Internet Engineering Task Force (IETF)
Request for Comments: 7975
Category: Standards Track
ISSN: 2070-1721

B. Niven-Jenkins, Ed.
Nokia
R. van Brandenburg, Ed.
TNO

October 2016

Request Routing Redirection Interface for Content Delivery Network (CDN) Interconnection

Abstract

The Request Routing interface comprises (1) the asynchronous advertisement of footprint and capabilities by a downstream Content Delivery Network (CDN) that allows an upstream CDN to decide whether to redirect particular user requests to that downstream CDN; and (2) the synchronous operation of an upstream CDN requesting whether a downstream CDN is prepared to accept a user request and of a downstream CDN responding with how to actually redirect the user request. This document describes an interface for the latter part, i.e., the CDNI Request Routing Redirection interface.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc7975.

Table of Contents

1. Introduction

A Content Delivery Network (CDN) is a system built on an existing IP network that is used for large-scale content delivery, via prefetching or dynamically caching content on its distributed surrogates (caching servers). [RFC6707] describes the problem area of interconnecting CDNs.

The CDNI Request Routing interface outlined in [RFC7336] is comprised of:

1. The asynchronous advertisement of footprint and capabilities by a downstream CDN (dCDN) that allows an upstream CDN (uCDN) to decide whether to redirect particular user requests to that dCDN.

2. The synchronous operation of a uCDN requesting whether a dCDN is prepared to accept a user request and of a dCDN responding with how to actually redirect the user request.

This document describes an interface for the latter part, i.e., the CDNI Request Routing Redirection interface (RI).

2. 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 [RFC2119].

This document reuses the terminology defined in [RFC6707].

The following additional terms are introduced by this document:

Application-Level Redirection: The act of using an application-specific redirection mechanism for the request routing process of a CDN. The Redirection Target (RT) is the result of a CDN's routing decision at the time it receives a content request via an application-specific protocol response. Examples of an application-level redirection are HTTP 302 Redirection and Real Time Messaging Protocol (RTMP) [RTMP] 302 Redirection.

DNS Redirection: The act of using DNS name resolution for the request routing process of a CDN. In DNS Redirection, the DNS name server of the CDN makes the routing decision based on a local policy and selects one or more Redirection Targets (RTs) and redirects the User Agent (UA) to the RT(s) by returning the details of the RT(s) in response to the DNS query request from the User Agent's DNS resolver.

HTTP Redirection: The act of using an HTTP redirection response for the request routing process of a CDN. The Redirection Target (RT) is the result of the routing decision of a CDN at the time it receives a content request via HTTP. HTTP Redirection is a particular case of application-level redirection.

Redirection Target (RT): The endpoint to which the User Agent is redirected. In CDNI, an RT may point to a number of different components, some examples include a surrogate in the same CDN as the request router, a request router in a dCDN, or a surrogate in a dCDN.

3. Interface Function and Operation Overview

The main function of the CDNI Redirection interface (RI) is to allow the request routing systems in interconnected CDNs to communicate to facilitate the redirection of User Agent requests between interconnected CDNs.

The detailed requirements for the Redirection interface and their relative priorities are described in Section 5 of [RFC7337].

The User Agent will make a request to a request router in the uCDN using one of either DNS or HTTP. The RI is used between the uCDN and one or more dCDNs. The dCDN's RI response may contain a Redirection Target with a type that is compatible with the protocol used between the User Agent and uCDN request router. The dCDN has control over the Redirection Target it provides. Depending on the returned Redirection Target, the User Agent's request may be redirected to:

- The final surrogate, which may be in the dCDN that returned the RI response to the uCDN or another CDN (if the dCDN delegates the delivery to another CDN); or

- A request router (in the dCDN or another CDN), which may use a different redirection protocol (DNS or HTTP) than the one included in the RI request.

The Redirection interface operates between the request routing systems of a pair of interconnected CDNs. To enable communication over the Redirection interface, the uCDN needs to know the URI (endpoint) in the dCDN to send CDNI request routing queries.

The Redirection interface URI may be statically preconfigured, dynamically discovered via the CDNI Control interface, or discovered via other means. However, such discovery mechanisms are not specified in this document, as they are considered out of the scope of the Redirection interface specification.

The Redirection interface is only relevant in the case of Recursive Request Redirection, as Iterative Request Redirection does not invoke any interaction over the Redirection interface between interconnected CDNs. Therefore, the scope of this document is limited to Recursive Request Redirection.

In the case of Recursive Request Redirection, in order to perform redirection of a request received from a User Agent, the uCDN queries the dCDN so that the dCDN can select and provide a Redirection Target. In cases where a uCDN has a choice of dCDNs, it is up to the uCDN to decide (for example, via configured policies) which dCDN(s) to query and in which order to query them. A number of strategies are possible, including selecting a preferred dCDN based on local policy, possibly falling back to querying an alternative dCDN(s) if the first dCDN does not return a Redirection Target or otherwise rejects the uCDN's RI request. A more complex strategy could be to query multiple dCDNs in parallel before selecting one and using the Redirection Target provided by that dCDN.

The uCDN->User Agent redirection protocols addressed in this document are: DNS redirection and HTTP redirection. Other types of application-level redirection will not be discussed further in this document. However, the Redirection interface is designed to be extensible and could be extended to support additional application-level redirection protocols.

For both DNS and HTTP redirection, either HTTP or HTTPS could be used to connect to the Redirection Target. When HTTPS is used to connect to the uCDN, if the uCDN uses DNS redirection to identify the RT to the User Agent, then the new target domain name may not match the domain in the URL dereferenced to reach the uCDN; without operational precautions, and in the absence of DNSSEC, this can make a legitimate redirection look like a DNS-based attack to a User Agent and trigger security warnings. When DNS-based redirection with HTTPS is used, this specification assumes that any RT can complete the necessary Transport Layer Security (TLS) handshake with the User Agent. Any operational mechanisms this requires, e.g., private key distribution to surrogates and request routers in dCDNs, are outside the scope of this document.

This document also defines an RI loop prevention and detection mechanism as part of the Redirection interface.

4. HTTP-Based Interface for the Redirection Interface

This document defines a simple interface for the Redirection interface based on HTTP [RFC7230], where the attributes of a User Agent's requests are encapsulated along with any other data that can aid the dCDN in processing the requests. The RI response encapsulates the attributes of the RT(s) that the uCDN should return to the User Agent (if it decides to utilize the dCDN for delivery) along with the policy for how the response can be reused. The examples of RI requests and responses below do not contain a complete set of HTTP headers for brevity; only the pertinent HTTP headers are shown.

The RI between the uCDN and dCDN uses the same HTTP interface to encapsulate the attributes of both DNS and HTTP requests received from User Agents, although the contents of the RI requests/responses contain data specific to either DNS or HTTP redirection.

This approach has been chosen because it enables CDN operators to only have to deploy a single interface for the RI between their CDNs, regardless of the User Agent redirection method. In this way, from an operational point of view, there is only one interface to monitor, manage, develop troubleshooting tools for, etc.

In addition, having a single RI where the attributes of the User Agent's DNS or HTTP request are encapsulated along with the other data required for the dCDN to make a request routing decision, avoids having to both 1) try to encapsulate or proxy DNS/HTTP/RTMP/ etc. requests and 2) find ways to somehow embed the additional CDNI Request Routing Redirection interface properties/data within those end-user DNS/HTTP/RTMP/etc. requests.

Finally, the RI is easily extendable to support other User Agent request redirection methods (e.g., RTMP 302 redirection) by defining additional protocol-specific keys for RI requests and responses along with a specification about how to process them.

The generic Recursive Request Redirection message flow between Request Routing systems in a pair of interconnected CDNs is as follows:

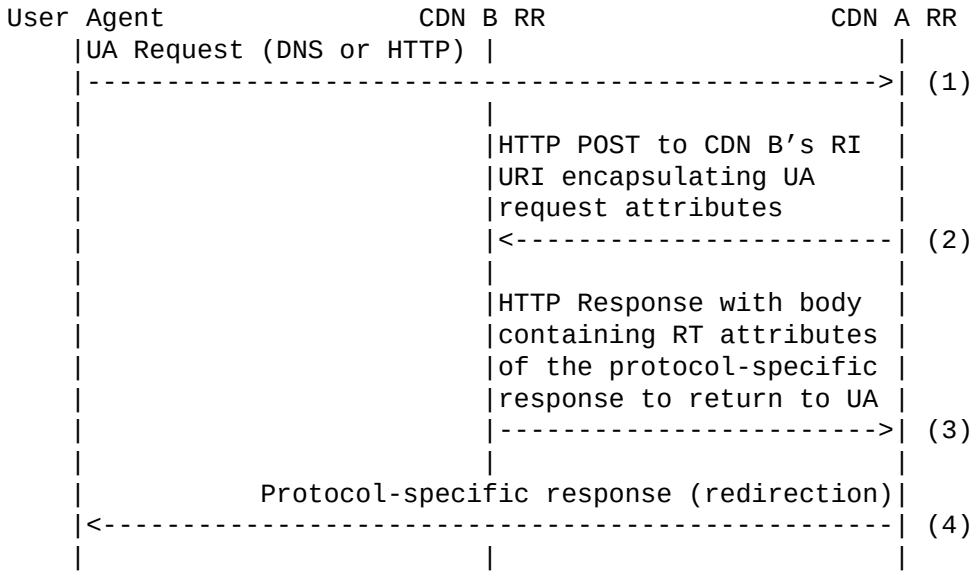

Figure 1: Generic Recursive Request Redirection Message Flow

1. The User Agent sends its (DNS or HTTP) request to CDN A. The Request Routing System of CDN A processes the request and, through local policy, recognizes that the request is best served by another CDN, specifically CDN B (or that CDN B may be one of a number of candidate dCDNs it could use).

2. The Request Routing System of CDN A sends an HTTP POST to CDN B's RI URI containing the attributes of the User Agent's request.

3. The Request Routing System of CDN B processes the RI request and, assuming the request is well-formed, responds with an HTTP "200" response with a message body containing the RT(s) to return to the User Agent as well as parameters that indicate the properties of the response (cacheability and scope).

4. The Request Routing System of CDN A sends a protocol-specific response (containing the returned attributes) to the User Agent, so that the User Agent's request will be redirected to the RT(s) returned by CDN B.

4.1. Information Passed in RI Requests and Responses

The information passed in RI requests splits into two basic categories:

1. The attributes of the User Agent's request to the uCDN.

2. Properties/parameters that the uCDN can use to control the dCDN's response or that can help the dCDN make its decision.

Generally, dCDNs can provide better routing decisions given additional information about the content request, e.g., the URI of the requested content or the User Agent's IP address or subnet. The information required to base a routing decision on can be highly dependent on the type of content delivered. A uCDN SHOULD only include information that is absolutely necessary for delivering that type of content. Cookies in particular are especially sensitive from a security/privacy point of view and in general SHOULD NOT be conveyed in the RI Requests to the dCDN. The information necessary to be conveyed for a particular type of request is expected to be conveyed out of band between the uCDN and dCDN. See Section 5.2 for more detail on the privacy aspects of using RI Requests to convey information about UA requests.

In order for the dCDN to determine whether it is capable of delivering any requested content, it requires CDNI metadata related to the content the User Agent is requesting. That metadata will describe the content and any policies associated with it. It is expected that the RI request contains sufficient information for the Request Router in the dCDN to be able to retrieve the required CDNI Metadata via the CDNI Metadata interface.

The information passed in RI responses splits into two basic categories:

1. The attributes of the RT to return to the User Agent in the DNS response or HTTP response.

2. Parameters/policies that indicate the properties of the response, such as, whether it is cacheable, the scope of the response, etc.

In addition to details about how to redirect the User Agent, the dCDN may wish to return additional policy information to the uCDN. For example, the dCDN may wish to return a policy that expresses "this response can be reused without requiring an RI request for 60 seconds provided the User Agent's IP address is in the range 198.51.100.0 -- 198.51.100.255".

These additional policies split into two basic categories:

- Cacheability information signaled via the HTTP response headers of the RI response (to reduce the number of subsequent RI requests the uCDN needs to make).

- The scope of a cacheable response signaled in the HTTP response body of the RI response, for example, whether the response applies to a wider range of IP addresses than what was included in the RI request.

The cacheability of the response is indicated using the standard HTTP Cache-Control mechanisms.

4.2. JSON Encoding of RI Requests and Responses

The body of RI requests and responses is a JSON object [RFC7159] that contains a dictionary of key:value pairs that MUST conform to [RFC7493]. Senders MUST encode all (top-level object and sub-object) keys specified in this document in lowercase. Receivers MUST ignore any keys that are unknown or invalid.

The following table defines the top-level keys and indicates whether they are applicable to RI requests, RI responses, or both:

Key	Request/Response	Description
dns	Both	The attributes of the UA's DNS request or the attributes of the RT(s) to return in a DNS response.
http	Both	The attributes of the UA's HTTP request or the attributes of the RT to return in an HTTP response.
scope	Response	The scope of the response (if it is cacheable). For example, whether the response applies to a wider range of IP addresses than what was included in the RI request.
error	Response	Additional details if the response is an error response.
cdn-path	Both	A List of Strings. Contains a list of the CDN Provider IDs of previous CDNs that have participated in the request routing for the associated User Agent request. On RI requests, it contains the list of previous CDNs that this RI request has passed through. On RI responses, it contains the list of CDNs that were involved in obtaining the final redirection included in the RI response. See Section 4.8.
max-hops	Request	Integer specifying the maximum number of hops (CDN Provider IDs) this request is allowed to be propagated along. This allows the uCDN to coarsely constrain the latency of the request routing chain.

Table 1: Top-Level Keys in RI Requests/Responses

A single request or response MUST contain only one of the dns or http keys. Requests MUST contain a cdn-path key and responses MAY contain a cdn-path key. If the max-hops key is not present, then there is no limit on the number of CDN hops that the RI request can be propagated along. If the first uCDN does not wish the RI request to be propagated beyond the dCDN it is making the request to, then the uCDN MUST set max-hops to 1.

The cdn-path MAY be reflected back in RI responses, although doing so could expose information to the uCDN that a dCDN may not wish to expose (for example, the existence of business relationships between a dCDN and other CDNs).

If the cdn-path is reflected back in the RI response, it MUST contain the value of cdn-path received in the associated RI request with the final dCDN's CDN Provider ID appended. Transit CDNs MAY remove the cdn-path from RI responses but MUST NOT modify the cdn-path in other ways.

The presence of an error key within a response that also contains either a dns or http key does not automatically indicate that the RI request was unsuccessful as the error key MAY be used for communicating additional (e.g., debugging) information. When a response contains an error key as well as either a dns or http key, the error-code SHOULD be 1xx (e.g., 100). See Section 4.7 for more details about encoding error information in RI responses.

All implementations that support IPv4 addresses MUST support the encoding specified by the 'IPv4address' rule in Section 3.2.2 of [RFC3986]. Likewise, implementations that support IPv6 addresses MUST support all IPv6 address formats specified in [RFC4291]. Server implementations SHOULD use IPv6 address formats specified in [RFC5952].

4.3. MIME Media Types Used by the RI Interface

RI requests MUST use a MIME media type of application/cdni as specified in [RFC7736], with the Payload Type (ptype) parameter set to 'redirection-request'.

RI responses MUST use a MIME media type of application/cdni as specified in [RFC7736], with the Payload Type (ptype) parameter set to 'redirection-response'.

4.4. DNS Redirection

The following sections provide detailed descriptions of the information that should be passed in RI requests and responses for DNS redirection.

4.4.1. DNS Redirection Requests

For DNS-based redirection, the uCDN needs to pass the following information to the dCDN in the RI request:

- The IP address of the DNS resolver that made the DNS request to the uCDN.
- The type of DNS query made (usually either A or AAAA).
- The class of DNS query made (usually IN).
- The fully qualified domain name for which DNS redirection is being requested.
- The IP address or prefix of the User Agent (if known to the uCDN).

The preceding information is encoded as a set of key:value pairs within the dns dictionary as follows:

Key	Value	Mandatory	Description
resolver-ip	String	Yes	The IP address of the UA's DNS resolver.
qtype	String	Yes	The type of DNS query made by the UA's DNS resolvers in uppercase. The value of this field SHALL be set to either 'A' or 'AAAA'.
qclass	String	Yes	The class of DNS query made in uppercase (IN, etc.).
qname	String	Yes	The fully qualified domain name being queried.
c-subnet	String	No	The IP address (or prefix) of the UA in Classless Inter-Domain Routing (CIDR) format.
dns-only	Boolean	No	If True, then dCDN MUST only use DNS redirection and MUST include RTs to one or more surrogates in any successful RI response. CDNs MUST include the dns-only property set to True on any cascaded RI requests. Defaults to False.

Table 2

An RI request for DNS-based redirection MUST include a dns dictionary. This dns dictionary MUST contain the following keys: resolver-ip, qtype, qclass, and qname; the value of each MUST be the value of the appropriate part of the User Agent's DNS query/request. For internationalized domain names containing non-ASCII characters, the value of the qname field MUST be the ASCII-compatible encoded (ACE) representation (A-label) of the domain name [RFC5890].

An example RI request (uCDN->dCDN) for DNS-based redirection is as follows:

```
POST /dcdn/ri HTTP/1.1
Host: rr1.dcdn.example.net
Content-Type: application/cdni; ptype=redirection-request
Accept: application/cdni; ptype=redirection-response

{
  "dns" : {
    "resolver-ip" : "192.0.2.1",
    "c-subnet" : "198.51.100.0/24",
    "qtype" : "A",
    "qclass" : "IN",
    "qname" : "www.example.com"
  },
  "cdn-path": ["AS64496:0"],
  "max-hops": 3
}
```

4.4.2. DNS Redirection Responses

For a successful DNS-based redirection, the dCDN needs to return one of the following to the uCDN in the RI response:

- The IP address(es) of (or the CNAME of) RTs that are dCDN surrogates (if the dCDN is performing DNS-based redirection directly to a surrogate); or

- The IP address(es) of (or the CNAME of) RTs that are Request Routers (if the dCDN will perform request redirection itself). A dCDN MUST NOT return an RT that is a Request Router if the dns-only key is set to True in the RI request.

The preceding information is encoded as a set of key:value pairs within the dns dictionary as follows:

Key	Value	Mandatory	Description
rcode	Integer	Yes	DNS response code (see [RFC6895]).
name	String	Yes	The fully qualified domain name the response relates to.
a	List of String	No	Set of IPv4 Addresses of RT(s).
aaaa	List of String	No	Set of IPv6 Addresses of RT(s).
cname	List of String	No	Set of fully qualified domain names of RT(s).
ttl	Integer	No	TTL in seconds of DNS response. Default is 0.

Table 3

A successful RI response for DNS-based redirection MUST include a dns dictionary and MAY include an error dictionary (see Section 4.7). An unsuccessful RI response for DNS-based redirection MUST include an error dictionary. If a dns dictionary is included in the RI response, it MUST include the rcode and name keys and it MUST include at least one of the following keys: a, aaaa, or cname. The dns dictionary MAY include both a and aaaa keys. If the dns dictionary contains a cname key, it MUST NOT contain either an a or aaaa key. For internationalized domain names containing non-ASCII characters, the value of the cname field MUST be the ACE representation (A-label) of the domain name.

An example of a successful RI response (dCDN->uCDN) for DNS-based redirection with both a and aaaa keys is listed below:

```
HTTP/1.1 200 OK
Date: Mon, 06 Aug 2012 18:41:38 GMT
Content-Type: application/cdni; ptype=redirection-response

{
  "dns" : {
    "rcode" : 0,
    "name" : "www.example.com",
    "a" : ["203.0.113.200", "203.0.113.201", "203.0.113.202"],
    "aaaa" : ["2001:DB8::C8", "2001:DB8::C9"],
    "ttl" : 60
  }
}
```

A further example of a successful RI response (dCDN->uCDN) for DNS-based redirection is listed below, in this case with a cname key containing the FQDN of the RT.

```
HTTP/1.1 200 OK
Date: Mon, 06 Aug 2012 18:41:38 GMT
Content-Type: application/cdni; ptype=redirection-response

{
  "dns" : {
    "rcode" : 0,
    "name" : "www.example.com",
    "cname" : ["rr1.dcdn.example"],
    "ttl" : 20
  }
}
```

4.5. HTTP Redirection

The following sections provide detailed descriptions of the information that should be passed in RI requests and responses for HTTP redirection.

The dictionary keys used in HTTP Redirection requests and responses use the following conventions for their prefixes:

- c- is prefixed to keys for information related to the Client (User Agent).
- cs- is prefixed to keys for information passed by the Client (User Agent) to the Server (uCDN).
- sc- is prefixed to keys for information to be passed by the Server (uCDN) to the Client (User Agent).

4.5.1. HTTP Redirection Requests

For HTTP-based redirection, the uCDN needs to pass the following information to the dCDN in the RI request:

- The IP address of the User Agent.
- The URI requested by the User Agent.
- The HTTP method requested by the User Agent.
- The HTTP version number requested by the User Agent.

The uCDN may also decide to pass the presence and value of particular HTTP headers included in the User Agent request to the dCDN.

The preceding information is encoded as a set of key:value pairs within the http dictionary as follows:

Key	Value	Mandatory	Description
c-ip	String	Yes	The IP address of the UA.
cs-uri	String	Yes	The Effective Request URI [RFC7230] requested by the UA.
cs-method	String	Yes	The method part of the request-line as defined in Section 3.1.1 of [RFC7230].
cs-version	String	Yes	The HTTP-version part of the request-line as defined in Section 3.1.1 of [RFC7230].
cs-(<headername>)	String	No	The field-value of the HTTP header field named <HeaderName> as a string, for example, cs-(cookie) would contain the value of the HTTP Cookie header from the UA request.

Table 4

An RI request for HTTP-based redirection MUST include an http dictionary. This http dictionary MUST contain the following keys: c-ip, cs-method, cs-version, and cs-uri; the value of each MUST be the value of the appropriate part of the User Agent's HTTP request.

The http dictionary of an RI request MUST contain a maximum of one cs-(<headername>) key for each unique header field-name (HTTP header field). <headername> MUST be identical to the equivalent HTTP header field-name encoded in all lowercase.

In the case where the User Agent request includes multiple HTTP header fields with the same field-name, it is RECOMMENDED that the uCDN combine these different HTTP headers into a single value according to Section 3.2.2 of [RFC7230]. However, because of the plurality of already defined HTTP header fields and the inconsistency of some of these header fields concerning the combination mechanism defined in RFC 7230, the uCDN MAY have to deviate from using the combination mechanism where appropriate. For example, it might only send the contents of the first occurrence of the HTTP Headers instead.

An example RI request (uCDN->dCDN) for HTTP-based redirection is as follows:

```
POST /dcdn/rrri HTTP/1.1
Host: rr1.dcdn.example.net
Content-Type: application/cdni; ptype=redirection-request
Accept: application/cdni; ptype=redirection-response

{
  "http": {
    "c-ip": "198.51.100.1",
    "cs-uri": "http://www.example.com",
    "cs-version": "HTTP/1.1",
    "cs-method": "GET"
  },
  "cdn-path": ["AS64496:0"],
  "max-hops": 3
}
```

4.5.2. HTTP Redirection Responses

For a successful HTTP-based redirection, the dCDN needs to return one of the following to the uCDN in the RI response:

- A URI pointing to an RT that is the selected dCDN surrogate(s) (if the dCDN is performing HTTP-based redirection directly to a surrogate); or

- A URI pointing to an RT that is a Request Router (if the dCDN will perform request redirection itself).

The preceding information is encoded as a set of key:value pairs within the http dictionary as follows:

Key	Value	Mandatory	Description
sc-status	Integer	Yes	The status-code part of the status-line as defined in Section 3.1.2 of [RFC7230] to return to the UA (usually set to 302).
sc-version	String	Yes	The HTTP-version part of the status-line as defined in Section 3.1.2 of [RFC7230] to return to the UA.
sc-reason	String	Yes	The reason-phrase part of the status-line as defined in Section 3.1.2 of [RFC7230] to return to the UA.
cs-uri	String	Yes	The URI requested by the UA/client.
sc-(location)	String	Yes	The contents of the Location header to return to the UA (i.e., a URI pointing to the RT(s)).
sc-(<headername>)	String	No	The field-value of the HTTP header field named <HeaderName> to return to the UA. For example, sc-(expires) would contain the value of the HTTP Expires header.

Table 5

Note: The sc-(location) key in the preceding table is an example of sc-(<headername>) that has been called out separately as its presence is mandatory in RI responses.

A successful RI response for HTTP-based redirection MUST include an http dictionary and MAY include an error dictionary (see Section 4.7). An unsuccessful RI response for HTTP-based redirection MUST include an error dictionary. If an http dictionary is included in the RI response, it MUST include at least the following keys: sc-status, sc-version, sc-reason, cs-uri, and sc-(location).

The http dictionary of an RI response MUST contain a maximum of one sc-(<headername>) key for each unique header field-name (HTTP header field). <headername> MUST be identical to the equivalent HTTP header field-name encoded in all lowercase.

The uCDN MAY decide to not return, override, or alter any or all of the HTTP headers defined by sc-(<headername>) keys before sending the HTTP response to the UA. It should be noted that in some cases, sending the HTTP Headers indicated by the dCDN transparently on to the UA might result in, for the uCDN, undesired behavior. As an example, the dCDN might include sc-(cache-control), sc-(last-modified), and sc-(expires) keys in the http dictionary, through which the dCDN may try to influence the cacheability of the response by the UA. If the uCDN would pass these HTTP headers on to the UA, this could mean that further requests from the uCDN would go directly to the dCDN, bypassing the uCDN and any logging it may perform on incoming requests. Therefore, the uCDN is recommended to carefully consider which HTTP headers to pass on, and which to either override or not pass on at all.

An example of a successful RI response (dCDN->uCDN) for HTTP-based redirection is a follows:

```
HTTP/1.1 200 OK
Date: Mon, 06 Aug 2012 18:41:38 GMT
Content-Type: application/cdni; ptype=redirection-response

{
  "http": {
    "sc-status": 302,
    "sc-version": "HTTP/1.1",
    "sc-reason": "Found",
    "cs-uri": "http://www.example.com"
    "sc-(location)":
      "http://sur1.dcdn.example/ucdn/example.com",
  }
}
```

4.6. Cacheability and Scope of Responses

RI responses may be cacheable. As long as a cached RI response is not stale according to standard HTTP Cache-Control or other applicable mechanisms, it may be reused by the uCDN in response to User Agent requests without sending another RI request to the dCDN.

An RI response MUST NOT be reused unless the request from the User Agent would generate an identical RI request to the dCDN as the one that resulted in the cached RI response (except for the c-ip field provided that the User Agent's c-ip is covered by the scope in the original RI response, as elaborated upon below).

Additionally, although RI requests only encode a single User Agent request to be redirected, there may be cases where a dCDN wishes to indicate to the uCDN that the RI response can be reused for other User Agent requests without the uCDN having to make another request via the RI. For example, a dCDN may know that it will always select the same surrogates for a given set of User Agent IP addresses and in order to reduce request volume across the RI or to remove the additional latency associated with an RI request, the dCDN may wish to indicate that set of User Agent IP addresses to the uCDN in the initial RI response. This is achieved by including an optional scope dictionary in the RI response.

Scope is encoded as a set of key:value pairs within the scope dictionary as follows:

Key	Value	Mandatory	Description
iprange	List of String	No	A List of IP subnets in CIDR notation that this RI response can be reused for, provided the RI response is still considered fresh.

Table 6

If a uCDN has multiple cached responses with overlapping scopes and a UA request comes in for which the User Agent's IP matches with the IP subnets in multiple of these cached responses, the uCDN SHOULD use the most recent cached response when determining the appropriate RI response to use.

The following is an example of a DNS redirection response from Section 4.4.2 that is cacheable by the uCDN for 30 seconds and can be returned to any User Agent with an IPv4 address in 198.51.100.0/24.

```
HTTP/1.1 200 OK
Date: Mon, 06 Aug 2012 18:41:38 GMT
Content-Type: application/cdni; ptype=redirection-response
Cache-Control: public, max-age=30

{
  "dns" : {
    "rcode" : 0,
    "name" : "www.example.com",
    "a" : ["203.0.113.200", "203.0.113.201"],
    "aaaa" : ["2001:DB8::C8", "2001:DB8::C9"],
    "ttl" : 60
  }
  "scope" : {
    "iprange" : ["198.51.100.0/24"]
  }
}
```

The following is an example of an HTTP redirection response from Section 4.5.2 that is cacheable by the uCDN for 60 seconds and can be returned to any User Agent with an IPv4 address in 198.51.100.0/24.

Note: The response to the UA is only valid for 30 seconds, whereas the uCDN can cache the RI response for 60 seconds.

```
HTTP/1.1 200 OK
Date: Mon, 06 Aug 2012 18:41:38 GMT
Content-Type: application/cdni; ptype=redirection-response
Cache-Control: public, max-age=60

{
  "http": {
    "sc-status": 302,
    "cs-uri": "http://www.example.com"
    "sc-(location)":
      "http://sur1.dcdn.example/ucdn/example.com",
    "sc-(cache-control)" : "public, max-age=30"
  }
  "scope" : {
    "iprange" : ["198.51.100.0/24"]
  }
}
```

4.7. Error Responses

From a uCDN perspective, there are two types of errors that can be the result of the transmission of an RI request to a dCDN:

1. An HTTP protocol error signaled via an HTTP status code, indicating a problem with the reception or parsing of the RI request or the generation of the RI response by the dCDN, and

2. An RI-level error specified in an RI response message.

This section deals with the latter type. The former type is outside the scope of this document.

There are numerous reasons for a dCDN to be unable to return an affirmative RI response to a uCDN. Reasons may include both dCDN internal issues such as capacity problems, as well as reasons outside the influence of the dCDN, such as a malformed RI request. To aid with diagnosing the cause of errors, RI responses SHOULD include an error dictionary to provide additional information to the uCDN as to the reason/cause of the error. The intention behind the error dictionary is to aid with either manual or automatic diagnosis of issues. The resolution of such issues is outside the scope of this document; this document does not specify any consequent actions a uCDN should take upon receiving a particular error-code.

Error information (if present) is encoded as a set of key:value pairs within a JSON-encoded error dictionary as follows:

Key	Value	Mandatory	Description
error-code	Integer	Yes	A three-digit numeric code defined by the server to indicate the error(s) that occurred.
reason	String	No	A string providing further information related to the error.

Table 7

The first digit of the error-code defines the class of error. There are 5 classes of errors distinguished by the first digit of the error-code:

1xx: Informational (no error): The response should not be considered an error by the uCDN, which may proceed by redirecting the UA according to the values in the RI response. The error-code and accompanying description may be used for informational purposes, e.g., for logging.

2xx: Reserved.

3xx: Reserved.

4xx: uCDN error: The dCDN cannot or will not process the request due to something that is perceived to be a uCDN error, for example, the RI request could not be parsed successfully by the dCDN. The last two-digits may be used to more specifically indicate the source of the problem.

5xx: dCDN error: Indicates that the dCDN is aware that it has erred or is incapable of satisfying the RI request for some reason, for example, the dCDN was able to parse the RI request but encountered an error for some reason. Examples include the dCDN not being able to retrieve the associated metadata or the dCDN being out of capacity.

The following error-codes are defined and maintained by IANA (see Section 6).

Error-codes with a "Reason" of "<reason>" do not have a defined value for their 'reason'-key. Depending on the error-code semantics, the value of this field may be determined dynamically.

Code	Reason	Description
100	<reason> (see Description)	Generic informational error-code meant for carrying a human-readable string
400	<reason> (see Description)	Generic error-code for uCDN errors where the dCDN cannot or will not process the request due to something that is perceived to be a uCDN error. The reason field may be used to provide more details about the source of the error.

500	<reason> (see Description)	Generic error-code for dCDN errors where the dCDN is aware that it has erred or is incapable of satisfying the RI request for some reason. The reason field may be used to provide more details about the source of the error.
501	Unable to retrieve metadata	The dCDN is unable to retrieve the metadata associated with the content requested by the UA. This may indicate a configuration error or that the content requested by the UA does not exist.
502	Loop detected	The dCDN detected a redirection loop (see Section 4.8).
503	Maximum hops exceeded	The dCDN detected the maximum number of redirection hops exceeding max-hops (see Section 4.8).
504	Out of capacity	The dCDN does not currently have sufficient capacity to handle the UA request.
505	Delivery protocol not supported	The dCDN does not support the (set of) delivery protocols indicated in the CDNI Metadata of the content requested by the UA.
506	Redirection protocol not supported	The dCDN does not support the requested redirection protocol. This error-code is also used when the RI request has the dns-only flag set to True and the dCDN is not supported or is not prepared to return an RT of a surrogate directly.

Table 8

The following is an example of an unsuccessful RI response (dCDN->uCDN) for a DNS-based User Agent request:

```
HTTP/1.1 500 Internal Server Error
Date: Mon, 06 Aug 2012 18:41:38 GMT
Content-Type: application/cdni; ptype=redirection-response
Cache-Control: private, no-cache

{
  "error" : {
    "error-code" : 504,
    "description" : "Out of capacity"
  }
}
```

The following is an example of a successful RI response (dCDN->uCDN) for an HTTP-based User Agent request containing an error dictionary for informational purposes:

```
HTTP/1.1 200 OK
Date: Mon, 06 Aug 2012 18:41:38 GMT
Content-Type: application/cdni; ptype=redirection-response
Cache-Control: private, no-cache

{
   "http": {
    "sc-status": 302,
    "sc-version": "HTTP/1.1",
    "sc-reason": "Found",
    "cs-uri": "http://www.example.com"
    "sc-(location)":
      "http://sur1.dcdn.example/ucdn/example.com",
   },
   "error" : {
    "error-code" : 100,
    "description" :
      "This is a human-readable message meant for debugging purposes"
  }
}
```

4.8. Loop Detection and Prevention

In order to prevent and detect RI request loops, each CDN MUST insert its CDN Provider ID into the cdn-path key of every RI request it originates or cascades. When receiving RI requests, a dCDN MUST check the cdn-path and reject any RI requests that already contain the dCDN's Provider ID in the cdn-path. Transit CDNs MUST NOT propagate to any downstream CDNs if the number of CDN Provider IDs in cdn-path (before adding its own Provider ID) is equal to or greater than max-hops.

The CDN Provider ID uniquely identifies each CDN Provider during the course of request routing redirection. It consists of the characters AS followed by the CDN Provider's AS number, then a colon (':') and an additional qualifier that is used to guarantee uniqueness in case a particular AS has multiple independent CDNs deployed; for example, "AS64496:0".

If a dCDN receives an RI request whose cdn-path already contains that dCDN's Provider ID, the dCDN MUST send an RI error response that SHOULD include an error-code of 502.

If a dCDN receives an RI request where the number of CDN Provider IDs in cdn-path is greater than max-hops, the dCDN MUST send an RI error response that SHOULD include an error-code of 503.

It should be noted that the loop detection and prevention mechanisms described above only cover preventing and detecting loops within the RI itself. Besides loops within the RI itself, there is also the possibility of loops in the data plane; for example, if the IP address(es) or URI(s) returned in RI responses do not resolve directly to a surrogate in the final dCDN, there is the possibility that a User Agent may be continuously redirected through a loop of CDNs. The specification of solutions to address data-plane request redirection loops between CDNs is outside of the scope of this document.

5. Security Considerations

Information passed over the RI could be considered personal or sensitive, for example, RI requests contain parts of a User Agent's original request and RI responses reveal information about the dCDN's policy for which surrogates should serve which content/user locations.

The RI interface also provides a mechanism whereby a uCDN could probe a dCDN and infer the dCDN's edge topology by making repeated RI requests for different content and/or UA IP addresses and correlating the responses from the dCDN. Additionally, the ability for a dCDN to indicate that an RI response applies more widely than the original request (via the scope dictionary) may significantly reduce the number of RI requests required to probe and infer the dCDN's edge topology.

The same information could be obtained in the absence of the RI interface, but it could be more difficult to gather as it would require a distributed set of machines with a range of different IP addresses, each making requests directly to the dCDN. However, the RI facilitates easier collection of such information as it enables a single client to query the dCDN for a redirection/surrogate selection on behalf of any UA IP address.

5.1. Authentication, Authorization, Confidentiality, and Integrity Protection

An implementation of the CDNI Redirection interface MUST support TLS transport as per [RFC2818] and [RFC7230]. The use of TLS for transport of the CDNI Redirection interface messages allows the dCDN and uCDN to authenticate each other. Once they have mutually authenticated each other, it allows:

- The dCDN and uCDN to authorize each other (to ensure they are transmitting/receiving CDNI Redirection messages to/from an authorized CDN);
- CDNI Redirection interface messages to be transmitted with confidentiality; and
- The integrity of the CDNI Redirection interface messages to be protected during the exchange.

In an environment where any such protection is required, mutually authenticated encrypted transport MUST be used to ensure confidentiality of the redirection information; to do so, TLS MUST be used (including authentication of the remote end) by the server side (dCDN) and the client side (uCDN) of the CDNI Redirection interface.

When TLS is used, the general TLS usage guidance in [RFC7525] MUST be followed.

5.2. Privacy

Information passed over the RI ought to be considered personal and sensitive. In particular, parts of a User Agent's original request, most notably the UA's IP address and requested URI, are transmitted over the RI to the dCDN. The use of mutually authenticated TLS, as described in the previous section, prevents any other party than the authorized dCDN from gaining access to this information.

Regardless of whether the uCDN and dCDN use the RI, a successful redirect from a uCDN to a dCDN will make that dCDN aware of the UA's IP address. As such, the fact that this information is transmitted across the RI does not allow the dCDN to learn new information. On the other hand, if a uCDN uses the RI to check with multiple candidate dCDNs, those candidates that do not end up getting redirected to do obtain information regarding end-user IP addresses and requested URIs that they would not have if the RI not been used.

While it is technically possible to mask some information in the RI Request, such as the last bits of the UA IP address, it is important to note that this will reduce the effectiveness of the RI in certain cases. CDN deployments need to strike a balance between end-user privacy and the features impacted by such masking. This balance is likely to vary from one deployment to another. As an example, when the UA and its DNS resolver is behind a Carrier-grade NAT, and the RI is used to find an appropriate delivery node behind the same NAT, the full IP address might be necessary. Another potential issue when using IP anonymization is that it is no longer possible to correlate an RI Request with a subsequent UA request.

6. IANA Considerations

6.1. CDNI Payload Type Parameter Registrations

IANA has registered the following two new Payload Types in the "Content Delivery Network Interconnection (CDNI) Parameters" registry for use with the application/cdni MIME media type.

Payload Type	Specification
redirection-request	RFC 7975
redirection-response	RFC 7975

Table 9

6.1.1. CDNI RI Redirection Request Payload Type

Purpose: The purpose of this payload type is to distinguish RI request messages.

Interface: RI

Encoding: See Section 4.4.1 and Section 4.5.1

6.1.2. CDNI RI Redirection Response Payload Type

Purpose: The purpose of this payload type is to distinguish RI response messages.

Interface: RI

Encoding: See Section 4.4.2 and Section 4.5.2

6.2. RI Error Response Registry

IANA has created a new "CDNI RI Error response code" subregistry within the "Content Delivery Network Interconnection (CDNI) Parameters" registry. The "CDNI RI Error response code" namespace defines the valid values for the error-code key in RI error responses. The CDNI RI Error response code MUST be a three-digit integer.

Additions to the "RI Error response registry" will be made via "Specification Required" as defined in [RFC5226].

The Designated Expert will verify that new error-code registrations do not duplicate existing error-code definitions (in name or functionality), ensure that the new error-code is in accordance with the error classes defined in Section 4.7 of this document, prevent gratuitous additions to the namespace, and prevent any additions to the namespace that would impair the interoperability of CDNI implementations.

New registrations are required to provide the following information:

Code: A three-digit numeric error-code, in accordance with the error classes defined in Section 4.7 of this document.

Reason: A string that provides further information related to the error that will be included in the JSON error dictionary with the 'reason'-key. Depending on the error-code semantics, the value of this field may be determined dynamically. In that case, the registration should set this value to '<reason>' and define its semantics in the description field.

Description: A brief description of the error-code semantics.

Specification: Reference to the specification that defines the error-code in more detail.

The entries in Table 8 are registered by this document, with the value of the 'Specification' field set to RFC 7975 (this document).

7. References

7.1. 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, <http://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, <http://www.rfc-editor.org/info/rfc3986>.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, DOI 10.17487/RFC4291, February 2006, <http://www.rfc-editor.org/info/rfc4291>.

[RFC5952] Kawamura, S. and M. Kawashima, "A Recommendation for IPv6 Address Text Representation", RFC 5952, DOI 10.17487/RFC5952, August 2010, <http://www.rfc-editor.org/info/rfc5952>.

[RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <http://www.rfc-editor.org/info/rfc7230>.

[RFC7159] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", RFC 7159, DOI 10.17487/RFC7159, March 2014, <http://www.rfc-editor.org/info/rfc7159>.

[RFC6895] Eastlake 3rd, D., "Domain Name System (DNS) IANA Considerations", BCP 42, RFC 6895, DOI 10.17487/RFC6895, April 2013, <http://www.rfc-editor.org/info/rfc6895>.

[RFC7493] Bray, T., Ed., "The I-JSON Message Format", RFC 7493, DOI 10.17487/RFC7493, March 2015, <http://www.rfc-editor.org/info/rfc7493>.

[RFC7525] Sheffer, Y., Holz, R., and P. Saint-Andre, "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", BCP 195, RFC 7525, DOI 10.17487/RFC7525, May 2015, <http://www.rfc-editor.org/info/rfc7525>.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, DOI 10.17487/RFC5226, May 2008, <http://www.rfc-editor.org/info/rfc5226>.

[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, <http://www.rfc-editor.org/info/rfc6707>.

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, DOI 10.17487/RFC2818, May 2000, <http://www.rfc-editor.org/info/rfc2818>.

[RFC5890] Klensin, J., "Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework", RFC 5890, DOI 10.17487/RFC5890, August 2010, <http://www.rfc-editor.org/info/rfc5890>.

[RTMP] Adobe Systems Incorporated, "Real-Time Messaging Protocol (RTMP) specification", December 2012, <http://www.adobe.com/go/spec_rtmp>.

7.2. Informative References

[RFC7337] Leung, K., Ed. and Y. Lee, Ed., "Content Distribution Network Interconnection (CDNI) Requirements", RFC 7337, DOI 10.17487/RFC7337, August 2014, <http://www.rfc-editor.org/info/rfc7337>.

[RFC7336] Peterson, L., Davie, B., and R. van Brandenburg, Ed., "Framework for Content Distribution Network Interconnection (CDNI)", RFC 7336, DOI 10.17487/RFC7336, August 2014, <http://www.rfc-editor.org/info/rfc7336>.

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

Acknowledgements

The authors would like to thank Taesang Choi, Francois Le Faucheur, Matt Miller, Scott Wainner, and Kevin J. Ma for their valuable comments and input to this document.

Contributors

The following persons have participated as co-authors to this document:

Wang Danhua
Huawei
Email: wangdanhua@huawei.com

He Xiaoyan
Huawei
Email: hexiaoyan@huawei.com

Ge Chen
China Telecom
Email: cheng@gsta.com

Ni Wei
China Mobile
Email: niwei@chinamobile.com

Yunfei Zhang
Email: hishigh@gmail.com

Spencer Dawkins
Huawei
Email: spencer@wonderhamster.org

Authors' Addresses

Ben Niven-Jenkins (editor)
Nokia
3 Ely Road
Milton, Cambridge CB24 6DD
United Kingdom

Email: ben.niven-jenkins@nokia.com

Ray van Brandenburg (editor)
TNO
Anna van Buerenplein 1
The Hague 2595DA
The Netherlands

Phone: +31-88-866-7000
Email: ray.vanbrandenburg@tno.nl