

**This specification is now in community review. You may file issues at <https://github.com/cta-wave/common-media-client-data/issues>**

## **Common Media Client Data CTA-5004 (community review)**

Media player clients can convey information to Content Delivery Networks (CDN) with each object request. This information can be useful in log analysis, QoS monitoring and delivery optimization. Session identification allows thousands of individual server log lines to be interpreted as a single user session, leading to a clearer picture of end-user quality of service. Bitrate, buffer and segment signaling allow CDNs to fine-tune and optimize their midgress traffic by intelligently reacting to the time constraints implicit in each request. Device and content IDs across a multi-CDN delivery surface allow performance problems to be cross-correlated with player software versions or specific devices. In combination, this transferred data should improve the quality-of-service offered by CDNs which in turn will improve the quality-of-experience enjoyed by consumers.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC2119 [1].

This document outlines a simple means by which every media player can communicate data with each media object request and have it received and processed consistently by every CDN. This client data can be sent by one of two means:

- As a custom header.
- As a query argument.

A request can carry a Common-Media-Client-Data header or a Common-Media-Client-Data query arg, but it MUST NOT carry both.

Note: usage of a custom header from a web browser user-agent will trigger a preflight OPTIONS request before each media object request. This will lead to an increased request rate against the server.

### **Header field definition**

The header has a reserved case-insensitive field name:

Common-Media-Client-Data

### Query Argument definition

The query argument is case-insensitive:

Common-Media-Client-Data

If the request already bears a query string, then an ampersand Unicode 0x26 character followed by the encoded Common-Media-Client-Data field should be appended.

### Data payload definition

The data payload is common to both means of transmission. It is a string, containing a series of key/value pairs and constructed according to the following rules:

1. All information in the payload MUST be represented as <key>=<value> pairs.
2. The key and value MUST be separated by an equals sign Unicode 0x3D.
3. Successive key/value pairs MUST be delimited by a comma Unicode 0x2C.
4. The key names described in this specification are reserved. Custom key names may be used, but they MUST carry a hyphenated prefix to ensure that there will not be a namespace collision with future revisions to this specification. Clients SHOULD use a reverse-DNS syntax when defining their own prefix.
5. All key names are case-sensitive.
6. Any value of type String MUST be enclosed by opening and closing double quotes Unicode 0x22. Double quotes and backslashes MUST be escaped using a backslash “\” Unicode 0x5C character. Any value of type Token does not require quoting.
7. All keys are OPTIONAL.
8. Key-value pairs may be sequenced in any order.
9. If the data payload is transmitted as a query argument, then the entire payload string MUST be URLEncoded per [1]. Data payloads transmitted via headers MUST NOT be URLEncoded.

The reserved keys and their definitions are defined in Table 1 below:

Description	Key Name	Unit	Value definition
Encoded bitrate	br	Integer kbps	The encoded bitrate of the audio or video object being requested. This may not be known precisely by the

			player, however it MAY be estimated based upon playlist/manifest declarations.
Buffer state	bs	Integer	<p>This key is used to indicate particular buffer states. If the buffer state is nominal during normal play, this key SHOULD not be sent.</p> <p>1: startup/initializing (ends when initial buffer target is reached)  2: seeking (if client can not distinguish between startup and seeking then they SHOULD use 1:startup for seeking as well)  3: buffer risk (if trending downward and buffer length is less than 1 object/segment target duration)  4: buffer empty (if buffer is completely drained)</p> <p>If the object type 'ot' key is sent along with this key, then the 'bs' value refers to the buffer associated with the particular object type. If no object type is communicated, then the buffer state applies to the current session.</p>
Content ID	cid	String	A unique string identifying the current content. Maximum length is 64 characters. This value is consistent across multiple different sessions and devices and is defined and updated at the discretion of the service provider.
Object duration	d	Integer milliseconds	The playback duration in milliseconds of the object being requested. If a partial segment is being requested, then this value MUST indicate the playback duration of that part and not that of its parent segment. This value can be an approximation of the estimated duration if the explicit value is not known.
Device ID	did	String	A unique string identifying the current device. Maximum length is 64 characters. This value is consistent across multiple different session and content IDs. This identifier may represent a combination of player software version and device hardware and it is defined and updated at the discretion of the service provider. This string must not identify an instance of a device.
Deadline	dl	Integer milliseconds	Deadline from the request time until the first sample of this Segment/Object needs to be available in order to not create a buffer underrun or any other playback problems.

			For a playback rate of 1, this may be equivalent to the player's remaining buffer length.
Measured throughput	mtp	Integer kilobits per second (kbps)	The throughput existing between client and server, as measured by the client. This value, however derived, SHOULD be the value that the client is using to make its next Adaptive Bitrate switching decision. If the client is connected to multiple servers concurrently, it must take care to report only the throughput measured against the receiving server.
Next object request	nor	String	Relative path of the next object to be requested. This can be used to trigger pre-fetching by the CDN. This MUST be a path relative to the current request. This string MUST be URL encoded [2].
Next range request	nrr	String of the form “<range-start>-<range-end>”	If the next request will be a partial object request, then this string denotes the byte range to be requested. If the ‘nor’ field is not set, then the object is assumed to match the object currently being requested. Formatting is similar to the HTTP Range header, except that the unit MUST be ‘byte’, the ‘Range:’ prefix is NOT required and specifying multiple ranges is NOT allowed. Valid combinations are: “<range-start>-” “<range-start>-<range-end>” “-<suffix-length>”
Object type	ot	Token - one of [m, a,v,av,i,c,tt,k,o]	The media type of the current object being requested: m = text file, such as a manifest or playlist a = audio only v = video only av = muxed audio and video i = init segment c = caption or subtitle tt = ISOBMFF timed text track k = cryptographic key, license or certificate. o = other  If the object type being requested is unknown, then this key MUST NOT be used.

Playback rate	pr	Decimal	1 if real-time, 2 if double speed, 0 if not playing. SHOULD only be sent if not equal to 1.
Requested maximum throughput	rtp	Integer kilobits per second (kbps)	The requested maximum throughput that the client considers sufficient for delivery of the asset. For example, a client would indicate that the current segment, encoded at 2Mbps, is to be delivered at no more than 10Mbps, by using rtp=10000.  Note: This can benefit clients by preventing buffer saturation through over-delivery and can also deliver a community benefit through fair-share delivery. The concept is that each client receives the throughput necessary for great performance, but no more. The CDN may not support the rtp feature.
Streaming format	sf	Token - one of [d,h,s,o]	The streaming format which defines the current request d = MPEG DASH h = HTTP Live Streaming (HLS) s = Smooth Streaming o = other  If the streaming format being requested is unknown, then this key MUST NOT be used.
Session ID	sid	String	A GUID identifying the current playback session. A playback session typically ties together segments belonging to a single media asset. Maximum length is 64 characters. It is RECOMMENDED to conform to the UUID specification [3].
Stream type	st	Token - one of [v,l]	v = all segments are available e.g. VOD l = segments become available over time e.g. LIVE
CMCD version	v	Integer	The version of this specification used for interpreting the defined key names and values. If this key is omitted, the client and server MUST interpret the values as being defined by version 1.

Table 1: Reserved Key and Value definitions

It is RECOMMENDED that a player supply sid on all media object requests in a session, including playlists/manifests, init files, captioning files and DRM key requests. Other keys should

be applied where they have contextual meaning. For example, a 'br' (bitrate) key on a manifest request is inappropriate and could be omitted.

### **Server processing requirements**

1. A server, upon receiving Common-Media-Client-Data, MUST interpret the keys according to their definition in this document.
2. Unknown keys, which the server does not understand, MUST be ignored.
3. Since there is no guarantee that keys are included, the server MUST be robust against the absence of individual keys on any given request.
4. The server MUST be able to correctly process the key-value pairs irrespective of the order in which they are defined.
5. If the sid key is present, the server SHOULD propagate that value to the server access logs. The server access logs SHOULD conform to RFC6302 [4].
6. The server MAY assume that the payload is held as either a header or a query arg, but never both at the same time.
7. The server, upon receiving the requested throughput (rtp) attribute, is not required to throttle the response at the requested value. It is merely a request from the client and the server may have other business requirements that dictate throttling at a different value, or not throttling the response at all.
8. The server, upon receiving the nor "next object request" or nrr "next range request" attributes, MAY optionally decide not to implement any pre-fetch action against that data. The client SHOULD NOT depend upon any pre-fetch action being taken - it is merely a request for optimization.
9. Servers SHOULD provide the necessary CORS responses to allow browser-based clients to send custom headers, specifically:
  - a. Access-Control-Allow-Headers response header with a value that contains "Common-Media-Client-Data"
  - b. Access-Control-Allow-Methods with a value that includes OPTIONS.
10. Servers SHOULD be aware that malicious clients may send false key data with the objective of either attacking the server or gaining an unfair delivery advantage. The server SHOULD validate incoming key data before any performance impacting behaviors are executed.

Note: any caching proxy should be aware that the Common-Media-Client-Data payload will be constantly changing and therefore has the potential to pollute cache keys. Implementers may wish to exclude common-media-client-data query arguments from any cache key.

### **Security considerations**

It cannot be assumed that all clients are benevolent, honest and accurate. However, the specification does not expose any security issues *that are not already exposed* to an edge server which answers all requests. A number of steps have been taken to mitigate security concerns:

1. The 'nor' key value must be a relative path to the current request. This makes it harder to inject false requests to arbitrary objects.
2. Only one object can be requested by the 'nor' key value - this lowers amplification opportunity.
3. All requests to the server are optionally executed by the server, meaning that a server can ignore them for security concerns (such as a rate-based threshold being exceeded) and still be compliant with the specification.

### Valid header examples

- Common-Media-Client-Data:  
sid="6e2fb550-c457-11e9-bb97-0800200c9a66",d=4004,rtp=15000,mtp:25430,br=3200,bs=1,ot=v
- Common-Media-Client-Data: sid="6e2fb550-c457-11e9-bb97-0800200c9a66"
- Common-Media-Client-Data:  
sid="6e2fb550-c457-11e9-bb97-0800200c9a66",rtp=15000
- Common-Media-Client-Data:  
sid="6e2fb550-c457-11e9-bb97-0800200c9a66",rtp=15000,d=4004,ot=v,sf=d
- Common-Media-Client-Data: d=4004,br=3200
- Common-Media-Client-Data: d=4004,com.example-myKey=myValue
- Common-Media-Client-Data:  
sid="6e2fb550-c457-11e9-bb97-0800200c9a66",d=2002
- Common-Media-Client-Data:  
sid="6e2fb550-c457-11e9-bb97-0800200c9a66",nor="..%2F300kbps%2Fsegment35.m4v"
- Common-Media-Client-Data:  
br=3200,bs=3,cid="ABCD-1234",d=4004,did="Android6.0-player-build-12.3",dl=18000,mtp=48175,nor="..%2F300kbps%2Fsegment35.m4v",nrr=12323-48763,ot=v,pr=1.08,rtp=12000,sf=d,sid="6e2fb550-c457-11e9-bb97-0800200c9a66",st=v,v=1

## Corresponding valid Query Arg examples

- ?Common-Media-Client-Data=sid%3D%E2%80%9D6e2fb550-c457-11e9-bb97-0800200c9a66%E2%80%9D%2Cd%3D4004%2Crtp%3D15000%2Cmtp%3A25430%2Cbr%3D3200%2Cbs%3D1%2Cot%3Dv
- ?Common-Media-Client-Data=sid%3D%E2%80%9D6e2fb550-c457-11e9-bb97-0800200c9a66
- ?Common-Media-Client-Data=sid%3D%E2%80%9D6e2fb550-c457-11e9-bb97-0800200c9a66%E2%80%9D%2Crtp%3D15000
- ?Common-Media-Client-Data=sid%3D%E2%80%9D6e2fb550-c457-11e9-bb97-0800200c9a66%E2%80%9D%2Crtp%3D15000%2Cd%3D4004%2Cot%3Dv%2Csf%3Dd
- ?Common-Media-Client-Data=d%3D4004%2Cbr%3D3200
- ?Common-Media-Client-Data=d%3D4004%2Ccom.example-myKey%3DmyValue
- ?Common-Media-Client-Data=sid%3D%E2%80%9D6e2fb550-c457-11e9-bb97-0800200c9a66%E2%80%9D%2Cd%3D2002
- ?Common-Media-Client-Data=sid%3D%E2%80%9D6e2fb550-c457-11e9-bb97-0800200c9a66%E2%80%9D%2Cnor%3D%E2%80%9D..%252F300kbps%252Fsegment35.m4v%E2%80%9D
- ?Common-Media-Client-Data=br%3D3200%2Cbs%3D3%2Ccid%3D%E2%80%9DABCD-1234%E2%80%9D%2Cd%3D4004%2Cdid%3D%E2%80%9DAndroid6.0-player-build-12.3%E2%80%9D%2Cdl%3D18000%2Cmtp%3D48175%2Cnor%3D%E2%80%9D..%252F300kbps%252Fsegment35.m4v%E2%80%9D%2Cnrr%3D12323-48763%2Cot%3Dv%2Cpr%3D1.08%2Crtp%3D12000%2Csf%3Dd%2Csid%3D%E2%80%9D6e2fb550-c457-11e9-bb97-0800200c9a66%E2%80%9D%2Cst%3Dv%2Cv%3D1

## External References

1. RFC2119 - <https://tools.ietf.org/html/rfc2119>
2. URL Encoding <https://url.spec.whatwg.org/#application/x-www-form-urlencoded>
3. UUID specification <https://www.ietf.org/rfc/rfc4122.txt>
4. RFC6302 <https://tools.ietf.org/html/rfc6302>