

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 25, 2019

O. Finkelman
Qwilt
S. Mishra
Verizon
October 22, 2018

CDNI Control Triggers Interface Extensions
draft-finkelman-cdni-triggers-sva-extensions-01

Abstract

The Open Caching working group of the Streaming Video Alliance is focused on the delegation of video delivery request from commercial CDNs to a caching layer at the ISP. In that aspect, Open Caching is a specific use case of CDNI, where the commercial CDN is the upstream CDN (uCDN) and the ISP caching layer is the downstream CDN (dCDN). The extensions specified in this document to the CDNI CI/T interface are derived from requirements raised by Open Caching but are applicable to CDNI use cases in general.

Requirements Language

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 [RFC 2119](#) [[RFC2119](#)].

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 25, 2019.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](https://trustee.ietf.org/license-info) and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

- [1. Introduction](#) [3](#)
- [1.1. Terminology](#) [3](#)
- [2. Interfaces Extensions Overview](#) [4](#)
- [2.1. CDNI Control Interface / Triggers Extensions](#) [4](#)
- [2.1.1. CI/T Objects](#) [4](#)
- [2.1.2. Trigger Specification](#) [4](#)
- [2.1.3. Content Selection](#) [4](#)
- [2.1.4. Trigger Extensibility](#) [5](#)
- [2.1.5. Error Propagation](#) [5](#)
- [2.2. CDNI Footprint and Capabilities Interface Extensions](#) [6](#)
- [3. CI/T Version 2](#) [6](#)
- [3.1. CI/T Objects V2](#) [6](#)
- [3.2. Properties of CI/T Version 2 objects](#) [9](#)
- [3.2.1. Trigger Specification Version 2](#) [9](#)
- [3.2.2. RegexpMatch](#) [10](#)
- [3.2.3. Playlist](#) [12](#)
- [3.2.4. MediaProtocol](#) [13](#)
- [3.2.5. CI/T Trigger Extensions](#) [13](#)
- [3.2.5.1. Enforcement Options](#) [13](#)
- [3.2.5.2. GenericExtensionObject](#) [16](#)
- [3.2.6. Error Description Version 2](#) [18](#)
- [3.2.7. Error codes](#) [19](#)
- [4. Trigger Extension Objects](#) [19](#)
- [4.1. LocationPolicy extension](#) [19](#)
- [4.2. TimePolicy Extension](#) [21](#)
- [5. Footprint and Capabilities](#) [23](#)
- [5.1. CI/T Versions Capability Object](#) [23](#)
- [5.1.1. CI/T Versions Capability Object Serialization](#) [24](#)
- [5.2. CI/T Playlist Protocol Capability Object](#) [24](#)
- [5.2.1. CI/T Playlist Protocol Capability Object
 Serialization](#) [24](#)

[5.3.](#) CI/T Trigger Extension Capability Object [25](#)
 5.3.1. CI/T Trigger Extension Capability Object
 Serialization [25](#)

[6.](#) IANA Considerations [26](#)
 [6.1.](#) CDNI Payload Types [26](#)
 [6.1.1.](#) CDNI ci-trigger-command.v2 Payload Type [26](#)
 [6.1.2.](#) CDNI ci-trigger-status.v2 Payload Type [27](#)
 [6.1.3.](#) CDNI CI/T LocationPolicy Trigger Extension Type [27](#)
 [6.1.4.](#) CDNI CI/T TimePolicy Trigger Extension Type [27](#)
 [6.1.5.](#) CDNI FCI CI/T Versions Payload Type [27](#)
 [6.1.6.](#) CDNI FCI CI/T Playlist Protocol Payload Type [27](#)
 [6.1.7.](#) CDNI FCI CI/T Extension Objects Payload Type [28](#)
 [6.2.](#) CDNI CI/T Trigger Error Codes types [28](#)
 [6.3.](#) CDNI Media protocol types [28](#)

[7.](#) Security Considerations [29](#)

[8.](#) Acknowledgments [29](#)

[9.](#) Contributors [29](#)

[10.](#) References [30](#)
 [10.1.](#) Normative References [30](#)
 [10.2.](#) Informative References [30](#)

Authors' Addresses [31](#)

[1.](#) Introduction

This document defines the objects and extensions required for granular content management operations. For that purpose it extends CDNI Control Interface/Triggers [[RFC8007](#)]. The basic operations are the ones defined in the RFC (i.e. purge, invalidate, pre-position). For consistency, this document follows the CDNI notation of uCDN (the commercial CDN) and dCDN (the ISP caching layer). When using the term CP in this document we refer to a video content provider.

The CDNI metadata interface is described in [[RFC8006](#)].

The CDNI footprint and capability interface is described in [[RFC8008](#)].

The CDNI control interface / triggers is described in [[RFC8007](#)].

[1.1.](#) Terminology

This document reuses the terminology defined in [[RFC6707](#)], [[RFC8006](#)], [[RFC8007](#)], and [[RFC8008](#)].

Additionally, the following terms are used throughout this document and are defined as follows:

- o HLS - HTTP Live Streaming

- o DASH - Dynamic Adaptive Streaming Over HTTP
- o MSS - Microsoft Smooth Streaming

2. Interfaces Extensions Overview

This document defines extensions for the CDNI Control Interface / Triggers [[RFC8007](#)] and defines FCI objects as per the CDNI Footprint and Capabilities Interface [[RFC8008](#)].

2.1. CDNI Control Interface / Triggers Extensions

2.1.1. CI/T Objects

This document specifies version 2 of the CI/T objects in order to support version 2 of the Trigger Specification as required below in [Section 2.1.2](#).

2.1.2. Trigger Specification

This document specifies version 2 of the Trigger Specification which is an enhancement of the Trigger Specification that includes all properties as defined in [section 5.2.1 of \[RFC8007\]](#) as well as the additional properties required by the use cases listed below in [Section 2.1.3](#) and [Section 2.1.4](#).

2.1.3. Content Selection

The trigger specification as defined in [section 5.2.1 of \[RFC8007\]](#) provides means to select content objects by matching a full content URL or patterns with wildcards. This document specifies two additional selection options.

- o Regular Expression - Using regex a uCDN can create more complex rules to select the content objects for the cases of invalidation and purge. For example, purging specific content within a specific directory path.
- o Content Playlist - Using video playlist files, a uCDN can trigger an operation that will be applied to a collection of distinct media files in a format that is natural for a streaming video content provider. A playlist may have several formats, specifically HTTP Live Streaming (HLS) *.m3u8 manifest [[RFC8216](#)], Microsoft Smooth Streaming (MSS) *.ismc client manifest [[MSS](#)], and Dynamic Adaptive Streaming over HTTP (DASH) *.mpd file [[ISO/IEC 23009-1:2014](#)] [[MPEG-DASH](#)].

2.1.4. Trigger Extensibility

The CDNI Control Interface / Triggers [[RFC8007](#)] defines a set of objects used by the trigger commands. In order to have better control and finer granularity, we define a mechanism for generic trigger extension object wrapper for managing individual CDNI trigger extensions in an opaque manner, as well as an initial set of trigger extension objects.

This document also registers CDNI Payload Types [[RFC7736](#)] under the namespace CIT for the initial set of trigger extension types:

- o CIT.LocationPolicy (for controlling the locations in which the trigger is executed)
- o CIT.TimePolicy (for scheduling a trigger to run in a specific time window)

Example use cases

- o Pre-position with cache location policy
- o Purge content with cache location policy
- o Pre-position at a specific time
- o Purge by content acquisition time (e.g. purge all content acquired in the past X hours)

2.1.5. Error Propagation

As triggers may be propagated over a chain of downstream CDNs and since, in some cases, triggers may be redistributed from dCDN-A to dCDN-B even if dCDN-A does not understand a specific extension, it is essential for the uCDN that sets the trigger to be able to trace back and error to the downstream where it occurred. This document specifies version 2 of the Error Description which is an enhancement of the Error Description as defined in [section 5.2.6 of \[RFC8007\]](#) and that includes all the original properties as well as the additional property "cdn" which is an identifier for the faulty CDN. When a downstream dCDN-A propagates a trigger to another downstream dCDN-B, it MUST also propagate back the errors received in the trigger status resource from dCDN-B. This makes sure that the trigger originating upstream CDN will receive an array of errors that occurred in all the CDNs along the execution path, each error carrying its own CDN identifier.

2.2. CDNI Footprint and Capabilities Interface Extensions

Extending the trigger mechanism with optional properties requires the ability for the dCDN to advertise which optional properties it supports.

The CDNI Footprint and Capabilities Interface [[RFC8008](#)] enables the dCDN to advertise the capabilities it supports across different footprints. This document introduces FCI objects to support the advertisement of these optional properties.

Example use cases

- o Trigger types: Advertise which trigger types are supported by the dCDN. CDNI defines three trigger types (purge, invalidate, pre-position), but it does not necessarily mean that all dCDNs support all of them. The uCDN may prefer to work only with dCDN that support what the uCDN needs.
- o Content selection rule types: Advertise which selection types are supported. For example, if adding content regex as a means to match on content URLs, not all dCDN would support it. For playlist mapping, advertise which types and versions of protocols are supported, e.g. HLS.vX/DASH.vY/MSS.vX, DASH templates. Note that the version string or schema are protocol specific.
- o Trigger extensions: Advertise which trigger extensions object types are supported by the dCDN.

3. CI/T Version 2

[RFC8007] does not define a version number and versioning scheme. We, therefore, designate the interface and objects as defined in [section 5 of \[RFC8007\]](#) as version 1. The following sections define version 2 of the CI/T objects and their properties as extensions of version 1.

3.1. CI/T Objects V2

Version 2 of the CI/T interface requires the support of the following objects:

- o CI/T Commands v2: A trigger command request using the payload type ci-trigger-command.v2. Version 2 MUST only use "trigger.v2" objects as defined in [Section 3.2.1](#), instead of "trigger" objects. All other properties of the trigger command v2 are as defined in [section 5.1.1 of \[RFC8007\]](#).

- o Trigger Status Resource v2: A trigger status resource response using the payload type ci-trigger-status.v2. Version 2 MUST only use "trigger.v2" objects as defined in [Section 3.2.1](#), instead of a "trigger" object, as well as "errors.v2" objects as defined in [Section 3.2.6](#), instead of a "errors" object. All other properties of the trigger status v2 are as defined in [section 5.1.2 of \[RFC8007\]](#). The errors array "errors.v2" is a list of all errors that occurred in any of the downstream CDNs along the execution path. When a downstream CDN, dCDN-A, propagates a trigger to another downstream CDN, dCDN-B, it MUST also propagate back all errors reported by dCDN-B in the trigger status resource and add them to its own trigger status resource.
- o Trigger Collections: The payload type ci-trigger-collection is used with no changes and as defined in 5.1.3 of [\[RFC8007\]](#).

Usage example of version 2 of trigger command

REQUEST:

```
POST /triggers HTTP/1.1
User-Agent: example-user-agent/0.1
Host: triggers.dcdn.example.com
Accept: */*
Content-Type: application/cdni; ptype=ci-trigger-command.v2
{
  "trigger.v2": { <properties of a trigger.v2 object> },
  "cdn-path": [ "AS64496:1" ]
}
```

RESPONSE:

```
HTTP/1.1 201 Created
Date: Wed, 04 May 2016 08:48:10 GMT
Content-Length: 467
Content-Type: application/cdni; ptype=ci-trigger-status.v2
Location: https://triggers.dcdn.example.com/triggers/0
Server: example-server/0.1

{
  "errors.v2": [ { <properties of 1st error.v2 object> },
                 ...,
                 { <properties of Nth error.v2 object> }
  ],
  "ctime": 1462351690,
  "etime": 1462351698,
  "mtime": 1462351690,
  "status": "pending",
  "trigger.v2": { <properties of a trigger.v2 object> }
}
```

Usage example of version 2 of trigger status for the trigger created in the above trigger command example:

REQUEST:

```

GET /triggers/0 HTTP/1.1
User-Agent: example-user-agent/0.1
Host: triggers.dcdn.example.com
Accept: */*

```

RESPONSE:

```

HTTP/1.1 200 OK
Content-Length: 467
Expires: Wed, 04 May 2016 08:49:10 GMT
Server: example-server/0.1
ETag: "6990548174277557683"
Cache-Control: max-age=60
Date: Wed, 04 May 2016 08:48:10 GMT
Content-Type: application/cdni; ptype=ci-trigger-status.v2

{
  "errors.v2": [ { <properties of 1st error.v2 object> },
                ...,
                { <properties of Nth error.v2 object> }
  ],
  "ctime": 1462351690,
  "etime": 1462351698,
  "mtime": 1462351690,
  "status": "pending",
  "trigger.v2": { <properties of a trigger.v2 object> }
}

```

3.2. Properties of CI/T Version 2 objects

This section defines the values that can appear in the top-level objects described in [Section 3.1](#), and their encodings.

3.2.1. Trigger Specification Version 2

Version 2 of the Trigger Specification adds the following properties on top of the existing properties of the trigger specification defined in [section 5.2.1 of \[RFC8007\]](#).

Property: content.regexs

Description: Regexs of content URLs to which the CI/T trigger command applies.

Type: A JSON array of RegexpMatch objects (see [Section 3.2.2](#)).

Mandatory: No, but at least one of "metadata.*" or "content.*" MUST be present and non-empty.

Property: content.playlists

Description: Playlists of content the CI/T trigger command applies to.

Type: A JSON array of Playlist objects (see [Section 3.2.3](#)).

Mandatory: No, but at least one of "metadata.*" or "content.*" MUST be present and non-empty.

Property: extensions

Description: Array of trigger extension data.

Type: Array of GenericTriggerExtension objects (see [Section 3.2.5.2](#)).

Mandatory-to-Specify: No. The default is no extensions.

Example of an invalidation trigger.v2 with a list of regex objects, a list of playlist objects, and extensions:

```
{
  "trigger.v2": {
    "type": "invalidate",
    "content.regexs": [ <list of RegexMatch objects> ],
    "content.playlists": [ <list of Playlist objects> ],
    "extensions": [ <list of GenericTriggerExtension objects> ]
  },
  "cdn-path": [ "AS64496:1" ]
}
```

[3.2.2](#). **RegexMatch**

A RegexMatch consists of a regular expression string a URI is matched against, and flags describing the type of match. It is encoded as a JSON object with following properties:

Property: regex

Description: A regular expression for URI matching.

Type: A regular expression to match against the URI, i.e against the path-absolute and the query string parameters

[RFC3986]. The regular expression string MUST be compatible with PCRE [PCRE841].

Note: Because '\\' has special meaning in JSON [RFC8259] as the escape character within JSON strings, the regular expression character '\\' MUST be escaped as '\\\\'.

Mandatory: Yes.

Property: case-sensitive

Description: Flag indicating whether or not case-sensitive matching should be used.

Type: JSON boolean. Either "true" (the matching is case sensitive) or "false" (the matching is case insensitive).

Mandatory: No; default is case-insensitive match (i.e., a value of "false").

Property: match-query-string

Description: Flag indicating whether to include the query part of the URI when comparing against the regex.

Type: JSON boolean. Either "true" (the full URI, including the query part, should be compared against the regex) or "false" (the query part of the URI should be dropped before comparison with the given regex).

Mandatory: No; default is "false". The query part of the URI MUST be dropped before comparison with the given regex. This makes the regular expression simpler and safer for cases in which the query parameters are not relevant for the match.

Example of a case sensitive, no query parameters, regex match against:

```
"^(https:\\\\video\\.example\\.com)\\/([a-z])\\/movie1\\/([1-7])\\/*(index.m3u8|\\d{3}.ts)$"
```

This regex matches URLs of domain video.example.com where the path structure is /(single lower case letter)/(name-of-title)/(single digit between 1 to 7)/(index.m3u8 or a 3 digit number with ts extension). For example: https://video.example.com/d/movie1/5/index.m3u8 or https://video.example.com/k/movie1/4/013.ts.


```
{
  "regex": "^(https:\\/\\/video\\.example\\.com)\\/([a-z])\\/movie1\\
  \\/[0-9]\\/*(index.m3u8|\\d{3}.ts)$",
  "case-sensitive": true,
  "match-query-string": false
}
```

[3.2.3.](#) Playlist

A Playlist consists of a full URL and a media protocol identifier. An implementation that supports a specific playlist media protocol MUST be able to parse playlist files of that protocol type and extract, possibly recursively, the URLs to all media objects and/or sub playlist files, and apply the trigger to each one of them separately.

Playlist is encoded as a JSON object with following properties:

Property: playlist

Description: A URL to the playlist file.

Type: A URL represented as a JSON string.

Mandatory: Yes.

Property: media-protocol

Description: Media protocol to be when parsing and interpreting this playlist.

Type: MediaProtocol (see [Section 3.2.4](#)).

Mandatory: Yes.

Example of a HLS playlist:

```
{
  "playlist": "https://www.example.com/hls/title/index.m3u8",
  "media-protocol": "hls"
}
```


3.2.4. MediaProtocol

Media Protocol objects are used to specify registered type of media protocol (see [Section 6.3](#)) used for protocol related operations like pre-position according to playlist.

Type: JSON string

Example:

"dash"

3.2.5. CI/T Trigger Extensions

A "trigger.v2" object, as defined in [Section 3.2.1](#) includes an optional array of trigger extension objects. A trigger extension contain properties that are used as directives for dCDN when executing the trigger command -- for example, location policies, time policies and so on. Each such CDNI Trigger extension is a specialization of a CDNI GenericTriggerExtension object. The GenericTriggerExtension object abstracts the basic information required for trigger distribution from the specifics of any given property (i.e., property semantics, enforcement options, etc.). All trigger extensions are optional, and it is thus the responsibility of the extension specification to define a consistent default behavior for the case the extension is not present.

3.2.5.1. Enforcement Options

The trigger enforcement options concept is in accordance with the metadata enforcement options as defined in [section 3.2 of \[RFC8006\]](#).

The GenericTriggerExtension object defines the properties contained within it as well as whether or not the properties are "mandatory-to-enforce". If the dCDN does not understand or support a mandatory-to-enforce property, the dCDN MUST NOT execute the trigger command. If the extension is not mandatory-to-enforce, then that GenericTriggerExtension object can be safely ignored and the trigger command can be processed in accordance with the rest of the CDNI Trigger spec.

Although a CDN MUST NOT execute a trigger command if a mandatory-to-enforce extension cannot be enforced, it could still be safe to redistribute that trigger (the "safe-to-redistribute" property) to another CDN without modification. For example, in the cascaded CDN case, a transit CDN (tCDN) could convey mandatory-to-enforce trigger extension to a dCDN. For a trigger extension that does not require customization or translation (i.e., trigger extension that is safe-

to-redistribute), the data representation received off the wire MAY be stored and redistributed without being understood or supported by the tCDN. However, for trigger extension that requires translation, transparent redistribution of the uCDN trigger values might not be appropriate. Certain triggers extensions can be safely, though perhaps not optimally, redistributed unmodified. For example, pre-position command might be executed in suboptimal times for some geographies if transparently redistributed, but it might still work.

Redistribution safety MUST be specified for each GenericTriggerExtension property. If a CDN does not understand or support a given GenericTriggerExtension property that is not safe-to-redistribute, the CDN MUST set the "incomprehensible" flag to true for that GenericTriggerExtension object before redistributing it. The "incomprehensible" flag signals to a dCDN that trigger metadata was not properly transformed by the tCDN. A CDN MUST NOT attempt to execute a trigger that has been marked as "incomprehensible" by a uCDN.

tCDNs MUST NOT change the value of mandatory-to-enforce or safe-to-redistribute when propagating a trigger to a dCDN. Although a tCDN can set the value of "incomprehensible" to true, a tCDN MUST NOT change the value of "incomprehensible" from true to false.

Table 1 describes the action to be taken by a tCDN for the different combinations of mandatory-to-enforce ("MtE") and safe-to-redistribute ("StR") properties when the tCDN either does or does not understand the trigger extension object in question:

MtE	StR	Extension object understood by tCDN	Trigger action
False	True	True	Can execute and redistribute.
False	True	False	Can execute and redistribute.
False	False	False	Can execute. MUST set "incomprehensible" to true when redistributing.
False	False	True	Can execute. Can redistribute after transforming the trigger extension (if the CDN knows how to do so safely); otherwise, MUST set "incomprehensible" to true when redistributing.
True	True	True	Can execute and redistribute.
True	True	False	MUST NOT execute but can redistribute..
True	False	True	Can execute. Can redistribute after transforming the trigger extension (if the CDN knows how to do so safely); otherwise, MUST set "incomprehensible" to true when redistributing.
True	False	False	MUST NOT serve. MUST set "incomprehensible" to true when redistributing.

Table 1: Action to be taken by a tCDN for the different combinations of MtE and StR properties

Table 2 describes the action to be taken by a tCDN for the different combinations of mandatory-to-enforce and "incomprehensible" ("Incomp") properties, when the dCDN either does or does not understand the trigger extension object in question:

MtE	Incomp	Extension object understood by dCDN	Trigger action
False	False	True	Can execute.
False	True	True	Can execute but MUST NOT interpret/apply any trigger extension marked as "incomprehensible".
False	False	False	Can execute.
False	True	False	Can execute but MUST NOT interpret/apply any trigger extension marked as "incomprehensible".
True	False	True	Can execute.
True	True	True	MUST NOT execute.
True	False	False	MUST NOT execute.
True	True	False	MUST NOT execute.

Table 2: Action to be taken by a dCDN for the different combinations of MtE and Incomp properties

3.2.5.2. GenericExtensionObject

A GenericTriggerExtension object is a wrapper for managing individual CDNI Trigger extensions in an opaque manner.

Property: generic-trigger-extension-type

Description: Case-insensitive CDNI Trigger extension object type.

Type: String containing the CDNI Payload Type [RFC7736] of the object contained in the generic-trigger-extension-value property (see table in Section 6.1).

Mandatory-to-Specify: Yes.

Property: generic-trigger-extension-value

Description: CDNI Trigger extension object.

Type: Format/Type is defined by the value of the generic-trigger-extension-type property above.

Mandatory-to-Specify: Yes.

Property: mandatory-to-enforce

Description: Flag identifying whether or not the enforcement of this trigger extension is mandatory.

Type: Boolean

Mandatory-to-Specify: No. Default is to treat the trigger extension as mandatory-to-enforce (i.e., a value of True).

Property: safe-to-redistribute

Description: Flag identifying whether or not this trigger extension can be safely redistributed without modification.

Type: Boolean

Mandatory-to-Specify: No. Default is to allow transparent redistribution (i.e., a value of True).

Property: incomprehensible

Description: Flag identifying whether or not any CDN in the chain of delegation has failed to understand and/or failed to properly transform this trigger extension object. Note: This flag only applies to trigger extension objects whose safe-to-redistribute property has a value of False.

Type: Boolean

Mandatory-to-Specify: No. Default is comprehensible (i.e., a value of False).

Example of a GenericTriggerExtension containing a specific trigger extension object:


```

{
  "generic-trigger-extension-type":
    <Type of this trigger extension object>,
  "generic-trigger-extension-value":
    {
      <properties of this trigger extension object>
    },
  "mandatory-to-enforce": true,
  "safe-to-redistribute": true,
  "incomprehensible": false
}

```

3.2.6. Error Description Version 2

Version 2 of the Error Description adds the "cdn" property on top of the existing properties of the trigger Error Description as defined in [section 5.2.6 of \[RFC8007\]](#). The "cdn" property identifies the CDN in which the error have occurred.

Property: cdn

Description: The CDN PID of the CDN where the error occurred.

Type: A non-empty JSON string, where the string is a CDN PID as defined in [section 4.6 of \[RFC8007\]](#).

Mandatory: Yes.

Example of an errors.v2 with a an error of unsupported location policy extension object:

```

{
  "errors.v2": [
    {
      "content.urls": [
        "https://newsite.example.com/index.html"
      ],
      "description": "unrecoginzed extension type CIT.LocationPolicy",
      "error": "eunsupported",
      "cdn": "AS64496:1"
    },
  ]
}

```


3.2.7. Error codes

This document adds the error code "eextension" to the error codes table defined in [section 5.2.6 of \[RFC8007\]](#). This error code designates that an error occurred while parsing a generic trigger extension, or that the specific extension is not supported by the CDN. A CDN that fails to parse or execute a generic extension object MUST report it using the "errors.v2" array within the trigger status resource, while setting the error code to "eextension" and providing an appropriate description. The "eextension" error code is a registered type of "CDNI CI/T Trigger Error Codes" (see [Section 6.2](#)).

4. Trigger Extension Objects

The objects defined below are intended to be used in the GenericTriggerExtension object's generic-trigger-extension-value field as defined in [section Section 3.2.5.2](#), and their generic-trigger-extension-type property MUST be set to the appropriate CDNI Payload Type as defined in [Section 6.1](#).

4.1. LocationPolicy extension

A content operation may be relevant for a specific geographical region, or need to be excluded from a specific region. In this case, the trigger should be applied only to parts of the network that are either "included" or "not excluded" by the location policy. Note that the restrictions here are on the cache location rather than the client location.

The LocationPolicy object defines which CDN or cache locations for which the trigger command is relevant.

Example use cases:

- o Pre-position: Certain contracts allow for pre-positioning or availability of contract in all regions except for certain excluded regions in the world, including caches. For example, some content cannot ever knowingly touch servers in a specific country, including cached content. Therefore, these regions MUST be excluded from a pre-positioning operation.
- o Purge: In certain cases, content may have been located on servers in regions where the content must not reside. In such cases a purge operation to remove content specifically from that region, is required.

Object specification

Property: locations

Description: An Access List that allows or denies (blocks) the trigger execution per cache location.

Type: Array of LocationRule objects (see [Section 4.2.2.1 of \[RFC8006\]](#))

Mandatory-to-Specify: Yes.

If a location policy object is not listed within the trigger command, the default behavior is to execute the trigger in all available caches and locations of the dCDN.

The trigger command is allowed, or denied, for a specific cache location according to the action of the first location whose footprint matches against that cache's location. If two or more footprints overlap, the first footprint that matches against the cache's location determines the action a CDN MUST take. If the "locations" property is an empty list or if none of the listed footprints match the location of a specific cache location, then the result is equivalent to a "deny" action.

The following is an example of pre-position trigger specification with a trigger-extensions array including a location policy that allows the trigger execution in the US but blocks its execution in Canada:


```

{
  "trigger": {
    "type": "preposition",
    "content.urls": [
      "https://www.example.com/a/b/c/1",
      "https://www.example.com/a/b/c/2"
    ],
    "extensions": [
      {
        "generic-trigger-extension-type": "CIT.LocationPolicy",
        "generic-trigger-extension-value":
          {
            "locations": [
              {
                "action": "allow",
                "footprints": [
                  {
                    "footprint-type": "countrycode",
                    "footprint-value": ["us"]
                  }
                ]
              },
              {
                "action": "deny",
                "footprints": [
                  {
                    "footprint-type": "countrycode",
                    "footprint-value": ["ca"]
                  }
                ]
              }
            ]
          },
        "mandatory-to-enforce": true,
        "safe-to-redistribute": true,
        "incomprehensible": false
      }
    ]
  },
  "cdn-path": [ "AS64496:1" ]
}

```

[4.2.](#) TimePolicy Extension

A uCDN may wish to perform content management operations on the dCDN in a specific schedule. The TimePolicy extensions allows the uCDN to instruct the dCDN to execute the trigger command in a desired time window.

Example use cases

- * Pre-position: A content provider wishes to pre-populate a new episode at off-peak time so that it would be ready on caches (for example home caches) at prime time when the episode is released for viewing. A scheduled operation enables the uCDN to direct the dCDN in what time frame to execute the trigger. The time values are in UNIX epoch.
- * Regional schedule: When used in combination with the Location Policy defined in [Section 4.1](#), the uCDN can trigger separate commands for different geographical regions, for each region using a different schedule. This allows the uCDN to control the execution time per region and, for example, direct the dCDN to execute at off-peak hours, as they are defined per region.

Object specification

Property: window

Description: A time frame in which the trigger should be executed.

Type: TimeWindow object (see [Section 4.2.3.2 of \[RFC8006\]](#))

Mandatory-to-Specify: Yes.

If a time policy object is not listed within the trigger command, the default behavior is to execute the trigger in a time frame most suitable to the dCDN taking under consideration other constrains and / or obligations.

Example of trigger specification with a scheduled time window between 09:00 01/01/2000 UTC and 17:00 01/01/2000 UTC:


```

POST /triggers HTTP/1.1
User-Agent: example-user-agent/0.1
Host: dcdn.example.com
Accept: */*
Content-Type: application/cdni; ptype=ci-trigger-command
Content-Length: 352

{
  "trigger": {
    "type": "preposition",
    "content.urls": [
      "https://www.example.com/a/b/c/1",
      "https://www.example.com/a/b/c/2"
    ],
    "extensions": [
      {
        "generic-trigger-extension-type": "CIT.TimePolicy",
        "generic-trigger-extension-value":
          {
            "window": {
              "start": 946717200,
              "end": 946746000
            }
          }
        "mandatory-to-enforce": true,
        "safe-to-redistribute": true,
        "incomprehensible": false
      }
    ],
  },
  "cdn-path": [ "AS64496:1" ]
}

```

5. Footprint and Capabilities

This section covers the FCI objects required for advertisement of the extensions and properties introduced in this document.

5.1. CI/T Versions Capability Object

The CI/T versions capability object is used to indicate support for one or more CI/T objects versions. Note that the default version as originally defined in [[RFC8007](#)] MUST be implicitly supported regardless of the versions listed in this capability object.

Property: versions

Description: A list of version numbers.

Type: An array of JSON strings

Mandatory-to-Specify: No. The default is version 1. A missing or an empty versions list means that only version 1 of the interface and objects is supported.

5.1.1. CI/T Versions Capability Object Serialization

The following shows an example of CI/T Versions Capability object serialization for a dCDN that supports versions 2 and 2.1 of the CI/T interface.

```
{
  "capabilities": [
    {
      "capability-type": "FCI.TriggerVersion",
      "capability-value": {
        "versions": [ "1", "2", "2.1" ]
      },
      "footprints": [
        <Footprint objects>
      ]
    }
  ]
}
```

5.2. CI/T Playlist Protocol Capability Object

The CI/T Playlist Protocol capability object is used to indicate support for one or more MediaProtocols listed in [Section 6.3](#) by the playlists property of the "trigger.v2" object.

Property: media-protocols

Description: A list of media protocols.

Type: A list of MediaProtocol (from the CDNI Triggers media protocol types [Section 6.3](#))

Mandatory-to-Specify: No. The default, in case of a missing or an empty list, is none supported.

5.2.1. CI/T Playlist Protocol Capability Object Serialization

The following shows an example of CI/T Playlist Protocol Capability object serialization for a dCDN that supports "hls" and "dash".


```

{
  "capabilities": [
    {
      "capability-type": "FCI.TriggerPlaylistProtocol",
      "capability-value": {
        "media-protocols": ["hls", "dash"]
      },
      "footprints": [
        <Footprint objects>
      ]
    }
  ]
}

```

5.3. CI/T Trigger Extension Capability Object

The CI/T Generic Extension capability object is used to indicate support for one or more GenericExtensionObject types.

Property: trigger-extension

Description: A list of supported CDNI CI/T GenericExtensionObject types.

Type: List of strings corresponding to entries from the "CDNI Payload Types" registry [[RFC7736](#)] that are under the CIT namespace, and that correspond to CDNI CI/T GenericExtensionObject objects.

Mandatory-to-Specify: No. The default, in case of a missing or an empty list, MUST be interpreted as "no GenericExtensionObject types are supported". A non-empty list MUST be interpreted as containing "the only GenericExtensionObject types that are supported".

5.3.1. CI/T Trigger Extension Capability Object Serialization

The following shows an example of CI/T Trigger Extension Capability object serialization for a dCDN that supports the "CIT.LocationPolicy" and the "CIT.TimePolicy" objects.


```
{
  "capabilities": [
    {
      "capability-type": "FCI.TriggerGenericExtension",
      "capability-value": {
        "trigger-extension": ["CIT.LocationPolicy", "CIT.TimePolicy"]
      },
      "footprints": [
        <Footprint objects>
      ]
    }
  ]
}
```

6. IANA Considerations

6.1. CDNI Payload Types

This document requests the registration of the following CDNI Payload Types under the IANA CDNI Payload Type registry defined in [[RFC7736](#)]:

Payload Type	Specification
ci-trigger-command.v2	RFcthis
ci-trigger-status.v2	RFcthis
CIT.LocationPolicy	RFcthis
CIT.TimePolicy	RFcthis
FCI.TriggerVersion	RFcthis
FCI.TriggerPlaylistProtocol	RFcthis
FCI.TriggerGenericExtension	RFcthis

[RFC Editor: Please replace RFcthis with the published RFC number for this document.]

6.1.1. CDNI ci-trigger-command.v2 Payload Type

Purpose: The purpose of this payload type is to distinguish version 2 of the CI/T command (and any associated capability advertisement)

Interface: CI/T

Encoding: see [Section 3.1](#)

6.1.2. CDNI ci-trigger-status.v2 Payload Type

Purpose: The purpose of this payload type is to distinguish version 2 of the CI/T status resource response (and any associated capability advertisement)

Interface: CI/T

Encoding: see [Section 3.1](#)

6.1.3. CDNI CI/T LocationPolicy Trigger Extension Type

Purpose: The purpose of this Trigger Extension type is to distinguish LocationPolicy CIT Trigger Extension objects.

Interface: CI/T

Encoding: see [Section 4.1](#)

6.1.4. CDNI CI/T TimePolicy Trigger Extension Type

Purpose: The purpose of this Trigger Extension type is to distinguish TimePolicy CI/T Trigger Extension objects.

Interface: CI/T

Encoding: see [Section 4.2](#)

6.1.5. CDNI FCI CI/T Versions Payload Type

Purpose: The purpose of this payload type is to distinguish FCI advertisement objects for CI/T Triggers Versions objects

Interface: FCI

Encoding: see [Section 5.1.1](#)

6.1.6. CDNI FCI CI/T Playlist Protocol Payload Type

Purpose: The purpose of this payload type is to distinguish FCI advertisement objects for CI/T Playlist Protocol objects

Interface: FCI

Encoding: see [Section 5.2.1](#)

6.1.7. CDNI FCI CI/T Extension Objects Payload Type

Purpose: The purpose of this payload type is to distinguish FCI advertisement objects for CI/T Extension objects

Interface: FCI

Encoding: see [Section 5.3.1](#)

6.2. CDNI CI/T Trigger Error Codes types

The IANA is requested to update the "CDNI CI/T Error Codes" subregistry (defined in [section 7.3 of \[RFC8007\]](#) and located at <https://www.iana.org/assignments/cdni-parameters>) with the following registration:

Error Code	Description	Specification
eextension	The dCDN failed to parse a generic extension object, or does not support this extension.	Section Section 3.2.7 of this document.

6.3. CDNI Media protocol types

The IANA is requested to create a new "CDNI MediaProtocol Types" subregistry in the "Content Delivery Networks Interconnection (CDNI) Parameters" registry. The "CDNI Media Protocol Types" namespace defines the valid Media Protocol object values in [Section 3.2.4](#), used by the Playlist object. Additions to the MediaProtocol namespace conform to the "Specification Required" policy as defined in [section 4.6 of \[RFC8126\]](#), where the specification defines the MediaProtocol Type and the protocol to which it is associated. The designated expert will verify that new protocol definitions do not duplicate existing protocol definitions and prevent gratuitous additions to the namespace.

The following table defines the initial MediaProtocol values corresponding to the HLS, MSS, and DASH protocols:

MediaProtocol Type	Description	Specification	Protocol Specification
hls	HTTP Live Streaming	RFCthis	RFC 8216 [RFC8216]
mss	Microsoft Smooth Streaming	RFCthis	MSS [MSS]
dash	Dynamic Adaptive Streaming over HTTP (MPEG-DASH)	RFCthis	MPEG-DASH [MPEG-DASH]

[RFC Editor: Please replace RFCthis with the published RFC number for this document.]

7. Security Considerations

All security considerations listed in [section 8 of \[RFC8007\]](#) and [section 7 of \[RFC8008\]](#) apply to this document as well.

This document defines the capability to use regular expression within the trigger spec for more granular content selection. The usage of regex introduced the risk of regex complexity attacks, a.k.a ReDos attacks. An attacker may be able to craft a regular expression that can exhaust server resources and may take exponential time in the worst case. An implementation MUST protect itself by at least accept triggers only from an authenticated party over a secured connection. An implementation SHOULD also protect itself by using secure programming techniques and decline trigger commands that use potentially risky regex, such techniques are readily available in secure programming literature and are beyond the scope of this document.

8. Acknowledgments

TBD

9. Contributors

The authors would like to thank all members of the "Streaming Video Alliance" (SVA) Open Caching Working Group for their contribution in support of this document.

10. References

10.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, [RFC 3986](#), DOI 10.17487/RFC3986, January 2005, <<https://www.rfc-editor.org/info/rfc3986>>.
- [RFC8006] Niven-Jenkins, B., Murray, R., Caulfield, M., and K. Ma, "Content Delivery Network Interconnection (CDNI) Metadata", [RFC 8006](#), DOI 10.17487/RFC8006, December 2016, <<https://www.rfc-editor.org/info/rfc8006>>.
- [RFC8007] Murray, R. and B. Niven-Jenkins, "Content Delivery Network Interconnection (CDNI) Control Interface / Triggers", [RFC 8007](#), DOI 10.17487/RFC8007, December 2016, <<https://www.rfc-editor.org/info/rfc8007>>.
- [RFC8008] Seedorf, J., Peterson, J., Previdi, S., van Brandenburg, R., and K. Ma, "Content Delivery Network Interconnection (CDNI) Request Routing: Footprint and Capabilities Semantics", [RFC 8008](#), DOI 10.17487/RFC8008, December 2016, <<https://www.rfc-editor.org/info/rfc8008>>.
- [RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", [BCP 26](#), [RFC 8126](#), DOI 10.17487/RFC8126, June 2017, <<https://www.rfc-editor.org/info/rfc8126>>.
- [RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, [RFC 8259](#), DOI 10.17487/RFC8259, December 2017, <<https://www.rfc-editor.org/info/rfc8259>>.

10.2. Informative References

- [MPEG-DASH] ISO, "Information technology -- Dynamic adaptive streaming over HTTP (DASH) -- Part 1: Media presentation description and segment format", ISO/IEC 23009-1:2014, Edition 2, 05 2014, <<http://www.iso.org/standard/65274.html>>.

- [MSS] Microsoft, "[MS-SSTR]: Smooth Streaming Protocol",
Protocol Revision 8.0, September 2017,
<<https://msdn.microsoft.com/en-us/library/ff469518.aspx>>.

- [PCRE841] Hazel, P., "Perl Compatible Regular Expressions",
Version 8.41, July 2017, <<http://www.pcre.org/>>.

- [RFC6707] Niven-Jenkins, B., Le Faucheur, F., and N. Bitar, "Content
Distribution Network Interconnection (CDNI) Problem
Statement", [RFC 6707](#), DOI 10.17487/RFC6707, September
2012, <<https://www.rfc-editor.org/info/rfc6707>>.

- [RFC7736] Ma, K., "Content Delivery Network Interconnection (CDNI)
Media Type Registration", [RFC 7736](#), DOI 10.17487/RFC7736,
December 2015, <<https://www.rfc-editor.org/info/rfc7736>>.

- [RFC8216] Pantos, R., Ed. and W. May, "HTTP Live Streaming",
[RFC 8216](#), DOI 10.17487/RFC8216, August 2017,
<<https://www.rfc-editor.org/info/rfc8216>>.

Authors' Addresses

Ori Finkelman
Qwilt
6, Ha'harash
Hod HaSharon 4524079
Israel

Phone: +972-72-2221647
Email: orif@qwilt.com

Sanjay Mishra
Verizon
13100 Columbia Pike
Silver Spring, MD 20904
USA

Email: sanjay.mishra@verizon.com

