CDNI
Internet-Draft
Intended status: Informational
Expires: March 30, 2013

W. Jin
M. Li
B. Khasnabish
ZTE Corporation

September 26, 2012

Content De-duplication for CDNi Optimization
draft-jin-cdni-content-deduplication-optimization-03

Abstract

Recent explosive growth of content delivery/distribution networks (CDNs) and their interconnection are causing unintended repetition of content storage in the same dCDN. This can be avoided by using a suitable de-duplication mechanism. This document explores the scenarios which create the problems, and then discusses the approaches to eliminate the duplicated transmission of the same content from uCDN(s) to dCDN in CDNi networks. To implement the optimization, some enhancements to the CDNi metadata model and interface are required.

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 March 30, 2013.

Table of Contents

1. Introduction

In some CDNi deployment, the dCDN occasionally caches the same content copy multiple times from the same Content Service Provider (CSP). For example, the CSP may have the agreement with two authoritative CDNs, and both could be the upstream CDNs of the same dCDN. Cascading of CDNs may result in a similar scenario as well. The top-layer uCDN establishes connections with two intermediate-layer uCDNs respectively, and both connect to the same bottom-layer dCDN. In such scenarios, the dCDN may receive the content via 'push' from one of the uCDN via pre-position procedure, and then may also request for downloading the same content from another uCDN via another pre-position procedure or upon user's content request.

Storing the same content multiple times not only wastes the dCDN's the memory or storage resources, it also causes wastage of transmission bandwidth that is used to deliver the same content repeatedly. Therefore, it is necessary to avoid delivering the same content from different uCDNs to dCDN repeatedly. In this draft, we list a set of scenarios which may cause repeated delivery of the same content. A feasible solution for content de-duplication is then discussed.

In order to address the content repetition problem, several issues need to be considered.

* How to detect content repetition by dCDN.

* How to avoid content repetition, when one or more uCDNs selects one dCDN to deliver the same content to multiple User Agents.

This document provides detailed analysis on the issues of content repetition. We realize that there is a need to develop an optimized mechanism to de-duplicate the content in CDNi network. In order to implement such optimization, enhancement to CDNi metadata model and interface may be required.

1.1. Terminology

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].

This document reuses the terminology defined in:

[draft-ietf-cdni-problem-statement-08],

[draft-ietf-cdni-requirements-03],

[draft-ietf-cdni-framework-01], and

[draft-ietf-cdni-use-cases-09].

Resource Id: a metadata object (e.g., partial or whole URL, or other format) which is generated by uCDN and identifies the storage of the content in the uCDN.

Content Id: a metadata object (e.g. a URN) which uniquely identifies the content in the scope of CDNi.

2. Deployment Scenarios

This section illustrates several CDNi deployment scenarios that typically lead to duplicated content in the same server.

2.1. Impact of Content Duplication on the Network System

Along with the explosive growth of digital information, the space occupied by data is increasing as well. In the past decade, the capacity of storage system provided by many industries has developed from dozens of gigabytes to hundreds of terabytes or even more. With the exponential growth of data, enterprises are facing with an increasing number of time points for quick data backup and recovery. The cost to manage and store data, as well as the space and consumption of data center, is also growing into a more and more serious situation. Survey exposes that up to 60% of the data stored in the application system is redundant and the proportion continues to grow along with the time moving forward.

As for CDN, the duplication of the content delivered will:

1. increase the complexity of CDN management. An increasing number of content tables to be maintained are needed, thus reducing the efficiency of content search;

2. demand larger CDN storage capacitywhich would be a waste of facility and investment;

3. lead to inaccurate statistics of hot content, which consequently results in the inaccuracy of the arithmetic for the hot content dispatching in the CDN;

4. increase the latency for the CDN content access. Such cases as local content could not be hit and the same content has to be acquired from other CDNs would often occur.

2.2. Current Data Deduplication Technologies

At present, data deduplication technologies are widely used in the storage backup and archiving system, in which the deduplication module is responsible for comparing and analyzing the data content, finding out redundant data and sending corresponding metadata back to the storage service interface. Finally, non-duplicated data will be stored into the storage medium. Main technologies can be divided into:

1. Identical Data Detection: identical data includes identical files and identical data block. Whole File Detection (WFD) technology uses Hash for data mining. Fixed-sized partition (FSP) detection technology, content-defined chunking (CDC) detection technology and sliding block technology are used for the lookup and deletion of duplicated data;

2. Resembling Data Detection: according to the resemblance characteristics of the data itself, shingle technology, bloom filter technology and mode match technology are used to find out the duplicated data that can not be detected by Identical Data Detection technologies.

Above technologies are applied under the prerequisite that the content is completed downloaded and stored. However, for CDN, comparison of the content after downloading is a waste of transmission traffic and computation used for comparison. What!_s more, with a widespread deployment of CDNi system, content duplication issue will not be caused by above reasons. Instead, it results from the redirection procedures used in CDNi during which URLs pointed to the same content would be changed. In this case, dCDN would download the same content several times due to the different URLs that in fact point to the same content. Therefore, in CDNi, current data deduplication technologies can not be used to address the issue and scenarios proposed in this draft.

2.3. Content Duplication Scenario Involved in this Draft

In this document, the content duplication results from the redirection procedures used in CDNi during which URLs pointed to the same content would be changed. In this case, dCDN would download the same content several times due to the different URLs that in fact point to the same content.

2.3.1. Scenario 1

As depicted in Figure 1, both CDN-A and CDN-B establish interconnections with CDN-C that acts as a dCDN. Thus, CDN-C will

cache the content for CDN-A and CDN-B. When both CDN provider A and CDN provider B have agreements with the same CSP for content delivery, CDN-C may be required by CDN-A and CDN-B separately to retrieve and cache the same content from the CSP. CDN-C is likely to suffer from content repetition troubles.

As the location of the content in a CDN is normally assigned by CDN itself, the URLs of the same content are likely different between CDNs. So it is not enough to determine whether the content to be retrieved and cached is the same only by the URL of the content.

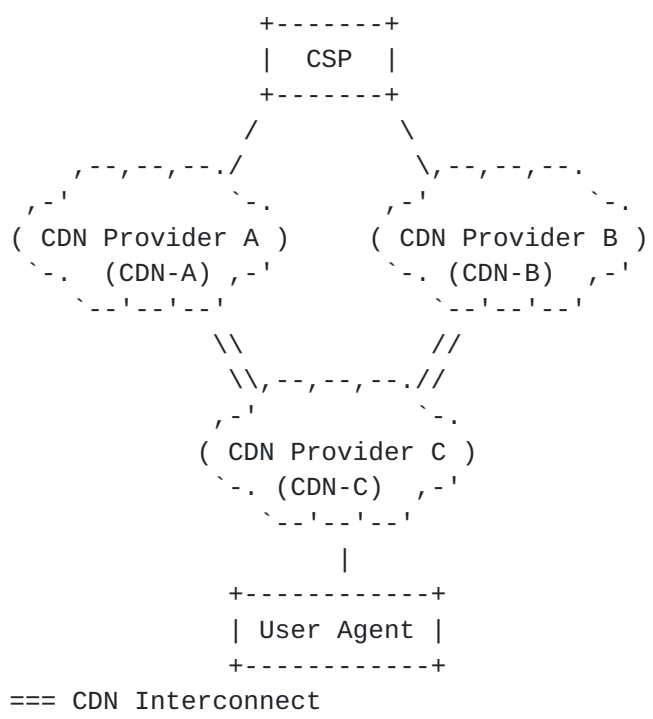

Figure 1 Interconnected CDNs with the same CSP and with one dCDN

2.3.2. Scenario 2

Now we consider the case of cascaded CDNs, as depicted in Figure 2. Note that the top-layer Upstream CDN-A has direct contract with CSP and interconnects with two middle-layer CDNs (CDN-B and CDN-C) that have the same bottom-layer Downstream CDN (CDN-D) interconnected to them. Consequently, there are two possible delivery paths for CDN-D to cache the contents of CSP, one is CDN-A -> CDN-B -> CDN-D, and the other is CDN-A -> CDN-C -> CDN-D. CDN-D may need to cache the same content by upstream CDNs(CDN-B and CDN-C) on different paths. If the URL of the content is changed by CDN-B or CDN-C, CDN-D cannot be aware of the contents to be cached and therefore this may lead to storing of duplicated contents.

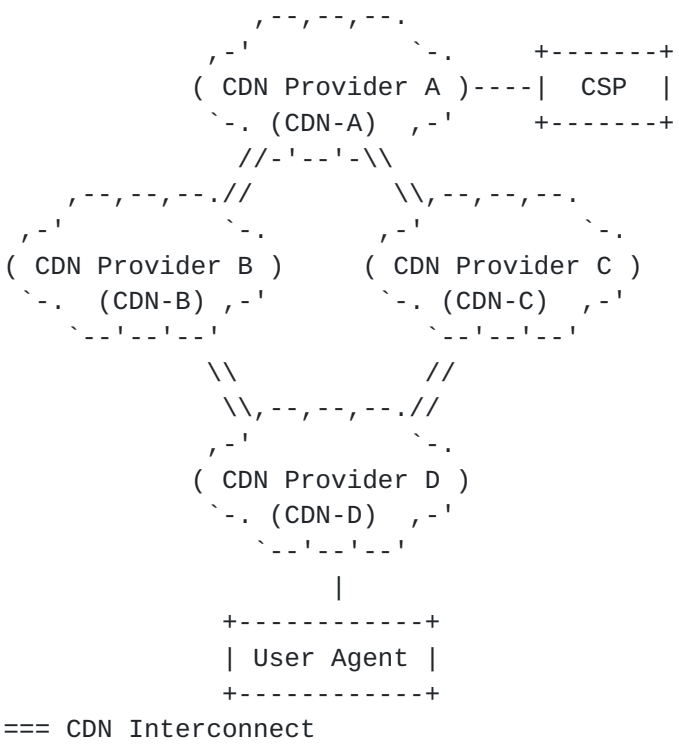

Figure 2 Cascaded CDNs

2.3.3. Scenario 3

As depicted in Figure 3, two interconnected CDNs - CDN-A(uCDN) and CDN-B(dCDN) - both have contracts with CSP. CDN-B plays two roles at the same time: downstream CDN of CDN-A and Authoritative CDN of CSP. When an end-user of CDN-A initiates content request from the CSP, CDN-A decides that CDN-B should be the serving CDN. Then CDN-A redirects the request to CDN-B. If CDN B does not have a local copy of the requested content yet (cache miss), CDN B ingests the content from CDN A. Normally CDN-B, as Authoritative CDN, is very likely to have already cached this content from original server. If CDN-B cannot identify the requested contents as the same content, this same content will be repeatedly retrieved and cached.

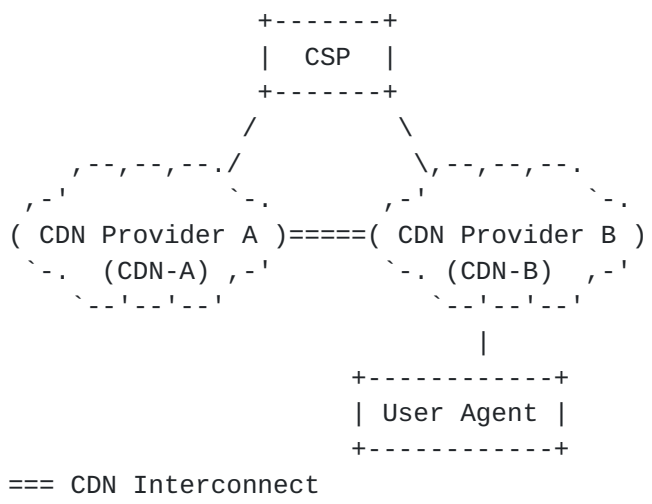

Figure 3 Interconnected CDNs with the same CSP

3. Content Naming for CDNi

It is well known that CDNs have their own content naming mechanisms, most of which are independent and separated from one another due to the use of different algorithms such as Hash algorithms. It implies that for the same content distributed by two CDNs, the corresponding content identifiers are likely to be quite different. [I-D.ietf-cdni-requirements-03] treats the information regarding CDN content naming as intra-CDN information and the CDNI solution MUST not require intra-CDN information to be exposed to other CDNs for effective and efficient delivery of the content. Therefore, establishing a uniform content naming mechanism is urgently needed for CDNi network. This mechanism which can be implemented by CDNI Metadata Distribution Protocol may have the following properties.

3.1. Uniqueness

CDNi content naming mechanism must guarantee the uniqueness of the content identification. Although URL is widely used for identifying network resource, it is not quite suitable for content identification in CDNi network where content de-duplication needs to be taken into consideration. Although the method of URL match is commonly used by many cache systems to detect the repetitive files with same name for avoiding content repetition, it will probably fail for CDNi case. This is due to the fact that different forwarding mechanisms are used in different CDNs involved. The user-originated requests are always snooped by the DPI (deep packet inspection) devices before transmitted to the original server, whereas the requests received by dCDN are always redirected by one or more uCDNs. Since there is no guarantee that the URLs will not be changed through the redirection

process, a new type of object needs to be defined to represent the uniqueness of content identifier.

For the CSP's the contents that are distributed into different interconnected CDNs, the related metadata objects may be somewhat different in many cases. For example, in Figure 2, when both CDN-A and CDN-B delegate the delivery of the same CSP's content, the content metadata such as (a) content description, (b) access policy, and (c) security policy may be not be exactly the same in both cases. Such metadata information is not suitable for content identification. Consequently, we need to define a new type of metadata object that helps uniquely identify the same content.

3.2. Ownership

CDNi content naming mechanism should embody the ownership of content identification. Typically, a CDN provider may have contracts with many CSPs for delivering their contents, as well as operate its own Content delivery network. However, it may happen that a lot of contents published by these CSPs may be very similar, and many of them may even be exactly the same. Therefore, the problem is whether these identical copies are from the same CSP or how the interconnected CDNs can verify that these contents are identical. (Note: Such copies are pointed by the same content identification only if they are from the same content source.) For a traditional (non-interconnected) CDN, there is no problem to distinguish them via its intra content naming mechanism. When a CDN interconnects with other CDNs, the condition becomes more complicated due to the lack of awareness of CSP's content when the CDN acts as a dCDN.

4. CDNi Content De-duplication Optimization Implementation

4.1. Constant URL

In general, URL-based mechanism can be used to implement content de-duplication in CDNi network. As referred in section 2, the URL description for a content may be different from uCDN to uCDN and is likely to be changed in the redirection process. An agreement to configure a specific URL between a pair of interconnected CDN is used in the draft [I-D.ietf-cdni-framework-01], however this method is not flexible enough for supporting de-duplication in complex CDNi network. So a feasible proposal is to specify a mechanism for CDNi network to guarantee that CSP's same contents cached in different uCDNs are identified by the same URL and that the URL is unchanged or at least the path part remains the same in the redirection process. The main problem of this mechanism is lack of resilience and we prefer an alternative mechanism as introduced in section 4.2 below.

4.2. Content Naming Mechanism

This section provides a detailed description of CDNi content naming mechanism using CDNi Metadata Protocol.

In general, CSP's content as well as its copies cached in interconnected CDNs are delivered to numerous End-Users. The draft [I-D.ietf-cdni-framework-01] assigns each copy an identifier in the form of an URL that is embedded with "CDN-Domain" which is used to distinguish whether a download request is from an end-user or from an uCDN. We use the term Resource Identifier to represent the storage pointing to the content copies in interconnected CDNs. However, unlike the usage in the draft [I-D.ietf-cdni-framework-01], in this document CDNi content naming mechanism specifies Resource Identifiers as the one which is only related to contents in uCDNs. Taking the example in section 2.3, we use Resource Identifier A to point to content originated through uCDN A.

Although the Resource Identifier is able to identify a content, the uniqueness of content identification can't be guaranteed, as Resource Identifier is used to identify the storage of the content in the uCDN and therefore will be changed during redirection processes between different uCDNs. In order to resolve it, we introduce the term Content Identifier that is assigned to associate with Resource Identifier to uniquely identify the content and is similar to the URN usage. Note that the Content Identifier MUST be globally unique. Figure 4 shows a metadata model that can be used for maintaining the relationship between the two types of Identifiers. Using this model, the dCDN is able to uniquely identify and route requests towards the same targeted content.

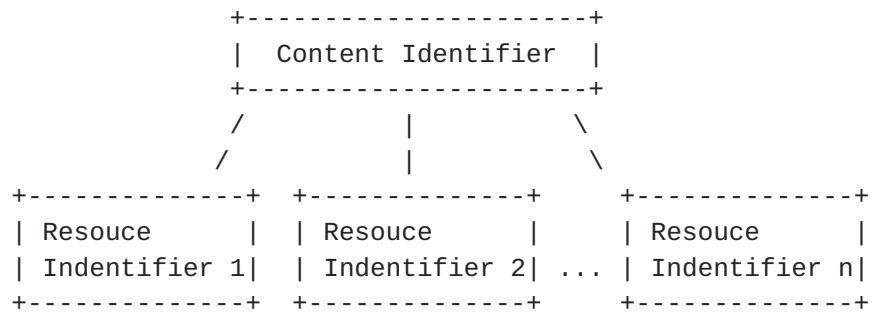

Figure 4 Metadata Model for Maintaining Relationship among Multi-IDs

Note: We need to develop an authoritative 'Entity' for creating and maintaining the Content Identifier.

4.3. Content ID

Actually, EIDR (Entertainment Identifier Registry), an industry non-profit organization, has already started the research and standardization of the globally unique content identifier and its generating mechanism. As stated in the whitepaper of [UNIVERSAL UNIQUE IDENTIFIERS IN MOVIE AND TELEVISION SUPPLY CHAIN MANAGEMENT], EIDR offers an inexpensive mechanism for uniquely identifying the complete range of audiovisual assets relevant to commerce including micro-assets such as clips and newer types of objects such as encodings. The EIDR data model can be readily extended to cover new and emerging objects and relationships as the industry evolves over time. EIDR naming system meets the requirements of coverage, flexibility, extensibility, scalability, cost-effectiveness, interoperability, etc.

The EIDR registry assigns a unique universal identifier for all registered assets. EIDR is an opaque ID with all information about the registered asset stored in the central registry. Its structure consists of a standard registry prefix, the unique suffix for each asset and a check digit. The suffix of an asset ID is of the form XXXX-XXXX-XXXX-XXXX-XXXX-C, where X is a hexadecimal digit and C is the ISO 7064 Mod 37, 36 check character.

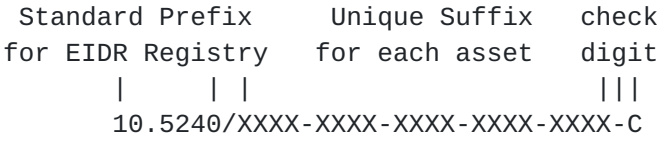

Figure 5 Structure of Content ID

The content provider submits content items for registration to the registry system, along with core metadata and information. The system uses a sophisticated system to insure that the object submitted to the registry has not already been registered and then generates an EIDR for the content. All above is done before the content is injected into CDNi system. After that, the content identifier, used as metadata of the content, is injected into CDNi system and transmitted between uCDN and dCDN via such interface as Control Interface, Metadata Interface.

The detail information of EIDR can be referred to in the whitepaper of [UNIVERSAL UNIQUE IDENTIFIERS IN MOVIE AND TELEVISION SUPPLY CHAIN MANAGEMENT].

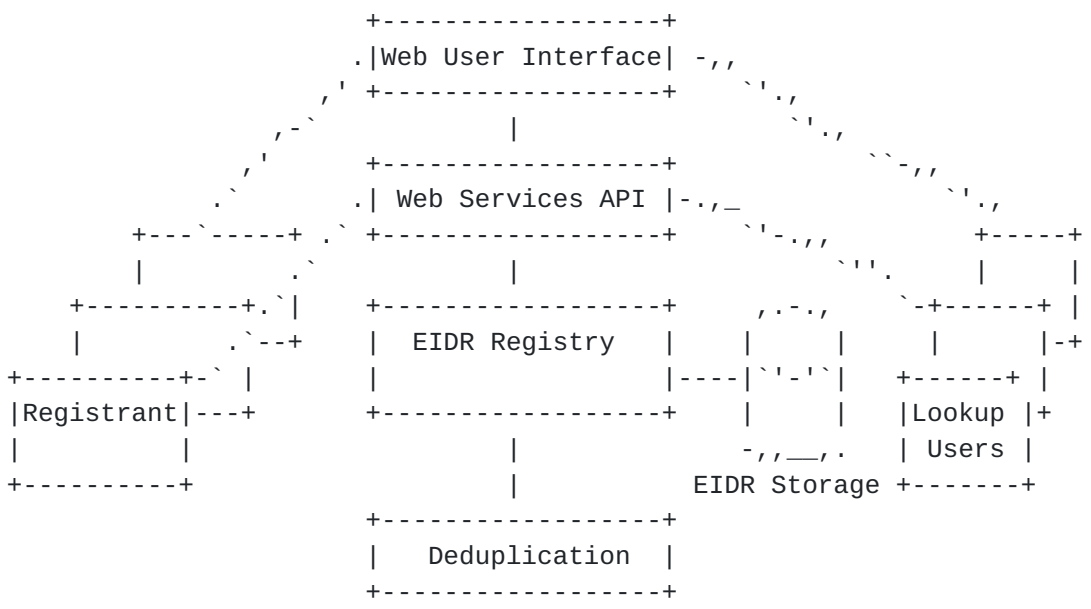

Figure 6 Entertainment Identifier Registry diagram

From the analysis of above section, the content identifier specified in EIDR is in line with the requirements of content deduplication proposed in this draft. In this document, it is suggested that we introduce content identifier format and a mechanism for generating it into CDNi with an objective to address content duplication issue.

4.4. Description of Content De-duplication

In this section the details of the solutions that use CDNi Content Naming mechanism for content de-duplication are discussed.

Content Identifier can be defined as a metadata object. The metadata object can be distributed with/without the actual content item from uCDN to dCDN for storage in the dCDN, if the uCDN wants to pre-position the content item to the dCDN before the actual user request. The content Identifier metadata object binds with the Resource Identifer to identify the storage of the content in the uCDN and forms a content identification model for the content item. By using this content identification model, an interconnected CDN is able to detect content repetition. The content status must be synchronously updated by the interconnected CDN. According to content status, the interconnected CDN can determine whether the resource copy is cached or not.

We present several procedures for optimized implementation of avoidance of CDNi content repetition.

4.4.1. Pre-Positioned Content Acquisition

The following flow illustrates how the two uCDNs successively pre-position the same content in the dCDN. In this flow, the content to be pre-positioned in the dCDN is identified by different Resource Identifiers corresponding to uCDN A and uCDN B.

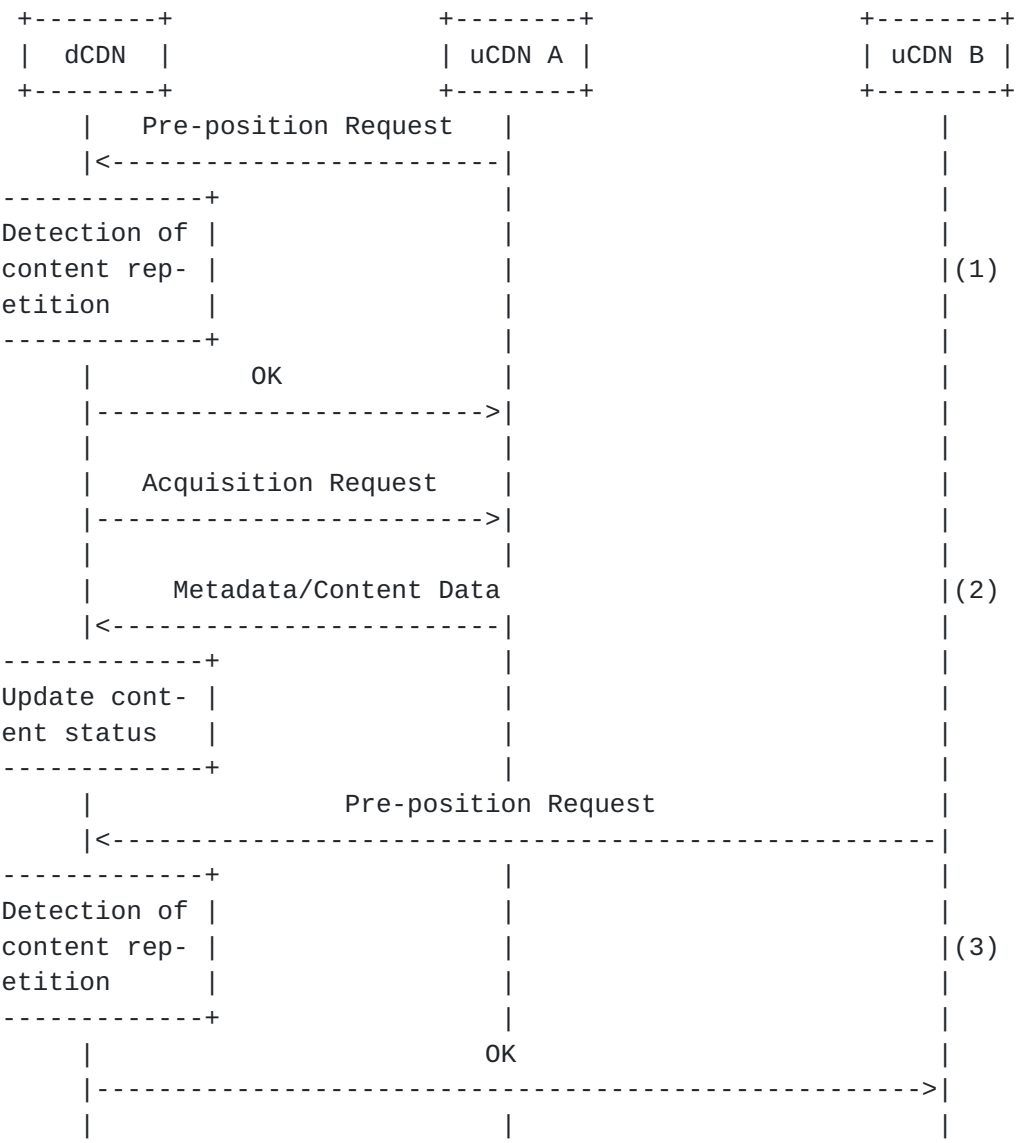

Figure 7 Acquisition of Pre-Positioned Content

The steps that are illustrated in the figure are as follows:

1. The uCDN A requests that the dCDN pre-positions a particular content item identified by its Resource Identifier and Content

Identifier. This message is sent via Trigger Interface. Receiving the message, the dCDN uses the Content Identifier to look up its stored content identification model to see whether the same content item is cached. With the result that the metadata does not exist, the dCDN replies to uCDN A with an OK message to notify that no such copy has been cached and content pre-position is required.

2. The dCDN acquires metadata of the content or the content itself from uCDN A. Once the content is pre-positioned, dCDN updates the content status and maintains the binding relation between Resource Identifier and Content Identifier metadata.

3. The uCDN B requests that dCDN pre-positions the same content item identified by its Resource Identifier and Content Identifier. Receiving the message, the dCDN uses the Content Identifier to look up its stored content identification model. Because such metadata exists, dCDN determines that the content is already cached. Then, dCDN replies to uCDN A with an OK message to notify that the same copy has been cached and the pre-position request should be cancelled. The dCDN locally binds the new Resource Identifier provided by uCDN B with the Content Identifier.

4.4.2. Dynamic Content Acquisition

The following flows illustrates how the dCDN performs content de-duplication in cases of a cache miss and a cache hit without content pre-positioning.

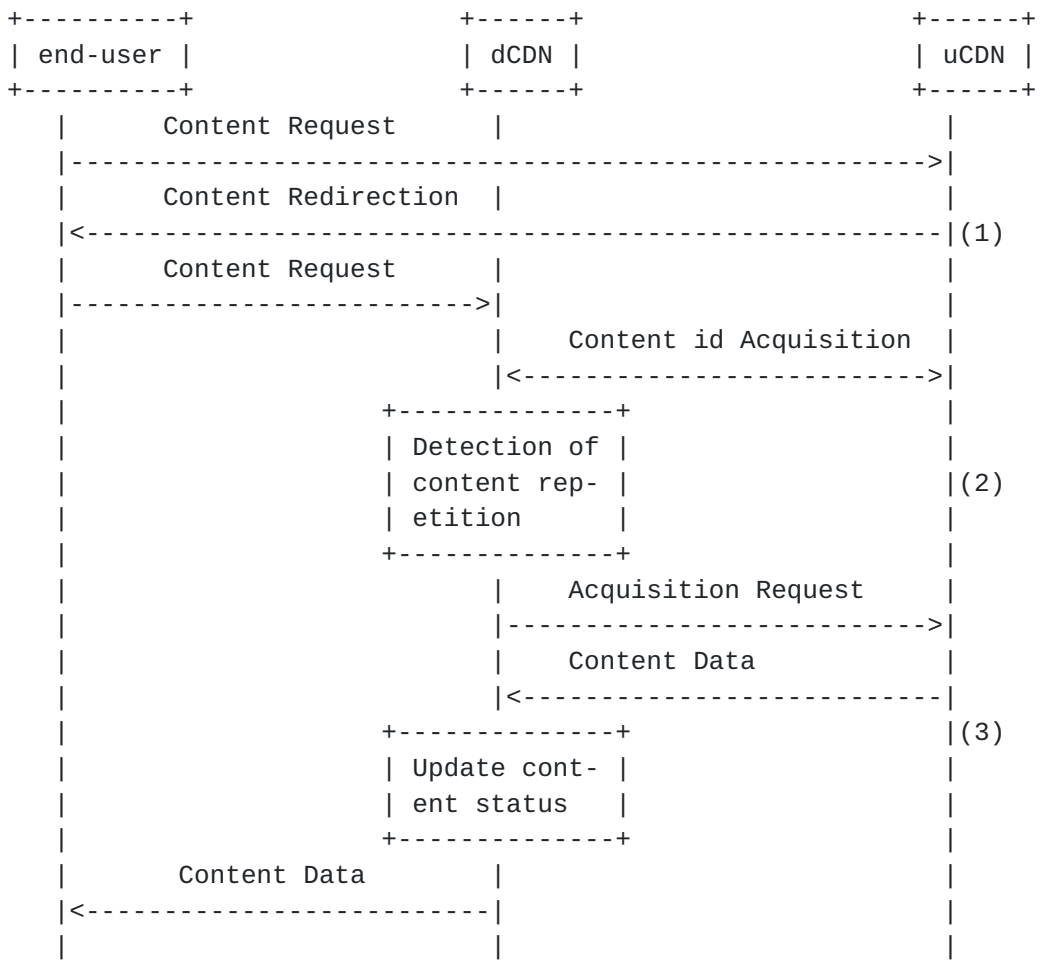

Figure 8 Dynamic Content Acquisition (cache miss case)

The steps that are illustrated in the figure are as follows:

1. A content request originated from an end-user is received at uCDN. The uCDN processes the request and recognizes that the end-user is best served by a dCDN. So uCDN redirects the request to the dCDN by sending redirection response to the end-user who then requests the content from the dCDN. The uCDN encapsulates Resource Identifier of the requested content item in the redirection response.

2. Receiving the request, the dCDN uses the Resource Identifier pointing to the requested resource to fetch the corresponding Content Identifier from the uCDN. The dCDN then uses the Content Identifier to look up its stored content identification model to check whether the content item is cached. With the result that such metadata does not exist, the case of a cache miss is determined by the dCDN, and therefore content needs to be downloaded from the uCDN before delivered to the end-user.

3. The dCDN acquires the requested content from uCDN A. Once the content is cached, dCDN updates the content status and maintains the binding relation between Resource Identifier and the Content Identifier metadata. The dCDN then delivers content data to the end-user.

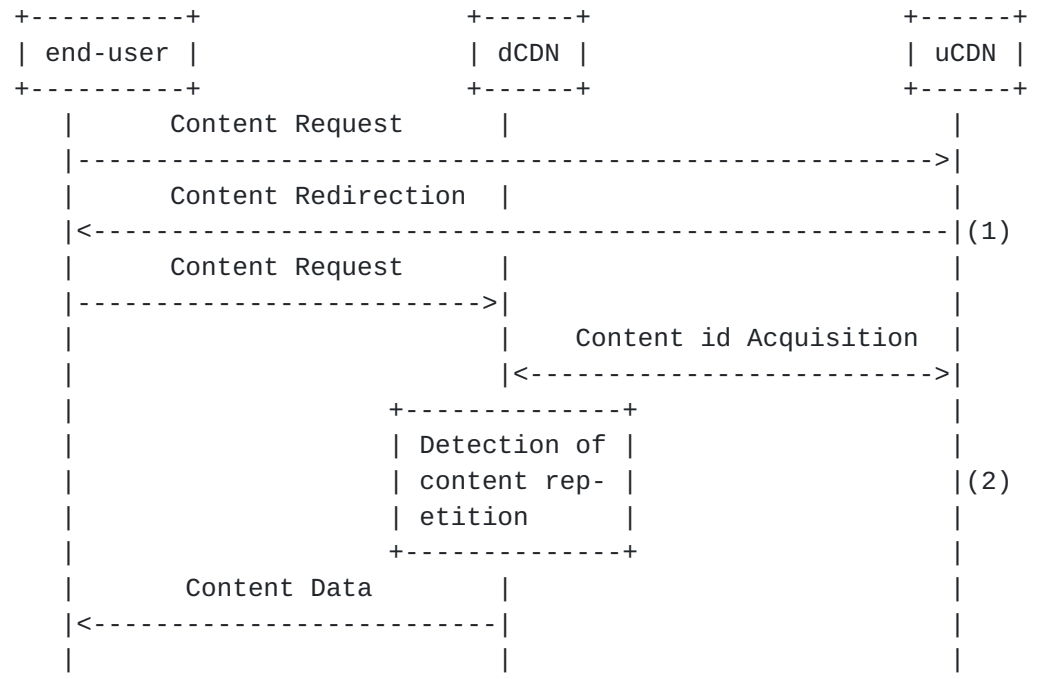

Figure 9 Dynamic Content Acquisition (cache hit case)

The steps that are illustrated in the figure are as follows:

Steps 1 and 2 are exactly the same as steps 1 and 2 of Figure 6, except that this time dCDN determines the case of a cache hit according to the existence of such record in the corresponding Content Identifier metadata.

This flow differs from the one in Figure 6 only in terms of not triggering dynamic content acquisition (step 3), since the content has already been cached by dCDN.

4.4.3. Content Purge and Invalidate

In general, the dCDN would assign separate location for each of its uCDN to store triggers. However, when it comes to complex CDNi deployment as discussed in this draft, dCDN is likely to receive multiple trigger operations coming from different uCDNs on the same content. If one of the uCDNs requests to invalidate certain content,

after receiving this Invalidated Trigger, dCDN finds the content using the Content Identifier of this content and marks Invalid in the Trigger Status Resource corresponding to the requesting uCDN. By doing so, this content is unavailable to this uCDN before it re-validates this content. The access to this content by other uCDNs will not be impacted.

If one of the uCDNs requests to purge certain content, after receiving this Purge Trigger, dCDN finds the content using the Content Identifier of this content and marks Invalid in the Trigger Status Resource corresponding to the requesting uCDN. By doing this, this uCDN is not able to make any operation on this content, while the content itself is not deleted from the cache of dCDN. The access to this content by other uCDNs will not be impacted. Only when all the uCDNs connected with this dCDN request to purge the same content and dCDN accepts all these requests will the content be purged from dCDN.

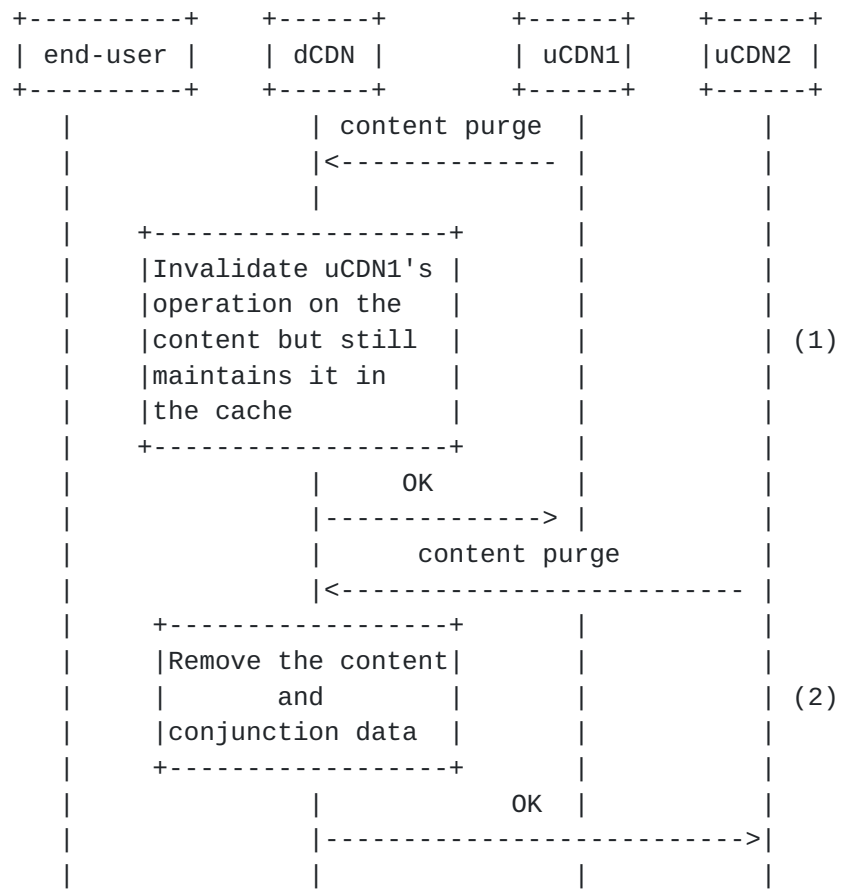

Figure 10 Removal of Content

A premise is that the content copy to be purged has already been cached in the dCDN from the uCDN. The Content Identifier of the content is linked with two Resource IDs from uCDN1 and uCDN2. The steps illustrated in the figure are as follows:

1. The uCDN1 requests that the dCDN remove some content resource identified by the Content Identifier due to the deployment policy or expiration of content's life-time. dCDN then invalidates uCDN's operation on the requested content but still maintains it in the cache so that other uCDNs that also have conjunction with this content can operate on this content normally. It then replies an OK response to uCDN1

2. If uCDN2 also requests that dCDN remove the same content, dCDN removes the contentand all the conjunction metadata. It then replies an OK response to uCDN2.

5. Security Considerations

To be added later

6. IANA Considerations

This document has no IANA Considerations.

7. Acknowledgments

The authors would like to thank Francois Le Faucheur, Kevin Ma, Theodore Zahariadis, Ben Niven-Jenkins, and Ramki Krishnam for valuable inputs, suggestions, and discussions.

8. References

8.1. Normative References

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

8.2. Informative References

[Data Deduplication Techniques]
Ao, L., Shu, JW., and MQ. Li, "Data Deduplication Techniques", May 2010.

[EIDR WHITEPAPER]
EIDR, EIDR., "UNIVERSAL UNIQUE IDENTIFIERS IN MOVIE AND TELEVISION SUPPLY CHAIN MANAGEMENT", October 2010.

[I-D.ietf-cdni-framework]
Peterson, L. and B. Davie, "Framework for CDN Interconnection", April 2012.

[I-D.ietf-cdni-problem-statement]
Niven-Jenkins, B., Faucheur, F., and N. Bitar, "Content Distribution Network Interconnection (CDNI) Problem Statement", May 2012.

[I-D.ietf-cdni-requirements]
Leung, K. and Y. Lee, "Content Distribution Network Interconnection (CDNI) Requirements", December 2011.

[I-D.murray-cdni-triggers]

Murray, R., Niven-Jenkins, B., and Velocix, "CDN Interconnect Triggers", March 2013.

[I-D.narten-iana-considerations-rfc2434bis]
Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", draft-narten-iana-considerations-rfc2434bis-09 (work in progress), March 2008.

[RFC2629] Rose, M., "Writing I-Ds and RFCs using XML", RFC 2629, June 1999.

[RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, July 2003.

Authors' Addresses

WeiYi Jin
ZTE Corporation
Nanjing, 210012
China

Phone: +86 025-52871364
Email: jin.weiyi@zte.com.cn

Mian Li
ZTE Corporation
Nanjing, 210012
China

Phone: +86 025-88014641
Email: li.mian@zte.com.cn

Bhumip Khasnabish
ZTE Corporation
New Jersey, 07960
USA

Phone: +001-781-752-8003
Email: bhumip.khasnabish@zteusa.com