Network Working Group   Danhua. Wang, Ed.
Internet-Draft   Huawei Technologies
Intended status: Standards Track   B. Niven-Jenkins, Ed.
Expires: October 17, 2013   Velocix (Alcatel-Lucent)
                                                          Xiaoyan. He
                                                               Huawei
                                                             Chen. Ge
                                                        China Telecom
                                                              Wei. Ni
                                                         China Mobile
                                                       April 15, 2013

        Request Routing Redirection Interface for CDN Interconnection
                    draft-ietf-cdni-redirection-00

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 term
"Distinguished CDN Domain" defined in [I-D.ietf-cdni-framework] is
also reused in this document.

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

[[Editor's note: Need to factor token authorisation into a future
draft when that work is more stable/mature within the WG.]]

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. If 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)
      |                              |                              |
```
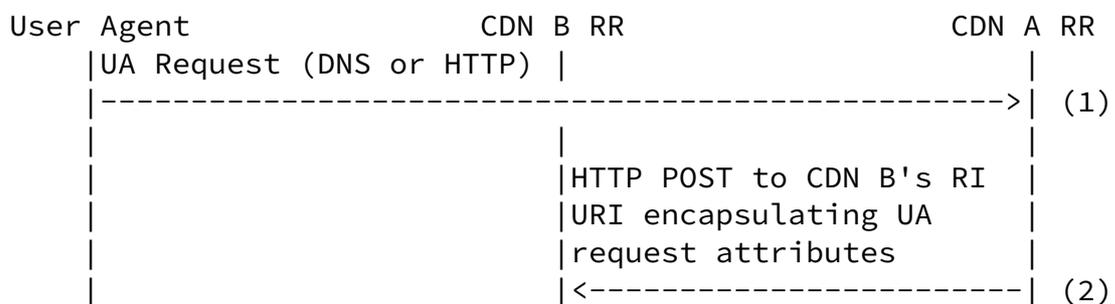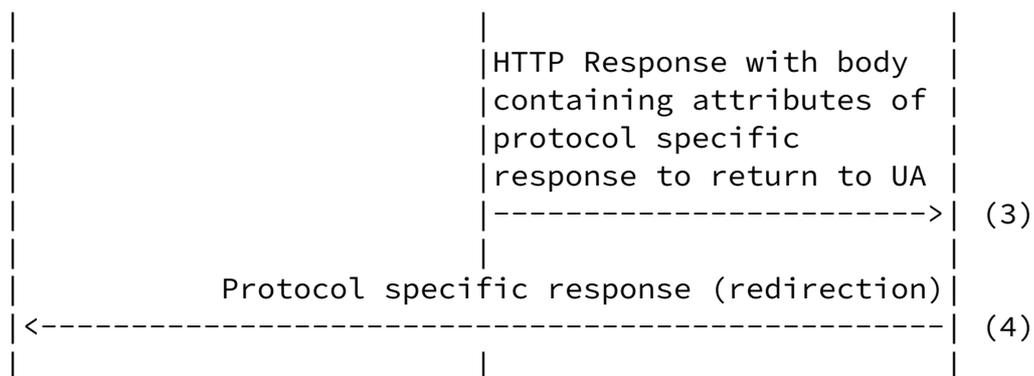
           Figure 1: Generic Recursive Request Redirection message flow

   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:

o 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).

o 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/section specifying the Media Types
   for RI requests/responses]]

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

   [[Editor's note: Need section detailing format/etc for scope and
   error keys]]

   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. DNS redirection

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

4.3.1. DNS Redirection requests

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

   o The IP address of the DNS resolver that made the DNS request to
      the Upstream CDN.

   o The type of DNS query made (A, AAAA, RCODEs, etc.).

o The class of DNS query made (usually IN). [[Editor's Note: Do we
   need to include class or can we always assume it is IN?]]

o The fully qualified domain name for which DNS redirection is being
   requested.

o The IP address or prefix of the User Agent (if known to the
   Upstream CDN, e.g. through draft-vandergaast-edns-client-subnet).

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.

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
       }
```

4.3.2. DNS Redirection responses

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

   o The IP address of (or a CNAME to) the RT (if the dCDN is
      performing DNS based redirection); or

   o The IP address 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	No	Set of IPv4 Addresses of
	String		RT(s).
aaaa	List of	No	Set of IPv6 Addresses of
	String		RT(s).

```
      | cname   | List of    | No         | Set of fully qualified    |
      |         | String     |            | domain names of RT(s).    |
      | ttl     | Integer    | No         | TTL of DNS response.      |
      |         |            |            | Default is 0.             |
      +---------+------------+------------+---------------------------+
```

   Response must contain at least one of a, aaaa, cname.

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

   [[Editor's note: Currently shows both A/AAAA & CNAME in single
   response, need to split to show the different use cases]]

   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"],
       "cname" : ["rr1.dcdn.example",
                  "rr2.dcdn.example"],
       "ttl" : 60
     }
   }

4.4. HTTP Redirection

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

4.4.1. HTTP Redirection requests

   For HTTP based redirection the uCDN MUST 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 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   Additional HTTP header 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/client              |
| cs-uri           | String  | Yes       | The URI requested by the |
|                  |         |           | UA/client.             |
| 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. Two    |
```

```
|                  |         |           | special <HeaderName>s  |
|                  |         |           | are defined: cs(Method) |
|                  |         |           | and cs(HTTP-Version)   |
|                  |         |           | which contain the      |
|                  |         |           | contents of the Method & |
|                  |         |           | HTTP-Version parts of  |
|                  |         |           | the Request-Line as    |
|                  |         |           | defined in Section 5.1 |
|                  |         |           | 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"
  },
  "cdn-path": ["AS65551:0"],
  "max-hops": 3
}
```

4.4.2. HTTP Redirection responses

For 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 of
			the HTTP response to
			return to the UA

			(usually 302).
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.  |
+--------------------+--------+----------+-----------------------+
```

[[Editor's Note: Should we change the format above to align with the cs() format for headers on the RI request and allow the dCDN to signal back any headers it wants in the response as sc(<HeaderName>)? How to handle sc-status in that case - as a "special" header or separate key? Probably need to give some advice on HTTP headers the uCDN may want to override/not pass through, e.g. Server:?]]

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,
    "cs-uri": "http://www.example.com"
    "sc-location":
      "http://sur1.dcdn.example/ucdn/example.com",
    "sc-cache-control" : "public, max-age=30"
  }
}
```

4.5. Indicating the cacheability and scope of responses

   [[Editor's note: Need to expand text a little.]]

   Cacheability is via the standard HTTP Cache-Control mechanisms.

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

```
+----------+---------+-----------+----------------------------+
| Key      | Value   | Mandatory | Description                |
+----------+---------+-----------+----------------------------+
| iprange  | List of | No        | A List of IP subnets in CIDR |
|          | String  |           | 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,
longest prefix matching of the User Agent's IP against the IP subnets
in the scope of each response SHOULD be used to select the most
appropriate RI response to use. [[Editor's note: is this always
true? What about the most recent response, should that override
older ones for the overlappign scope?]]

Example of DNS redirection response from Section 4.3.2 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

{
  "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 4.4.2 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"]
  }
}
```

4.6. Error responses

[[Editor's note: Probably need more explanation & examples of errors
that shouldn't be propagated to the User Agent?]]

RI error response examples.

RI error response (dCDN->uCDN) for DNS based User Agent requests:

```
HTTP/1.1 500 Server Error
Date: Mon, 06 Aug 2012 18:41:38 GMT
Content-Type: application/vnd.cdni.rrri.error+json
Cache-Control: private, no-cache

{
  "dns" : {
    "rcode" : 4                              # DNS response code (e.g.
                                             # doesn't support AAAA)
    "name" : "www.example.com",              # domain name response
                                             # relates to
  },
  "error" : {
    "code" : TBD,                            # Give each error type its
                                             # own numeric code
    "description" :                          # Give more informative
    "IPv6/AAAA queries are not supported"    # description than just
  }                                          # protocol specific error
                                             # codes
}
```

RI error response (dCDN->uCDN) for HTTP based User Agent requests:

```
HTTP/1.1 500 Server Error
Date: Mon, 06 Aug 2012 18:41:38 GMT
Content-Type: application/vnd.cdni.rrri.error+json
Cache-Control: private, no-cache

{
   "http": {
      "rcode": 400,                    # HTTP response code
      "url": "http://www.example.com", # URL response
```

```
                                              # relates to
        }
      "error" : {
      "code" : TBD,                           # Give each error type its
                                              # own numeric code
       "description" : TBD                    # Give more informative
                                              # description than just
    }                                         # protocol specific error
                                              # codes
    }
```

4.7. 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 MUST
   send a RI response with an error code of [[TBD]].

   It should be noted that the loop detection & prevention mechanisms
   described above only cover preventing and detecting loops within the
   RI itself. In the cases where the IP address(es) or URI(s) returned
   in RI responses do not resolve directly to a surrogate in the final

dCDN it is also possible to have redirection loops where Request
Routers in different CDNs direct User Agents in a loop.

5. Security Considerations

[[Editor's note: Not sure if this current text is really security
considerations or whether it is better placed elsewhere in the
document.]]

In HTTP based Recursive Request Redirection, the end user's web
browsers will not send cookies if the content request is redirected
to a URL in a different domain rather than the original CP's domain,
e.g. the Downstream CDN's domain. If the browser is expected to
send any cookies associated with the original CP's domain, this will
cause problem that the CP's policy is not enforced by the CDN.

The section 5.2 of draft [I-D.peterson-cdni-strawman] has discussed a
similar question and given a solution.

6. IANA Considerations

This document makes no request of IANA.

7. Acknowledgements

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

8. Outstanding considerations

Along with the various Editor's notes in the document, the following
items still need to be addressed:

o What extra properties/fields are required to cover all DNS/HTTP
   redirection cases?

o Do we need Queries other than A/AAAA & response other than A/AAAA/
   CNAME?

o Response scopes other than IP address? (AS? URL match?)

o Better Security Considerations section.

   o Description/specification for how to extend the protocol with
      additional optimonal parameters/attributes.

9. Contributing Authors

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

   Spencer Dawkins
   Huawei

   Email: spencer@wonderhamster.org

   Yunfei Zhang

   Email: hishigh@gmail.com

10. References

10.1. 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.

10.2. Informative References

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

   [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.

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

Authors' Addresses

   Wang Danhua (editor)
   Huawei Technologies
   No. 101 Software Avenue
   Nanjing, Jiangsu Province 210001
   P.R.China

   Phone: +86-25-56624734
   Fax: +86-25-56624702
   Email: wangdanhua@huawei.com

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

Email: ben@velocix.com

He Xiaoyan
Huawei
B2, Huawei Industrial Base
518129
P.R.China

Email: hexiaoyan@huawei.com

Ge Chen
China Telecom
109 West Zhongshan Ave,Tianhe District
Guangzhou
P.R. China

Email: cheng@gsta.com

Ni Wei
China Mobile
No.32 Xuanwumen West Street Xicheng District
Beijing   100053
P.R. China

Email: niwei@chinamobile.com