Constrained RESTful Environments (CoRE) Link Format

Abstract

   This specification defines Web Linking using a link format for use by
   constrained web servers to describe hosted resources, their
   attributes, and other relationships between links. Based on the HTTP
   Link Header field defined in RFC 5988, the Constrained RESTful
   Environments (CoRE) Link Format is carried as a payload and is
   assigned an Internet media type. "RESTful" refers to the
   Representational State Transfer (REST) architecture. A well-known
   URI is defined as a default entry point for requesting the links
   hosted by a server.

Status of This Memo

   This is an Internet Standards Track document.

   This document is a product of the Internet Engineering Task Force
   (IETF). It represents the consensus of the IETF community. It has
   received public review and has been approved for publication by the
   Internet Engineering Steering Group (IESG). Further information on
   Internet Standards is available in Section 2 of RFC 5741.

   Information about the current status of this document, any errata,
   and how to provide feedback on it may be obtained at
   http://www.rfc-editor.org/info/rfc6690.

Copyright Notice

Table of Contents

1. Introduction

   The Constrained RESTful Environments (CoRE) realizes the
   Representational State Transfer (REST) architecture [REST] in a
   suitable form for the most constrained nodes (e.g., 8-bit
   microcontrollers with limited memory) and networks (e.g., IPv6 over
   Low-Power Wireless Personal Area Networks (6LoWPANs) [RFC4919]).
   CoRE is aimed at Machine-to-Machine (M2M) applications such as smart
   energy and building automation.

   The discovery of resources hosted by a constrained server is very
   important in machine-to-machine applications where there are no
   humans in the loop and static interfaces result in fragility. The
   discovery of resources provided by an HTTP [RFC2616] web server is
   typically called "Web Discovery" and the description of relations
   between resources is called "Web Linking" [RFC5988]. In the present
   specification, we refer to the discovery of resources hosted by a
   constrained web server, their attributes, and other resource
   relations as CoRE Resource Discovery.

   The main function of such a discovery mechanism is to provide
   Universal Resource Identifiers (URIs, called links) for the resources
   hosted by the server, complemented by attributes about those
   resources and possible further link relations. In CoRE, this
   collection of links is carried as a resource of its own (as opposed
   to HTTP headers delivered with a specific resource). This document
   specifies a link format for use in CoRE Resource Discovery by
   extending the HTTP Link Header format [RFC5988] to describe these
   link descriptions. The CoRE Link Format is carried as a payload and
   is assigned an Internet media type. A well-known relative URI
   "/.well-known/core" is defined as a default entry point for
   requesting the list of links about resources hosted by a server and
   thus performing CoRE Resource Discovery. This specification is
   applicable for use with Constrained Application Protocol (CoAP)
   [COAP], HTTP, or any other suitable web transfer protocol. The link
   format can also be saved in file format.

1.1. Web Linking in CoRE

   Technically, the CoRE Link Format is a serialization of a typed link
   as specified in [RFC5988], used to describe relationships between
   resources, so-called "Web Linking". In this specification, Web
   Linking is extended with specific constrained M2M attributes; links
   are carried as a message payload rather than in an HTTP Link Header
   field, and a default interface is defined to discover resources
   hosted by a server. This specification also defines a new relation

type "hosts" (from the verb "to host"), which indicates that the
resource is hosted by the server from which the link document was
requested.

In HTTP, the Link Header can be used to carry link information about
a resource along with an HTTP response. This works well for the
typical use case for a web server and browser, where further
information about a particular resource is useful after accessing it.
In CoRE, the main use case for Web Linking is the discovery of which
resources a server hosts in the first place. Although some resources
may have further links associated with them, this is expected to be
an exception. For that reason, the CoRE Link Format serialization is
carried as a resource representation of a well-known URI. The CoRE
Link Format does reuse the format of the HTTP Link Header
serialization defined in [RFC5988].

1.2. Use Cases

Typical use cases for Web Linking on today's web include, e.g.,
describing the author of a web page or describing relations between
web pages (next chapter, previous chapter, etc.). Web Linking can
also be applied to M2M applications, where typed links are used to
assist a machine client in finding and understanding how to use
resources on a server. In this section a few use cases are described
for how the CoRE Link Format could be used in M2M applications. For
further technical examples, see Section 5. As there is a large range
of M2M applications, these use cases are purposely generic. This
specification assumes that different deployments or application
domains will define the appropriate REST Interface Descriptions along
with Resource Types to make discovery meaningful.

1.2.1. Discovery

In M2M applications, for example, home or building automation, there
is a need for local clients and servers to find and interact with
each other without human intervention. The CoRE Link Format can be
used by servers in such environments to enable Resource Discovery of
the resources hosted by the server.

Resource Discovery can be performed either unicast or multicast.
When a server's IP address is already known, either a priori or
resolved via the Domain Name System (DNS) [RFC1034][RFC1035], unicast
discovery is performed in order to locate the entry point to the
resource of interest. In this specification, this is performed using
a GET to "/.well-known/core" on the server, which returns a payload
in the CoRE Link Format. A client would then match the appropriate
Resource Type, Interface Description, and possible media type

[RFC2045] for its application. These attributes may also be included
in the query string in order to filter the number of links returned
in a response.

Multicast Resource Discovery is useful when a client needs to locate
a resource within a limited scope, and that scope supports IP
multicast. A GET request to the appropriate multicast address is
made for "/.well-known/core". In order to limit the number and size
of responses, a query string is recommended with the known
attributes. Typically, a resource would be discovered based on its
Resource Type and/or Interface Description, along with possible
application-specific attributes.

1.2.2. Resource Collections

RESTful designs of M2M interfaces often make use of collections of
resources. For example, an index of temperature sensors on a data
collection node or a list of alarms on a home security controller.
The CoRE Link Format can be used to make it possible to find the
entry point to a collection and traverse its members. The entry
point of a collection would always be included in "/.well-known/core"
to enable its discovery. The members of the collection can be
defined either through the Interface Description of the resource
along with a parameter resource for the size of the collection or by
using the link format to describe each resource in the collection.
These links could be located under "/.well-known/core" or hosted, for
example, in the root resource of the collection.

1.2.3. Resource Directory

In many deployment scenarios, for example, constrained networks with
sleeping servers or large M2M deployments with bandwidth limited
access networks, it makes sense to deploy resource directory entities
that store links to resources stored on other servers. Think of this
as a limited search engine for constrained M2M resources.

The CoRE Link Format can be used by a server to register resources
with a resource directory or to allow a resource directory to poll
for resources. Resource registration can be achieved by having each
server POST their resources to "/.well-known/core" on the resource
directory. This, in turn, adds links to the resource directory under
an appropriate resource. These links can then be discovered by any
client by making a request to a resource directory lookup interface.

1.3. Terminology

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

   This specification makes use of the Augmented Backus-Naur Form (ABNF)
   [RFC5234] notation, including the core rules defined in Appendix B of
   that document.

   This specification requires readers to be familiar with all the terms
   and concepts that are discussed in [RFC5988] and [RFC6454]. In
   addition, this specification makes use of the following terminology:

   Web Linking
      A framework for indicating the relationships between web
      resources.

   Link
      Also called "typed links" in [RFC5988]. A link is a typed
      connection between two resources identified by URI and is made up
      of a context URI, a link relation type, a target URI, and optional
      target attributes.

   Link Format
      A particular serialization of typed links.

   CoRE Link Format
      A particular serialization of typed links based on the HTTP Link
      Header field serialization defined in Section 5 of [RFC5988] but
      carried as a resource representation with a media type.

   Attribute
      Properly called "Target Attribute" in [RFC5988]. A key/value pair
      that describes the link or its target.

   CoRE Resource Discovery
      When a client discovers the list of resources hosted by a server,
      their attributes, and other link relations by accessing
      "/.well-known/core".

2. Link Format

   The CoRE Link Format extends the HTTP Link Header field specified in
   [RFC5988]. The format does not require special XML or binary
   parsing, is fairly compact, and is extensible -- all important
   characteristics for CoRE. It should be noted that this link format
   is just one serialization of typed links defined in [RFC5988]; others

include HTML links, Atom feed links [RFC4287], or HTTP Link Header
fields. It is expected that resources discovered in the CoRE Link
Format may also be made available in alternative formats on the
greater Internet. The CoRE Link Format is only expected to be
supported in constrained networks and M2M systems.

Section 5 of [RFC5988] did not require an Internet media type for the
defined link format, as it was defined to be carried in an HTTP
header. This specification thus defines the Internet media type
'application/link-format' for the CoRE Link Format (see Section 7.3).
Whereas the HTTP Link Header field depends on [RFC2616] for its
encoding, the CoRE Link Format is encoded as UTF-8 [RFC3629]. A
decoder of the format is not expected to validate UTF-8 encoding (but
is not prohibited from doing so) and doesn't need to perform any
UTF-8 normalization. UTF-8 data can be compared bitwise, which
allows values to contain UTF-8 data without any added complexity for
constrained nodes.

The CoRE Link Format is equivalent to the [RFC5988] link format;
however, the ABNF in the present specification is repeated with
improvements to be compliant with [RFC5234] and includes new link
parameters. The link parameter "href" is reserved for use as a query
parameter for filtering in this specification (see Section 4.1) and
MUST NOT be defined as a link parameter. As in [RFC5988], multiple
link descriptions are separated by commas. Note that commas can also
occur in quoted strings and URIs but do not end a description. In
order to convert an HTTP Link Header field to this link format, first
the "Link:" HTTP header is removed, any linear whitespace (LWS) is
removed, the header value is converted to UTF-8, and any percent-
encodings are decoded.

```
 Link            = link-value-list
 link-value-list = [ link-value *[ "," link-value ]]
 link-value      = "<" URI-Reference ">" *( ";" link-param )
 link-param      = ( ( "rel" "=" relation-types )
                 / ( "anchor" "=" DQUOTE URI-Reference DQUOTE )
                 / ( "rev" "=" relation-types )
                 / ( "hreflang" "=" Language-Tag )
                 / ( "media" "=" ( MediaDesc
                       / ( DQUOTE MediaDesc DQUOTE ) ) )
                 / ( "title" "=" quoted-string )
                 / ( "title*" "=" ext-value )
                 / ( "type" "=" ( media-type / quoted-mt ) )
                 / ( "rt" "=" relation-types )
                 / ( "if" "=" relation-types )
                 / ( "sz" "=" cardinal )
                 / ( link-extension ) )
  link-extension = ( parmname [ "=" ( ptoken / quoted-string ) ] )
```

```
                    / ( ext-name-star "=" ext-value )
    ext-name-star = parmname "*" ; reserved for RFC-2231-profiled
                                 ; extensions.  Whitespace NOT
                                 ; allowed in between.
    ptoken        = 1*ptokenchar
    ptokenchar    = "!" / "#" / "$" / "%" / "&" / "'" / "("
                    / ")" / "*" / "+" / "-" / "." / "/" / DIGIT
                    / ":" / "<" / "=" / ">" / "?" / "@" / ALPHA
                    / "[" / "]" / "^" / "_" / "`" / "{" / "|"
                    / "}" / "~"
    media-type    = type-name "/" subtype-name
    quoted-mt     = DQUOTE media-type DQUOTE
    relation-types = relation-type
                    / DQUOTE relation-type *( 1*SP relation-type ) DQUOTE
    relation-type = reg-rel-type / ext-rel-type
    reg-rel-type  = LOALPHA *( LOALPHA / DIGIT / "." / "-" )
    ext-rel-type  = URI
    cardinal      = "0" / ( %x31-39 *DIGIT )
    LOALPHA       = %x61-7A   ; a-z
    quoted-string = <defined in [RFC2616]>
    URI           = <defined in [RFC3986]>
    URI-Reference = <defined in [RFC3986]>
    type-name     = <defined in [RFC4288]>
    subtype-name  = <defined in [RFC4288]>
    MediaDesc     = <defined in [W3C.HTML.4.01]>
    Language-Tag  = <defined in [RFC5646]>
    ext-value     = <defined in [RFC5987]>
    parmname      = <defined in [RFC5987]>
```

2.1. Target and Context URIs

Each link conveys one target URI as a URI-reference inside angle
brackets ("<>"). The context URI of a link (also called the base URI
in [RFC3986]) is determined by the following rules in this
specification:

(a) The context URI is set to the anchor parameter, when specified.

(b) Origin of the target URI, when specified.

(c) Origin of the link format resource's base URI.

2.2. Link Relations

Since links in the CoRE Link Format are typically used to describe
resources hosted by a server, the new relation type "hosts" is
assumed in the absence of the relation parameter (see Section 7.2).
The "hosts" relation type (from the verb "to host") indicates that

   the target URI is a resource hosted by the server (i.e., server hosts
   resource) indicated by the context URI. The target URI MUST be a
   relative URI of the context URI for this relation type.

   To express other relations, links can make use of any registered
   relation by including the relation parameter. The context of a
   relation can be defined using the anchor parameter. In this way,
   relations between resources hosted on a server or between hosted
   resources and external resources can be expressed.

2.3. Use of Anchors

   As per Section 5.2 of [RFC5988], a link description MAY include an
   "anchor" parameter, in which case the context is the URI included in
   that attribute. This is used to describe a relationship between two
   resources. A consuming implementation can, however, choose to ignore
   such links. It is not expected that all implementations will be able
   to derive useful information from explicitly anchored links.

3. CoRE Link Attributes

   The following CoRE-specific target attributes are defined in addition
   to those already defined in [RFC5988]. These attributes describe
   information useful in accessing the target link of the relation and,
   in some cases, can use the syntactical form of a URI. Such a URI MAY
   be dereferenced (for instance, to obtain a description of the link
   relation), but that is not part of the protocol and MUST NOT be done
   automatically on link evaluation. When the values of attributes are
   compared, they MUST be compared as strings.

3.1. Resource Type 'rt' Attribute

   The Resource Type 'rt' attribute is an opaque string used to assign
   an application-specific semantic type to a resource. One can think
   of this as a noun describing the resource. In the case of a
   temperature resource, this could be, e.g., an application-specific
   semantic type like "outdoor-temperature" or a URI referencing a
   specific concept in an ontology like
   "http://sweet.jpl.nasa.gov/2.0/phys.owl#Temperature". Multiple
   Resource Types MAY be included in the value of this parameter, each
   separated by a space, similar to the relation attribute. The
   registry for Resource Type values is defined in Section 7.4.

   The Resource Type attribute is not meant to be used to assign a
   human-readable name to a resource. The "title" attribute defined in
   [RFC5988] is meant for that purpose. The Resource Type attribute
   MUST NOT appear more than once in a link.

3.2. Interface Description 'if' Attribute

   The Interface Description 'if' attribute is an opaque string used to
   provide a name or URI indicating a specific interface definition used
   to interact with the target resource. One can think of this as
   describing verbs usable on a resource. The Interface Description
   attribute is meant to describe the generic REST interface to interact
   with a resource or a set of resources. It is expected that an
   Interface Description will be reused by different Resource Types.
   For example, the Resource Types "outdoor-temperature", "dew-point",
   and "rel-humidity" could all be accessible using the Interface
   Description "http://www.example.org/myapp.wadl#sensor". Multiple
   Interface Descriptions MAY be included in the value of this
   parameter, each separated by a space, similar to the relation
   attribute. The registry for Interface Description values is defined
   in Section 7.4.

   The Interface Description could be, for example, the URI of a Web
   Application Description Language (WADL) [WADL] definition of the
   target resource "http://www.example.org/myapp.wadl#sensor", a URN
   indicating the type of interface to the resource "urn:myapp:sensor",
   or an application-specific name "sensor". The Interface Description
   attribute MUST NOT appear more than once in a link.

3.3. Maximum Size Estimate 'sz' Attribute

   The maximum size estimate attribute 'sz' gives an indication of the
   maximum size of the resource representation returned by performing a
   GET on the target URI. For links to CoAP resources, this attribute
   is not expected to be included for small resources that can
   comfortably be carried in a single Maximum Transmission Unit (MTU)
   but SHOULD be included for resources larger than that. The maximum
   size estimate attribute MUST NOT appear more than once in a link.

   Note that there is no defined upper limit to the value of the 'sz'
   attributes. Implementations MUST be prepared to accept large values.
   One implementation strategy is to convert any value larger than a
   reasonable size limit for this implementation to a special value
   "Big", which in further processing would indicate that a size value
   was given that was so big that it cannot be processed by this
   implementation.

4. Well-Known Interface

   Resource discovery in CoRE is accomplished through the use of a well-
   known resource URI that returns a list of links about resources
   hosted by that server and other link relations. Well-known resources

have a path component that begins with "/.well-known/" as specified
in [RFC5785]. This specification defines a new well-known resource
for CoRE Resource Discovery: "/.well-known/core".

A server implementing this specification MUST support this resource
on the default port appropriate for the protocol for the purpose of
resource discovery. It is, however, up to the application which
links are included and how they are organized. The resource
"/.well-known/core" is meant to be used to return links to the entry
points of resource interfaces on a server. More sophisticated link
organization can be achieved by including links to CoRE Link Format
resources located elsewhere on the server, for example, to achieve an
index. In the absence of any links, a zero-length payload is
returned. The resource representation of this resource MUST be the
CoRE Link Format described in Section 2.

The CoRE resource discovery interface supports the following
interactions:

o Performing a GET on "/.well-known/core" to the default port
   returns a set of links available from the server (if any) in the
   CoRE Link Format. These links might describe resources hosted on
   that server or on other servers or express other kinds of link
   relations as described in Section 2.

o Filtering may be performed on any of the link format attributes
   using a query string as specified in Section 4.1. For example,
   [GET /.well-known/core?rt=temperature-c] would request resources
   with the Resource Type temperature-c. A server is not, however,
   required to support filtering.

o More capable servers such as proxies could support a resource
   directory by requesting the resource descriptions of other end-
   points or allowing servers to POST requests to "/.well-known/
   core". The details of such resource directory functionality is,
   however, out of the scope of this specification and is expected to
   be specified separately.

4.1. Query Filtering

   A server implementing this specification MAY recognize the query part
   of a resource discovery URI as a filter on the resources to be
   returned. The path and query components together should conform to
   the following level-4 URI Template [RFC6570]:

       /.well-known/core{?search*}

   where the variable "search" is a 1-element list that has a single
   name/value pair, where

   o name is either "href", a link-param name defined in this
      specification, or any other link-extension name, and

   o value is either a Complete Value String that does not end in an
      "*" (%2A), or a Prefix Value String followed by an "*" (%2A).

   The search name "href" refers to the URI-reference between the "<"
   and ">" characters of a link. Both Value Strings match a target
   attribute only if it exists. Value Strings are percent-decoded
   ([RFC3986], Section 2.1) before matching; similarly, any target
   attributes notated as quoted-string are interpreted as defined in
   Section 2.2 of [RFC2616]. After these steps, a Complete Value String
   matches a target attribute if it is bitwise identical. A Prefix
   Value String matches a target attribute if it is a bitwise prefix of
   the target attribute (where any string is a prefix of itself). Empty
   Prefix Value Strings are allowed; by the definition above, they match
   any target attribute that does exist. Note that relation-type target
   attributes can contain multiple values, and each value MUST be
   treated as a separate target attribute when matching.

   It is not expected that very constrained nodes support filtering.
   Implementations not supporting filtering MUST simply ignore the query
   string and return the whole resource for unicast requests.

   When using a transfer protocol like the Constrained Application
   Protocol (CoAP) that supports multicast requests, special care needs
   to be taken. A multicast request with a query string SHOULD NOT be
   responded to if filtering is not supported or if the filter does not
   match (to avoid a needless response storm). The exception is in
   cases where the IP stack interface is not able to indicate that the
   destination address was multicast.

The following are examples of valid query URIs:

o ?href=/foo matches a link-value that is anchored at /foo

o ?href=/foo* matches a link-value that is anchored at a URI that
   starts with /foo

o ?foo=bar matches a link-value that has a target attribute named
   foo with the exact value bar

o ?foo=bar* matches a link-value that has a target attribute named
   foo, the value of which starts with bar, e.g., bar or barley

o ?foo=* matches a link-value that has a target attribute named foo

5. Examples

   A few examples of typical link descriptions in this format follows.
   Multiple resource descriptions in a representation are separated by
   commas. Linefeeds are also included in these examples for
   readability. Although the following examples use CoAP response
   codes, the examples are applicable to HTTP as well (the corresponding
   response code would be 200 OK).

   This example includes links to two different sensors sharing the same
   Interface Description. Note that the default relation type for this
   link format is "hosts" in links with no rel= target attribute. Thus,
   the links in this example tell that the Origin server from which
   /.well-known/core was requested (the context) hosts the resources
   /sensors/temp and /sensors/light (each a target).

   REQ: GET /.well-known/core

   RES: 2.05 Content
   </sensors/temp>;if="sensor",
   </sensors/light>;if="sensor"

   Without the linefeeds inserted here for readability, the format
   actually looks as follows.

   </sensors/temp>;if="sensor",</sensors/light>;if="sensor"

   This example arranges link descriptions hierarchically, with the
   entry point including a link to a sub-resource containing links about
   the sensors.

```
REQ: GET /.well-known/core

RES: 2.05 Content
</sensors>;ct=40

REQ: GET /sensors

RES: 2.05 Content
</sensors/temp>;rt="temperature-c";if="sensor",
</sensors/light>;rt="light-lux";if="sensor"
```

An example query filter may look like:

```
REQ: GET /.well-known/core?rt=light-lux

RES: 2.05 Content
</sensors/light>;rt="light-lux";if="sensor"
```

Note that relation-type attributes like 'rt', 'if', and 'rel' can
have multiple values separated by spaces. A query filter parameter
can match any one of those values, as in this example:

```
REQ: GET /.well-known/core?rt=light-lux

RES: 2.05 Content
</sensors/light>;rt="light-lux core.sen-light";if="sensor"
```

This example shows the use of an "anchor" attribute to relate the
temperature sensor resource to an external description and to an
alternative URI.

```
REQ: GET /.well-known/core

RES: 2.05 Content
</sensors>;ct=40;title="Sensor Index",
</sensors/temp>;rt="temperature-c";if="sensor",
</sensors/light>;rt="light-lux";if="sensor",
<http://www.example.com/sensors/t123>;anchor="/sensors/temp"
;rel="describedby",
</t>;anchor="/sensors/temp";rel="alternate"
```

If a client is interested in finding relations about a particular
resource, it can perform a query on the anchor parameter:

REQ: GET /.well-known/core?anchor=/sensors/temp

RES: 2.05 Content
<http://www.example.com/sensors/temp123>;anchor="/sensors/temp"
;rel="describedby",
</t>;anchor="/sensors/temp";rel="alternate"

The following example shows a large firmware resource with a size
attribute. The consumer of this link would use the 'sz' attribute to
determine if the resource representation is too large and if block
transfer would be required to request it. In this case, a client
with only a 64 KiB flash might only support a 16-bit integer for
storing the 'sz' attribute. Thus, a special flag or value should be
used to indicate "Big" (larger than 64 KiB).

REQ: GET /.well-known/core?rt=firmware

RES: 2.05 Content
</firmware/v2.1>;rt="firmware";sz=262144

6. Security Considerations

This specification has the same security considerations as described
in Section 7 of [RFC5988]. The "/.well-known/core" resource MAY be
protected, e.g., using Datagram Transport Layer Security (DTLS) when
hosted on a CoAP server as per [COAP], Section 9.1.

Some servers might provide resource discovery services to a mix of
clients that are trusted to different levels. For example, a
lighting control system might allow any client to read state
variables, but only certain clients to write state (turn lights on or
off). Servers that have authentication and authorization features
SHOULD support authentication features of the underlying transport
protocols (HTTP or DTLS/TLS) and allow servers to return different
lists of links based on a client's identity and authorization. While
such servers might not return all links to all requesters, not
providing the link does not, by itself, control access to the
relevant resource -- a bad actor could know or guess the right URIs.
Servers can also lie about the resources available. If it is
important for a client to only get information from a known source,
then that source needs to be authenticated.

Multicast requests using CoAP for the well-known link-format
resources could be used to perform denial of service on a constrained
network. A multicast request SHOULD only be accepted if the request
is sufficiently authenticated and secured using, e.g., IPsec or an
appropriate object security mechanism.

CoRE Link Format parsers should be aware that a link description may
be cyclical, i.e., contain a link to itself. These cyclical links
could be direct or indirect (i.e., through referenced link
resources). Care should be taken when parsing link descriptions and
accessing cyclical links.

7. IANA Considerations

7.1. Well-Known 'core' URI

This memo registers the 'core' well-known URI in the Well-Known URIs
registry as defined by [RFC5785].

URI suffix: core

Change controller: IETF

Specification document(s): RFC 6690

Related information: None

7.2. New 'hosts' Relation Type

This memo registers the new "hosts" Web Linking relation type as per
[RFC5988].

Relation Name: hosts

Description: Refers to a resource hosted by the server indicated by
the link context.

Reference: RFC 6690

Notes: This relation is used in CoRE where links are retrieved as a
"/.well-known/core" resource representation and is the default
relation type in the CoRE Link Format.

Application Data: None

7.3. New 'link-format' Internet Media Type

   This memo registers the a new Internet media type for the CoRE Link
   Format, 'application/link-format'.

   Type name: application

   Subtype name: link-format

   Required parameters: None

   Optional parameters: None

   Encoding considerations: Binary data (UTF-8)

   Security considerations:

   Multicast requests using CoAP for the well-known link-format
   resources could be used to perform denial of service on a constrained
   network. A multicast request SHOULD only be accepted if the request
   is sufficiently authenticated and secured using, e.g., IPsec or an
   appropriate object security mechanism.

   CoRE Link Format parsers should be aware that a link description may
   be cyclical, i.e., contain a link to itself. These cyclical links
   could be direct or indirect (i.e., through referenced link
   resources). Care should be taken when parsing link descriptions and
   accessing cyclical links.

   Interoperability considerations: None

   Published specification: RFC 6690

   Applications that use this media type: CoAP server and client
   implementations for resource discovery and HTTP applications that use
   the link-format as a payload.

   Additional information:

   Magic number(s):

   File extension(s): *.wlnk

   Macintosh file type code(s):

   Intended usage: COMMON

   Restrictions on usage: None

Author: CoRE WG

Change controller: IETF

7.4. Constrained RESTful Environments (CoRE) Parameters Registry

This specification establishes a new Constrained RESTful Environments
(CoRE) Parameters registry, which contains two new sub-registries of
Link Target Attribute values (defined in [RFC5988]), one for Resource
Type (rt=) Link Target Attribute values and the other for Interface
Description (if=) Link Target Attribute values. No initial entries
are defined by this specification for either sub-registry.

For both sub-registries, values starting with the characters "core"
are registered using the IETF Review registration policy [RFC5226].
All other values are registered using the Specification Required
policy, which requires review by a designated expert appointed by the
IESG or their delegate.

The designated expert will enforce the following requirements:

o Registration values MUST be related to the intended purpose of
   these attributes as described in Section 3.

o Registered values MUST conform to the ABNF reg-rel-type definition
   of Section 2, meaning that the value starts with a lowercase
   alphabetic character, followed by a sequence of lowercase
   alphabetic, numeric, ".", or "-" characters, and contains no white
   space.

o It is recommended that the period "." character be used for
   dividing name segments and that the dash "-" character be used for
   making a segment more readable. Example Interface Description
   values might be "core.batch" and "core.link-batch".

o URIs are reserved for free use as extension values for these
   attributes and MUST NOT be registered.

Registration requests consist of the completed registration template
below, with the reference pointing to the required specification. To
allow for the allocation of values prior to publication, the
designated expert may approve registration once they are satisfied
that a specification will be published.

Note that Link Target Attribute Values can be registered by third
parties if the Designated Expert determines that an unregistered Link
Target Attribute Value is widely deployed and not likely to be
registered in a timely manner.

The registration template for both sub-registries is:

o Attribute Value:

o Description:

o Reference:

o Notes: [optional]

Registration requests should be sent to the core-parameters@ietf.org
mailing list, marked clearly in the subject line (e.g., "NEW RESOURCE
TYPE - example" to register an "example" relation type or "NEW
INTERFACE DESCRIPTION - example" to register an "example" Interface
Description).

Within at most 14 days of the request, the Designated Expert(s) will
either approve or deny the registration request, communicating this
decision to the review list and IANA. Denials should include an
explanation and, if applicable, suggestions as to how to make the
request successful.

Decisions (or lack thereof) made by the Designated Expert can be
first appealed to Application Area Directors (contactable using the
app-ads@tools.ietf.org email address or directly by looking up their
email addresses on http://www.iesg.org/ website) and, if the
appellant is not satisfied with the response, to the full IESG (using
the iesg@ietf.org mailing list).

8. Acknowledgments

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.

   [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.

   [RFC3629]   Yergeau, F., "UTF-8, a transformation format of ISO
               10646", STD 63, RFC 3629, November 2003.

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

   [RFC4288]   Freed, N. and J. Klensin, "Media Type Specifications and
               Registration Procedures", BCP 13, RFC 4288, December 2005.

   [RFC5226]   Narten, T. and H. Alvestrand, "Guidelines for Writing an
               IANA Considerations Section in RFCs", BCP 26, RFC 5226,
               May 2008.

   [RFC5234]   Crocker, D. and P. Overell, "Augmented BNF for Syntax
               Specifications: ABNF", STD 68, RFC 5234, January 2008.

   [RFC5646]   Phillips, A. and M. Davis, "Tags for Identifying
               Languages", BCP 47, RFC 5646, September 2009.

   [RFC5987]   Reschke, J., "Character Set and Language Encoding for
               Hypertext Transfer Protocol (HTTP) Header Field
               Parameters", RFC 5987, August 2010.

   [RFC5988]   Nottingham, M., "Web Linking", RFC 5988, October 2010.

   [RFC6570]   Gregorio, J., Fielding, R., Hadley, M., Nottingham, M.,
               and D. Orchard, "URI Template", RFC 6570, March 2012.

9.2. Informative References

   [COAP]   Shelby, Z., Hartke, K., Bormann, C., and B. Frank,
               "Constrained Application Protocol (CoAP)", Work in
               Progress, July 2012.

   [REST]   Fielding, R., "Architectural Styles and the Design of
               Network-based Software Architectures", 2000,
               <http://www.ics.uci.edu/~fielding/pubs/dissertation/
               top.htm>.

   [RFC1034]   Mockapetris, P., "Domain names - concepts and facilities",
               STD 13, RFC 1034, November 1987.

   [RFC1035]   Mockapetris, P., "Domain names - implementation and
               specification", STD 13, RFC 1035, November 1987.

   [RFC2045]   Freed, N. and N. Borenstein, "Multipurpose Internet Mail
               Extensions (MIME) Part One: Format of Internet Message
               Bodies", RFC 2045, November 1996.

   [RFC2231]   Freed, N. and K. Moore, "MIME Parameter Value and Encoded
               Word Extensions: Character Sets, Languages, and
               Continuations", RFC 2231, November 1997.

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

   [RFC4919]   Kushalnagar, N., Montenegro, G., and C. Schumacher, "IPv6
               over Low-Power Wireless Personal Area Networks (6LoWPANs):
               Overview, Assumptions, Problem Statement, and Goals",
               RFC 4919, August 2007.

   [RFC5785]   Nottingham, M. and E. Hammer-Lahav, "Defining Well-Known
               Uniform Resource Identifiers (URIs)", RFC 5785,
               April 2010.

   [RFC6454]   Barth, A., "The Web Origin Concept", RFC 6454,
               December 2011.

   [W3C.HTML.4.01]
               Raggett, D., Le Hors, A., and I. Jacobs, "HTML 4.01
               Specification", World Wide Web Consortium
               Recommendation REC-html401-19991224, December 1999,
               <http://www.w3.org/TR/1999/REC-html401-19991224>.

   [WADL]   Hadley, M., "Web Application Description Language (WADL)",
               2009, <http://java.net/projects/wadl/sources/svn/content/
               trunk/www/wadl20090202.pdf>.

Author's Address

   Zach Shelby
   Sensinode
   Kidekuja 2
   Vuokatti 88600
   Finland

   Phone: +358407796297
   EMail: zach@sensinode.com