

CONTRIBUTORS

The following people contributed to this document:

* Guillaume Bichot (Broadpeak)
* Christoph Neumann (Broadpeak)
* Chris Lemmons (Comcast)
* Pankaj Chaudhari (Disney Streaming)
* Robert Colantuoni (Disney Streaming)
* Glenn Goldstein (Lumen)
* Will Power (Lumen)
* Rajeev RK (picoNETS)
* Shmuel Asafi (Qwilt)
* Yoav Gressel (Qwilt)
* Nir Sopher (Qwilt)
* Arnon Warshavsky (Qwilt)
* Eric Klein (Sirius XM)
* Francisco Cano Hila (Telefonica)
* Alfonso Siloniz (Telefonica)
* Ben Rosenblum (Vecima)

Notice:

This document has been created by the Streaming Video Technology Alliance. It is offered to the Alliance Membership solely as a basis for agreement and is not a binding proposal on the companies listed as resources above. The Alliance reserves the rights to at any time add, amend or withdraw statements contained herein. Nothing in this document is in any way binding to the Alliance or any of its members. The user’s attention is called to the possibility that implementation of the Alliance agreement contained herein may require the use of inventions covered by the patent rights held by third parties. By publication of this Alliance document, the Alliance makes no representation or warranty whatsoever, whether expressed or implied, that implementation of the specification will not infringe any third party rights, nor does the Alliance make any representation or warranty whatsoever, whether expressed or implied, with respect to any claim that has been or may be asserted by any third party, the validity of any patent rights related to any such claim, or the extent to which a license to use any such rights may or may not be available or the terms hereof.

©2022-2023 Streaming Video Technology Alliance

This document and translation of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction other than the following, (1) the above copyright notice and this paragraph must be included on all such copies and derivative works, and (2) this document itself may not be modified in any way, such as by removing the copyright notice or references to the Alliance, except as needed for the purpose of developing Alliance Specifications.

By downloading, copying, or using this document in any manner, the user consents to the terms and conditions of this notice. Unless the terms and conditions of this notice are breached by the user, the limited permissions granted above are perpetual and will not be revoked by the Alliance or its successors or assigns.

This document and the information contained herein is provided on an “AS IS” bases and THE ALLIANCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY, TITLE OR FITNESS FOR A PARTICULAR PURPOSE.

ABSTRACT

This specification defines configuration metadata objects related to controlling edge access to resources via content delivery networks (CDNs) and Open Caching systems. Configuring Cross-Origin Resource Sharing (CORS) access rules and the dynamic generation of CORS headers is a key feature of typical configurations, as are the ability to define response body compression rules and client connection timeouts.

Versioning

| **Version** | 1 | **Date** | draft |
| --- | --- | --- | --- |
| Extracted from Part 2 of the Configuration Interface Version 1.1 Specification with the following changes:   * Expanded documentation on CORS handling * MI.ClientConnectionControl added | | | |

Requirements

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [[RFC2119]](https://www.ietf.org/rfc/rfc2119.txt).

TABLE OF CONTENTS

[**1 OBJECTIVES AND SCOPE 6**](#_51yn7vkybcs7)

[**2 INTRODUCTION 7**](#_3znysh7)

[**3 ACRONYMS AND DEFINITIONS 8**](#_2et92p0)

[**4 MI.CrossoriginPolicy 9**](#_j0bxom85rm4)

[4.1 MI.AccessControlAllowOrigin 12](#_4bkljf3s2m4a)

[**5 MI.AllowCompress 14**](#_vr4qe3ob12bs)

[**6 MI.ClientConnectionControl 16**](#_29x3h84fw9yb)

[**7 CONCLUSION 17**](#_dlufk3jez8i1)

[**8 SPECIFICATION AND STANDARD REFERENCES 18**](#_4d34og8)

# 

# 1 OBJECTIVES AND SCOPE

The objectives of this document are:

* Define CORS configuration metadata
* Define response body compression options
* Define client connection timeout options

The scope this document ADDRESSES:

* Metadata object definitions
* Usage examples

The scope this document DOES NOT ADDRESS:

* Best practice guidelines

# 2 INTRODUCTION

CDNs typically require a set of configuration metadata to provide directives for the processing of responses downstream (at the edge and in the user agent). This document specifies GenericMetadata objects to meet those requirements, defining edge processing rules such as CORS handling, response compressions, and client connection failures.

# 3 ACRONYMS AND DEFINITIONS

This document uses the following acronyms and other terms:

| **Abbreviation** | **Description** |
| --- | --- |
| CDN | Content Delivery Network |
| CDNI | CDN Interconnection - A set of specifications published by the IETF to facilitate traffic delegation between entities in the content delivery ecosystem. |
| CORS | Cross-Origin Resource Sharing |
| dCDN | Downstream CDN - A role within the CDNI ecosystem for entities that receive delegated traffic. The role can be fulfilled by content providers, commercial CDNs, ISPs, or Open Caching systems. |
| HTTP | Hypertext Transfer Protocol |
| HTTPS | Hypertext Transfer Protocol Secure |
| IETF | Internet Engineering Task Force |
| MEL | Metadata Expression Language |
| MI | CDNI Metadata Interface - Enables a dCDN to retrieve metadata from a uCDN with directives for processing and caching requests. |
| OCN | Open Caching Node |
| Origin | Equivalent term to Source |
| Source | The endpoint from which CDNs retrieve content upon cache misses |
| SVTA | Streaming Video Technology Alliance |
| UA | User Agent |
| uCDN | Upstream CDN - A role within the CDNI ecosystem for entities that delegate traffic to dCDNs. The role can be fulfilled by content providers, commercial CDNs, ISPs, or Open Caching systems. |
| W3C | World Wide Web Consortium |

# 4 MI.CrossoriginPolicy

Delegation of traffic between CDNs over an Open Caching node (OCN) based on Hypertext Transfer Protocol (HTTP) redirection changes the domain name in the client requests. This represents a cross-origin request that must be managed appropriately using (CORS headers in the responses.

The dynamic generation of CORS headers is typical in modern HTTP request processing and avoids CORS validation forwarded to the CDN origin servers, particularly with the preflight OPTIONS requests. The CDN Interconnection (CDNI) metadata model requires extensions to specify how a CDN or Open Caching node should generate and evaluate these headers.

Required capabilities:

* Set a default value for CORS response headers independent of the origin request header value.
* Match the origin request header with a list of valid values to return or not return the CORS response headers.
* Set a list of custom headers that can be exposed to the client (expose headers).
* Support preflight requests using the OPTIONS method, including custom header validation, expose headers, and methods.
* Support credentials validation within CORS.

Simple CORS requests are those where both HTTP method and headers in the request don´t require a preflight request. The user agent (UA) request can include an origin header set to the URL domain of the webpage that the player runs. Depending on the metadata configuration, the logic to apply by the Open Caching node is:

* Validation of the origin header - Metadata can include a list of valid domains to validate the request origin header. If it does not match, the CORS header MUST NOT be included in the response.
* WIldcard usage - Depending on the configuration, the resultant CORS header to include in the response will be the same as the request origin header, or a wildcard.
* If no validation of request is included in the origin header, set a default value for CORS response headers independent of the origin request header value.

When a UA makes a request that includes a method or headers that require a CORS preflight request, the UA will use the OPTIONS method to the resource, including the origin header. If CORS is enabled and the request passes the origin validation, the OCN SHOULD respond with the set of headers that indicate what is permitted for that resource, including one or more of the following:

* Allowed methods
* Allowed credentials
* Allowed request headers
* Maximum age that the OPTIONS request is valid
* Headers that can be exposed to the client

When an upstream CDN (uCDN) configures any of those advanced parameters, it is requesting the dCDN generate synthetic responses to OPTIONS requests. Therefore, no conditional request is performed to the uCDN origin. The uCDN SHOULD configure these values taking that into account. If some of the advanced parameters are empty, the dCDN would not send the corresponding header into the UA OPTIONS request.

In cases where the uCDN only configures the MI.AccessControlAllowOrigin subobject, the dCDN will not generate synthetic responses to OPTIONS requests. Instead, the dCDN will validate with the uCDN every OPTIONS request to obtain the response.

MI.CrossoriginPolicy is a GenericMetadata object that allows for the specification of dynamically generated CORS headers.

Property: allow-origin

* Description: Validation of simple CORS requests.
* Type: Object MI.AccessControlAllowOrigin
* Mandatory-to-Specify: Yes

Property: expose-headers

* Description: A list of values the OCN will include in the Access-Control-Expose-Headers response header to a preflight request.
* Type: Array of strings
* Mandatory-to-Specify: No

Property: allow-methods

* Description: A list of values the OCN will include in the Access-Control-Allow-Methods response header to a preflight request.
* Type: Array of strings
* Mandatory-to-Specify: No

Property: allow-headers

* Description: A list of values the OCN will include in the Access-Control-Allow-Headers response header to a preflight request.
* Type: Array of strings
* Mandatory-to-Specify: No

Property: allow-credentials

* Description: The value the OCN will include in the Access-Control-Allow-Credentials response header to a preflight request.
* Type: Boolean
* Mandatory-to-Specify: No

Property: max-age

* Description: The value the OCN will include in the Access-Control-Max-Age response header to a preflight request.
* Type: Integer
* Mandatory-to-Specify: No

Property: no-origin-response-headers

* Description: In the case of a request that has no Origin field, return this set of headers with the response.
* Type: Array of MI.HTTPHeader
* Mandatory-to-Specify: No

Property: apply-to-all-methods

* Description: By default, the CORS configuration refers to OPTIONS requests. Setting this flag to “True” applies the entire CORS configuration to the other methods as well.
* Type: Boolean
* Mandatory-to-Specify: No. The default is “False”.

## 

## 4.1 MI.AccessControlAllowOrigin

The MI.AccessControlAllowOrigin object has the following properties:

Property: allow-list

* Description: List of valid URLs that will be used to match the request origin header. The origin header is an HTTP extension. Its value is a version of the Referer header in some specific requests, and used for cross-origin requests. Permitted values are schema://hostname[:port]
* Type: Array of PatternMatch objects, defined in [[RFC8006] section 4.1.5](https://datatracker.ietf.org/doc/html/rfc8006#section-4.1.5)
* Mandatory-to-Specify: Yes

Property: wildcard-return

* Description: If “True”, the OCN will include a wildcard (\*) in the Access-Control-Allow-Origin response header. If “False”, the OCN will reflect the request origin header in the Access-Control-Allow-Origin response header.
* Type: Boolean
* Mandatory-to-Specify: Yes

The examples below demonstrate how to configure response headers dynamically for CORS validation.

The following is an example of a simple CORS validation configuration:

| {  "generic-metadata-type": "MI.CrossoriginPolicy",  "generic-metadata-value": {  "allow-origin": {  "allow-list": [  {  "pattern": "\*"  }  ],  "wildcard-return": true  }  } } |
| --- |

The following is an example of validation of a preflight request when some of the headers included in the subsequent object request are not included in the CORS specification safelist:

| {  "generic-metadata-type": "MI.CrossoriginPolicy",  "generic-metadata-value": {  "allow-origin": {  "allow-list": [  {  "pattern": "\*://sourcepage.example.com"  },  "wildcard-return": false  },  "allow-methods": [ "GET", "POST" ],  "allow-credentials": true,  "allow-headers": [ "X-PINGOTHER", "Content-Type" ],  "expose-headers": [ "X-User", "Authorization" ],  "max-age": 3600  }  } } |
| --- |

# 

# 5 MI.AllowCompress

Downstream CDNs often have the ability to compress HTTP response bodies in cases where the client has declared that it can accept compressed responses (via an Accept-Encoding header), but the source/origin has returned an uncompressed response.

The specific compression algorithm used by the dCDN is negotiated by the client’s Accept-Encoding header according to [[RFC9110]](https://datatracker.ietf.org/doc/html/rfc9110) (including “q=” preferences) and the compression capabilities available on the dCDN.

In addition, HeaderTransform allows the uCDN to normalize, or modify, the Accept-Encoding header to allow for fine-grain control over the selection of the compression algorithm (e.g., gzip, compress, deflate, br, etc.).

MI.AllowCompress is a new GenericMetadata object that allows the dCDN to compress content before sending it to the client.

Property: allow-compress

* Description: If set to “True”, the dCDN will try to compress the response to the client based on the Accept-Encoding request header.
* Type: Boolean. The values are “True” or “False”.
* Mandatory-to-Specify: No. The default is “False”.

The following examples illustrate the use of MI.AllowCompress in the context of the Processing Stages Model that allowed for metadata to be applied conditionally based on evaluation of HTTP request headers. See the Processing Stages Metadata Specification [[SVTA2032]](https://svta.org/documents/SVTA2032) and Metadata Model Expression Language (MEL) Specification [[SVTA2031]](https://svta.org/documents/SVTA2031).

The following is an example of an MI.AllowCompress that allows manifests (\*.m3u8) to be compressed by the dCDN:

| {  "match": {  "expression": "req.h.uri \*= '\*.m3u8'"  },  "stage-metadata": {  "generic-metadata": [  {  "generic-metadata-type": "MI.AllowCompress",  "generic-metadata-value": {  "allow-compress": true  }  }  ]  } } |
| --- |

The following is an example of an MI.AllowCompress that allows manifests (\*.m3u8) to be compressed by the dCDN but normalizes the client’s Accept-Encoding header:

| {  "match": {  "expression": "req.h.accept-encoding \*= '\*gzip\*'"  },  "stage-metadata": {  "generic-metadata": [  {  "generic-metadata-type": "MI.AllowCompress",  "generic-metadata-value": {  "allow-compress": true  }  }  ]  } } |
| --- |

# 6 MI.ClientConnectionControl

Configuration metadata is required to define how connections against a client are maintained by a dCDN. Since the clients are typically owned/operated by a uCDN, giving this control to a uCDN allows it to accommodate device-specific constraints and performance optimization. A dCDN can also benefit from this configuration metadata to meet its security and resource consumption requirements.

MI.ClientConnectionControl is a new GenericMetadata object that specifies how a dCDN manages its connections to clients/players.

Property: connection-keep-alive-time-ms

* Description: Specifies the time, in milliseconds, to keep an idle connection open.
* Type: Integer
* Mandatory-to-Specify: No. When not specified, a default value selected by the dCDN will be used.

The following example shows how a connection setup and a keep alive timeout can be set for client connections to a dCDN:

| {  "generic-metadata-type": "MI.ClientConnectionControl",  "generic-metadata-value": {  "connection-keep-alive-time-ms": 3  }  } |
| --- |

# 

# 7 CONCLUSION

This specification has defined configuration metadata objects related to controlling edge access to resources via CDNs and Open Caching systems. It has added to the set of basic CDNI configuration metadata objects defined in [[RFC8006]](https://datatracker.ietf.org/doc/html/rfc8006), and can be extended in the future with additional Edge Control Metadata object definitions.

# 8 SPECIFICATION AND STANDARD REFERENCES

| **Spec Number** | **Specification Name** | **Org** | **More Information** |
| --- | --- | --- | --- |
| RFC2119 | Key words for use in RFCs to Indicate Requirement Level | IETF | <https://datatracker.ietf.org/doc/html/rfc2119> |
| RFC8006 | Content Delivery Network Interconnection (CDNI) Metadata | IETF | <https://datatracker.ietf.org/doc/html/rfc8006> |
| RFC9110 | HTTP Semantics | IETF | <https://datatracker.ietf.org/doc/html/rfc9110> |
| SVTA2031 | Configuration Interface Part 2a: Metadata Expression Language (MEL) | SVTA | <https://svta.org/documents/SVTA2031> |
| SVTA2032 | Configuration Interface Part 2b: Processing Stages Metadata | SVTA | <https://svta.org/documents/SVTA2032> |

ABOUT THE STREAMING VIDEO TECHNOLOGY ALLIANCE

Comprised of members from across the video ecosystem, the Streaming Video Technology Alliance (SVTA) is a global association that works to solve critical streaming video challenges in an effort to improve end-user experience and adoption. The organization focuses on three main activities: first is to educate the industry on challenges, technologies, and trends through informative, publicly-available resources such as whitepapers, articles, and e-books; second is to foster collaboration among different video ecosystem players through working groups, quarterly meetings, and conferences; third is to define solutions for streaming video challenges by producing specifications, best practices, and other technical documentation. For more information, please visit [svta.org](http://www.streamingvideoalliance.org).

Streaming Video Technology Alliance

5177 Brandin Court

Fremont, CA 94538 USA

(510) 492-4000

[svta.org](https://www.svta.org/) © 2022-2023 Streaming Video Technology Alliance