated by a "-" dash.
- **mediatime-range** - a range defined by start and end media times, each expressed as milliseconds and separated by a "-" dash.
- **location-range** - a range defined by start and end media MOQT Location tuples and expressed as Start Group ID, Start Object ID, End Group ID, End Object ID, each separated by a "-" dash

#72 Example URLs

Example range-defining query arguments:

1. wallclock-range=1761759637565-1761759836189
2. mediatime-range=0-13421
3. location-range=34-0-2145-16

Example URLs

1. URL pointing at a catalog using a Webtransport connection

`https://example.com/relay-app/relayID?ns=customerID/broadcastID`

2. URL pointing at a non-catalog track, using a QUIC connection

`moqt://example.com/relay-app/relayID?ns=customerID/broadcastID&t=video`

3. URL pointing at a subclip of a catalog

`https://example.com/relay-app/relayID?ns=customerID/broadcastID&location-range=34-0-64-16`

4. URL pointing at a catalog and supplying a token

**`https://example.com/relay-app/relayID?ns=customerID/broadcastID
&c4m=gqhkYWxnIGVzaGFyqGR0eXBNhdZ9hdWQAY3VybGZlbWlzcwZleWV2aW5u
ZWlhdlGVwQWNyZW5lYnJmcmVqMTIzNDU2NzgwMHZpc3VlZF9hdD0xNzMwNDM`**

URL format #87 - fragment -based



Note:

1. *This proposal uses the namespace+name syntax defined in MOQT for logging.*
2. *Any query args are sent to both relay and player.*
3. *Per Section 3.5 of RFC 3986, a fragment identifier allows indirect identification of a secondary resource by reference to a primary resource. Unlike the rest of the URI, the fragment is not sent to the server in an HTTP request. It is processed entirely by the "user agent".*

#87 Example URLs

Example range-defining query arguments:

1. wallclock-range=1761759637565-1761759836189
2. mediatime-range=0-13421
3. location-range=34-0-2145-16

Example URLs

1. URL pointing at a catalog using a Webtransport connection

`https://example.com/relay-app/relayID#customerID-broadcastID`

2. URL pointing at a non-catalog track, using a QUIC connection

`moqt://example.com/relay-app/relayID#customerID-broadcastID--video`

3. URL pointing at a subclip of a catalog

`https://example.com/relay-app/relayID?location-range=34-0-64-16#customerID-broadcastID`

4. URL pointing at a catalog and supplying a token

`https://example.com/relay-app/relayID?c4m=gqhkYWxnIGVzaGFyqGR0eXBNhdZ9hdWQAY3VybGZlbWlzcwZleWV2aW5uZWlhZGVwQWNyZW5lYnJmcmVqMTIzNDU2NzgwMHZpc3VlZF9hwNDM#ns=customerID=broadcastID`

Formal ABNF for #87

The following ABNF defines the MSF URL structure, importing core rules from [RFC5234] and [RFC3986].

```
msf-url      = msf-scheme "://" authority path-abempty [ "?" query ] [ "#" msf-fragment ]
msf-scheme   = "moqt" / "https"
msf-fragment = msf-namespace "--" msf-track-name
msf-namespace = msf-encoded-string *( "-" msf-encoded-string )
msf-track-name = msf-encoded-string
msf-encoded-string = *( unreserved / "." 2HEXDIG )
unreserved   = ALPHA / DIGIT / "-" / "_"
msf-query-arg = wallclock-arg / mediatime-arg / location-arg / token-arg / other-arg
wallclock-arg = "wallclock-range=" 1*DIGIT [ "-" 1*DIGIT ]
mediatime-arg = "mediatime-range=" 1*DIGIT [ "-" 1*DIGIT ]
location-arg  = "location-range=" start-loc [ "-" end-loc ]
start-loc     = group-id "-" object-id
end-loc       = group-id "-" object-id
group-id      = 1*DIGIT
object-id     = 1*DIGIT
token-arg     = "c4m=" 1*( ALPHA / DIGIT / "-" / "_" ) [ "=" / "==" ]
```

Both formats side-by-side - let's choose one.

#72 - query-arg based

<https://example.com/relay-app/relayID?ns=app/broadcastID&t=catalog&token=1234>

#87 - fragment based

<https://example.com/relay-app/relayID?token=1234#app-broadcastID--catalog>

```

{
  "version": 1,
  "generatedAt": 1746104606044,
  "tracks": [
    {
      "name": "hd",
      "renderGroup": 1,
      "packaging": "loc",
      "isLive": true,
      "targetLatency": 1500,
      "role": "video",
      "codec": "av01",
      "width": 1920,
      "height": 1080,
      "bitrate": 5000000,
      "framerate": 30,
      "altGroup": 1
    },
    {
      "name": "md",
      "renderGroup": 1,
      "packaging": "loc",
      "isLive": true,
      "targetLatency": 1500,
      "role": "video",
      "codec": "av01",
      "width": 720,
      "height": 640,
      "bitrate": 3000000,
      "framerate": 30,
      "altGroup": 1
    }
  ],
}

```

```

{
  "name": "sd",
  "renderGroup": 1,
  "packaging": "loc",
  "isLive": true,
  "targetLatency": 1500,
  "role": "video",
  "codec": "av01",
  "width": 192,
  "height": 144,
  "bitrate": 500000,
  "framerate": 30,
  "altGroup": 1
},
{
  "name": "audio",
  "renderGroup": 1,
  "packaging": "loc",
  "isLive": true,
  "targetLatency": 1500,
  "role": "audio",
  "codec": "opus",
  "sampleRate": 48000,
  "channelConfig": "2",
  "bitrate": 32000
}
]
}

```

```

{
  "version": 1,
  "generatedAt": 1746104606044,
  "altGroups": [
    {
      "altGroup": 1,
      "renderGroup": 1,
      "packaging": "loc",
      "isLive": true,
      "targetLatency": 1500,
      "role": "video",
      "codec": "av01",
      "framerate": 30
    }
  ],
  "tracks": [
    {
      "name": "hd",
      "altGroup": 1,
      "width": 1920,
      "height": 1080,
      "bitrate": 5000000,
    },
    {
      "name": "md",
      "altGroup": 1,
      "width": 720,
      "height": 640,
      "bitrate": 3000000,
    }
  ],
}

```

```

{
  "name": "sd",
  "altGroup": 1,
  "width": 192,
  "height": 144,
  "bitrate": 500000,
},
{
  "name": "audio",
  "renderGroup": 1,
  "packaging": "loc",
  "isLive": true,
  "targetLatency": 1500,
  "role": "audio",
  "codec": "opus",
  "sampleRate": 48000,
  "channelConfig": "2",
  "bitrate": 32000
}
]
}

```

Issues #106 - don't have a flat list of tracks.

Issue #106: Variable support in catalog

For scalable delivery, we want to cache catalogs and provide the same content to all viewers, while still allowing each to be customized for advertising, A/B watermarking, QoE reporting or logging purposes. The % character will be prohibited for non-variable usage.

<https://example.com/relay-app/relayID?token=1234&id=bob&event=xyz#app-live--catalog>

```
{  
  "name": "cmcdv2-%id%",  
  "namespace": "advertising-decisions/live-sports/%event%",  
  "packaging": "eventtimeline",  
  "eventTimelineType": "com.example.iab.vast",  
  "c4m": "%token%"  
}, ...
```

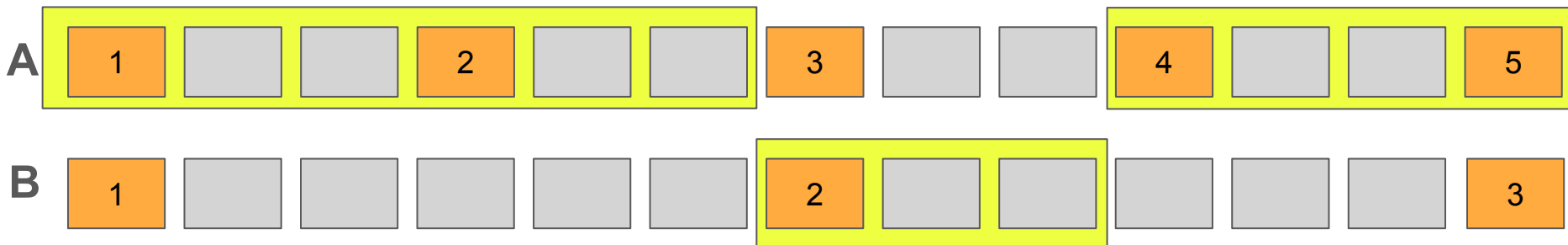
Issue #112: Instructing a player on where to publish QoE or log data #112

One of the benefits of a bidirectional protocol such as MOQT is that a player can publish QOE and logging data directly back to the delivery system that is feeding it media, rather than beaconing it to a 3rd party location.

We can introduce a new **PUBLISH** catalog item, which holds an **array of track definitions**. For each track, we specify the **namespace, name, track type and token** that it should use.

```
"publish": [  
  {  
    "name": "cmcdv2-%playerID%",  
    "namespace": "datacollection.com/qoe",  
    "packaging": "eventtimeline",  
    "eventTimelineType": "org.cta.wave.cmcd.v2",  
    "c4m": "%token1%"  
  },  
  {  
    "name": "logging-%playerID%",  
    "namespace": "central-log-collector.com/ingest",  
    "packaging": "eventtimeline",  
    "eventTimelineType": "org.ietf.moq.moqt.logging",  
    "c4m": "%token2%"  
  }  
]
```

Issue #110: zapping and quick switching



Both tracks are **time aligned at Object boundaries, not only Group boundaries.**

Track A then has a keyframe every N (3) objects, Track B every M (6) objects.

The player can switch from A to B when $(ID_A - 1) * N / M + 1$ is an integer.

Try to switch at Group 2 (ID_A) gives 1.5, so cannot switch

Try to switch at Group 3 gives 2, so can switch and the target group ID in B is 2.

We can enable switching across arbitrary intervals by adding a **Track Spacing Parameter** to the catalog entry for each track. Default is 1 when every group is aligned.

Issue#105: Think about template based media timeline track design

An example media timeline is shown below:

```
[  
  [0, [0, 0], 1759924158381],  
  [2002, [1, 0], 1759924160383],  
  [4004, [2, 0], 1759924162385],  
  [6006, [3, 0], 1759924164387],  
  [8008, [4, 0], 1759924166389],  
  [10010, [5, 0], 1759924168391],  
  [12012, [6, 0], 1759924170393],  
  [14014, [7, 0], 1759924172395],  
  [16016, [8, 0], 1759924174397],  
  [18018, [9, 0], 1759924176399],  
  [20020, [10, 0], 1759924178401],  
  [22022, [11, 0], 1759924180403],  
  [24024, [12, 0], 1759924182405],  
  [26026, [13, 0], 1759924184407],  
  [28028, [14, 0], 1759924186409]  
]
```

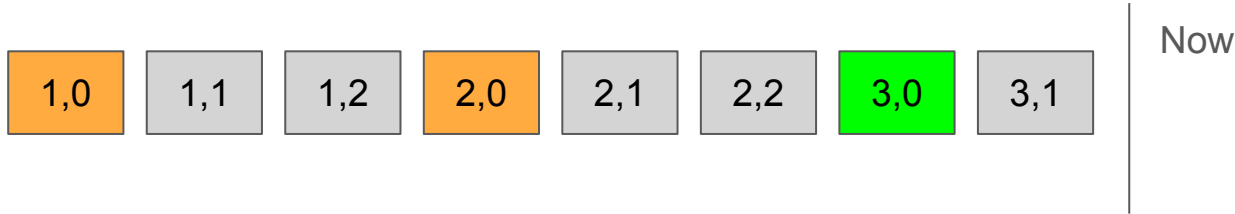
For constant duration GOPs, this can be replaced with a simple template entry in the catalog.

StartTime, delta
Start Location, delta
Start media time, delta

1759924158381,2002
[0,0], [1,0]
0, 2002

This allows any timeline track to be calculated without having to list each item.

Issue #100 - How to get the latest full catalog?



With MSF, the Publisher sends a full catalog (orange) at each group start and then sends delta updates as subsequent objects within that group.

A subscriber wants to have the "latest version" of the full catalog i.e the green object in the timeline above.

The way to do this is to SUBSCRIBE catalog (to get the future updates) + Relative Joining FETCH catalog (0).

CMSF Issue #17: Explicit signalling of DRM/C2PA key-rotation or various init segment updates

We need a mechanism for clients to be notified when the init segment changes due to key rotation or any other reason. The goal is to ensure that clients can easily switch to new tracks and reliably obtain the appropriate init segment for that track, including when a key rotation occurs. Some ideas:

1. Have a separate track that sends the init segment whenever an update occurs. The group raised concerns regarding synchronization between these two tracks.
2. Include the init segment as an extension header.
3. Explicitly signal key rotation in the extension header.
4. Explicitly signal that an object requires a new init segment via a simple extension header indicating that the latest init segment is required to decode the object (optionally including a reference to the location of that init segment in the separate track).

Thank you for your time.