Network Working Group B. Niven-Jenkins
Internet-Draft D. Ferguson
Intended status: Standards Track Velocix (Alcatel-Lucent)
Expires: March 15, 2012 G. Watson
BT
September 12, 2011

CDN Interconnect Metadata
draft-jenkins-cdni-metadata-00

Abstract

This document focuses on the CDNI Metadata interface, which enables the CDNI Metadata function in a Downstream CDN to obtain CDNI Metadata from an Upstream CDN so that the Downstream CDN can properly process and respond to Redirection Requests received over the CDNI Request Routing protocol and Request Routing and Content Requests received directly from User Agents.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on March 15, 2012.

Copyright Notice

Copyright (c) 2011 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction

[I-D.jenkins-cdni-problem-statement] introduces the Problem scope for CDN Interconnection (CDNI) and lists the four categories of interfaces that may be used to compose a CDNI solution (Control, Metadata, Request Routing, Logging). [I-D.davie-cdni-framework] expands on the information provided in [I-D.jenkins-cdni-problem-statement] and describes each of the interfaces and the relationships between them in more detail.

This document focuses on the CDNI Metadata interface, which enables the CDNI Metadata function in a Downstream CDN to obtain CDNI Metadata from an Upstream CDN so that the Downstream CDN can properly process and respond to:

- Redirection Requests received over the CDNI Request Routing protocol.
- Request Routing and Content Requests received directly from User Agents.

Specifically this document proposes:

- A set of data objects that are used to describe the different aspects of CDNI Metadata along with a Data Model for CDNI Metadata that expresses the relationships between the CDNI Metadata objects (Section 2).
- An initial set of properties for the data objects in the CDNI Metadata data model (Section 3 through Section 5).
- A RESTful web service for the transfer of CDNI Metadata data objects (i.e. the CDNI Metadata interface) (Section 6).

Note: In order to make this document more accessible, it does not attempt to articulate all the CDNI Metadata that would be required to satisfy all CDNI use cases. Rather, a smaller set of properties are proposed. These should be sufficient to implement a minimal interconnect of basic HTTP delivery. Readers are encouraged to focus on the overall structure of the protocol rather than on the details or omission of particular properties. Additional properties may be added in future drafts or may be better placed in separate documents that focus on the CDNI Metadata required to interconnect delivery of individual end-user protocols.

1.1. Terminology

This document reuses the terminology defined in [I-D.jenkins-cdni-problem-statement].

2. CDNI Metadata Data Model

The CDNI Metadata interface specified in this document utilizes two types of Data Object:

- Addressable Data Objects, which are resources that may be retrieved via their own URIs.
- Embedded Data Objects, which are contained within a property of an Addressable Data Object.

Table 1 below defines the addressable data objects used by the CDNI Metadata interface.

Data Object	Description
SiteFeed	A SiteFeed object lists the Sites that may be delegated to the Downstream CDN.
Site	A Site object represents a customer-facing hostname that is used to serve content through including the rules for serving that content.
SelectionACL	A SelectionACL object restricts the Surrogates that can provide service for the associated Site to Surrogates in thre listed Locations.
DeliveryACL	A DeliveryACL object restricts the User Agent IP addresses that can access the associated Site or a URI pattern with the associated Site to User Agents in the listed Locations.
Location	A Location object represents a set of IP Address ranges.

Table 1: Content Distribution Metadata Data Objects

The only embedded data object defined is the DeliveryGlob object within a Site object, which describes the rules to apply for particular pattern based path matches within a Site.

The relationships between the different addressable CDNI Content Distribution Metadata objects are described in Figure 1 and Table 2 below and the properties of each object are described in Section 3.

```
                   +------------+
                   |SelectionACL+----------+
                   +------------+          |
                          ^                |
                          |                |
                          |                v
   +--------+          +--+-+         +--------+
   |SiteFeed+--------->|Site|         |Location|
   +--------+          +--+-+         +--------+
                          |                ^
                          |                |
                          v                |
                    +-----------+          |
                    |DeliveryACL+----------+
                    +-----------+
   Key: ----> = References
```

Figure 1: Relationships between CDNI Metadata Objects

Data Object	Objects it References
SiteFeed	0 or more Site objects.
Site	0 or 1 SelectionACL objects. 0 or 1 DeliveryACL objects.
SelectionACL	1 or more Location objects.
DeliveryACL	1 or more Location objects.
Location	None.

Table 2: Relationships between CDNI Metadata Objects

3. CDNI Metadata Addressable Data Object Descriptions

Each of the sub-sections below describes the properties associated with the data objects defined in Table 1.

The definition of each object is split into an ordered list of relationships and an unordered set of properties. Relationships are to other addressable data objects that are retrievable via the CDNI Metadata interface. The only exception is the DeliveryProtocol relationship which is to an 'opaque' URI representing the particular feature set of a delivery protocol defined in a specification and implemented in code.

Note: The names used for relationships (for example DeliveryProtocol) are illustrative and selected for readability and intuitive

understanding of their meaning. If retained, a future version of this document should list them out in the IANA considerations section and request they are registered in the Link registry.

3.1. SiteFeed

Relationship: Lists
Description: A Site that the Upstream CDN may delegate the delivery of to a Downstream CDN.
Allowed Target Types: Site
Cardinality: 0..*

Properties: None

3.2. Site

Relationship: DeliveryProtocol
Description: The Protocol to use for delivering this Site's content.
Allowed Target Types: Protocol
Cardinality: 1

Relationship: RestrictsDelivery
Description: The DeliveryACL is used to restrict which End User IP addresses are allowed access to the content of this Site. If not present, there are no restrictions on which End Users may receive the associated content (i.e. any End User IP address can access the content).
Allowed Target Types: DeliveryACL
Cardinality: 0..1

Relationship: RestrictsSelection
Description: The SelectionACL is used to restrict which Surrogates are allowed to serve the content of this Site. If not present, there are no restrictions on which Surrogates may deliver the associated content (i.e. any server can serve the content).
Allowed Target Types: SelectionACL
Cardinality: 0..1

Property: active
Description: Whether delivery should be enabled for this Site.
Type: Boolean
Mandatory: No (default True)

Property: delivery.globs

```
         Description: Path specific rules.  First match applies.
         Type: List of DeliveryGlob
         Mandatory: No (default apply the properties defined in the Site
         object to all paths)

      Property: delivery.hostname
         Description: The customer facing hostname for this site.
         Type: Hostname
         Mandatory: Yes

      Property: delivery.query.remove
         Description: A list of query parameter names.  The listed query
         parameters must be removed before checking for presence in the
         cache or forwarding the request to the origin
         Type: List of Strings
         Mandatory: No (default pass full URI through)

      Property: origin.basic.active
         Description: Whether to use HTTP Basic authentication to the
         Origin.
         Type: Boolean
         Mandatory: No (default False)

      Property: origin.basic.password
         Description: HTTP Basic auth password.  Required if
         origin.basic.active is true.
         Type: String
         Mandatory: No (default no password)

      Property: origin.basic.username
         Description: HTTP Basic auth username.  Required if
         origin.basic.active is true.
         Type: String
         Mandatory: No (default no username)

      Property: origin.cookie.active
         Description: Whether to use cookie-based authentication when
         contacting the Origin server.
         Type: Boolean
         Mandatory: No (default False)

      Property: origin.cookie.value
         Description: Cookie value to be returned to origin for cookie-
         based authentication.  Required if origin.cookie.active is
         True.
         Type: String
```

Mandatory: No (default no cookie value)

Property: origin.endpoints

Description: Origins from which the Downstream CDN can acquire content. These are not necessarily the actual origin servers operated by the CSP but might be a set of Surrogates/servers in the Upstream CDN.

Type: Set of Endpoints

Mandatory: Yes

3.3. SelectionACL

Relationship: SelectionAllow

Description: Surrogates in the referenced Location are allowed to serve the content of this Site.

Allowed Target Types: Location

Cardinality: 0..*

Relationship: SelectionDeny

Description: Surrogates in the referenced Location are not allowed to serve the content of this Site.

Allowed Target Types: Location

Cardinality: 0..*

Properties: None

Note: The order of Relationships within a SelectionACL is the order in which the ACL MUST be processed. If a SelectionACL object does not contain any of the above relationships (i.e. the object is empty) the result is the equivalent of matching against a SelectionDeny entry (i.e. any server is allowed to serve the associated content). If the end of a SelectionACL is reached without matching any of its entries the result is the equivalent of matching against a SelectionDeny entry (i.e. no Surrogates are allowed to serve the content of the Site).

3.4. DeliveryACL

Relationship: DeliveryAllow

Description: User Agents in the referenced Location are allowed to receive the content of this Site/DeliveryGlob.

Allowed Target Types: Location

Cardinality: 0..*

Relationship: DeliveryDeny

Description: User Agents in the referenced Location are not allowed to receive the content of this Site/DeliveryGlob.

Allowed Target Types: Location
Cardinality: 0..*

Relationship: DeliveryAuth
Description: The referenced URI represents an Authorisation Server that must be queried to determine whether or not to allow clients to receive the content of this Site/DeliveryGlob.
Allowed Target Types: URI
Cardinality: 0..1

Properties: None

Note: The order of Relationships within a DeliveryACL is the order in which the ACL MUST be processed. If a DeliveryACL object does not contain any of the above relationships (i.e. the object is empty) the result is the equivalent of matching against a DeliveryDeny entry (i.e. any User Agent IP address is allowed to receive the associated content). If the end of an DeliveryACL is reached with matching any of its entries the result is the equivalent of matching against a DeliveryDeny entry (i.e. Delivery to the User Agent is not allowed).

3.5. Location

Relationships: None

Property: location.ip
Description: A set of IP Addresses.
Type: List of IPRange.
Mandatory: Yes

[Editor's Note: Location as specified above only supports the Class 1a names described in [I-D.jenkins-cdni-names]. Need to add support for Class 1b names to a later version.]

4. CDNI Metadata Embedded Data Objects Descriptions

Each of the sub-sections below describes the data objects that are embedded within one or more of the data objects described in Section 3.

As in the previous section, the definition of each object is split into its properties and its relationships. Relationships may be to other objects that are retrievable via the CDNI Metadata interface.

4.1. DeliveryGlob

Relationship: RestrictsDelivery
Description: The DeliveryACL is used to restrict which End User IP addresses are allowed access to the content matched by this DeliveryGlob. If not present, there are no restrictions on which End Users may receive the associated content (i.e. any End User IP address can access the content).
Allowed Target Types: DeliveryACL
Cardinality: 0..1

Property: pattern.string
Description: String to match against the requested path, i.e. a [RFC3986] path-absolute.
Type: Pattern.
Mandatory: Yes

Property: pattern.flags
Description: Flags to control the pattern match.
Type: PatternFlags.
Mandatory: No (default Case-sensitive infix matching)

Property: delivery.proxy
Description: If set to True then this pattern should be proxied and not cached.
Type: Boolean.
Mandatory: No (default False)

5. CDNI Metadata Simple Data Type Descriptions

This section describes the simpler data types that are used for properties of Addressable Data Objects and Embedded Data Objects.

5.1. Protocol

This type only appears in links. Links with this type are not machine readable but rather represent particular feature sets of a protocol defined in a specification and implemented in code. The URI contained in the link needs to be defined for each delivery protocol with an associated interoperable feature set.

The following examples are illustrative:

- http://url.cdni.ietf.example/protocol/delivery/http/rfcABCD
- http://url.cdni.ietf.example/protocol/delivery/rtmp/rfcEFGH

- http://url.vendorY.ietf.example/protocol/delivery/rtmp/releaseP.Q

[Editor's Note: It may be more appropriate to use the 'tag' URI scheme [RFC4151] for these URIs.]

5.2. Endpoint

A hostname (with optional port) or an IP address (with optional port).

Note: Client 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].

5.3. IPRange

One of:

- A range of consecutive IP addresses (IPv4 or IPv6) expressed as Address1-Address2 which does not have to be to power of two aligned, for example the range 192.0.2.1-192.0.2.10 is valid. The first Address in the range MUST be 'lower' than the final address in the range.
- A valid IP subnet (IPv4 or IPv6) expressed using CIDR notation.
- A single IP address (IPv4 or IPv6).

Note: Client 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].

5.4. Pattern

A pattern for string matching. The string may contain the wildcards * and ?:

- * matches any sequence of characters (including the empty string).
- ? matches exactly one character.

Escaping: The three literals \ , * and ? should be escaped as \\, * and \?

5.5. PatternFlags

A set of flags indicating how a pattern match is made. The flags are:

- o Case-insensitive - Perform a case insensitive match (absence indicates case-sensitive match).
- o Prefix - Match against the start of the string (absence indicates that a match may start anywhere in the string).
- o Suffix - Match against the end of the string (absence indicates that a match may end anywhere in the string).

Absence of both Prefix and Suffix results in a match against any part of the string (infix).

5.6. URI

A URI as specified in [RFC3986].

6. CDNI Metadata interface

This section specifies an interface to enable a Downstream CDN to retrieve CDNI Metadata objects from an Upstream CDN.

The CDNI Metadata interface is built on the principles of RESTful web services, in particular the use of hypermedia as the engine of application state, and follows patterns established in The Atom Syndication Format [RFC4287]. This means that requests and responses over the interface are built around the transfer of representations of hyperlinked resources. A resource in the context of the CDNI Metadata interface is an Addressable Object in the Data Model (as described in Section 2 through Section 3).

Requests are made over HTTP. The HTTP Method defines the operation the request would like to perform. Servers implementing the CDNI Metadata interface MUST support the HTTP GET and HEAD methods. The corresponding HTTP Response returns the status of the operation in the HTTP Status Code and returns the current representation of the resource (if appropriate) in the Response Body. HTTP Responses from servers implementing the CDNI Metadata interface that contain a response body SHOULD include an ETag to enable validation of cached versions of returned resources.

The CDNI Metadata interface specified in this document is a read-only interface. Therefore support for other HTTP methods such as PUT, POST and DELETE etc. is not specified. Server implementations of this interface SHOULD reject all methods other than GET and HEAD.

The only representation specified in this document is JSON.

The interface can be used by a Downstream CDN to retrieve CDNI Metadata objects either dynamically as required by the Downstream CDN to process received requests (for example in response to receiving a CDNI Request Routing request from an Upstream CDN or in response to receiving a request for content from a User Agent) or in advance of being required.

In the general case a CDNI Metadata server makes each instance of an addressable CDNI Metadata object available via a unique URI that returns a representation of that instance of that CDNI Metadata object. When an object needs to reference another addressable CDNI Metadata object (for example a Site object referencing a DeliveryACL object) it does so by including a link to the referenced object.

The URI for the SiteFeed object needs to be either discovered by or configured in the downstream CDN. All other objects/resources are then discoverable from the SiteFeed object by following the links in the SiteFeed object and the referenced Site objects. CDNI Metadata servers are therefore free to assign whatever structure they desire to the URIs for CDNI Metadata objects and CDNI Metadata clients MUST NOT make any assumptions regarding the structure of CDNI Metadata URIs or the mapping between CDNI Metadata objects and their associated URIs. Therefore any URIs present in the examples below are purely illustrative and are not intended impose a definitive structure on CDNI Metadata interface implementations.

As the CDNI Metadata interface builds on top of HTTP, CDNI Metadata servers may make use of any HTTP feature when implementing the CDNI Metadata interface, for example a CDNI Metadata server may make use of HTTP's caching mechanisms to indicate that the returned response/ representation can be reused without re-contacting the CDNI Metadata server.

6.1. MIME Media Types

Table 3 lists the MIME Media Type for each object (resource) that is retrievable through the CDNI Metadata interface as well as the MIME Media Type for the DeliveryProtocol relationship.

Note: A prefix of "vnd.cdni" is used, however it is expected that a more appropriate prefix will be used if this document is accepted by the CDNI WG.

Data Object	MIME Media Type
SiteFeed	application/vnd.cdni.metadata.site.feed+json
Site	application/vnd.cdni.metadata.site+json
SelectionACL	application/vnd.cdni.metadata.acl.selection+json
DeliveryACL	application/vnd.cdni.metadata.acl.delivery+json
Location	application/vnd.cdni.metadata.location+json
DeliveryProtocol	application/vnd.cdni.metadata.protocol+json

Table 3: MIME Media Types for CDNI Metadata resources

6.2. JSON Encoding of Objects

The "base" encoding for a CDNI Metadata object is a JSON object containing a dictionary of (key,value) pairs where the keys are the property names and the values are the associated property values.

The keys of the dictionary are the names of the properties associated with the object and are therefore dependent on the specific object being encoded (i.e. dependent on the MIME Media Type of the returned resource). Likewise, the values associated with each key are dependent on the specific object being encoded (i.e. dependent on the MIME Media Type of the returned resource).

Dictionary keys in JSON are case sensitive and therefore any dictionary key defined by this document (for example the names of CDNI Metadata object properties) MUST always be represented in lowercase.

In addition to the properties of the object, the following three additional keys defined below may be present.

Key: base
Description: Provides a prefix for any relative URLs in the object. This is similar to the XML base tag [XML-BASE]. If absent, all URLs in the remainder of the document must be absolute URLs.
Type: URI
Mandatory: No

Key: links
Description: The relationships of this object to other addressable objects.

Type: List of Relationships.
Mandatory: Yes

Key: inline

Description: Dictionary containing additional objects that are inlined within this object. The keys in the dictionary are then then used to generate URI fragments which are used to refer to inlined objects and the value is the Object itself.
Type: Dictionary of Objects.
Mandatory: No

Example SiteFeed Object:

```
{
  "base": "http://metadata.cdni.example.com/sites/",
  "links": [
    {
      "title": "videos.example.net",
      "href": "example1",
      "rel": "Lists",
      "type": "application/vnd.cdni.metadata.site+json"
    },
    {
      "title": "trailers.example.net",
      "href": "http://metadata2.cdni.example.com/sites/example2",
      "rel": "Lists",
      "type": "application/vnd.cdni.metadata.site+json"
    }
  ],
}
```

6.3. JSON Encoding of Embedded Types

6.3.1. PatternFlags

JSON: A number calculated by adding together the values associated with each flag that is set:

- o 1 - Case-insensitive
- o 2 - Prefix
- o 4 - Suffix

Example of case-insensitive prefix match:

```
"pattern.flags": 3
```

6.3.2. Protocol

TBD

6.3.3. Relationship / Link

JSON: A dictionary with the following keys:

- href - The URI of the of the addressable object being referenced.
- rel - The Relationship between the referring object and the object it is referencing.
- type - The MIME Media Type of the referenced object. See Section 6.1 for the MIME Media Types of objects specified in this document.
- title - An title for the for the Relationship/link. For the "Lists" Relationships contained in a SiteFeed object the title key MUST be present and MUST be the value of delivery.hostname for the referenced Site object. For all other Relationships/links the title key is optional.

Note: The above structure follows the pattern of atom:link in [RFC4287].

Example Relationship to a CDNI Metadata location within a SelectionACL object:

```
{
  "href": "http://metadata.cdni.example.com/locations/everywhere",
  "rel": "SelectionAllow",
  "type": "application/vnd.cdni.metadata.location+json"
}
```

6.4. Retrieval of CDNI Metadata resources

In the general case a CDNI Metadata server makes each instance of an addressable CDNI Metadata object available via a unique URI and therefore in order to retrieve CDNI Metadata, a CDNI Metadata client first makes a HTTP GET request for the URI of the SiteFeed which provides the CDNI Metadata client with a list of Sites (along with their public facing hostnames) that the upstream CDN may delegate to the downstream CDN.

In order to retrieve the CDNI Metadata for a particular Site the CDNI Metadata client processes the received SiteFeed object and finds the corresponding entry (using the title key to match against the required public facing hostname if required) in the SiteFeed for the Site it requires and can then can make a GET request for the URI specified in the href key of that Site's entry in the SiteFeed.

In order to retrieve the SelectionACLs and DeliveryACLs associated with that Site the CDNI Metadata client processes the received Site object (as well as any DeliveryGlob objects embedded in the Site's delivery.globs key) and can then can make a GET request for the URIs specified in the href key of links within that Site or its DeliveryGlobs that have a Relationship of RestrictsSelection and RestrictsDelivery respectively.

Finally in order to retrieve the Locations associated with each ACL the CDNI Metadata client processes the received ACL object(s) and can then can make a GET request for the URIs specified in the href key of links within that ACL that have a Relationship of Location.

6.4.1. Bulk Retrieval of CDNI Metadata resources

In addition to the general case where a CDNI Metadata server makes each instance of an addressable CDNI Metadata object available via a unique URI, in response to a request for a CDNI Metadata object a CDNI Metadata Servers MAY include one or more of the addressable objects referenced by the requested object inside the inline dictionary of the referenced objects.

Inlined objects are referenced using a link in the normal way with the exception that the href component of the link begins with # and follows the fragment resolution protocol defined in [I-D.zyp-json-schema].

Use of the inline dictionary to include multiple addressable objects has the advantage of reducing the number of requests a CDNI Metadata client needs to make (and therefore reduces the additional latency introduced by making multiple requests) in order to retrieve all the CDNI Metadata it requires to process CDNI Request Routing requests and User Agent requests for content.

However use of the inline dictionary has the disadvantage that the cacheability of any inlined objects are tied to the cacheability of the object they are inlined in. Some objects (for example Locations) may not change very often and therefore could be cached for a longer period of time than the objects that reference them. Or an object (for example a SelectionACL) may be referenced by multiple other objects and benefit from being cached separately in order to reduce the size of the responses to requests for objects that reference it.

Example of an inline dictionary containing a DeliveryACL object:

```
"inline": {
  "deliveryeverywhereacl": {
    "links": [
      {
        "href":
          "http://metadata.cdni.example.com/locations/everywhere",
        "rel": "DeliveryAllow",
        "type": "application/vnd.cdni.metadata.location+json"
      }
    ]
  }
}
```

Example of an inline dictionary containing a DeliveryACL object where the Location referenced by the DeliveryACL is also inlined:

```
"inline": {
  "deliveryeverywhereacl": {
    "links": [
      {
        "href": "#/inline/everywhere",
        "rel": "DeliveryAllow",
        "type": "application/vnd.cdni.metadata.location+json"
      }
    ]
  },
  "everywhere": {
    "location.ip": ["0.0.0.0", "::/0"]
  }
}
```

Note: An alternative to using an inline key as described above would be for the CDNI Metadata server to return a multipart/mixed response. Using a multipart/mixed response would have the advantage that inlined objects would not have to be tied to the cacheability of the object they are inlined in (as they could have their own Cache-Control headers) and could be cached separately from the object they are inlined with (as they could have their own Location header). However, using a multipart/mixed response would have the disadvantage of requiring more complex processing by the CDNI Metadata client.

6.5. Examples

The following sections provide examples of different CDNI Metadata objects encoded as JSON.

6.5.1. SiteFeed

Example SiteFeed:

```
   {
     "base": "http://metadata.cdni.example.com/sites/",
     "links": [
       {
         "title": "videos.example.net",
         "href": "example1",
         "rel": "Lists",
         "type": "application/vnd.cdni.metadata.site+json"
       },
       {
         "title": "trailers.example.net",
         "href": "http://metadata2.cdni.example.com/sites/example2",
         "rel": "Lists",
         "type": "application/vnd.cdni.metadata.site+json"
       }
     ],
   }
```

6.5.2. Site

Example Site with inlined objects:

```
   {
     "base": "http://metadata.cdni.example.com/",
     "links": [
       {
         "href": "http://url.cdni.ietf.org/protocol/delivery/http/1.1",
         "rel": "DeliveryProtocol",
         "type": "application/vnd.cdni.metadata.protocol+json"
       },
       {
         "href": "#/inline/delivereverywhereacl",
         "rel": "RestrictsDelivery",
         "type": "application/vnd.cdni.acl.delivery+json"
       },
       {
         "href": "#/inline/servefromanywhereacl",
         "rel": "RestrictsSelection",
         "type": "application/vnd.cdni.acl.selection+json"
       }
     ],
     "active": "true",
     "delivery.hostname": "videos.example.net"
     "origin.hostnames": ["origin.videos.example.net.cdni.example.com",
```

```
       "origin.example.net"],
     "delivery.globs": [
       {
         "links": [
           {
             "href": "#/inline/qaonlyacl",
             "rel": "RestrictsDelivery",
             "type": "application/vnd.cdni.acl.delivery+json"
           }
         ],
         "pattern.string": "*/test_files/*",
         "pattern.flags": 1,
         "delivery.proxy": false
       }
     ],
     "inline": {
       "deliveryeverywhereacl": {
         "links": [
           {
             "href": "#/inline/everywhere",
             "rel": "DeliveryAllow",
             "type": "application/vnd.cdni.metadata.location+json"
           }
         ]
       },
       "servefromanywhereacl": {
         "links": [
           {
             "href": "#/inline/everywhere",
             "rel": "SelectionAllow",
             "type": "application/vnd.cdni.metadata.location+json"
           }
         ]
       },
       "qaonlyacl": {
         "links": [
           {
             "href": "#/inline/qa1",
             "rel": "DeliveryAllow",
             "type": "application/vnd.cdni.metadata.location+json"
           },
           {
             "href": "#/inline/qa2",
             "rel": "DeliveryAllow",
             "type": "application/vnd.cdni.metadata.location+json"
           },
           {
             "href": "#/inline/everywhere",
```

```
          "rel": "DeliveryDeny",
          "type": "application/vnd.cdni.metadata.location+json"
        }
      ]
    },
    "everywhere": {
      "location.ip": ["0.0.0.0", "::/0"]
    },
    "qa1": {
      "location.ip": ["192.0.2.1-192.0.2.10"]
    },
    "qa2": {
      "location.ip": ["198.51.100.1-198.51.100.15",
        "198.51.100.200-198.51.100.254"]
    }
  }
}
```

7. IANA Considerations

TBD.

8. Security Considerations

TBD.

9. Acknowledgements

TBD.

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.

[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.

10.2. Informative References

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

[I-D.jenkins-cdni-names]
Niven-Jenkins, B., "Thoughts on Naming and Referencing of Data Objects within Content Distribution Network Interconnection (CDNI) solutions", draft-jenkins-cdni-names-00 (work in progress), February 2011.

[I-D.jenkins-cdni-problem-statement]
Niven-Jenkins, B., Faucheur, F., and N. Bitar, "Content Distribution Network Interconnection (CDNI) Problem Statement", draft-jenkins-cdni-problem-statement-02 (work in progress), March 2011.

[I-D.lefaucheur-cdni-requirements]
Leung, K., Lee, Y., Faucheur, F., Viveganandhan, M., and G. Watson, "Content Distribution Network Interconnection (CDNI) Requirements", draft-lefaucheur-cdni-requirements-02 (work in progress), July 2011.

[I-D.zyp-json-schema]
Zyp, K. and G. Court, "A JSON Media Type for Describing the Structure and Meaning of JSON Documents", draft-zyp-json-schema-03 (work in progress), November 2010.

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

[RFC4151] Kindberg, T. and S. Hawke, "The 'tag' URI Scheme", RFC 4151, October 2005.

[RFC4287] Nottingham, M., Ed. and R. Sayre, Ed., "The Atom Syndication Format", RFC 4287, December 2005.

[XML-BASE]
Marsh, J., Ed. and R. Tobin, Ed., "XML Base (Second Edition) - http://www.w3.org/TR/xmlbase/", January 2009.

Appendix A. Relationship to the CDNI Requirements

Section 6 of [I-D.lefaucheur-cdni-requirements] lists the requirements for the CDNI Metadata Distribution interface. This section outlines which of those requirements are met by the CDNI Metadata interface specified in this document.

Requirements R49 through R57 (inclusive) and R61 through R63 (inclusive) are directly met by the interface specified in this document.

Requirements R59 and R60 can be trivally met by defining additional properties for the CDNI Metadata objects defined in this document.

It is the opinion of the authors that requirement R58 is better handled at Request Routing time by the CDNI Request Routing interface, rather than directly being met by the CDNI Metadata interface.

Authors' Addresses

Ben Niven-Jenkins
Velocix (Alcatel-Lucent)
326 Cambridge Science Park
Milton Road, Cambridge CB4 0WG
UK

Email: ben@velocix.com

David Ferguson
Velocix (Alcatel-Lucent)
326 Cambridge Science Park
Milton Road, Cambridge CB4 0WG
UK

Email: david@velocix.com

Grant Watson
BT
pp GDC 1 PP14, Orion Building, Adastral Park
Martlesham, Ipswich IP5 3RE
UK

Email: grant.watson@bt.com