LRDD: Link-based Resource Descriptor Discovery
draft-hammer-discovery-06

Abstract

LRDD (pronounced ‘lard’) provides a process for obtaining information about a resource identified by a Uniform Resource Identifier (URI). The ‘information about a resource’ - a resource descriptor - provides machine-readable information that aims to increase interoperability and enhance interactions with the resource. LRDD provides a narrow and well-defined set of rules for obtaining and processing link-based descriptors (found in multiple sources such as HTTP headers, document markup, and resource descriptors) which are often required for security and consistent client behavior.

Editorial Note (to be removed by RFC Editor)

Please discuss this draft on the apps-discuss@ietf.org [1] mailing list.

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 November 29, 2010.

Copyright Notice

Copyright (c) 2010 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. Specification Status ........................................... 3
2. Introduction .................................................... 3
   2.1. Example .................................................... 4
   2.2. Notational Conventions ..................................... 7
3. Link Source Priority Order ........................................ 7
4. Resource Descriptor Construction .................................. 8
5. Link Sources ....................................................... 9
   5.1. host-meta Document ......................................... 10
   5.2. HTTP Response Headers ...................................... 10
   5.3. Document Markup ........................................... 11
6. Security Considerations ........................................... 12
7. IANA Considerations .............................................. 12
   7.1. The ‘lrdd’ Relation Type .................................... 12
Appendix A. Acknowledgments ......................................... 13
Appendix B. Document History ........................................ 13
8. References ......................................................... 15
   8.1. Normative References ....................................... 15
   8.2. Informative References .................................... 16
Author’s Address ..................................................... 16
1. Specification Status

This specification was retired in favor of a single, simpler document. Some of the functionality was retained merged into [I-D.hammer-hostmeta] where future work is done.

2. Introduction

LRDD defines a simple process for locating resource descriptors for URI-identified resources. Resource descriptors are machine-readable documents that provide information about resources (resource metadata) for the purposes of promoting interoperability and assisting in interacting with unknown resources that support known interfaces.

For example, a web page about an upcoming meeting can provide in its descriptor the location of the meeting organizer’s free/busy information to potentially negotiate a different time. A social network profile page descriptor can identify the location of the user’s address book as well as accounts on other sites. A web service implementing an API with optional components can advertise which of these are supported.

Given the wide range of metadata needs, no single descriptor format or retrieval method can adequately accommodate every use case. While there are many methods for obtaining resource descriptors (e.g., links, HTTP headers, WebDAV’s PROPFIND [RFC4918], HTTP OPTIONS, URIQA’s MGET [URIQA]), LRDD utilizes the Web Linking framework defined in [I-D.nottingham-http-link-header]. LRDD defines a narrow profile of Web Linking for obtaining and processing link-based descriptors to accommodate the common discovery needs of many Web protocols.

In LRDD, the resource descriptor is not a single document but the aggregation of links obtained from three link sources (when applicable and available). The resource descriptor is constructed by determining the order in which to process the three link sources and aggregating the links and metadata contained in each source:

- host-meta document - links generated by applying the resource URI to the link templates provided by the host’s host-meta document as defined in [I-D.hammer-hostmeta].
- HTTP response header - links included in the HTTP response header to an HTTP [RFC2616] "HEAD" or "GET" resource representation request using the "Link" header field defined in [I-D.nottingham-http-link-header].
2.1. Example

In this example, an article published by a website allows readers to post comments and provide the web address of their own blog. The article page includes an avatar (a photo of the reader) when displaying comments. The photo is obtained by performing discovery on the web address provided by the reader, if supported by the reader’s blog.

After receiving a comment from Jane which has her own blog at "http://jane.example.com/blog", the website performs LRDD discovery to try and obtain Jane’s photo.

First, the website determines the source priority order - the order in which it looks for links in the various sources - for Jane’s blog by fetching its host-meta document from "https://jane.example.com/.well-known/host-meta":

```xml
<?xml version='1.0' encoding='UTF-8'?>
<XRD xmlns='http://docs.oasis-open.org/ns/xri/xrd-1.0'
     xmlns:hm='http://host-meta.net/ns/1.0'>
  <hm:Host>jane.example.com</hm:Host>
  <Property type='http://lrdd.net/priority/resource' />
  <Link rel='lrdd' template='http://jane.example.com?lrdd={uri}' />
  <Link rel='contents' template='http://example.com?c={uri}' />
</XRD>
```

which indicates the blog is using Resource-priority (giving higher priority to links provided by the resource itself over those defined...
by the global host policy). Since Jane’s blog uses Resource- 
priority, the website looks for "avatar" type links in this order: 
<Link> elements in the blog document’s markup, HTTP Link headers in 
the blog document HTTP response, and last in the host-meta document 
using link templates. Note that the "avatar" relation type is used 
for illustration purposes only and at this time is not a registered 
relation type.

Figure 1

To obtain a markup representation of Jane’s blog, the website makes 
an HTTP "GET" request to "http://jane.example.com/blog":

GET /blog HTTP/1.1
Host: jane.example.com
Accept: text/html

And receives back (HTML schema simplified for display purposes):

HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
Link: <http://jane.example.com/author>; rel='author'

<HTML>
  <HEAD>
    <LINK href='http://jane.example.com/image' rel='avatar' />
  </HEAD>
  <BODY>
    <H1>Jane’s Blog</H1>
  </BODY>
</HTML>

The document’s HTML markup includes the desired link. The website 
can fetch Jane’s photo from "http://jane.example.com/image".

After obtaining Jane’s photo, the website looks for a short 
description of Jane to include with her comment. It performs another 
LRDD discovery, this time looking for an "author" link.

Repeating the same process, the website looks for qualified <LINK> 
elements in Jane’s blog HTML markup and finds none. It then looks 
for a LRDD document – a descriptor document containing additional 
information about the resource – which uses the "lrdd" link relation. 
It finds none in the HTML document.
In Resource-priority, the next source is the HTTP header included in the resource representation response. The HTTP header (shown above with the HTML response) includes a qualified link to Jane’s author page.

HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
Link: <http://jane.example.com/author>; rel='author'

Before the website can display Jane’s photo and description, it needs to find the copyright license used by Jane’s blog. It performs another LRDD discovery looking for a link with a "copyright" relation type. It fails to find such link in the HTML markup as well as a markup link to a LRDD document. It tries and fails to find a "copyright" link in the HTTP response header or a link to a LRDD document.

After exhausting the first two sources, the website proceeds to the host-meta source (Figure 1), and looks for link templates. It fails to find a link to a copyright statement, but it does find a link to a LRDD document:

    <Link rel='lrdd' template='http://jane.example.com?lrdd={uri}' />

The website obtains the LRDD document for Jane’s blog by applying the blog’s URI to the template:


and obtains the LRDD document for Jane’s blog:

<?xml version='1.0' encoding='UTF-8'?>
<XRD xmlns='http://docs.oasis-open.org/ns/xri/xrd-1.0'>
    <Subject>http://jane.example.com/blog</Subject>
    <Property type='http://example.com/version'>2.0</Property>

    <Link rel='copyright' href='http://jane.example.com/copyright' />
</XRD>
The LRDD document provided by the host-meta link template includes a link to the copyright statement. The website now has all the information it needs to display Jane’s comment along with her photo and description on the article page.

2.2. Notational Conventions

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].

3. Link Source Priority Order

The client MUST first determine the host source priority order. To ensure consistent client behavior and to enable hosts to set their own security policy with regard to metadata authority, LRDD provides two processing profiles:

- Host priority - priority is given to links set by the host policy (via host-meta and HTTP response headers) over those set by each individual resources (via the document markup). The client MUST process the three sources in the following order: host-meta document, HTTP response header, and document markup.

- Resource priority - priority is given to the individual resource over the policies of the host. The client MUST process the three sources in the following order: document markup, HTTP response header, and host-meta document.

Host priority is the default source priority order. Hosts that wish to use resource priority MUST declare it by setting the LRDD priority property in their host-meta document: "http://lrdd.net/priority/resource". The priority property element does not have a value.

For example:

```xml
<?xml version='1.0' encoding='UTF-8'?>
<XRD xmlns='http://docs.oasis-open.org/ns/xri/xrd-1.0'
     xmlns:hm='http://host-meta.net/ns/1.0'>
  <hm:Host>example.com</hm:Host>
  <Property type='http://lrdd.net/priority/resource'/>  
</XRD>
```
4. Resource Descriptor Construction

To construct the resource descriptor, the client determines the source priority order as described in Section 3, and processes each of the three sources (Section 5) as follows:

1. Obtain and processes the source to produce an ordered list of links as described for each source in Section 5.

2. Add any link found to the resource descriptor links list in order, except for any link using the "lrdd" relation type.

3. For each link using the "lrdd" relation type and "application/xrd+xml" media type (if present):
   A. Obtain the LRDD document by following the scheme-specific rules for the LRDD document URI. If the document URI scheme is "http" or "https", the document is obtained via an HTTP "GET" request to the identified URI. If the HTTP response status code is 301, 302, or 307, the client MUST follow the redirection response and repeat the request with the provided location. If a redirection response results in another 301, 302, or 307 response, the client SHOULD repeat the request with the provided location as many times as reasonable, making sure not to enter into a loop if a location URI repeats itself. The client MUST only process the document if it was received with an HTTP 200 (OK) status code and is a valid XRD document per [OASIS.XRD-1.0].
   B. Add any link found in the LRDD document to the resource descriptor links list in order, except for any link using the "lrdd" relation type. When adding links, the client SHOULD retain any extension attributes and child elements if present (e.g. <Property> or <Title> elements).
   C. Add any resource properties found in the LRDD document to the resource descriptor metadata list in order (e.g. <Alias> or <Property> child elements of the <XRD> root element).
For example, the resource descriptor of Jane’s blog as described in Section 2.1 and expressed as an XRD document (for illustration purposes only) is:

```xml
<?xml version='1.0' encoding='UTF-8'?>
<XRD xmlns='http://docs.oasis-open.org/ns/xri/xrd-1.0'>
  <Subject>http://jane.example.com/blog</Subject>
  <Property type='http://example.com/version'>2.0</Property>
  <Link rel='avatar' href='http://jane.example.com/image' />
  <Link rel='author' href='http://jane.example.com/author' />
  <Link rel='contents' href='http://example.com?c=http%3A%2F%2Fjane.example.com%2Fblog' />
  <Link rel='copyright' href='http://jane.example.com/copyright' />
</XRD>
```

Using the same example, if Jane’s host-meta used host priority instead of resource priority, the resource descriptor of Jane’s blog would be:

```xml
<?xml version='1.0' encoding='UTF-8'?>
<XRD xmlns='http://docs.oasis-open.org/ns/xri/xrd-1.0'>
  <Subject>http://jane.example.com/blog</Subject>
  <Property type='http://example.com/version'>2.0</Property>
  <Link rel='contents' href='http://example.com?c=http%3A%2F%2Fjane.example.com%2Fblog' />
  <Link rel='copyright' href='http://jane.example.com/copyright' />
  <Link rel='author' href='http://jane.example.com/author' />
  <Link rel='avatar' href='http://jane.example.com/image' />
</XRD>
```

5. Link Sources

Each of the link sources supported by LRDD presents a different set of requirements and benefits. The criteria used to determine which sources a server supports are based on a combination of factors:

- The ability to offer and obtain a representation of the resource by dereferencing its URI.
The availability of a markup document representation supporting
<LINK> elements compatible with [I-D.nottingham-http-link-header].

The availability of an HTTP representation of the resource and the
ability to provide and access link information in its response
header.

The ability to offer and process a host-meta document.

5.1. host-meta Document

The host-meta document source is available for any resource
identified by a URI with an authority that supports the host-meta
document as defined in [I-D.hammer-hostmeta]. This method does not
require obtaining any representation of the resource, and operates
solely using the resource URI.

Links between the resource URI and other resources are expressed
using link templates as defined by [I-D.hammer-hostmeta] section 3.2.
By applying the host-wide templates to an individual resource URI, a
resource-specific link is generated which can be used to express
links without the need to access or provide a representation for the
resource.

The client processes the host-meta document, searches for link
templates and applies the resource URI to produce a set of links.
The client MUST retain the order of links as present in the host-meta
document. Any links contained in the host-meta document not using
the "template" attribute MUST be ignored and excluded from the
resource descriptor (they are still valid for other purposes).

For example, the resource URI "http://example.com/x" and the
following host-meta link template:

    <Link rel='author' template='http://example.com?author={uri}' />

generate an "author" link between "http://example.com/x" and
"http://example.com?author=http%3A%2F%2Fexample.com%2Fx".

5.2. HTTP Response Headers

The HTTP response header source is limited to resources for which an
HTTP "GET" or "HEAD" request returns a valid representation of the
resource. This source uses the "Link" header field defined in
[I-D.nottingham-http-link-header] and requires the retrieval of a
resource representation header.
For example:

```
Link: <http://example.com?author=http%3A%2F%2Fexample.com%2Fx>; rel='author'
```

The client obtains the HTTP response header links by making an HTTP (or HTTPS) "GET" or "HEAD" request to the resource URI.

If the HTTP response status code is 301, 302, or 307, the client MUST follow the redirection response and repeat the request with the provided location. If a redirection response results in another 301, 302, or 307 response, the client SHOULD repeat the request with the provided location as many times as reasonable, making sure not to enter into a loop if a location URI repeats itself.

The client MUST only process the header if the HTTP response includes a 200, 204, 206, or 304 status code. The client processes the response header and searches for "Link" header fields. The client MUST retain the order of links as present in the response header.

Link headers can include multiple relation types in a single "rel" attribute (for example "rel="license copyright"""). The client MUST properly process such multiple relation types "rel" attributes as defined by [I-D.nottingham-http-link-header].

5.3. Document Markup

The document markup source is limited to resources with an available markup representation that supports typed relations using the `<LINK>` element, such as HTML [W3C.REC-html401-19991224], XHTML [W3C.REC-xhtml1-20020801], and Atom [RFC4287]. Other markup formats are permitted as long as the semantics of their `<LINK>` elements are fully compatible with the link framework defined in [I-D.nottingham-http-link-header].

For example:

```
<LINK href='http://example.com?author=http%3A%2F%2Fexample.com%2Fx' rel='author'>
```

The client obtains the document markup by retrieving a representation of the resource using the applicable transport for that resource URI. When using HTTP (or HTTPS), the client obtains the document markup by making a "GET" request to the resource URI.
If the HTTP response status code is 301, 302, or 307, the client MUST follow the redirection response and repeat the request with the provided location. If a redirection response results in another 301, 302, or 307 response, the client SHOULD repeat the request with the provided location as many times as reasonable, making sure not to enter into a loop if a location URI repeats itself.

The client MUST only process the document markup if the HTTP response includes a 200 (OK) status code. The client processes the document markup and searches for "LINK" elements. The client MUST retain the order of links as present in the document markup.

The client MUST obey the document markup schema and ignore any invalid elements (such as <LINK> elements outside the <HEAD> section of an HTML document). This is done to avoid unintentional markup from other parts of the document to be used for discovery purposes, which can have vast impact on usability and security.

Some <LINK> elements allow multiple relation types in a single "rel" attribute (for example "rel='license copyright'’"). The client MUST properly process such multiple relation "rel" attributes as defined by the format specification.

6. Security Considerations

The methods used to perform discovery are not secure, private or integrity-guaranteed, and due caution should be exercised when using them. Applications that perform LRDD SHOULD consider the attack vectors opened by automatically following, trusting, or otherwise using links gathered from document markups, HTTP response headers, or host-meta documents.

7. IANA Considerations

7.1. The ‘lrdd’ Relation Type

This specification registers the "lrdd" relation type in the Link Relation Type Registry defined by [I-D.nottingham-http-link-header]:

Relation Name: lrdd
Description: Identifies a resource descriptor for the link’s context used by the LRDD protocol.
Appendix A. Acknowledgments

Inspiration for this memo derived from previous work on a descriptor format called XRDS-Simple, which in turn derived from another descriptor format, XRDS. Previous discovery workflows include Yadis which is currently used by the OpenID community. While suffering from significant shortcomings, Yadis was a breakthrough approach to performing discovery using extremely restricted hosting environments, and this memo has strived to preserve as much of that spirit as possible.

The author wishes to thanks the OASIS XRI TC and WebFinger communities for their support, encouragement, and enthusiasm for this work. Special thanks go to Phil Archer, Lisa Dusseault, Joseph Holsten, Mark Nottingham, John Panzer, Drummond Reed, and Jonathan Rees for their invaluable feedback.

Appendix B. Document History

[[ to be removed by the RFC editor before publication as an RFC ]]

-06

- Marked as deprecated with some functionality moved to host-meta.

-05

- Many editorial changes, cleanup.

- Changed the processing flow to focus on a aggregation. Clients can always stop when they find what they need.

- Clarified redirections and HTTP status codes for valid document markups and response headers.

- Removed restrictions on multiple LRDD documents.

-04

- Changed focus to a narrow and well-defined discovery process.

- Removed analysis appendix and discussion of discovery types and removed informative references.
- Expanded the descriptor definition to include links as well as LRDD documents, moving away from the single-document approach.

- Moved the Link-Pattern field and template syntax to new host-meta draft.

- Updated references.

- Added example.

-03

- Added protocol name LRDD (pronounced ‘lard’).

- Fixed Link-Pattern examples to include missing semicolons.

-02

- Changed focus from an HTTP-based process to Link-based process.

- Completely revised and restructured document for better clarity.

- Realigned the methods to produce consistent results and changed the way redirections and client-errors are handled.

- Updated to use newer version of site-meta, now called host-meta, including a new plaintext-based format to replace the previous XML format.

- Renamed Link-Template to Link-Pattern to avoid future conflict with a previously proposed Link-Template HTTP header.

- Removed support for the ‘scheme’ Link-Template parameter.

- Replaced restrictions with interoperability recommendations.

- Added IANA considerations per new host-meta registry requirements.

-01

- Rename ‘resource discovery’ to ‘descriptor discovery’.

- Added informative reference to Metalink.

- Clarified that the resource descriptor URI can use any URI scheme, not just ‘http’ or ‘https’.
o Removed comment regarding redirects when using <LINK> Elements.

o Clarified that HTTPS must be used with 'https' URIs for both Link headers and host-meta retrieval.

o Removed DNS verification step for host-meta with schemes other then 'http' and 'https'. Replaced with a general discussion of authority and a security consideration comment.

o Organized host-meta section into another sub-section level.

o Enlarged the template vocabulary from a single ‘uri’ variable to include smaller URI components.

o Added informative reference to RFC 2295 in analysis appendix.

-00

o Initial draft.

8. References

8.1. Normative References

[I-D.hammer-hostmeta]
Hammer-Lahav, E., "host-meta: Web Host Metadata",
draft-hammer-hostmeta-08 (work in progress), May 2010.

[I-D.nottingham-http-link-header]
Nottingham, M., "Web Linking",
draft-nottingham-http-link-header-10 (work in progress),
May 2010.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H.,
Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext


[W3C.REC-html401-19991224]
Hors, A., Raggett, D., and I. Jacobs, "HTML 4.01
Specification", World Wide Web Consortium
Recommendation REC-html401-19991224, December 1999,
8.2. Informative References


URIs


Author’s Address

Eran Hammer-Lahav
Yahoo!

Email: eran@hueniverse.com
URI:  http://hueniverse.com