Network Working Group B. Niven-Jenkins, Ed.
Internet-Draft Velocix (Alcatel-Lucent)
Intended status: Standards Track R. van Brandenburg, Ed.
Expires: October 18, 2014 TNO
April 16, 2014

Request Routing Redirection Interface for CDN Interconnection
draft-ietf-cdni-redirection-02

Abstract

The Request Routing Interface comprises of (1) the asynchronous advertisement of footprint and capabilities by a downstream CDN that allows a 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

Copyright Notice

Table of Contents

1. Introduction

A Content Delivery Network (CDN) is a system built on an existing IP network which 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 [I-D.ietf-cdni-framework] comprises of:

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

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 (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 the routing decision of a CDN 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 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 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): A Redirection Target is the endpoint to which the user agent is redirected. In CDNI, a 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 downstream CDN or a surrogate in a downstream CDN, etc.

3. Interface function and operation overview

The CDNI request routing/Redirection Interface (RI) is one of the main building blocks required in order to interconnect CDNs. The main function of the Redirection Interface 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 [I-D.ietf-cdni-requirements].

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 User Agent and uCDN request router. The dCDN has control over the Redirection Target it provides and depending on the returned Redirection Target, the User Agent's request may be redirected to:

- o The final Surrogate, which may be in the dCDN or another dCDN (if dCDN delegates the delivery to another CDN).
- o A request router (in dCDN or another CDN) that will be using a redirection protocol (DNS or HTTP) which may or may not be the same as original redirection protocol.

The Redirection Interface operates between the Request Routing systems of a pair of interconnected CDNs. To enable communication over the Redirection Interface, the two interconnected CDNs need to know the end point (URI) in the other CDN to query. For example, an Upstream CDN needs to know the URI (end point) in a Downstream CDN to send its CDNI request routing queries to.

The Redirection Interface URI may be statically pre-configured, 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.

CDNI solutions must support both of the request routing mechanisms illustrated in section 2.1 of [I-D.ietf-cdni-framework], namely Iterative Request Redirection and Recursive Request Redirection. However, the Iterative Request Redirection method does not invoke any interaction over the Redirection Interface between interconnected CDNs. Therefore, the Redirection Interface is only relevant in the case of Recursive Request Redirection and so this document will not discuss Iterative Request Redirection further.

In the case of Recursive Request Redirection, in order to perform redirection of a request received from a User Agent, the Upstream CDN queries the Downstream CDN so that the Downstream CDN can select and provide a Redirection Target. In cases where a uCDN has a choice of dCDNs it is down 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 reject 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 Upstream CDN->User Agent redirection protocols addressed in this draft are: DNS redirection and HTTP redirection. Other types of application level redirection will not be discussed further in this draft. However the Redirection Interface is designed to be extensible and could be extended to support additional application level redirection protocols.

Also, according to the CDNI generic and request routing interface requirements, the CDNI solution shall support mechanisms to prevent and detect RI request loops. To meet such requirements, this document defines a loop prevention and detection mechanism as part of the Redirection Interface.

4. HTTP based RESTful interface for the Redirection Interface

This document defines a simple RESTful interface for the Redirection Interface based on HTTP [RFC2616], where the attributes of a User Agent's requests are encapsulated along with any other data that can aid the downstream CDN in processing the requests. The RI response encapsulates the attributes of the RT(s) that the upstream CDN should return to the User Agent (if it decides to utilize the Downstream CDN for delivery) along with the policy for how the response can be reused.

The same RESTful interface is used for both DNS and HTTP redirection of User Agent's requests, 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 (RESTful) 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 downstream CDN to make a request routing decision, avoids having to try and encapsulate or proxy DNS/HTTP/RTMP /etc requests and 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).

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

```
User Agent                CDN B RR                  CDN A RR
    |UA Request (DNS or HTTP) |                         |
    |-------------------------------------------------->| (1)
    |                         |                         |
    |                         |HTTP POST to CDN B's RI  |
    |                         |URI encapsulating UA     |
    |                         |request attributes       |
    |                         |<------------------------| (2)
    |                         |                         |
    |                         |HTTP Response with body  |
    |                         |containing attributes of |
    |                         |protocol specific        |
    |                         |response to return to UA |
    |                         |------------------------>| (3)
    |                         |                         |
    |           Protocol specific response (redirection)|
    |<--------------------------------------------------| (4)
    |                         |                         |
```

Figure 1: Generic Recursive Request Redirection message flow

1. The User Agent sends its request, either DNS request or HTTP request, to CDN A. The Request Routing System of CDN A processes the request and, through local policy, it recognizes that the request is best served by another CDN, specifically CDN B (or that CDN B is 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 request and assuming the request is well formed, etc. 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 & responses

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

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

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

To assist the routing decision of a Downstream CDN, the Upstream CDN shall convey as much information as possible to the Downstream CDN, for example the URI of the requested content and the User Agent's location information, when those are known by the uCDN Request Routing system.

In order for the Downstream CDN 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 Downstream CDN to be able to retrieve the require 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 of how to redirect the User Agent, the Downstream CDN may wish to return additional policy to the Upstream CDN to help the Upstream CDN with future RI requests. For example the Downstream CDN may wish to return a policy that expresses "this response can be reused without requiring a RI request for 60 seconds provided the User Agent's IP address is in the range 192.0.2.0 - 192.0.2.255".

These additional policies split into two basic categories:

- An indication of the cacheability of the response carried in the HTTP response headers (to reduce the number of subsequent RI requests the uCDN needs to make).

- The scope of the response (if it is cacheable) carried within the body of the HTTP 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 & responses

The body of RI requests and responses is a JSON object containing a dictionary of keys. Keys MUST always be encoded in lowercase. Unknown keys MUST be ignored but the response MUST NOT be considered invalid unless the syntax of the request is invalid.

The following keys are defined:

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 a 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 the CDN Provider IDs of previous CDNs this RI request has passed through. When cascading a RI request the transit CDN appends its own CDN Provider ID to the list in cdn-path so that downstream CDNs can detect loops in the RI request chain. Transit CDNs should check the cdn-path and not cascade the RI request to downstream CDNs that are already listed in cdn-path. The cdn-path MUST be reflected back in RI responses.
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 crudely constrain the latency of the request routing chain.

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.

[[Editor's note: Need some text on minimum attributes to be able to (at least parse) - e.g. A/AAAA/CNAME, etc)]]

Note: All implementations MUST support IPv4 addresses encoded as specified by the 'IPv4address' rule in Section 3.2.2 of [RFC3986] and 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 SHOULD use a MIME Media Type of application/cdni.redirectionrequest

RI responses SHOULD use a MIME Media Type of application/cdni.redirectionresponse.

4.4. DNS redirection

The following sections provide more 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 Upstream CDN.
- The type of DNS query made (A, AAAA, RCODEs, etc.).
- 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 Upstream CDN).

The information above 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 (A, AAAA, etc.).
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 of the UA in CIDR format.
dns-only	Boolean	No	If True then dCDN MUST only use DNS redirection to a surrogate and MUST include the dns-only property set to True on any cascaded RI requests. Defaults to False.

A RI request for DNS-based redirection MUST include a dns dictionary. This dns dictionary MUST contain the following keys: resolver-ip, qtype, qclass, qname and the value of each MUST be the value of the appropriate part of the User Agent's DNS query/request.

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

```
POST /dcdn/ri HTTP/1.1
Host: rr1.dcdn.example.net
Accept: application/vnd.cdni.ri.response+json

{
  "dns" : {
    "resolver-ip" : "192.0.2.1",
    "c-subnet" : "198.51.100.0/24",
    "qtype" : "A",
    "qclass" : "IN",
    "qname" : "www.example.com"
  },
  "cdn-path": ["AS65551:0"],
  "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:

- o The IP address(es) of (or a CNAME to) the RT (if the dCDN is performing DNS based redirection); or
- o The IP address(es) of (or a CNAME to) a RT which is a Request Router (if the dCDN is performing HTTP based redirection).

The information above 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.
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 of DNS response. Default is 0.

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 at least one of the following keys: a, aaaa, cname. The dns dictionary MAY include both 'a' and 'aaaa' keys.

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/vnd.cdni.ri.response+json

{
  "dns" : {
    "rcode" : 0,
    "name" : "www.example.com",
    "a" : ["192.0.2.200", "192.0.2.201"],
    "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 a set of two RT FQDNs.

```
HTTP/1.1 200 OK
Date: Mon, 06 Aug 2012 18:41:38 GMT
Content-Type: application/vnd.cdni.ri.response+json

{
  "dns" : {
    "rcode" : 0,
    "name" : "www.example.com",
    "cname" : ["rr1.dcdn.example",
               "rr2.dcdn.example"],
    "ttl" : 20
  }
}
```

4.5. HTTP Redirection

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

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:

- o The IP address of the User Agent.

- o The URL requested by the User Agent.

The uCDN may also decide to pass additional information to the dCDN in the RI request, such as:

- o The HTTP method or version number of the User Agent's request.
- o The presence and value of particular HTTP headers included in the User Agent request.

The information above 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 URI requested by the UA.
cs-method	String	Yes	The Method part of the Request-Line as defined in Section 5.1 of [RFC2616].
cs-version	String	Yes	The HTTP-Version part of the Request-Line as defined in Section 5.1 of [RFC2616].
cs(<HeaderName>)	String	No	The contents of the HTTP header named <HeaderName> as a string, for example cs(Cookie) would contain the content of the HTTP Cookie: header.

A 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 and the value of each MUST be the value of the appropriate part of the User Agent's DNS query/ request.

In order to be a valid JSON object, the http dictionary of a RI request MUST contain a maximum of one cs(<HeaderName>) key for each unique HeaderName (see section 4 of [RFC7159]). In the case where the User Agent request includes multiple message-header fields with the same field-name, it is up to the uCDN to determine how to handle this. One option would be to only send the contents of the first occurence of that HTTP Header in the User Agent request. Another would be to combine the different HTTP Headers into a single value according to Section 4.2 of [RFC2616].

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

```
POST /dcdn/rrri HTTP/1.1
Host: rr1.dcdn.example.net
Accept: application/vnd.cdni.rrri.response+json

{
  "http": {
    "c-ip": "198.51.100.1",
    "cs-uri": "http://www.example.com",
    "cs-version": "HTTP/1.1",
    "cs-method": "GET"
  },
  "cdn-path": ["AS65551:0"],
  "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:

- o A URL pointing to the selected RT (if the dCDN is redirecting the User Agent directly to a surrogate); or

- o A URL pointing to a RT which is a Request Router (if the dCDN is not redirecting the User Agent directly to a surrogate).

The information above 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 6.1 of [RFC2616] 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 6.1 of [RFC2616] to return to the UA.
sc-reason	String	Yes	The Reason-Phrase part of the Status-Line as defined in Section 6.1 of [RFC2616] 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(Cache-Control)	String	No	The contents of the Cache-Control header to return to the UA.
sc(<HeaderName>)	String	No	The contents of the HTTP header named <HeaderName> to return to the UA. For example, sc(Cookie) would contain the content of the HTTP Cookie: header.

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, sc(Location).

In order to be a valid JSON object, the http dictionary of a RI response MUST contain a maximum of one sc(<HeaderName>) key for each unique HeaderName.

The uCDN MAY decide to not return, override or alter some 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 behaviour. As an example, the dCDN might include sc(LastModified) 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. The uCDN is therefore 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:

```
HTTP/1.1 200 OK
Date: Mon, 06 Aug 2012 18:41:38 GMT
Content-Type: application/vnd.cdni.ri.response+json

{
  "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",
    "sc(Cache-Control)" : "public, max-age=30"
  }
}
```

4.6. Cacheability and scope of responses

RI responses may be cacheable and may be reused by the uCDN in response to User Agent requests without the uCDN issuing another RI request to the dCDN if the RI response is considered cacheable & not stale according to the standard HTTP Cache-Control, etc mechanisms.

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 the User Agent's c-ip is covered by the scope in the original RI response).

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.

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 approriate RI response to use.

The following is an example of 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 198.51.100.0/16.

```
HTTP/1.1 200 OK
Date: Mon, 06 Aug 2012 18:41:38 GMT
Content-Type: application/vnd.cdni.ri.response+json
Cache-Control: public, max-age=30

{
  "dns" : {
    "rcode" : 0,
    "name" : "www.example.com",
    "a" : ["192.0.2.200", "192.0.2.201"],
    "aaaa" : ["2001:DB8::C8", "2001:DB8::C9"],
    "cname" : ["rr1.dcdn.example",
               "rr2.dcdn.example"],
    "ttl" : 60
  }
  "scope" : {
    "iprange" : ["198.51.100.0/16"]
  }
}
```

Example of 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 198.51.100.0/16.

```
HTTP/1.1 200 OK
Date: Mon, 06 Aug 2012 18:41:38 GMT
Content-Type: application/vnd.cdni.ri.response+json
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" : ["198.51.100.0/16"]
  }
}
```

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: An HTTP protocol error signaled via an HTTP error code, indicating a problem with the reception of the RI request at the dCDN, and a 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 not being able 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 may include an optional 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 diagnostics of issues. The resolution of such issues is outside the scope of this document and this document therefore does not specify the consequent actions a uCDN should take upon receiving a particular error code.

[[Editor's note: We've tried to keep error specification light weight & provide the hooks needed to help with debugging without trying to be overly prescriptive over how it gets used as we'd like to avoid the rat hole of specifying every possible error condition and consequent actions.]]

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	No	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.

The first digit of the error-code defines the class of error. Thera are 4 values for the first digit:

1xx: No error (informational): 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: Malformed RI request: The RI request could not be parsed succesfully by the dCDN. The last two-digits may be used to more specifically indicate the source of the problem.

3xx: uCDN or path error: The dCDN was able to parse the RI request but encountered an error due to reasons outside the influence of the dCDN. Examples include the dCDN not being able to retrieve the associated metadata or the dCDN detecting a redirection loop.

4xx: Request denied by dCDN: The dCDN was able to parse the RI request but is currently not able to deliver the content on behalf of the uCDN due to internal reasons. Examples include the dCDN being out of capacity or the dCDN not supporting the delivery method required by the CDNI Metadata.

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

Code	Reason	Description
100	<reason> (see Description)	Generic informational error-code meant for carrying a human-readable string
200	<reason> (see Description)	Generic malformed RI request error-code. The reason field may be used to provide more details about the source of the error.
300	<reason> (see Description)	Generic uCDN or path error. The reason field may be used to provide more details about the source of the error.
301	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 the content requested by the UA not existing.
302	Loop detected	The dCDN detected a redirection loop (see Section 4.8).
303	Maximum hops exceeded	The dCDN detected the maximum number of redirection hops exceeding max-hops (see Section 4.8).
400	<reason> (see Description)	Generic request denied error. The reason field may be used to provide more details about the source of the error.
401	Out of capacity	The dCDN currently doesn't have sufficient capacity to handle the UA request.
402	Delivery protocol not supported	The dCDN does not support the (set of) delivery protocols indicated in the CDNI Metadata of the content requested content by the UA.

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

```
HTTP/1.1 200 OK
Date: Mon, 06 Aug 2012 18:41:38 GMT
Content-Type: application/vnd.cdni.rrri.error+json
Cache-Control: private, no-cache

{
  "error" : {
    "code" : 400,
    "description" : "Out of capacity"
  }
}
```

The following is an example of a successful RI response (dCDN->uCDN) for a 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/vnd.cdni.rrri.error+json
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",
    "sc(Cache-Control)" : "public, max-age=30"
   },
   "error" : {
    "code" : 100,
    "description" :
       "This is a human-readable message meant for debugging purposes"
  }
}
```


4.8. Loop detection & 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 SHOULD check the cdn-path and reject any RI requests which already contain the downstream CDN's Provider ID in the cdn-path. Transit CDNs SHOULD check the cdn-path and not cascade the RI request to downstream CDNs that are already listed in cdn-path. CDNs MUST NOT propagate to any downstream CDNs if the number of CDN Provider IDs in cdn-path (including the CDN's 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 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 "AS65551:0".

If a downstream CDN receives a RI request whose cdn-path already contains that downstream CDN's Provider ID the downstream CDN SHOULD send a RI response with an error code of 302.

If a downstream CDN receives a RI request where the number of CDN Provider IDs in cdn-path (including the CDN's own Provider ID) is equal to or greater than max-hops, the downstream CDN SHOULD send a RI response with an error code of 303.

It should be noted that the loop detection & prevention mechanisms described above only cover preventing and detecting loops within the RI itself. As well as loops with 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 continuosly redirected through a loop of CDNs. The specification of solutions to address data plane request redirection loops between CDNs is out 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 a RI response applies more widely that 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.

In order to prevent passive interception of RI messages the RI communications channel should be suitably secured (e.g. use of TLS).

In order to reduce the risk of information leakage to unauthorized parties, RI clients and servers SHOULD use suitable authentication prior to trusting the contents of RI messages.

6. IANA Considerations

[[Editor's Note: TODO: Add error code registry]]

[[Editors' Note: Need to insert some text to register the Media Types we use with IANA?]]

7. Contributors

[RFC Editor Note: Please move the contents of this section to the Authors' Addresses section prior to publication as an RFC.]

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

Wang Danhua, Huawei, Email: wangdanhua@huawei.com

He Xiaoyan, Huawi, 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

8. Acknowledgements

The authors would like to thank Taesang Choi, Francois le Faucheur and Scott Wainner for their valuable comments and input to this document.

9. References

9.1. Normative References

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

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, February 2006.

[RFC5952] Kawamura, S. and M. Kawashima, "A Recommendation for IPv6 Address Text Representation", RFC 5952, August 2010.

[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.

9.2. Informative References

[RFC6707] Niven-Jenkins, B., Le Faucheur, F., and N. Bitar, "Content Distribution Network Interconnection (CDNI) Problem Statement", RFC 6707, September 2012.

[RFC7159] Bray, T., "The JavaScript Object Notation (JSON) Data Interchange Format", March 2014.

[I-D.ietf-cdni-framework] Peterson, L. and B. Davie, "Framework for CDN Interconnection", draft-ietf-cdni-framework-03 (work in progress), February 2013.

[I-D.ietf-cdni-requirements] Leung, K. and Y. Lee, "Content Distribution Network Interconnection (CDNI) Requirements", draft-ietf-cdni-requirements-06 (work in progress), April 2013.

Authors' Addresses

Ben Niven-Jenkins (editor)
Velocix (Alcatel-Lucent)
3 Ely Road
Milton, Cambridge CB24 6DD
UK

Email: ben@velocix.com

Ray van Brandenburg (editor)
TNO
Brassersplein 2
Delft 2612CT
the Netherlands

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