CORS CDMI Extension

Version 2.0

ABSTRACT: This CDMI Extension is intended for developers who are considering a standardized way to add functionality to CDMI. When multiple compatible implementations are demonstrated and approved by the Technical Working Group, this extension will be incorporated into the CDMI standard.

This document has been released and approved by the SNIA. The SNIA believes that the ideas, methodologies, and technologies described in this document accurately represent the SNIA goals and are appropriate for widespread distribution. Suggestion for revision should be directed to http://www.snia.org/feedback/.

SNIA Working Draft

November 4, 2020

USAGE

DISCLAIMER

The information contained in this publication is subject to change without notice. The SNIA makes no warranty of any kind with regard to this specification, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The SNIA shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this specification.

Suggestions for revisions should be directed to https://www.snia.org/feedback/.

Contents

Clause 1

CORS CDMI Extension

1.1 Overview

HTTP user agents, including HTTP proxies, web browsers, and secure web applications, commonly apply same-origin restrictions to network requests. These restrictions prevent a client-side web application loaded from one origin from obtaining or modifying data retrieved from another origin. These restrictions also limit unsafe HTTP requests that can be performed against destinations that differ from the running application's origin. Many web browsers also treat requests with custom headers as unsafe HTTP requests. To operate in a multi-origin environment, CDMI servers must comply with the W3C Cross-Origin Resource Sharing (CORS) Specification.

This extension specifies how CDMI works with the CORS Specification.

1.2 Instructions to the Editor

To merge this extension into the CDMI 2.0.0 specification, make the following changes:

1. Insert into preamble/normative_references.txt, as follows:

RFC 6454, *The Web Origin Concept* - see [rfc6454]

REC-cors-20140116, *Cross-Origin Resource Sharing* - see [REC-cors-20140116]

2. Insert into preamble/terms.txt, as follows:

x.x

Cross-Origin Request Sharing (CORS) **|br|** A method by which resources hosted on one domain can be permitted to be accessed by resources hosted on a second, different domain **|br|**

3. Insert into references/normative.bib and references/refs.bib, as follows:

@Misc{rfc6454, author = {Adam Barth}, title = {{The Web Origin Concept}}, howpublished = {RFC 6454}, month = dec, year = {2011}, abstract = {This document defines the concept of an “origin”, which is often used as the scope of authority or privilege by user agents. Typically, user agents isolate content retrieved from different origins to prevent malicious web site operators from interfering with the operation of benign web sites. In addition to outlining the principles that underlie the concept of origin, this document details how to determine the origin of a URI and how to serialize an origin into a string. It also defines an HTTP header field, named “Origin”, that indicates which origins are associated with an HTTP request.}, doi = {10.17487/RFC6454}, number = {6454}, owner = {Peter van Liesdonk}, pagetotal = {20}, publisher = {RFC Editor}, series = {Request for Comments}, url = {https://rfc-editor.org/rfc/rfc6454.txt},

}

4. Insert into references/refs.bib, as follows:

@Misc{REC-cors-20140116, author = {Anne van Kesteren}, title = {Cross-Origin Resource Sharing}, howpublished = {TR/2014/REC-cors-20140116/}, month = jan, year = {2014}, url = {https://www.w3.org/TR/2014/REC-cors-20140116/}, abstract = {This document defines a mechanism to enable client-side cross-origin requests. Specifications that enable an API to make cross-origin requests to resources can use the algorithms defined by this specification. If such an API is used on http://example.org resources, a resource on http://hello-world.example can opt in using the mechanism described by this specification (e.g., specifying Access-Control-Allow-Origin: http://example.org as response header), which would allow that resource to be fetched cross-origin from http://example.org.}, publisher = {World Wide Web Consortium}

}

5. Add an entry to the end of the table starting on line 135 of cdmi_advanced/cdmi_capability_object.txt, as follows:

Table 1: System-wide capabilities

Capability name	Type	Definition
`cdmi_cors`	JSON string	If present and “`true`”, indicates that the cloud storage system supports CORS.

6. Add an entry to the end of the table starting on line 451 of cdmi_advanced/cdmi_capability_object.txt, as follows:

Table 2: Capabilities for data system metadata

Capability name	Type	Definition
`cdmi_cors_methods`	JSON array of JSON strings	When the cloud storage system supports the `cdmi_cors_methods` data system metadata as defined in `ref_support_for_data_system_metadata`, the `cdmi_cors_methods` capability shall be present and contain a list of HTTP methods supported. When this capability is absent, or present and is an empty JSON array, `cdmi_cors_methods` data system metadata shall not be used. When a cloud storage system supports CORS, the system-wide capability of `cdmi_cors` specified in `ref_cloud_storage_system-wide_capabilities` shall be present and set to “true”.
`cdmi_cors_origins`	JSON String	When the cloud storage system supports the `cdmi_cors_origins` data system metadata as defined in `ref_support_for_data_system_metadata`, the `cdmi_cors_origins` capability shall be present and set to the string value “`true`”. When this capability is absent, or present and set to the string value “`false`”, `cdmi_cors_origins` data system metadata shall not be used.
`cdmi_cors_headers`	JSON String	When the cloud storage system supports the `cdmi_cors_headers` data system metadata as defined in `ref_support_for_data_system_metadata`, the `cdmi_cors_headers` capability shall be present and set to the string value “`true`”. When this capability is absent, or present and set to the string value “`false`”, `cdmi_cors_headers` data system metadata shall not be used.

7. Add an entry to the end of the table starting on line 216 of cdmi_advanced/cdmi_metadata.txt, as follows:

Table 3: Data system metadata

Metadata name	Type	Description	Requirement
cdmi_cors_methods	JSON array of JSON strings	If this data system metadata item is present, it indicates that contained CORS request methods are permitted, and shall be present in the "Access-Control-Allow-Methods" header. When this data system metadata item is absent, the "Access-Control-Allow-Methods" header shall also be absent. Supported request methods are expressed as JSON strings, as defined in section 9 of [rfc2616].	Optional
cdmi_cors_origins	JSON array of JSON strings	If this data system metadata item is present, it indicates that the contained CORS source origins are permitted, and shall be present in the "Access-Control-Allow-Origin" header. When this data system metadata item is absent, the "Access-Control-Allow-Origin" header shall also be absent. Supported source origins are expressed as JSON strings, as defined in section 4 of :cite:`rfc6454`, with "``*``" indicating that all source origins are permitted.	Optional
cdmi_cors_headers	JSON array of JSON strings	If this data system metadata item is present, it indicates that the contained headers are permitted, and shall be present in the "Access-Control-Allow-Headers" header. When this data system metadata item is absent, the "Access-Control-Allow-Headers" header shall also be absent. Supported request headers are expressed as JSON strings, as defined in section 5.3 of :cite:`rfc2616`.	Optional

8. Create new clause, "cdmi_cors.txt" after existing clause 25 "Data Object Versions", as follows.

Clause 2

Cross-Origin Request Sharing (CORS)

2.1 Overview

HTTP clients that conform to the W3C Cross-Origin Resource Sharing (CORS) Specification, such as web browsers and web applications, apply restrictions on network requests sent to a different domain (origin) then the initiating resource is hosted from. These network requests require additional headers (and sometimes additional “preflight” requests) to ensure that the server will permit a cross-domain operation to be performed.

As CDMI is based on HTTP, in order to operate in a multi-origin environment, CDMI servers shall provide these headers, as defined in the W3C Cross-Origin Resource Sharing Specification, in order to permit cross-origin HTTP operations.

2.2 Non-Preflight Operations

Non-preflight operations (GET, HEAD and POST) require the HTTP client to include a new “`Origin`” header, indicating the domain where the initiating resource is hosted, and requires the server to return an “`Access-Control-Allow-Origin`” header to indicate if the operation is permitted.

EXAMPLE 1: Non-preflight request

```
--> GET /cdmi/2.0.0/data.txt HTTP/1.1
--> Host: cloud.example.com
--> Accept: */*
--> Origin: http://app.example.com

<-- HTTP/1.1 200 OK
<-- Access-Control-Allow-Origin: *
<-- Content-Type: text/plain
<-- Content-Length: 6
<--
<-- [text]
```

2.3 Preflight Operations

Preflight operations (PUT, DELETE and OPTIONS, as well as when custom headers are used) require the HTTP client to perform a “preflight” operation. This preflight operation is a OPTIONS operation with “`Access-Control-Request-Method`” and “`Access-Control-Request-Headers`” headers, and is responded to with “`Access-Control-Allow-Origin`”, “`Access-Control-Allow-Methods`”, and “`Access-Control-Allow-Headers`” headers.

If the preflight operation is successful, and the returned headers indicate that the CORS request is permitted, the HTTP client will then perform the original operation, and shall include an “`Origin`” header, and requires the server to return an “`Access-Control-Allow-Origin`” header, as with non-preflight operations.

EXAMPLE 2: Preflight request

```
--> OPTIONS /cdmi/2.0.0/data.txt HTTP/1.1
--> Host: cloud.example.com
--> Accept: */*
--> Origin: http://app.example.com
--> Access-Control-Reqeuest-Method: PUT
--> Access-Control-Request-Headers: Content-Type

<-- HTTP/1.1 200 OK
<-- Access-Control-Allow-Origin: http://app.example.com
<-- Access-Control-Allow-Methods: GET, PUT, DELETE, OPTIONS
<-- Access-Control-Allow-Headers: Content-Type
<-- Access-Control-Max-Age: 3600
```

This instructs the client that the PUT operation is permitted.

EXAMPLE 3: CORS-enabled PUT

```
--> PUT /cdmi/2.0.0/data.txt HTTP/1.1
--> Host: cloud.example.com
--> Content-Type: text/plain;charset=utf-8
--> Content-Length: 37
--> Origin: http://app.example.com
-->
--> This is the Value of this Data Object

<-- HTTP/1.1 201 Created
<-- Access-Control-Allow-Origin: *
```

2.4 Cross-Origin Rules

1. A request shall be considered a CORS request if the “Origin” header is present
2. A request shall be considered a CORS preflight request if the request type is “OPTIONS” and the “Origin” header is present
3. For all CORS requests, if the “cdmi_cors_methods” metadata contains the request method, and the “cdmi_cors_origins” metadata matches against the “Origin” request header value (or the metadata item contains “*”), permit the operation and include an “Access-Control-Allow-Origin” response header with the contents of the request “Origin” request header value (or “*” if matched against “*” in the metadata item).
4. For all CORS preflight requests, if the “cdmi_cors_methods” metadata matches against the “Access-Control-Request-Method” request header value, and the “cdmi_cors_origins” metadata matches against the “Origin” request header value (or the metadata item contains “*”), respond to the request with:
   - An “Access-Control-Allow-Origin” response header with the contents of the request “Origin” request header value (or “*” if matched against “*” in the metadata item)
   - An “Access-Control-Allow-Methods” response header with the intersection of the methods specified in the “Access-Control-Request-Method” request header and the methods in the “cdmi_cors_methods” metadata item
   - An “Access-Control-Allow-Headers” response header with the intersection of the headers specified in the “Access-Control-Request-Headers” request header and the headers in the “cdmi_cors_headers” metadata item
   - An “Access-Control-Max-Age” response header with a value sufficiently high enough to permit the HTTP Client to perform the corresponding HTTP operation.