

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 26, 2012

E. Stephan
G. Bertrand
F. Fieau
R. Pages
France Telecom Orange
October 24, 2011

metadata for CDNs Interconnection
draft-stephan-cdni-usecases-metadata-00

Abstract

There are use cases where the delivery of contents by a CDN on the behalf of another requires the exchange of extra information between these CDNs. This memo proposes a RESTful framework for the exchange of content distribution metadata between an upstream and a downstream CDN. Then it applies this framework to the use cases selected by the CDNi WG [[I-D.ietf-cdni-use-cases](#)] to identify relevant operations, objects and information elements of the CDNi metadata interface and to verify that the RESTful approach suits for these operations.

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 <http://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 26, 2012.

Copyright Notice

Copyright (c) 2011 IETF Trust and the persons identified as the

document authors. All rights reserved.

This document is subject to [BCP 78](http://trustee.ietf.org/license-info) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://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.

This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.

Table of Contents

1.	Introduction	4
1.1.	Terminology	4
2.	Content Distribution Metadata Framework	6
2.1.	Introduction to Content Distribution Metadata	6
2.1.1.	HTTP Adaptive Streaming	6
2.1.2.	IPTV CoD content distribution metadata	7
2.1.3.	SNIA CDMI	7
2.2.	Framework for exchanging CDmD	7
2.2.1.	RESTful design	7
2.2.2.	Push of CDmD	8
2.2.3.	Datamodeling Language	8
2.2.4.	On-the-fly vs Batch CDmD Exchange	9
2.2.5.	Objects Extension	10
2.2.6.	Operations	10
2.2.7.	Main Objects	11
2.3.	Bootstrapping of the CDmD interface	15
2.4.	Comparison of CDNi CDmD and SNIA/CDMI	16
3.	Metadata exchanged for CDNi use cases	16
3.1.	Example of Initialization of the CDmD exchange	17
3.2.	Footprint Extension Use Cases	18
3.2.1.	Geographic Extension	18
3.2.2.	Inter-Affiliates Interconnection	20
3.2.3.	On-Net Delivery	21
3.2.4.	Nomadic Users	21
3.3.	Offload Use Cases	24
3.3.1.	Overload Handling and Dimensioning	24
3.3.2.	Resiliency	26
3.4.	CDN Capability Use Cases	30
3.4.1.	Device and Network Technology Extension	30
3.4.2.	Technology and Vendor Interoperability	33
3.4.3.	Dynamic QoE and QoS improvement	35
4.	Discussions	37
4.1.	JSON reference	37
4.2.	Network and infrastructure Metadata	37
5.	Inputs for the next version	37
5.1.	Requirement	38
6.	IANA Considerations	38
7.	Security Considerations	38
8.	Acknowledgements	38
9.	References	38
9.1.	Normative References	38
9.2.	Informative References	39
Appendix A.	GPX XML Schema fields extensibility	40
	Authors' Addresses	41

1. Introduction

There are use cases where the delivery of contents by a CDN on the behalf of another requires the exchange of extra information between these CDNs. This memo proposes a RESTful framework for the exchange of content distribution metadata between an upstream and a downstream CDN. Then it applies this framework to the use cases selected by the CDNi WG [[I-D.ietf-cdni-use-cases](#)] to identify relevant operations, objects and information elements of the CDNi metadata interface and to verify that the RESTful approach suits for these operations.

This document does not study all the metadata to be exchanged on the CDNi interface. It focuses on the specification of the exchange of content distribution metadata between an upstream CDN and a downstream CDN when the upstream CDN decides to distribute contents through this downstream CDN.

This draft compliments the works done by Ben in [[I-D.jenkins-cdni-metadata](#)] and Kevin in [[I-D.ma-cdni-metadata](#)].

1.1. Terminology

The document reuses terminology defined by CDNi WG documents [[I-D.ietf-cdni-problem-statement](#)] and [[I-D.ietf-cdni-requirements](#)].

The call flows of this document uses the following abbreviations for the interfaces (Control, Metadata, Request Routing, Logging) of the CDNi solution:

- Metadata interface: Mi
- Request Routing interface: R Ri
- Content Acquisition interface: CAi
- Control interface: CTi

Content distribution metadata:

Information communicated by the upstream CDN to the downstream CDN that must be enforced by the downstream CDN to distribute contents on the behalf of the upstream CDN. The document uses the abbreviations 'CDmD'.

Push or Pull of metadata:

The upstream CDN associates the delegation of the distribution of contents with metadata to provide the downstream CDN with the

information needed to distribute the content according to the rules determined by the upstream CDN.

- A metadata is pushed on the Cdni Mi when the operation is initiated by the upstream CDN.
- A metadata is pulled on the Cdni Mi when the operation is initiated by the downstream CDN.

Time-to-service:

The delay between the request of a service and the delivery of the service (e.g. the delay between the selection of a VoD by an eyeball and the display of the first picture of this VoD).

information element (IE):

Piece of information of the information model.

Objects:

IE which can be individually operated through the metadata interface: created, read, modified and deleted.

Main objects:

IE which are addressable and can be individually created or deleted.

Ancillary objects:

Addressable IE which can be downloaded or modified individually.

Scalar:

IE that can not be individually addressed.

Sparse and dense mode:

The intensity of usage of a CDN interconnection may differ dramatically:

- A CDN interconnection is said 'sparse' when dCDN rarely distribute contents on the behalf of uCDN.
- A CDN interconnection is said 'dense' when dCDN intensively distribute contents on the behalf of uCDN.

2. Content Distribution Metadata Framework

This section presents existing metadata related to the distribution of contents then it defines the framework for exchanging metadata on the CDNi Mi interface.

2.1. Introduction to Content Distribution Metadata

Metadata is defined literally as 'data about data'. Practically it represents structured data about resources that help operating these resources. They may be stored in a file which captures the characteristics of a resource or dynamically generated like the sitemap of a WEB site. Metadata are designed according to the operational requirements of a domain. MPEG21, MPEG 7, TVAnyTime, DublinCore and Atom are notorious examples for the content domain.

Following are metadata directly tied to the content distribution domain.

2.1.1. HTTP Adaptive Streaming

HTTP adaptive streaming services distribute content in segments. The segments provide boundaries for performing rate adaptation. The segments are described as URL in manifest files.

There are several HTTP adaptive streaming:

- HTTP Live Streaming (HLS) [[I-D.pantos-http-live-streaming](#)] uses a hierarchy of m3u8 playlists pointing to individual segment files or other m3u8 playlists;
- Smooth Streaming [IIS-Smooth-Streaming] uses a set of XML files to describe the media source;
- HTTP Dynamic Streaming [Flash-Media-Manifest] uses the flash media manifest file XML format;
- Dynamic Adaptive Streaming over HTTP [MPEG-DASH] [[ETSI-3GP-DASH-R10](#)] also supports an XML manifest format;

These manifest files contain metadata about the organization of the content being delivered. This may include the location of the content as well as the order in which it is likely to be retrieved.

CDNs uses the CDNi metadata interface to exchange information on HAS streams to prepare their delivery. This encompasses the exchange of information about the delivery and the update of the manifest file.

2.1.2. IPTV CoD content distribution metadata

IPTV services distributes the technical information of the distribution of a content in metadata. They provide the terminal with the technical information for accessing to the content.

[DVB-IP-TV] specification includes the specification of the metadata for scheduling and for the controlling the regionalization of the distribution of a content.

Part 'A' of the specification of TV-Anytime metadata [[TVA-CD-mD](#)] specifies XML objects describing the parameters of a content distribution. 'ProgramLocationType' describes the content and where it must be distributed. 'ScheduleEventType' describes when the content must be distributed. These metadata are used in OIPF specifications. Before receiving a content an Open IPTV Terminal (OITF) receives technical information from the OIPF IPTV platform. The information is sent at the format of TV-Anytime XSD.

[I-D.thompson-cdni-atis-scenarios] presents the CoD service specified by ATIS [[ATIS-0800042](#)]. The metadata (namely Media Resource Metadata) are specified in a XML schema.

2.1.3. SNIA CDMI

CDMI [[SNIA-CDMI-1.0](#)] defines the functional interface that applications will use to create, retrieve, update and delete data elements.

2.2. Framework for exchanging CDmD

This section presents the framework.

2.2.1. RESTful design

In a CDNi interconnection dCDN provides the content distribution service to uCDN. Contents and Content Distribution metadata are going from uCDN to dCDN.

The framework is based on a RESTful model:

- dCDN is the server and uCDN is the client;
- The default protocol of CDNi Metadata interface is HTTP [[RFC2616](#)] over a secure transport layer like TLS 1.1 [[RFC4346](#)].
- uCDN uses the HTTP requests PUT, POST, GET, HEAD and DELETE methods to manage all the live cycle of the metadata;

- CDMi metadata interface does not define new methods than those of HTTP;
- The CDMi uploaded by uCDN are not accessible to other CDNs which interconnect with dCDN too.

2.2.2. Push of CDMi

CDMi operations occurs at various states of the distribution of a content:

- CDMi interconnection bootstrapping;
- Set up of a the delivery of a content or a group of content;
- Modification of the CDMi of a content or of a group of content;
- Reception of a request routing from an eyeball (before, during or after its redirection by uCDN to dCDN);
- Purge of a content;

[Section 3](#) of the draft checks that the push mode covers the situations presented in the use cases selected by the WG CDMi. It allows uCDN to synchronize the CDMi operations with the CSP queries.

Discussion:

Pull mode might be used too but it does not cover any situations and comes at the expense of an increase of the time-to-service (see [[I-D.stiemerling-cdni-routing-cons](#)]) and of the burstiness of the signalling on the CDMi interfaces. Furthermore when uCDN has to reflect content distribution demands of its CSP it does not provide uCDN with a simple mechanism to send CDMi to dCDN. Consequently the usage of pull mode requires some push mode too and increases the metadata interface complexity.

2.2.3. Datamodeling Language

Existing content distribution metadata are generally specified in XML schema. As they correspond to information describing one content sent to an eyeball they do not correspond exactly to the one that should be exchanged by 2 CDNs:

- Content acquisition between 2 CDNs differs from content delivery, especially direct delivery. A content may consists in a single file from the acquisition perspective (e.g. zipped) while being delivered to the end-user in small files, e.g. VOD HAS

example;

- Scalability: CD mD are send mostly individually to eyeball terminals. An efficient CDNi metadata interface requires the grouping of the exchange of metadata between the 2 CDNs;

The data modeling is based on XML schema. Several languages are candidate for exchanging information.

- XML: Intensively used; XML come along with W3C XML Schema for specifying interfaces and XPATH to transform and selectively extract data out of XML document ;
- IETF YANG/NETCONF, [RFC 6020](#); human readable, XML and RELAX HG interoperability;
- OASIS Relax Ng;
- JSON: Intensively used; human readable; Nothing equivalent to XSD schema and XPATH;

2.2.4. On-the-fly vs Batch CDmD Exchange

CDmD exchange (same applies for Content acquisition and purge) is said to be Batch when the preparation or the endding of the distribution of a content over the CDN interconnection is not timely correlated with the delivery of the content to the eyeball.

CDmD exchange is said to be on-the-fly when it is triggered by the arrival of a request routing query from an eyeball on the RRi of uCDN or dCDN.

Information elements:

On-the-fly actions are very demanding in terms of processing and signaling. A CDN may not support them or all of them. the capability object reflects the capability of a given CDN to support them:

- CDmD Exchange mode: the mode supported, either batch or on-the-fly;
- Content acquisition mode: the mode supported, either batch or on-the-fly;
- CDmD deletion mode: the mode supported, either batch or on-the-fly;

-CDmD purge mode: the mode supported, either batch or on-the-fly;

Discussion:

The list of flags above is not be exhaustive. Overspecifying the capabilities (i.e. splitting the description of one capability in 2 flags) will not harm the CDNi performance. As an example CDmD Exchange mode can be split in 'CDmD' reception mode' (able to receive on the fly or not) and 'CDmD sending mode' (may send on the fly or not).

uCDN must exchange similar information to the dCDN must inform of its capabilities to.

2.2.5. Objects Extension

CDNs interoperate to distribute content services that may require specific metadata. Consequently each main object is extensible and includes per

design a field 'extension' for this purpose. GPX XML schema uses this approach (See Annex A). The extension field can be used to carry opaque information as requested by [[I-D.ma-cdni-metadata](#)].

Discussion

In GPX schema each field is extensible. Will all the objects of the CDNi Mi datamodel be extensible?

2.2.6. Operations

This section presents Mi operations triggered by uCDN (with effect in the dCDN).

2.2.6.1. Creation of Object

uCDN uses the HTTP method PUT to create metadata in main objects in dCDN.

on success dCDN returns 201 'Created'.

When dCDN detects a format error it returns an HTTP error 400 'Bad Request'

When dCDN does not support the metadata object received it returns the HTTP error 403 'Forbidden'.

When dCDN received a PUT method for the creation of a object that is

not supported it shall return the method 405 'Method Not Allowed'.

When dCDN receives an object that uCDN is not authorize to create it returns the HTTP error 401 'Unauthorized'.

2.2.6.2. Update of an object

uCDN uses the method POST to update an addressable object in dCDN.

on success dCDN returns 200 'OK'.

[edit] add error cases

2.2.6.3. Consultation of an Object

uCDN requests the value of an object using a HTTP GET method.

When the object exists and is addressable dCDN returns the value of object.

when the object does not exist uCDN returns 410 'Gone'

2.2.6.4. Deletion of an Object

uCDN uses the HTTP method DELETE to remove the CD metadata from the dCDN. dCDN removes the object and returns HTTP '201' OK.

The deletion of CDmD metadata is performed by content or by set of contents. As an example, individual deletion happens when a CoD content is removed from a catalogue, and a set of deletion happens when a CoD is stored in fragments corresponding to chapters.

Check of deletion:

uCDN checks the deletion with a HTTP GET asking for the object that have be deleted. dCDN should return 410 'Gone'.

Discussion:

The purge of the content is not in the scope of the MI interface. Nevertheless the deletetion of the CDmD of a content is coupled with the purge of the content.

2.2.7. Main Objects

The metadata interface is designed to operate main objects. It provides operations to fully (create, delete..) manage main objects, to modify objects but it does not support atomic management of scalar

parameters. As an example the WG may decide that the 'start of delivery' of a content metadata can not be modified atomically, i.e. without modifying the content object. It is up to the CDNi WG to determine the level of each information elements. Nevertheless for the sake of interoperability the number of operations has to be limited:

The data model is made of objects connected by the operations needed by the interconnection. A main object is an item that can be created and deleted independently of the others. As an example, geographic areas are complex objects that are managed individually.

The study made in [section 3](#) identifies the following object organisation. It does not specify a datamodel. It gathers the information elements identified in [section 3](#) to ease the reading and the specification of the call flow and of the operations:

Main object 'content' :

- A group of pieces of content.
- Operations:
 - uCDN creates a content;
 - uCDN adds a piece of content in a content;
 - uCDN deletes a piece of content from a content;
 - uCDN updates the parameters
- IEs:
 - content activation;
 - content deactivation;
 - expiration times;
 - cache invalidation and removal intervals;
 - source URL
 - backup source URL;

- auto_purge_delay;
- minimal_storage_duration: duration of the storage of a content;
- immediat_acquisition: flag requesting that dCDN must start the acquisition triggered on reception of the content metadata;
- 'extension'

Main object 'region' :

- standard administrative names of geographic area like country, region,... like defined ISO 3166-2.
- Operations:
 - uCDN gets the name of regions predefined by dCDN.
 - uCDN creates a logical region made of a subset of names predefined by uCDN;
 - uCDN gets, modifies, deletes a logical region.
 - uCDN gets the regions where dCDN supports on net delivery;
 - uCDN gets the network prefixes of a region where dCDN supports on net delivery;
- IEs
 - Names of geographic areas like countries, part of country, district, city...
 - on_net flag: indicates the support or not of on-net delivery : 'full', 'partial' or 'none';
 - on_net_geoIP;
 - Capacity:peak (unit, value), sustainable: (unit, value); duration (start, end);
 - 'extension'

Object 'Geoloc':

- Detailed network information on a region. Lookup information resolving the localisation of a host based on its network address.

CDNs embed geoIP database and are usually able to resolve geoIP localisation without exchanging Geolocalisation information. Nevertheless several use cases require the exchange of the addresses of the eyeball networks that a CDN covers.

- Operations:

- uCDN gets the Geoloc of a region.

- IEs

- IP prefixes

- 'extension'

Main object 'dCDNcaps'

- Operations:

- uCDN gets dCDN capabilities.

- IEs

- URL_format;

- token_format

- regions: details of dCDN footprints. A set of names of geographic area like country, region, ..

- capacity

- interconnection capability,

- CDmD_exchange_mode: 'batch', 'on-the-fly' or 'both';

- CDmD_deletion_mode: 'batch', 'on-the-fly' or 'both';

- CDmD purge mode: 'batch', 'on-the-fly' or 'both';

- Content acquisition mode: 'batch', 'on-the-fly' or 'both';

- 'extension'

- 'extension'

Main object 'uCDNcaps'

- Operations:

- uCDN creates and sets an uCDNcaps object on dCDN Mi interface.

- IEs

- on_dCDN_failure_FQDN: FQDN of the uCDN RRi which handles the redirection on failure of the redirection by dCDN.

- backupCDN: FQDN of a content acquisition backup;

- bursty_interconnection: a flag saying that the interconnection is very bursty;

- 'extension'

2.3. Bootstrapping of the CDmD interface

This section makes the assumption that uCDN and dCDN have an technical agreement which defines the usage of dCDN resources by uCDN and exchanged bootstrapping information like RRi and CAi server addresses of whole CDNi bootstrapping on the control interface or manually.

The initialization of the Metadata interface includes the steps below :

1. dCDN grants uCDN with the right to connect, upload and download CDmD on its metadata interface;
2. dCDN initiates its capabilities objects according to the technical agreement it has with uCDN;
3. uCDN connects to dCDN CDmD server;
4. uCDN gets dCDN capabilities;
5. uCDN setups high level CDmD in dCDN;
6. Setup of objects needed during the duration of the interconnection (e.g. a group of content that may be modified but unlikely deleted);

Discussion:

The boundary between the information needed by the control interface and the Mi interface is not clear. Some IEs may be needed on both (e.g. the RRi server of a region).

2.4. Comparison of CDNi CDmD and SNIA/CDMI

Following is a list of points above which are already specified by the the CDMI interface [[SNIA-CDMI-1.0](#)]:

- Both are client/server and RESTful;
- Both requires metadata extensibility ([section 16.2](#));
- CDMi reuses HTTP 1.1 primitives and error codes to implement the operations;
- Secure Transport;
- both must balance between XML and JSON;
- Same operations:
 - discovery of capabilities;
 - creation
 - deletion;
 - read;

Most of the aspects of the CDNi Mi are included in CDMI. A deeper insight is needed to determine if CDNi Mi can be specified as a subset of CDMI where uCDN uses the Data Storage Interface and dCDN acts in the role of providing a Data Storage Interface.

3. Metadata exchanged for CDNi use cases

[I-D.ietf-cdni-use-cases] presents realistic situations between 2 CDNs where the downstream CDN ingests and delivers contents on the behalf of the upstream CDN. This section studies the exchange of metadata for these situations. Each subsection presents the exchange of content distribution metadata for one use case.

Only one example of call flow is shown per use case. DNS steps are not represented to simplify the call flows. They are discussed in

[[I-D.peterson-cdni-strawman](#)].

3.1. Example of Initialization of the CDmD exchange

This section makes the assumption that uCDN and dCDN have a technical agreement which defines the usage of dCDN resources by uCDN. It gathers CDmD exchanges used by several call flows.

The initialization of the Metadata interface is as follow :

1. dCDN grants uCDN with the right to connect, upload and download CDmD on its metadata interface;
2. dCDN initiates its capabilities objects according to the technical agreement it has with uCDN;
3. uCDN connects to dCDN CDmD server;
4. uCDN gets dCDN capabilities;
5. uCDN setups high level CDmD in dCDN;
6. optionally, uCDN setups objects it is likely to need during the duration of the interconnection (e.g. a group of content that may be modified but unlikely deleted);

Following is an example of call flow for the initialization of the metadata interface.

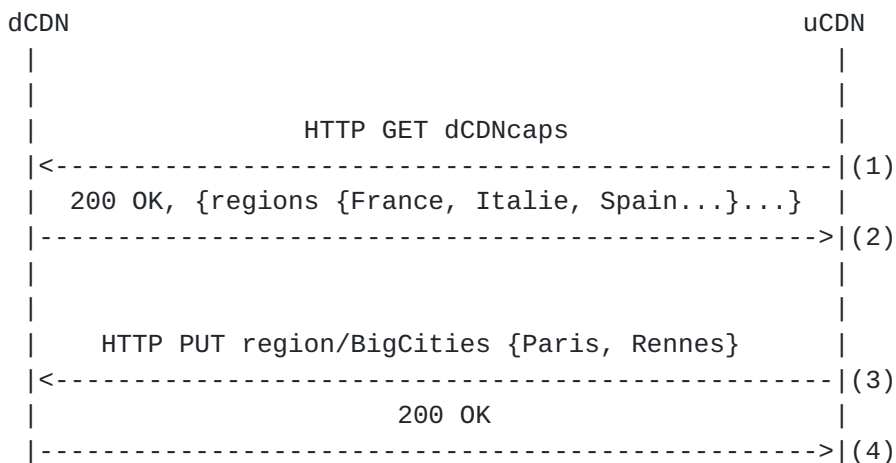


Figure 1, Initialization of the Mi interface

1. uCDN requests the details of the footprint where it is allowed to distribute contents;

2. dCDN returns the list of the countries;
3. uCDN creates a logical region named 'BigCities'. It initialize this region with 2 towns 'Paris' and 'Rennes' which are not geographically close;
4. dCDN accepts the creation;

3.2. Footprint Extension Use Cases

At first approach the CD metadata required to setup a footprint extension are intuitive. The uCDN simply indicates the geographic area to dCDN. Nevertheless there is a need of CDmD for controlling explicitly the delivery because dCDN is not aware of the constraints that apply to the contents.

3.2.1. Geographic Extension

in this use case uCDN interconnects with dCDN to extend its footprint to 2 countries covered by dCDN. Once the footprint extension setup achieved the RRi function of uCDN will be able to redirect the requests coming from the prefixes of these 2 countries to dCDN RRi function.

Assumption:

the initialization of the Mi has be done previously as per the call flow of [section 2.4](#): uCDN got the list of the countries of the footprint. dCDN initialized the main objects Italie and Spain.

Information elements:

Geographic capabilities are high level Geographic information like the name of well known areas, country, region, district, city...

Operations:

- Get the list of regions;
- Create a content;
- Initialize the contents to be distributed in a region;
- Create a region;

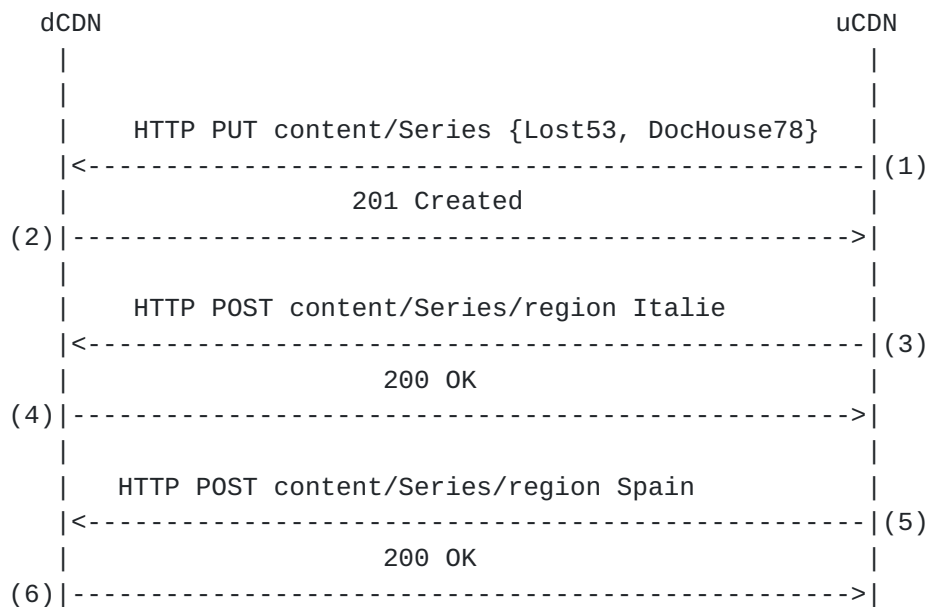


Figure 2, Geographic Extension

The initialization of the Mi is done according to the first call flow presented.

1. uCdn create the content 'Series' and initializes it with the pieces of contents 'Lost53' and 'DocHouse78'
2. dCDN accepts the creation;
3. uCDN adds this content in the region 'Italie' predefined by dCDN.
4. dCDN accepts the adding;
5. uCDN adds this content in the region 'Spain' predefined by dCDN.
6. dCDN accepts the adding;

Discussion:

uCDN might create a logical region 'South' initialized with 'Italy' and 'Spain' before step 1. This avoids steps 3 and 4.

Grouping CDmD may lead to complex processing and signaling when dCDN rejects the delivery of a subset of the contents.

3.2.2. Inter-Affiliates Interconnection

This use case covers the interconnection of CDNs managed by subsidiaries of a large group. For instance, it includes the interconnection of Orange France's and TP's CDNs, which are both part of the Orange group. The trust relationship among the interconnected CDNs is strong in this use case. Therefore, the CDNs can exchange internal dimensioning information like the number of caches, or detailed routing information, CDN detailed capabilities or performance information.

Information elements:

capacity { number of caches, sustainable sessions per second; sustainable delivery throughput...};

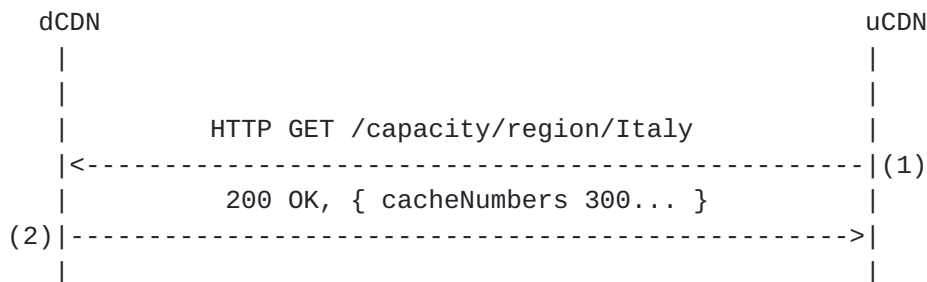


Figure 3, Inter-Affiliates Interconnection

1. uCDN requests the number of caches available on a region.
2. dCDN returns the information

Discussion:

Step1 gets directly the IE 'capacity' . In fact requesting the dCDNcaps at a whole may be enough to implement. This look like a XPATH request. Do we need a flag to present the level of granularity of the GET request ?

Such metadata exchanges enable tied interworking between the 2 CDNs.

At some extend, this kind of information may quickly overlap with monitoring information.

The object capacity must include a field 'extension' to permit enrichment.

3.2.3. On-Net Delivery

In this use case uCDN wants to deliver content to eyeballs directly connected to dCDN networks.

Information elements:

- on-net flag: indicate if a region include has an on-net footprint;
- on-net geoIP: the prefixes of network the dCDN eyeballs;

Operation:

- Get on_net regions;
- get on_net region prefixes;

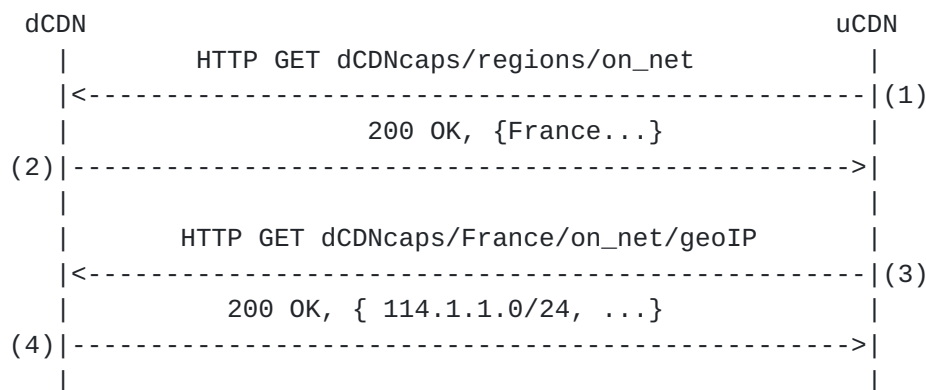


Figure 4, On-Net Delivery

1-2. uCDN downloads the regions to which the dCDN can deliver on-net. This information enables the uCDN to know which areas are served.

3-4. uCDN gets the prefixes. Then it adapts its CDN selection policies to guarantee that the content will always be delivered on net, with the best performance (i.e. uCDN redirects to dCDN only the content requests coming from eyeball located on dCDN networks) .

3.2.4. Nomadic Users

This use case considers that the initialisation of the Mi has been done before as per [section 3.1](#).

Asumption:

For optimization reasons uCDN and dCDN did not provision the distribution of the content by dCDN. Consequently the CDmD of the content and the content are exchanged on-the-fly between the 2 CDNs and the content is purged at the end of the delivery.

information element:

- o synchronous purge: a flag which indicate that the purge must be done at the end of the delivery after a reasonable (from cache algo point of view) delay
- o a timer of a a reasonable delay;
- o on the fly acquisition: flag indicating that dCDN supports or not on the fly content acquisition;

Operations:

- Push of per content metadata in dCDN;



Figure 5, On the fly CDmD exchange for Nomadic use case

1. A end-user requests a content that uCDN is in charge of.
2. The request arrives on the request routing function of uCDN; The request routing logic of uCDN determines that the end-user is located on the footprint of dCDN; dCDN pushes the CDmD of this content toward dCDN. CDmD includes information describing the constraint attached to the nomadic case (limited to one user, ...)

3. dCDN accepts the delivery.
4. dCDN starts the acquisition of the content.
5. UCDN redirects the request to dCDN.
6. CDN stores the (first part) of the content
7. The end-user request is received by dCDN.
8. dCDN starts the distribution of the content to the end-user.
9. The delivery achieved;
10. dCDN deletes of the CDmD
11. dCDN delete the content

Discussion:

In this use case only the user considered in (5) must be served by dCDN. this does not means that dCDN must check the identity of the user in (7) because if any identity verification must be performed it should be checked by uCDN in (5) before the redirection.

Most of the use cases require an Mi operation to explicitly delete the metadata attached to a content;

3.3. Offload Use Cases

This section gives examples of call flow for the offload use cases.

3.3.1. Overload Handling and Dimensioning

The initialization of the Mi has been done in [section 3.1](#).

The CDN interconnection is setup to cover unexpected peak of traffic in uCDN.

Information element:

- Content object;
- Capacity:peak (unit, value), sustainable: (unit, value);
duration (start, end);
- bursty interconnection flag: a flag to clearly distinguish very bursty interconnection from more stable ones.

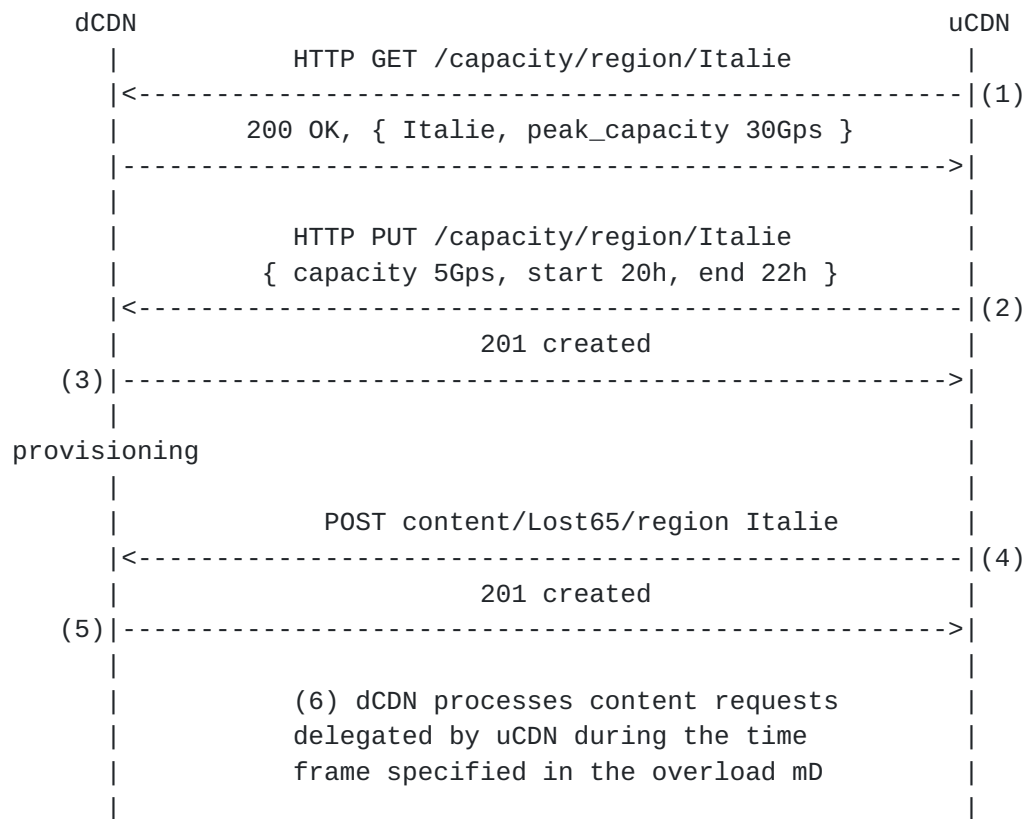


Figure 6, Overload handling, planned traffic peak

1. uCDN downloads dCDN overload traffic handling capabilities (may be downloaded during bootstrapping). This information enables the uCDN to know how much traffic it can delegate to the dCDN.

2. uCDN creates capacity request mD. This enables the dCDN to know how much traffic and the period of traffic the uCDN will delegate to the dCDN and (e.g., for planned traffic peaks, or planned offload for maintenance operations).

3. dCDN acknowledges that it understands and agrees to the overload mD. Then it provisions the resources.

4. uCDN creates geographic content delivery restrictions CDmD pushing the CSP related policies to dCDN. This enables the dCDN to know to which areas it must or must not deliver every piece of content.

5. dCDN acknowledges that it understands and accepts the CDmD.

Notes: if the dCDN cannot or does not want to fulfill the capacity request, it will response with an HTTP error message such as a 403 forbidden. This use case covers the failure of delivery resources. It assumes that the uCDN is still able to redirect requests to the

dCDN, but not to serve all the requests by itself.

Discussion:

This use case highlights the information elements of a regular CDNi interconnection (even if peak and sustainable value are extremely different in an Offload situation).

3.3.2. Resiliency

3.3.2.1. Failure of Content Delivery Resources

uCDN interconnects with dCDN1 and dCDN2 to guarantee service continuity when dCDN1 can not handle the delivery.

The call flow below presents the situation where dCDN1 encounters an overload problem or a failure before beginning the delivery and cannot serve the content to the UE.

Assumptions:

dCDN1 and dCDN2 have similar footprint. uCDN already pushed the CDmD of the content named Lost65 both on dCDN1 and dCDN2. In normal situation dCDN1 distributes the content 'Lost65' and already made the acquisition of the content Lost65.

Information elements:

on_dCDN_failure_FQDN: FQDN of the uCDN RRI which handles the redirection on failure of the redirection function of dCDN.

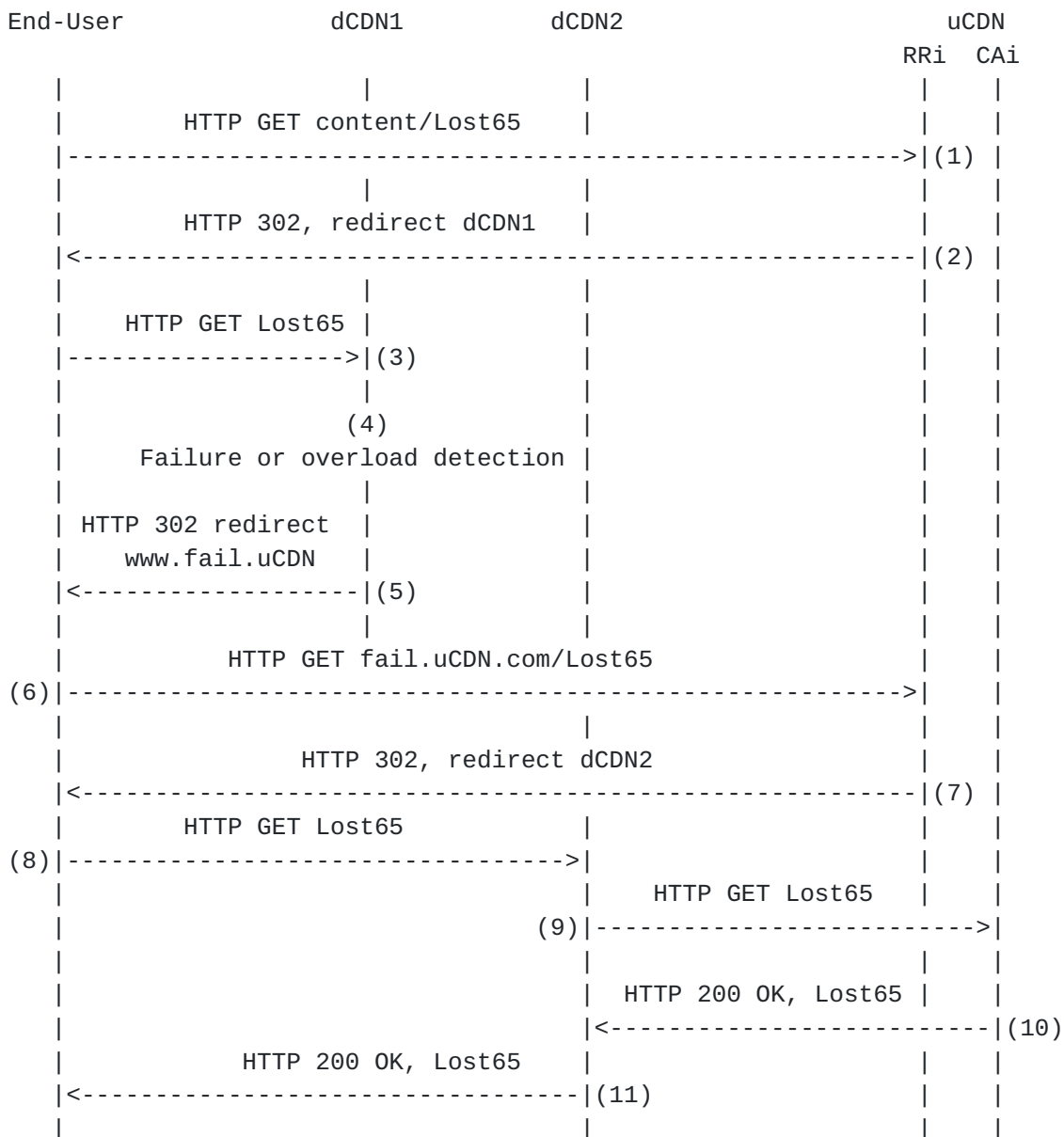


Figure 7, Failure of Content Delivery Resources

1. A end-user requests a content that uCDN is in charge of;
2. uCDN redirects the request to dCDN1;
3. The end-user request is received by dCDN1;
4. dCDN detects a failure or a lack of ressource on its delivery;
5. dCDN1 redirects the EU request to a dedicaced FQDN RRi of uCDN which handles the failure of delivery;
6. The end-user request is received by uCDN on this special FQDN

(Editor notes: solution already presented in another draft: include the reference);

7. uCDN redirects the request to dCDN2;
8. The end-user request is received by dCDN2;
9. dCDN2 starts the acquisition of the content;
10. dCDN2 stores the content;
11. dCDN2 sends the content to the EU;

Discussion:

Direct redirection between dCDN1 and dCDN2 may reduce the redirection duration. it requires the exchange of more CDmD and hides the failure of the delivery to uCDN.

In this use case the detection of the failure happens before the selection of the delivery node. In case of failure during the delivery, dCDN should send a message to uCDN on the control interface.

3.3.2.2. Failure of Content Acquisition

Assumption:

uCDN interconnects with dCDN1 and dCDN2 for the delivery of the content 'Lost65'.

dCDN2 already made its acquisition and keep a copy.

uCDN setups dCDN2 as a backup solution for the acquisition of 'Lost65'.

Information element :

- backupCDN: FQDN of a content acquisition backup says that the upstream CDN provides (or not) a backup for the acquisition of the content it is requesting the distribution.
- storage_duration: duration of the storage of a content. Flag of the content object to request that the dCDN keep a copy of the content during a period of time after acquisition (or after the last delivery);

- `immediat_acquisition`: flag requesting that dCDN must start the acquisition triggered on reception of the content metadata;

operations:

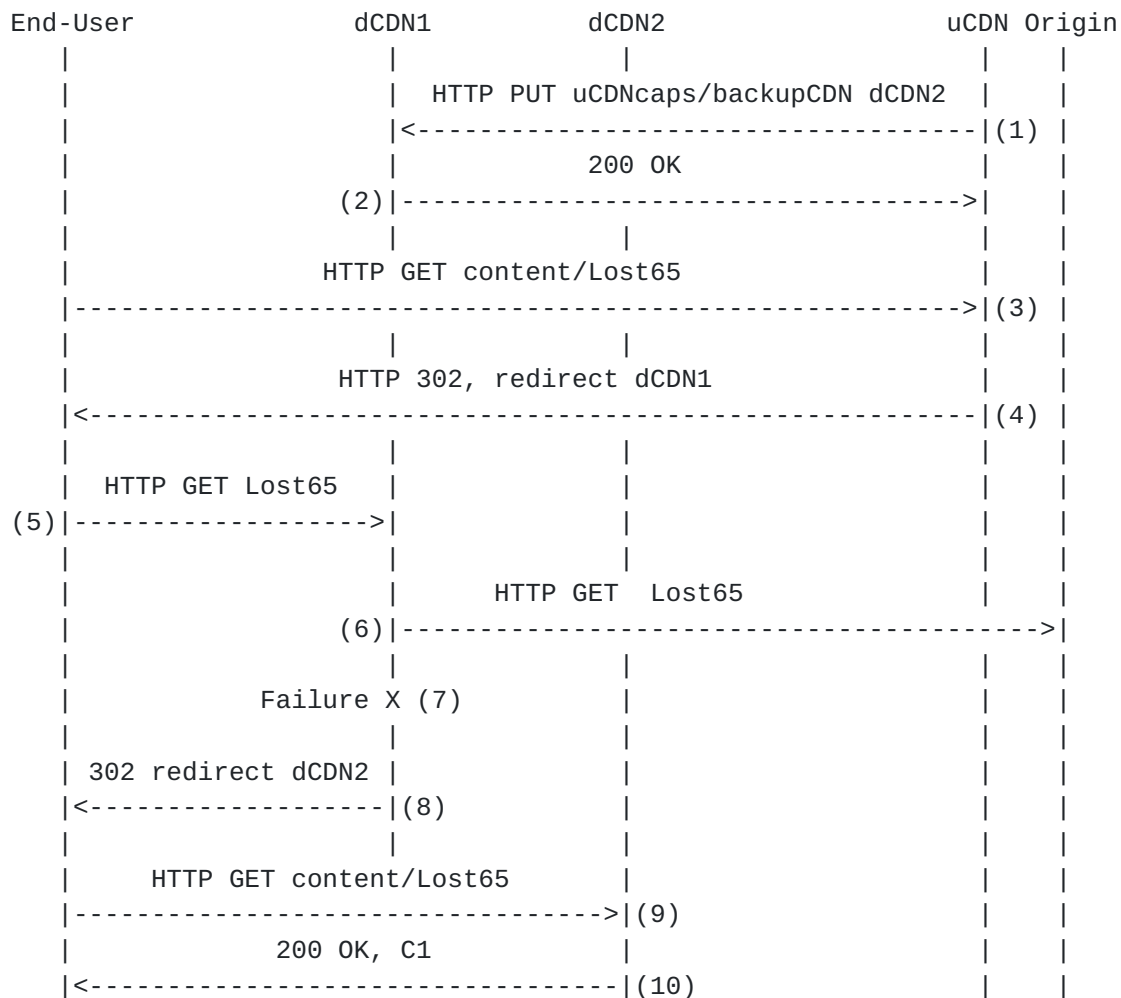


Figure 8, Failure of Content Acquisition

1. uCDN setups the backup CDN.
2. dCDN1 accepts.
3. End-user requests the content Lost65.
4. uCDN redirects the request to dCDN1.
5. The end-user request is received by dCDN1
6. dCDN1 starts the acquisition of the content.

7. The acquisition fails
8. dCDN1 redirects the UE to dCDN2 because uCDN provided a backup.
9. The end-user request is received by dCDN2.
10. dCDN2 deliver the content to the UE.

Discussion:

The failure of content acquisition may be solved without redirection, simply with the selection of a backup acquisition server.

3.4. CDN Capability Use Cases

3.4.1. Device and Network Technology Extension

Assumptions:

uCDN only streams content using HTTP smooth streaming protocol. dCDN supports other protocols like MPEG-DASH. An End-user requests a MPEG-DASH content that is managed by uCDN. The uCDN and the dCDN have exchanged their capabilities prior the UE content request. uCDN has pushed content related metadata before the EU request.

Information elements:

- delivery methods supported (HTTP smooth streaming, MPEG-DASH,...);
- networks type supported (fiber, xDSL, WiFi, 3G, LTE,...);

The figure below presents the typical CD mD call flow:

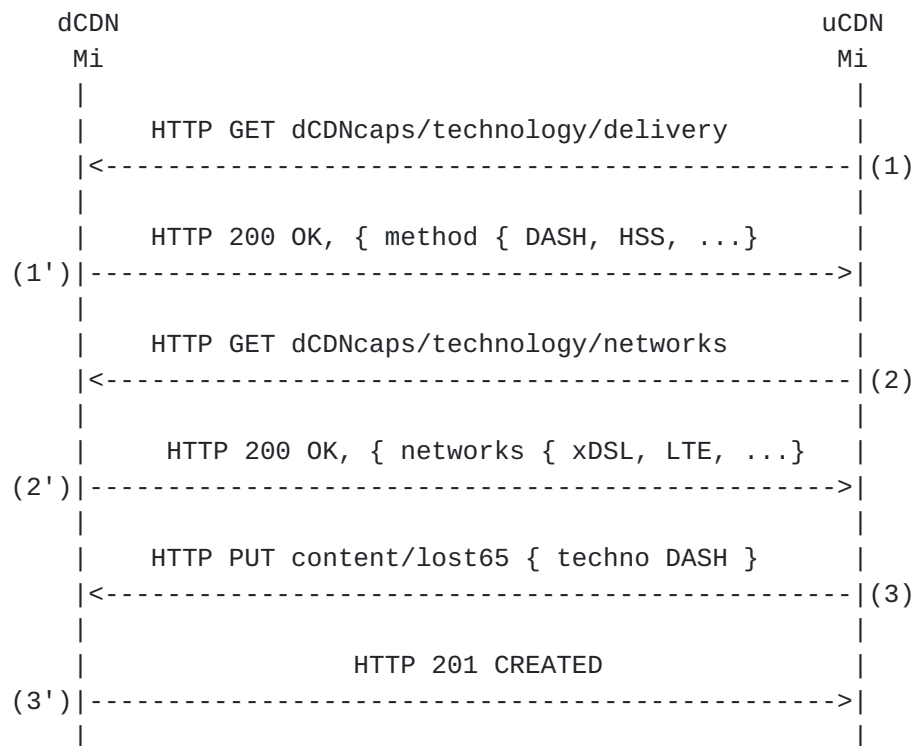


Figure 9, Device and Network Technology Extension

1. uCDN downloads dCDN delivery technology capabilities.
2. uCDN downloads dCDN network type capabilities.
3. uCDN pushes the CDmD of this content toward dCDN. CDmD includes information describing the constraint attached to the CDN capability use case.

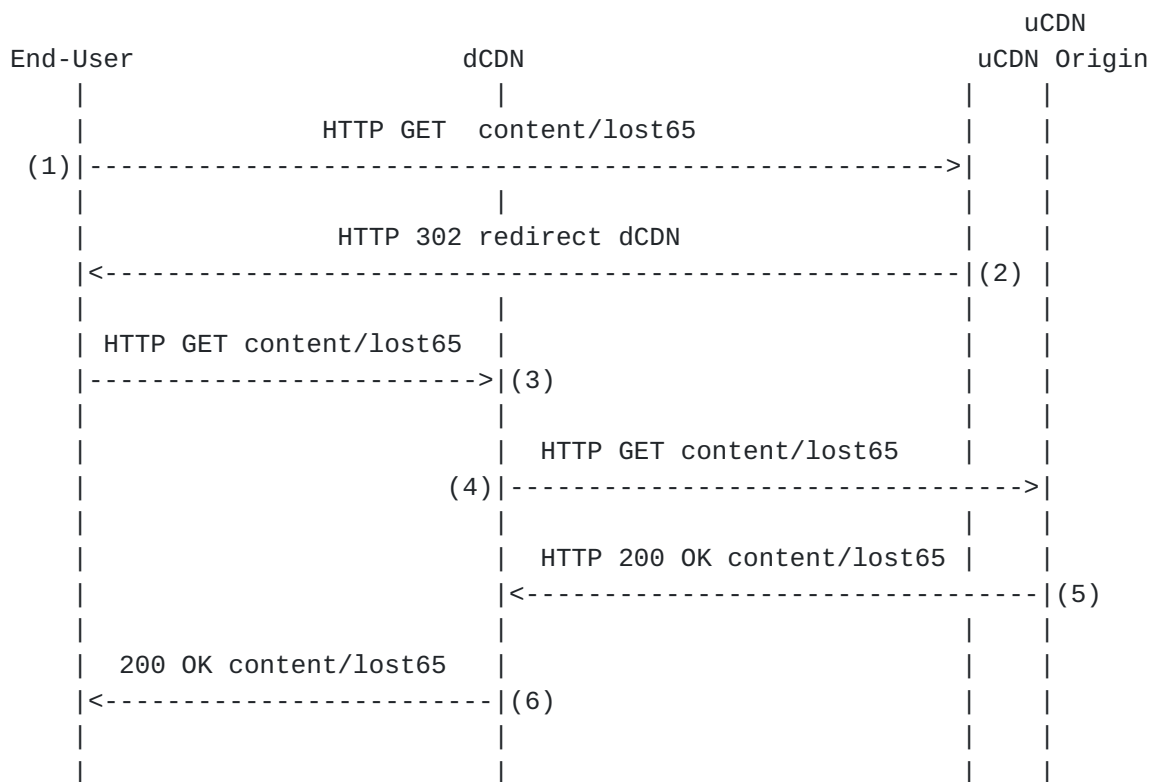


Figure 10, Device and Network Technology Extension

call flow:

1. A end-user requests a content that uCDN is in charge of.
2. uCDN redirects the request to dCDN.
3. The User Equipment request is received by dCDN.
4. dCDN starts the acquisition of the content.
5. CDN stores the content
6. The delivery achieved

Discussion :

Synchronous vs asynchronous : The exchange of the capabilities between the CDN may be done before receiving a content request or when receiving a content request. This case adds delay to the handling of the content request. The content related metadata may be provisioned before the request.

3.4.2. Technology and Vendor Interoperability

In a CDN interconnection, CDN may need to exchange their configuration : CDN parameters, functions configuration. Such information can be provided at service bootstrapping, but can also be provided on fly using the metadata interface.

information elements:

Description of a CDN:

origin server

origin servers list (server name, IP)

ingestion interface: IP address of the ingestion interface of origin server

routing server

list: routing server name lists

routing interface ip: provides the routing interface IP of the routing server name

token

format: hash algorithm used

parameters: parameters URL used to calculate token (domain, keys, values)

key: used to calculate token

url

format: format of the URL

parameters: name of the parameters

function: name of the function (represented by parameter).
Such function should be understood by the uCDN.

Call flow:

In this use case, we assume that uCDN wants to get the URL and token

format in order to be able to check URL and token and adapt them if necessary when redirecting end-users to dCDN.

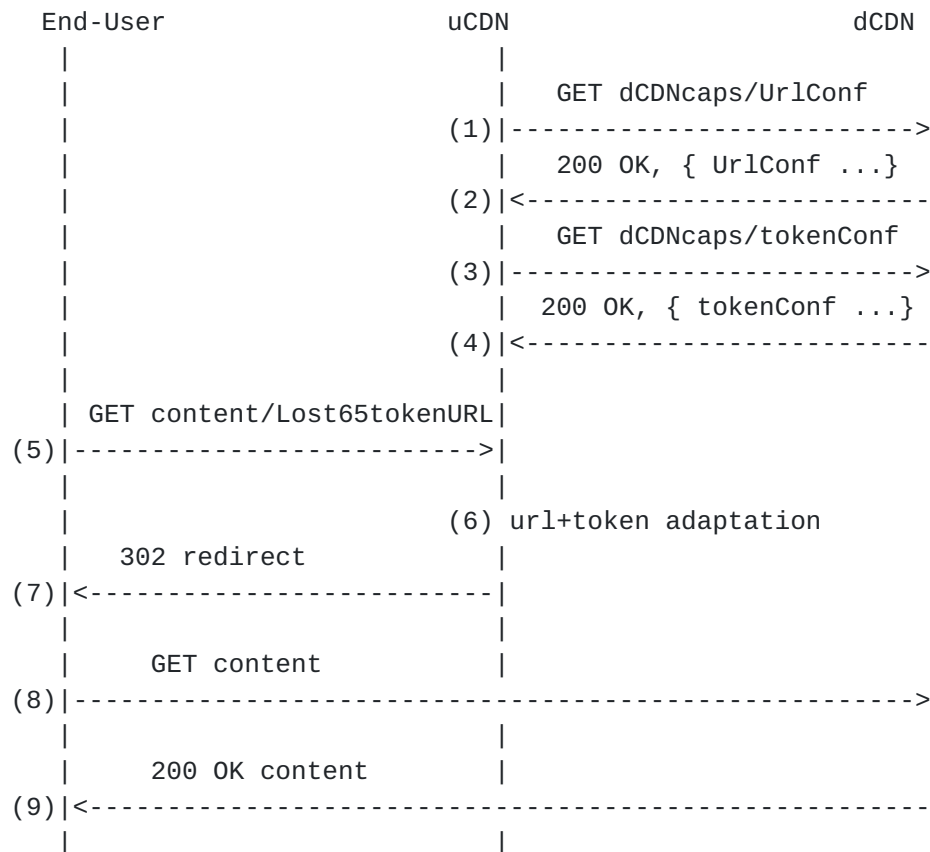


Figure 11, Technology and Vendor Interoperability

Call flow:

1. uCDN requests the dCDN URL configuration metadata;
2. dCDN answers with its URL configuration metadata;
- 3 . uCDN requests the dCDN token configuration metadata;
4. dCDN answers with the token configuration metadata (e.g. CDN model hash algorithm = MD5, ciphering key = a1f4d0eab4df...)
5. A end-user requests a content that uCDN is in charge of.
6. According to information received, the uCDN adapts its policies : redirection, ingestion

7. uCDN redirects the request to dCDN.
8. End-user requests the dCDN to deliver content
9. dCDN accepts the delivery.

Discussion:

Security considerations:

Most of this information can be restricted to specific CDNi members and subject to access control rights. Thus members SHALL be authenticated before accessing those data. Requested CDN (uCDN or dCDN) MAY refuse access to information by issuing a 401unauthorized response.

3.4.3. Dynamic QoE and QoS improvement

In this use case, the uCDN will check if the delivery QoE of the dCDN is compliant with QoE of the CDNi SLA. If not compliant, then the uCDN will redirect next requests to other dCDNs.

Assumption:

uCDN indicates the SLA that dCDN must ensure high level QoE (no sessions failures, or glitches at client side).

information elements:

- Policy: Policy provides CDN policy to ensure QoE (typically can tell which specific function can be ensured by the CDN)
- QoE: Key Performance indicator assessing the QoE (could be computed for instance on% glitches, % sessions failures, % access to service delay, etc.).

call flow:

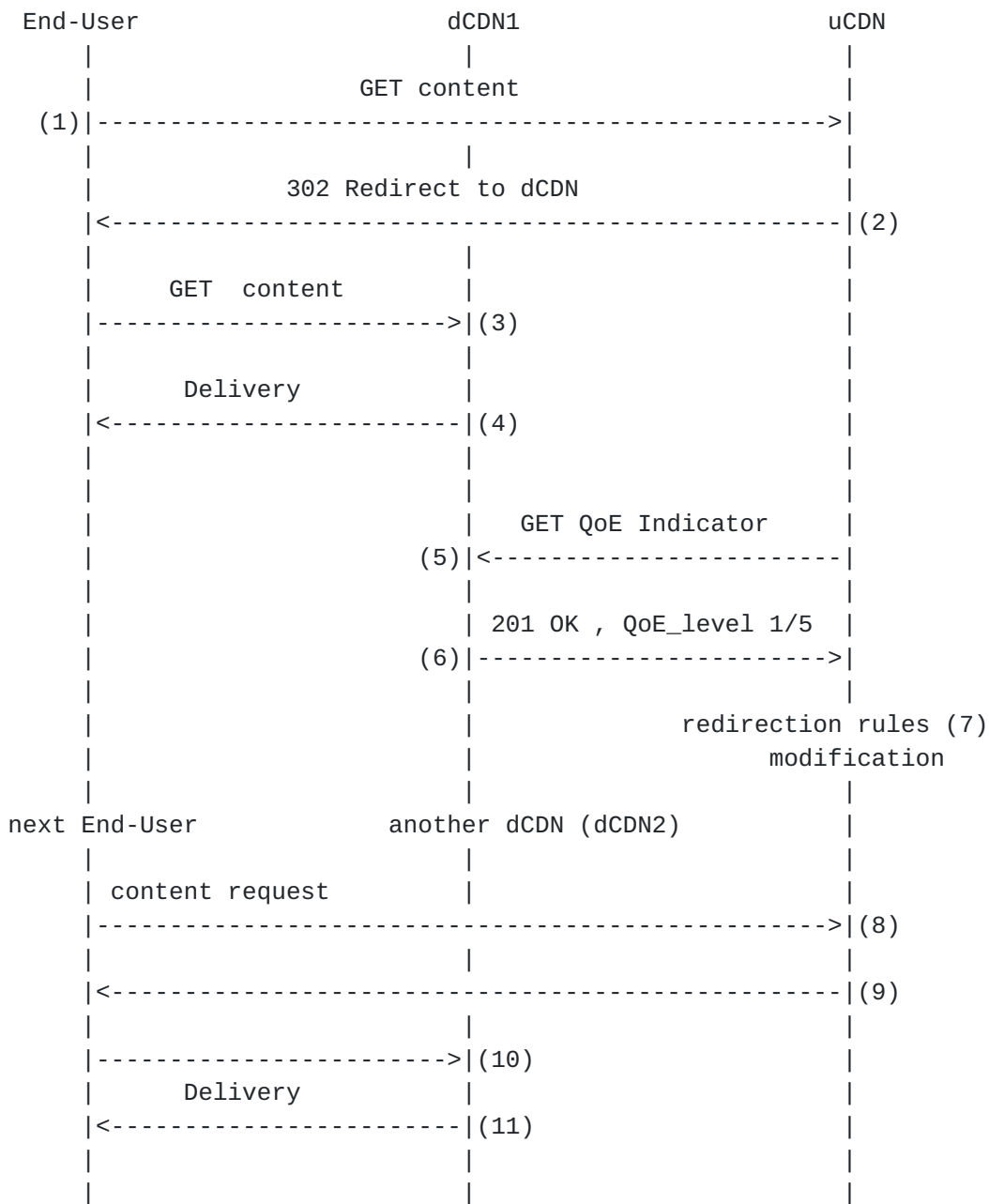


Figure 12, QoE and QoS improvement

1. A end-user requests a content that uCDN is in charge of.
2. uCDN redirects the request to dCDN1.
3. End-user requests dCDN1 to deliver content
4. dCDN1 accepts the delivery.
5. uCDN requests the QoE indicator from dCDN1 for ongoing delivery

sessions.

6. dCDN1 sends the QoE level indicator.

9. uCDN adapts its redirection rules.

10. Another end-user requests a content that uCDN is in charge of.

11. uCDN redirects the request to dCDN2 with the initial QoE.

12. End-user requests the dCDN2 to deliver content

13. dCDN2 accepts the delivery.

Discussion:

Retrieving QoE information may need some adaptation in the player at client side.

Statistics data imply logs data processing at CDN side. Statistics are carried over a the monitoring interface, and therefore are not metadata. There computing takes time and may delay the detection of the decrease of QoE.

4. Discussions

This section gather points to present during the meeting or to discuss on the ML.

4.1. JSON reference

What is the reference of the JSON language ? is it only [[RFC4627](#)] ?

Is there a JSON framework for specifying things like XML Schema ?

4.2. Network and infrastructure Metadata

2 CDNs may desire to exchange information on the location, the routing, the reachability, the availability and the load of their ressources. These metadata are not content distribution metadata.

5. Inputs for the next version

5.1. Requirement

Add the link toward individual entries of the requirement draft at the format [Req #].

Identify and specify new requirements for [\[I-D.ietf-cdni-requirements\]](#).

6. IANA Considerations

None by now.

7. Security Considerations

This section needs more works:

Content distribution Metadata, include information that may ease DoS towards CSP or CDN infrastructures.

Privacy: Content distribution Metadata may carry information on the location of the terminal

8. Acknowledgements

The authors would like to thank Yannick Le Louedec for its reviews and comments.

Part of this work is funded by the EU FP7 Ocean project.

9. References

9.1. Normative References

[I-D.ietf-cdni-problem-statement]
Niven-Jenkins, B., Faucheur, F., and N. Bitar, "Content Distribution Network Interconnection (CDNI) Problem Statement", [draft-ietf-cdni-problem-statement-00](#) (work in progress), September 2011.

[I-D.ietf-cdni-requirements]
Leung, K. and Y. Lee, "Content Distribution Network Interconnection (CDNI) Requirements", [draft-ietf-cdni-requirements-01](#) (work in progress), October 2011.

[I-D.ietf-cdni-use-cases]

Bertrand, G., Emile, S., Watson, G., Burbridge, T., Eardley, P., and K. Ma, "Use Cases for Content Delivery Network Interconnection", [draft-ietf-cdni-use-cases-00](#) (work in progress), September 2011.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", [RFC 2616](#), June 1999.

[RFC4346] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.1", [RFC 4346](#), April 2006.

9.2. Informative References

[ATIS-0800042]

ATIS, "ATIS IPTV Content on Demand Service", 2010, <ATIS IPTV Content on Demand Service>.

[DVB-IP-TV]

Digital Video Broadcasting (DVB), "Transport of MPEG-2 TS Based DVB Services over IP Based Networks", 2003, <http://www.etsi.org/deliver/etsi_ts/102000_102099/102034/01.04.01_60/ts_102034v010401p.pdf>.

[ETSI-3GP-DASH-R10]

ETSI, "Progressive Download and Dynamic Adaptive Streaming over HTTP", 2010, <<http://www.3gpp.org/ftp/Specs/html-info/26247.htm>>.

[I-D.davie-cdni-framework]

Davie, B. and L. Peterson, "Framework for CDN Interconnection", [draft-davie-cdni-framework-00](#) (work in progress), July 2011.

[I-D.jenkins-cdni-metadata]

Niven-Jenkins, B., Ferguson, D., and G. Watson, "CDN Interconnect Metadata", [draft-jenkins-cdni-metadata-00](#) (work in progress), September 2011.

[I-D.ma-cdni-metadata]

Ma, K., "Content Distribution Network Interconnection (CDNI) Metadata Interface", [draft-ma-cdni-metadata-00](#) (work in progress), October 2011.

[I-D.pantos-http-live-streaming]

Pantos, R., "HTTP Live Streaming",
[draft-pantos-http-live-streaming-07](#) (work in progress),
September 2011.

[I-D.peterson-cdni-strawman]

Peterson, L. and J. Hartman, "A Simple Approach to CDN
Interconnection", [draft-peterson-cdni-strawman-01](#) (work in
progress), May 2011.

[I-D.stiemerling-cdni-routing-cons]

Stiemerling, M., "Considerations on Request Routing for
CDNI", [draft-stiemerling-cdni-routing-cons-00](#) (work in
progress), July 2011.

[I-D.thompson-cdni-atis-scenarios]

Thompson, B. and A. Kobayashi, "ATIS Internet Sourced
Content Initiative and Relevance to CDNI",
[draft-thompson-cdni-atis-scenarios-00](#) (work in progress),
March 2011.

[RFC4627] Crockford, D., "The application/json Media Type for
JavaScript Object Notation (JSON)", [RFC 4627](#), July 2006.

[SNIA-CDMI-1.0]

ATIS, "Cloud Data Management Interface", 2010,
<<http://www.xmlmind.com/xmleditor/namespace/clipboard>>.

[TVA-CD-md]

Mtv-anytime, "Metadata Specification Version 1.3", 2003,
<<http://www.tv-anytime.org/workinggroups/wg-md.html>>.

Appendix A. GPX XML Schema fields exensibility

Following is an example of extension in XML. GPX 1.1 (See
<http://www.topografix.com/GPX/1/1/>) is a popular datamodel for
geolocalization information exchange. GPX includes GPS localization
(way point) and navigation information (routes and tracks). Each
object includes an 'extensions' element.

localization information elements

```
<xsd:complexType name="gpxType">
  <xsd:sequence>
    <xsd:element name="metadata" type="metadataType"/>
    <xsd:element name="wpt" type="wptType"/>
    <xsd:element name="rte" type="rteType"/>
    <xsd:element name="trk" type="trkType"/>
    <xsd:element name="extensions" type="extensionsType" />
  </xsd:sequence>
</xsd:complexType>
```

Metadata information elements

```
<xsd:complexType name="metadataType">
  <xsd:sequence>
    <!-- elements must appear in this order -->
    <xsd:element name="name" type="xsd:string"/>
    <xsd:element name="desc" type="xsd:string"/>
    <xsd:element name="author" type="personType"/>
    <xsd:element name="copyright" type="copyrightType"/>
    <xsd:element name="link" type="linkType"/>
    <xsd:element name="time" type="xsd:dateTime"/>
    <xsd:element name="keywords" type="xsd:string"/>
    <xsd:element name="bounds" type="boundsType"/>
    <xsd:element name="extensions" type="extensionsType"/>
  </xsd:sequence>
</xsd:complexType>
```

Figure #, GPX XML Schema Extension field

Authors' Addresses

Stephan Emile
France Telecom Orange
2 avenue Pierre Marzin
Lannion F-22307
France

Email: emile.stephan@orange.com

Bertrand Gilles
France Telecom Orange
38-40 rue du General Leclerc
Issy Les Moulineaux F-92794
France

Email: gilles.bertrand@orange.com

Fieau Frederic
France Telecom Orange
2 avenue Pierre Marzin
Lannion F-22307
France

Email: fieau.frederic@orange.com

Pages Remi
France Telecom Orange
38-40 rue du General Leclerc
Issy Les Moulineaux F-92794
France

Email: remi.pages@orange.com

